Dynamic Application and Development

Skylar One version 12.5.21

Table of Contents

Dynamic Application Development Overview **20**

What Is A Dynamic Application? 22

Types of Dynamic Applications 22

Core Elements of a Dynamic Application 24

Optional Features for Dynamic Applications 26

Managing Dynamic Applications 29

Viewing the List of Dynamic Applications 29

Performing Other Tasks in the Dynamic Application Manager Page 32

Viewing the Dynamic Applications Associated with a Device 34

Viewing the Status of a Dynamic Application 35

Found 36

Collect 36

How Skylar One Manages Collect Status 37

Starting Collection 37

Collection Objects that are Excluded from Maintenance 38

Status of Objects for Deviation 38

Manually Associating a Dynamic Application with a Device 39

Editing the Credential Associated with a Dynamic Application 40

Performing Other Administrative Tasks for an Aligned Dynamic Application 41

Enabling or Disabling Objects 41

Restarting Automatic Maintenance of Collection Objects 42

Editing the Poll Frequency for a Dynamic Application on the Current Device 42

Stopping Data Collection for a Dynamic Application 43

Resetting Statistical Data for a Dynamic Application 43

Resetting Persistent Session Objects for a Dynamic Application 44

Testing Data Collection for a Dynamic Application 45

Removing Data Collected by a Dynamic Application 45

Bulk Un-aligning Dynamic Applications 46

Setting Thresholds for Dynamic Applications 46

Dynamic Applications and Discovery 47

How Does Skylar One Align Dynamic Applications During Discovery? 47

Queuing Discovery from the Dynamic Applications Manager Page 48
System Settings That Affect Dynamic Application Discovery 48
Viewing Dynamic Application Data and Alerts 51
Performance Dynamic Applications 52
Viewing Performance Dynamic Application Data 53
Viewing Details with the Data Table 53
Configuring the Data for the Report 53
Generating a Report from a Performance Graph 54
Defining the Date Range for a Report 55
Configuration Dynamic Applications 55
Selecting Data to View 56
Viewing Data 56
Generating a Report of the Data 57
Viewing Historical Data 57
Editing the Application 58
Journal Dynamic Applications 58
Selecting Data to View 58
Viewing Data 59
Generating a Report of the Data 59
Editing the Application 60
Viewing Alerts Generated by Dynamic Applications 60
Searching the List of Log Entries for a Dynamic Application 61
Viewing Active Events for a Dynamic Application 62
Viewing Details About an Active Event 64
Viewing Cleared Events from a Dynamic Application 65
Dynamic Application Settings 67
Dynamic Application Properties 67
The Abandon Collection Property 72
Creating a Dynamic Application 73
Creating the Container for a Dynamic Application 76
Duplicating an Existing Dynamic Application 76
Creating a Dynamic Application Using the SNMP Walker Tool 77

Collection Objects ... 77
What is a Collection Object? ... 77
How Does a Collection Object Work? ... 77
How Collection Objects are Used by Skylar One ... 78
Viewing the Collection Objects in a Dynamic Application ... 79
Creating a Collection Object ... 80
Basic Settings ... 81
Collection Settings ... 83
Grouping & Indexing Settings ... 84
Configuration Dynamic Application Settings ... 85
Custom Attribute Settings ... 88
Standard Deviation Settings ... 90
Dynamic Component Mapping Settings ... 91
Creating a Discovery Object ... 92
Editing a Collection Object ... 94
Performing Bulk Actions on Collection Objects ... 95
Caching ... 95
What is Caching? ... 95
Caching Responses ... 96
Consuming Cached Responses ... 96
Configuring Collection Objects in Cache-Consuming Dynamic Applications ... 97
SOAP Dynamic Applications ... 97
WMI Dynamic Applications ... 97
PowerShell Applications ... 98
XML Dynamic Applications ... 98
XSLT Dynamic Applications ... 98
Collector Affinity for Dynamic Applications That Use Caching ... 99
Dynamic Component Mapping ... 99
What is Dynamic Component Mapping? ... 99
Where in Skylar One are Component Devices Displayed? ... 100
Configuring a Dynamic Application to Create Component Devices ... 100
Configuring Collection Objects as Component Identifiers ... 101

Duplicate MAC Addresses for Component Devices 103
Component Devices, Collector Groups, and Load Balancing 104
Collector Affinity 105
Failover for Collector Groups for Component Devices 106
Merging Component Devices 106
Availability for Component Devices 107
Moving Component Devices Between Root Devices 108
Automatically Aligning Dynamic Applications with Component Devices 109
Automatically Assigning a Device Class to Each Component Device 110
Configuring a Device Class to Match Collected Values 110
Configuring Component Identifiers for the Collection Objects 110
How Does Skylar One Use Component Identifiers to Automatically Assign a Device Class to Each Component Device? 111
Aligning a Default Device Class with a Dynamic Application 111
Defining a Default Device Class in the Device Class Editor 112
Defining a Default Device Class in the Dynamic Application 112
Variables in Dynamic Applications for Component Devices 112
Caching and Component Mapping 113
Dynamic Application Relationships 113
Configuring Collection Objects for Relationships 114
Configuring Identity-Based Relationships 116
Viewing a Map of Component Devices 117
Viewing a Component Map 118
Adding Devices from Another Component Tree to the Component Map 118
Viewing Relationships Maps in the Organization View 120
Viewing an Organizational Map 120
Viewing Relationships in Customized Maps 121
Viewing Relationships for a Single Device 122
Presentation Objects for Performance Dynamic Applications 124
What is a Presentation Object? 124
Viewing the Presentation Objects in a Dynamic Application 124
Creating a Presentation Object 125

Presentation Object Formulas 126
Vitals Linking 127
Editing a Presentation Object 127
Deleting a Presentation Object 128
Alerts and Thresholds 128
What is an Alert? 128
What is a Threshold? 128
Viewing the Thresholds in a Dynamic Application 128
Creating a Threshold 129
Editing a Threshold 130
Deleting a Threshold 130
Viewing the Alerts in a Dynamic Application 131
Creating an Alert 131
Alert Formulas 132
Evaluate 133
Operators 134
Dividing Integers 135
Using Quotes in an Alert 135
The result() Function 135
The threshold() Function 137
The active() Function 138
The avg() Function 138
The deviation() Function 139
The find() Function 142
The global() Function 142
The log() Function 144
The prior() Function 144
The round() Function 144
The sum() Function 145
Date & Time Variables 145
The tab_idx Variable 147
Creating an Event Policy for an Alert 148

Editing an Alert 149
Deleting an Alert 149
Validating an Alert 150
Running the Alert Validator Command Line Tool 150
Example 151
Indexing 152
What is Dynamic Application Indexing? 152
How Indexing Affects Configuration Tables 153
How Indexing Affects Performance Graphs 153
How Indexing Affects Alerts 155
The Default Indexing Method 156
Instance Values and Index Creation for SNMP Dynamic Applications 157
Instance Values and Index Creation for Snippet Dynamic Applications 157
Instance Values and Index Creation for Other Dynamic Applications 158
Designating an Index 158
System Settings that Affect Indexing 162
Grouping Dynamic Application Data Using Collection Labels 162
What are Collection Labels and Collection Groups? 163
Viewing the List of Collection Labels 163
Creating a Collection Group 164
Creating a Collection Label 164
What is Normalization? 164
What are Duplicates and How Does Skylar One Manage Them? 167
What is Precedence? 168
Aligning a Presentation Object with a Collection Label 168
Viewing and Managing the List of Presentation Objects Aligned with a Collection Label 169
Viewing and Editing Duplicate Presentation Objects by Collection Label 170
Viewing and Managing the List of Devices Aligned with a Collection Label 170
Editing Duplicate Presentation Objects by Device 171
Editing Duplicate Presentation Objects for a Single Device 172
Editing a Collection Label 172
Deleting a Collection Label 173

Viewing Reports About Collection Labels on a Single Device ... 173
Viewing Dashboards About Collection Labels ... 173
Snippet Dynamic Application Development ... 174
What is a Snippet Dynamic Application? ... 175
Relationship Between the Dynamic Application Builder and Snippet Dynamic Applications ... 175
Common Dynamic Application Elements ... 176
Execution Environments ... 176
Prerequisites ... 176
About this Chapter ... 176
Performance and Configuration Snippets ... 177
Collection Objects ... 177
Snippet Code ... 177
Populating Collection Objects ... 178
Available Variables ... 181
Available Variables for Bulk Performance and Bulk Configuration Snippets ... 182
Using Credentials ... 183
Reporting Collection Status ... 186
Generating Alerts ... 186
Generating an Internal Alert ... 186
Generating a Custom Alert ... 187
Self-Reporting Collection Objects ... 188
Writing Debug Information to silo.log ... 188
Developing a Journal Snippet Dynamic Application ... 189
Requirements ... 189
Creating the Dynamic Application ... 190
Creating the Collection Objects ... 191
Snippet Code ... 192
Creating the Presentation Objects ... 199
Creating a Credential and Using the Dynamic Application ... 200
Full Snippet Code Listing ... 202
Caching and Cache-consuming Snippets ... 206
Caching a Result in a Snippet ... 207

Using a Cached Result in a Snippet 207
Excluded Modules and Additional Functions 208
Excluded Python Modules 208
Snippet Functions for Database, SNMP, and CURL Handles 209
Database Handles 209
SNMP Handles 210
cURL Handles 210
Snippet Functions for Polling Nagios Agents 211
Snippet Functions for Performing WMI and WBEM Requests 212
Caching Results Between Polling Periods 214
Caching Values 214
Retrieving Cached Values 215
Example Snippet Dynamic Applications 216
Example 1: A Random Number Generator Dynamic Application 216
Creating the Dynamic Application 216
Creating the Collection Objects 217
Snippet Code 218
Using the Dynamic Application 219
Full Snippet Code Listing 220
Example 2: Using Snippets to Collect SNMP Data 221
Creating the Dynamic Application 221
Creating a Container for the Snippet 222
Creating the Collection Objects 222
Full Snippet Code 223
Creating the Presentation Object 224
Snippet Walkthrough 225
Using the Dynamic Application 227
Aligning the Dynamic Application with a Device 228
Viewing Performance Data 229
Example 3: Using Telnet to Collect Performance Data 230
Creating the Dynamic Application 231
Creating the Collection Objects 232

Snippet Code ... 233
Using the Dynamic Application ... 234
Full Snippet Code Listing ... 236
Using Snippets to Collect Database Data ... 237
Creating the Dynamic Application ... 238
Creating a Container for the Snippet ... 238
Creating the Collection Objects ... 238
Snippet Code Walkthrough ... 240
Adding the Snippet Code ... 242
Using the Dynamic Application ... 244
Define the Credential ... 244
Align the Dynamic Application with a Device ... 244
Viewing Data ... 245
SNMP Dynamic Applications ... 246
What is SNMP? ... 247
Prerequisites ... 247
What is an SNMP Dynamic Application? ... 247
Elements of an SNMP Dynamic Application ... 247
What is Concurrent SNMP Collection? ... 247
Concurrent SNMP Collection ... 248
Using Concurrent SNMP Collection ... 248
Enabling Concurrent SNMP Collection ... 249
Enabling a Collector Group to Use Concurrent SNMP Collection ... 249
SNMP Collection Objects ... 250
Collection Objects ... 250
Extending the SNMP OID Field ... 251
Using Multiple OIDs and/or Existing Collection Objects to Define a Single Collection Object 251
Using Python Operators to Define a Collection Object ... 252
Using Variables in the SNMP Field to Specify Component Devices ... 252
Using Caching to Define a Collection Object ... 254
Filtering Results to Define a Collection Object ... 255
Defining "Fallback" Values for a Collection Object ... 256

Using Index Values and Parsed Index Values to Define a Collection Object 256
Viewing MIBs and Using the SNMP Walker Tool 257
The Management Information Base 258
Viewing MIBs 258
Viewing the Contents of a MIB file 259
Importing a MIB 259
Compiling a MIB 260
Viewing Error Logs for a MIB 261
Rebuilding the SNMP Tree 263
Exporting a MIB 263
Compiling Multiple MIB Files or Deleting Multiple MIB Files 263
The OID Browser 264
Drilling Down in OIDs 264
Searching the List of OIDs 264
Adding an OID to a Dynamic Application from the OID Browser 265
Synchronizing the Latest Compiled MIBs to Collectors 265
The SNMP Walker Tool 265
Adding an OID to a Dynamic Application 267
Creating an SNMP Performance Dynamic Application 268
Defining the Dynamic Application Properties 268
Adding the Discovery Object 268
Adding the Collection Objects 269
Creating the Presentation Object 271
Manually Aligning the Dynamic Application to a Device 272
Viewing the Report 272
Database Dynamic Applications 274
Prerequisites 275
What is a Database Dynamic Application? 275
How Do I Allow Skylar One to Access the Database? 276
What Can I Monitor with a Database Dynamic Application? 276
Can I Create My Own Database Dynamic Applications? 276
Elements of a Database Dynamic Application 276

Collection Objects ... 277
Creating a Dynamic Application ... 277
Creating Collection Objects for a Database Dynamic Application ... 277
Examples of Collection Objects ... 278
Example of a Database Performance Dynamic Application ... 279
Defining the Basic Properties for the Dynamic Application ... 280
Defining the Discovery Object for the Dynamic Application ... 280
Defining the Collection Objects ... 281
Defining the Presentation Objects ... 284
Defining the Threshold Objects ... 286
Defining the Alerts ... 288
Creating a Credential for the "MySQL: DBPerformance" Dynamic Application ... 291
Aligning the Dynamic Application with a Device ... 292
Viewing Reports for the Dynamic Application ... 293
Viewing Alerts for the Dynamic Application ... 293
Changing the Threshold for a Subscriber Device ... 293
Example of a Dynamic Application of Type "Database Configuration" ... 294
Defining the Basic Properties for the Dynamic Application ... 294
Defining the Discovery Object for the Dynamic Application ... 295
Defining the Collection Objects ... 296
Creating a Credential for the MySQL:DBConfiguration Dynamic Application ... 298
Aligning the Dynamic Application with a Device ... 298
Viewing the Configuration Report for the Dynamic Application ... 299
Example of a Dynamic Application with an Identity-Based Relationship ... 300
Building Dynamic Applications with an Identity-Based Relationship ... 300
Defining the Basic Properties for the Dynamic Application ... 301
Defining the Collection Objects ... 301
Creating the Dynamic Application that Creates the Relationship ... 302
Aligning the Dynamic Applications ... 303
Viewing Dynamic Application Relationships ... 304
Internal Collection Dynamic Applications ... 305
What is an Internal Collection Dynamic Application (ICDA)? ... 305

How Do ICDAs Work? 306
Types of ICDAs 306
Snippets and Execution Environments 307
Data Collected by ICDAs 307
Internal Collection Inventory Application Types 308
Filesystem Inventory 308
Interface Inventory 308
Internal Collection Performance Application Types 309
Availability 309
SNMP Detail 309
Filesystem Performance 309
Interface Performance 310
Port Performance 310
Process Inventory 310
Process Performance 310
Process Performance 311
Service Inventory 311
Service Performance 311
Creating Internal Collection Dynamic Applications (ICDAs) 311
Viewing the List of ICDAs 312
Creating an ICDA 312
Creating a Container for the ICDA Snippet 313
Creating the Collection Objects 314
Adding the Snippet Code 314
Snippet Examples for ICDAs 315
Availability 315
Discovery 316
Filesystem Inventory 317
Filesystem Type 319
Interface Octets 320
Latency 321
XML, SOAP, and XSLT Dynamic Application 322

XML, SOAP, and XSLT Protocols ... 323
What is XML? ... 323
Elements of XML, SOAP, and XSLT Dynamic Applications ... 323
SOAP and XSLT Requests ... 324
Viewing the Requests in a Dynamic Application ... 324
Creating a Request ... 325
Defining XSLT Request Code ... 326
The XSLT Request Code Field ... 326
The Cached XSLT Request Field ... 327
Defining XSLT Parser Code ... 327
Substitution Characters ... 329
Collection Object Substitution Characters ... 329
Substitution Characters from Credentials ... 329
Substitution Characters from Component Devices ... 330
Editing a Request ... 330
Deleting a Request ... 330
Collection Objects ... 331
Protocol-Specific Fields for Collection Objects ... 331
XML Dynamic Applications ... 331
SOAP Dynamic Applications ... 332
XSLT Dynamic Applications ... 332
Specifying XML Tags and SOAP Tags ... 332
Parsing an XML Element ... 332
Specifying XSLT Tags ... 334
Session ID Objects ... 334
Creating an XML Dynamic Application ... 335
Creating the Dynamic Application ... 335
Defining XML Data-Points to Monitor ... 336
Defining the Collection Objects ... 339
Creating the Presentation Object ... 340
Testing the Dynamic Application ... 341
Creating a Credential ... 341

Manually Aligning the Dynamic Application to the Test Device 341
Viewing the Reports 342
Dynamic Component Mapping & Caching with XSLT 342
Design 345
Creating the "Example Dynamic Component Mapping General" Dynamic Application 346
Defining the Dynamic Application Properties 346
Adding the XSLT Request 346
Adding the Discovery Object 348
Creating the "Example Dynamic Component Mapping Discovery" Dynamic Application 349
Defining the Dynamic Application Properties 349
Adding the XSLT Requests 350
Adding the Collection Objects 353
Creating a Device Class for the Component Devices 356
Creating the "Example Component Performance" Dynamic Application 357
Defining the Dynamic Application Properties 357
Adding the XSLT Request 357
Adding the Collection Objects 360
Editing the Presentation Objects 361
Automatically Aligning the Dynamic Application to Component Devices 361
Using the Dynamic Applications 362
Configuring a Test Device 362
Editing the Device Class for the Test Device 362
Creating a Credential 363
Discovering the Test Device 363
Verifying the Dynamic Application Alignments 364
Viewing the Device Components Registry 364
Viewing the Component Device Map 364
Viewing the Performance Graphs 365
Expanding this Example 365
WMI and PowerShell Dynamic Applications 369
What is WMI? 371
What is PowerShell? 371

Prerequisites ... 371
WMI and PowerShell Dynamic Applications ... 371
Elements of WMI and PowerShell Dynamic Application ... 371
WMI Requests ... 372
Defining a WMI Request ... 373
Example WMI Code ... 374
Defining a WBEM Request ... 374
Example WBEM Request Code ... 375
Editing a WMI Request ... 376
Editing a WBEM Request ... 376
Deleting a WMI or WBEM Request ... 376
PowerShell Requests ... 376
Defining a PowerShell Request ... 377
Editing a PowerShell Request ... 378
Deleting a PowerShell Request ... 379
Converting Legacy PowerShell Requests ... 379
WMI and PowerShell Collection Objects ... 380
WMI-Specific Fields for Collection Objects ... 380
PowerShell-Specific Fields for Collection Objects ... 381
Configuring Devices for Monitoring with WMI ... 381
Configuring WMI on Windows 2008 Servers ... 381
Step 1: Configuring Services ... 382
Step 2: Configuring the Windows Firewall ... 383
Step 3: Configuring a User Account and Permissions ... 384
Configuring Namespace and DCOM Security Permissions ... 384
Configuring User Account Control to Allow Elevated Permissions ... 392
Step 4: Configuring a Fixed Port for WMI ... 394
Configuring WMI for Windows Desktop Systems ... 394
Step 1: Configuring Services ... 395
Step 2: Configuring Windows Firewall ... 399
Step 3: Setting the Default Namespace Security ... 399
Step 4: Setting the DCOM Security Level ... 405

Step 5: Disabling User Account Control 412
Step 6: Configuring a fixed port for WMI 413
Configuring Devices for Monitoring with PowerShell 413
Prerequisites 413
Configuring PowerShell 414
Step 1: Configuring the User Account for Skylar One 414
Option 1: Creating an Active Directory Account with Administrator Access 415
Option 2: Creating a Local User Account with Administrator Access 415
Option 3: Creating a Non-Administrator Local User Account 415
Optional: Configuring the User Account for Remote PowerShell Access to Microsoft Exchange Server 417
Optional: Configuring the User Account for Remote PowerShell Access to Hyper-V Servers 418
Creating a User Group and Adding a User in Active Directory 418
Setting the Session Configuration Parameters and Group Permissions 418
Optional: Configuring the User Account for Access to Windows Failover Cluster 419
Step 2: Configuring a Server Authentication Certificate 419
Option 1: Using the Microsoft Management Console to Create a Self-Signed Authentication Certificate 420
Option 2: Using the MakeCert Tool to Create a Self-Signed Authentication Certificate 421
Option 3: Using PowerShell Commands to Create a Self-Signed Authentication Certificate . 422
Step 3: Configuring Windows Remote Management 422
Option 1: Using a Script to Configure Windows Remote Management 422
Option 2: Manually Configuring Windows Remote Management 425
Option 3: Using a Group Policy to Configure Windows Remote Management 428
Configuring an HTTPS Listener with GPO Configuration 446
Using Forward and Reverse DNS for Windows Remote Management 447
Step 4: Configuring a Windows Management Proxy 447
Step 5: Increasing the Number of PowerShell Dynamic Applications That Can Run Simultaneously 449
Optional PowerShell CLI Parameters 449
Creating a PowerShell Credential 450
Error Messages for PowerShell Collection 454
Concurrent PowerShell Collection 455

Prerequisites ... 455
Scope ... 455
Enabling and Disabling Concurrent PowerShell for Collector Groups ... 456
Enabling and Disabling Concurrent PowerShell on All Collector Groups ... 456
Enabling and Disabling Concurrent PowerShell on a Specific Collector Group ... 457
The Skylar One: Concurrent PowerShell Monitoring PowerPack ... 457
Aligning the "ScienceLogic: PowerShell Service Log Parser" Dynamic Application ... 457
Manually Aligning the Dynamic Application ... 458
Configuring the Device Template ... 459
Applying the Device Template ... 459
Aligning the "ScienceLogic: PowerShell Collector Performance" Dynamic Application ... 460
Enabling HTTPS Between Skylar One and the PowerShell Data Collector ... 461
Enabling and Disabling the Python PowerShell Remoting Protocol Client ... 462
Optional PowerShell CLI Parameters ... 463
Users with Windows 2008 R2 Servers ... 464
Scale Recommendations ... 464
Additional Scale Tips ... 464
Credentials for WMI and PowerShell Devices ... 465
Configuring a WMI Credential ... 465
Configuring a WMI Credential in the Classic Skylar One User Interface ... 466
Configuring a PowerShell Credential ... 467
Configuring a PowerShell Credential in the Classic Skylar One User Interface ... 470
Example: Creating a WMI Performance Dynamic Application ... 472
Defining the WMI Request ... 472
Adding the WMI Request ... 472
Adding the Collection Objects ... 473
Creating the Presentation Objects ... 475
Creating a Credential ... 477
Manually Aligning the Dynamic Application to a Device ... 478
Viewing the Performance Reports ... 478
Example: Creating a PowerShell Performance Dynamic Application ... 479
Creating the Dynamic Application ... 479

Adding the PowerShell Command 479
Adding the Collection Object 480
Creating the Presentation Object 481
Creating a Credential 481
Manually Aligning the Dynamic Application to a Device 483
Viewing the Performance Report 483

Chapter

1

Dynamic Application Development Overview

Overview

This chapter describes Dynamic Applications, which are customizable policies, created for a specific vendor and a specific type of device or system.

Dynamic Applications are customizable monitoring policies that define what data Skylar One collects from a device or system, how that data is collected, and how it is presented. Each Dynamic Application targets a specific vendor and device type, and uses one of several collection protocols – including SNMP, WMI, PowerShell, database queries, XML/SOAP/XSLT, and Python-based snippet code – to gather performance metrics, configuration data, and other operational information. This guide covers all Dynamic Application types and their respective development workflows, including raw snippet development and development using the Snippet Framework.

This chapter covers the following topics:

Managing Dynamic Applications 29

Viewing Dynamic Application Data and Alerts 51

Dynamic Application Settings 67

Creating a Dynamic Application 73

Collection Objects 77

Caching 95

Dynamic Component Mapping 99

Dynamic Application Relationships 113

Presentation Objects for Performance Dynamic Applications 124

Alerts and Thresholds *128*

Indexing *152*

Grouping Dynamic Application Data Using Collection Labels *162*

What Is A Dynamic Application?

Dynamic Applications are customizable policies, created for a specific vendor and a specific type of device or system, that tell Skylar One:

- What data to collect from devices
- How to present the data that has been collected
- When to generate alerts and events based on the data that has been collected

Skylar One includes Dynamic Applications for the most common hardware and software. You can customize these default Dynamic Applications to best work in your environment. You can also create custom Dynamic Applications.

For example, a Dynamic Application that is designed to collect information about file system usage on a device might define that:

- Skylar One should collect the total size of each file system and amount of space used for each file system.
- Skylar One should use the two collected values to present a graph of "percentage used" values for each file system.
- Skylar One should generate an alert when the amount of space used on a file system exceeds a specified percentage of the total size of that file system.

Types of Dynamic Applications

There are many different types of Dynamic Applications that can be created and used in Skylar One. The Dynamic Application type defines:

- The *protocol* Skylar One should use to collect the data.
- An *archetype*, which defines generally what type of data is being collected.

All Dynamic Application types use one of the following *protocols*:

- ***Bulk Snippet***. Skylar One will collect data by executing custom code written in the Python programming language. A *single instance of the Dynamic Application* uses custom-written Python code to collect data from *multiple devices*. For a standard Snippet Dynamic Application, Skylar One spawns a separate instance of the Dynamic Application for each device or component device. The Bulk Snippet is useful for systems that include a large number of component devices.
- ***Database***. Skylar One will collect data by connecting to a database and executing a query.
- ***Internal Collection***. Skylar One will collect data by performing an "Extended Internal Collection". Basic hardware and system data that was previously collected by Skylar One using an internal SNMP process can now be collected with Internal Collection Dynamic Applications (ICDAs). ICDAs do not use SNMP to provide a way to collect basic hardware and system data from devices that do not support SNMP.
- ***PowerShell***. Skylar One will collect data from a Windows device by performing PowerShell commands.

- ***Snippet***. Skylar One will collect data by executing custom code written in the Python programming language.
- ***SNMP***. Skylar One will collect data by performing SNMP requests.
- ***SOAP***. Skylar One will collect data by sending HTTP POST requests to a SOAP web service.
- ***WMI***. Skylar One will collect data by sending requests to a Windows Management Instrumentation service running on an external Windows device.
- ***XML***. Skylar One will collect data by sending an HTTP GET request for an XML document.
- ***XSLT***. Similar to the ***SOAP*** protocol, Skylar One will collect data by sending HTTP POST requests to a SOAP web service. Skylar One will also perform XSLT transformations to generate the requests and parse the responses.

All Dynamic Application types use one of the following *archetypes*:

- ***Performance***. Skylar One will display the collected data in a historical graph. Skylar One will also automatically calculate daily normalized data for each graph.
- ***Configuration***. Skylar One will display the current values of the collected data in tabular format. Skylar One will store tables of previously collected data to show when the values have changed.
- ***Journal***. Skylar One will display collected data in log format. Each log entry can contain multiple collected values and can change over time. The ***Journal*** archetype is available only when using the ***Snippet*** protocol.

Each Dynamic Application type has unique configuration options.

Core Elements of a Dynamic Application

All Dynamic Applications, regardless of protocol, include the following core elements:

- ***Archetypes*** that define what type of data is being collected and how it will be displayed in Skylar One. All Dynamic Applications use one of three archetypes: Performance (displays collected data in historical graphs with daily normalized data), Configuration (displays current values in tabular format with change tracking), or Journal (displays collected data in log format; Snippet only).
    - **Database Dynamic Applications**: Database Dynamic Applications can be either the Performance or Configuration archetypes.
    - **Snippet Dynamic Applications**: Snippet Dynamic Applications can be any of the three archetypes: Performance, Configuration, and Journal.
    - **XML/SOAP/XSLT Dynamic Applications**: XML, SOAP, and XSLT Dynamic Applications can be either the Performance or Configuration archetypes.
    - **SNMP Dynamic Applications**: SNMP Dynamic Applications can be either the Performance or Configuration archetypes.
    - **WMI and PowerShell Dynamic Applications**: WMI and PowerShell Dynamic Applications can use either the Performance or Configuration archetypes.
- ***Properties*** that define general settings for the Dynamic Application. For example, the name of the Dynamic Application, type of Dynamic Application (protocol and archetype), and frequency at which Skylar One should collect data using the Dynamic Application. Allow for version control, release notes, collection, and retention settings.
- ***Collection Objects*** that the individual data points that will be retrieved by the Dynamic Application, called collection objects. Collection objects define what type of data is being collected (gauge, counter, etc.) and how it is grouped. For Dynamic Applications with an archetype of Configuration, collection objects also define how each value will be displayed in the table of data.
    - **Database Dynamic Applications**: Collection objects for database Dynamic Applications differ from collection objects in other types of Dynamic Applications.
    - **Snippet Dynamic Applications**: Collection objects for Snippet Dynamic Applications have settings that are unique to Snippet Dynamic Applications.
    - **XML/SOAP/XSLT Dynamic Applications**: XML, SOAP, and XSLT Dynamic Applications have collection object settings that are unique to the respective protocol.

- ***Requests or Snippets*** that define how Skylar One should request data from a device. For Dynamic Applications types that use Requests or Snippets, each Collection Object is associated with the Request or Snippet that will populate that Collection Object. The implementation of this mechanism varies by protocol:
    - **Requests** are used in Dynamic Applications that use the SOAP, XML, or XSLT protocol
    - **Snippets** are used in Dynamic Applications that use the Snippet protocol
    - **SQL Queries** are used in Dynamic Applications that use the Database protocol
    - **SNMP OIDs** are used in Dynamic Applications that use the WMI or PowerShell protocol
    - **WMI Queries or PowerShell commands** are used in Dynamic Applications that use the WMI or PowerShell protocol
    - **Predefined collections** are used in Internal Collection Dynamic Applications (ICDA)
- ***Presentations*** that define how collected values will be displayed in Performance and Journal Dynamic Applications in Skylar One. Presentation Objects are not used for Configuration Dynamic Applications.
- ***Presentation Objects*** that define how Skylar One should present the collected data. For Dynamic Applications with an archetype of Performance or Journal, Presentation Objects define how the collected values are displayed by Skylar One. Presentation Objects are not used for Dynamic Applications with an archetype of Configuration.
- ***Alerts*** that include formulas that Skylar One will evaluate each time data is collected. Each formula examines the current values for one or more ***Collection Objects***. If the formula evaluates to *true*, Skylar One generates an alert. Additionally, you can align an event policy with each ***Alert***, and Skylar One can trigger that event if the alert evaluates to *true*.
- ***Thresholds*** that define variables that can be used in ***Alerts***. Each ***Threshold*** defines an allowed range of values for the variable and a default value. The value of the ***Threshold*** variable can be changed on a per-device basis by users of the Dynamic Application, without having to modify the Dynamic Application.

> **NOTE**: The threshold appears in the **Device Thresholds** page for each device that is aligned with the Dynamic Application. The threshold can be edited in the **Device Thresholds** page for each device without affecting the behavior of the Dynamic Application for other devices.

- ***Credentials*** that define how authentication should occur for each Dynamic Application on each device. The credential type and authentication method vary by protocol. Each Dynamic Application requires credentials to be configured with protocol-specific parameters:

    - **Database Dynamic Applications** use database credentials. Supported database types include MySQL, MSSQL, Oracle, PostgreSQL, DB2, Sybase, Informix, and Ingress. Credentials must include the database name, username, password, and connection parameters.

    - **SNMP Dynamic Applications** use SNMP credentials. Supported SNMP versions include SNMPv1, SNMPv2c, and SNMPv3. Credentials must include the appropriate community string (SNMPv1/v2c) or authentication parameters (SNMPv3).

    - **XML/SOAP/XSLT Dynamic Applications** use SOAP/XML credentials. When an XML, SOAP, or XSLT Dynamic Application is aligned with a device, you must align a SOAP/XML credential to that Dynamic Application for collection to occur. Credentials must specify the URL, HTTP method (GET for XML, POST for SOAP, or GET/POST for XSLT), proxy settings if applicable, encoding, and up to four embed values that can be substituted into requests or responses.

    - **Snippet Dynamic Applications** can use any credential type for authentication; however, the snippet code must be written to use the credential information passed to the snippet by Skylar One. This flexibility allows Snippet Dynamic Applications to use Database, SNMP, SOAP/XML, Basic/Snippet, PowerShell, or SSH/Key credentials depending on the protocol the snippet code is designed to work with.

    - **WMI Dynamic Applications** use Basic/Snippet credentials. Credentials must include a username in the format `DOMAIN\username`, `password`, and `hostname/IP address` of the device. An optional port number can be specified. For WBEM requests, supply the port used by the WBEM service on the device.

    - **PowerShell Dynamic Applications** use PowerShell credentials. Credentials must include the account type (local or domain user), username, password, hostname/IP address, and timeout value (in milliseconds). Skylar One will stop trying to communicate with the authenticated server after the specified timeout.

    - **Internal Collections Dynamic Applications** (ICDAs) use the Snippet protocol, so they can use any credential type for authentication. Skylar One passes credential information to the ICDA snippet code, which must be written to use that information appropriately.

Optional Features for Dynamic Applications

You can enable the following optional features for a Dynamic Application:

- ***Dynamic Component Mapping***. Skylar One can use Dynamic Application data that has been collected from a single management system, such as a VMware ESX server, to create a device record for each entity (such as a virtual machine) managed by that single management system. Dynamic Applications with Configuration archetype can be configured to use Dynamic Component Mapping. Dynamic Applications with Performance archetype cannot be used to create component devices.For more information, see the *Dynamic Component Mapping* section.
    - **Snippet Dynamic Applications**. Dynamic Component Mapping allows Skylar One to use Dynamic Application data that has been collected from a single management system, such as a VMware ESX server, and to create multiple device records for the entities managed by that single management system. The settings for configuring Dynamic Component Mapping in a Snippet Dynamic Application with Configuration or Performance archetype are the same as the Dynamic Component Mapping settings for other Dynamic Application protocols. Snippet Dynamic Applications with Journal archetype cannot be configured to use Dynamic Component Mapping.
    - **XML/SOAP/XSLT Dynamic Applications**: Dynamic Component Mapping allows Skylar One to useDynamic Application data that has been collected from a single management system, such as a VMware ESX server, to create multiple device records for the entities managed by that single management system.
- ***Caching***. When Skylar One requests information from a device during Dynamic Application collection, Skylar One can optionally ***cache*** the response from the device. If a response is cached, other Dynamic Applications that use the same protocol can use the cached response to populate collection objects. Caching responses reduces the number of requests performed by Skylar One and speeds up collection. For more information, see the *Caching* section.
    - **Snippet Dynamic Applications**. Dynamic Component Mapping allows Skylar One to use Dynamic Application data that has been collected from a single management system, such as a VMware ESX server, to create multiple device records for the entities managed by that single management system.
- ***Collector Affinity***. Skylar One enables you to specify which Data Collectors are allowed to collect data from a device for a Dynamic Application. This ensures that a group of collection jobs that cannot be split up among multiple Data Collectors–for instance, Dynamic Applications that produce and consume cache–will run on a single Data Collector. For more information, see the *Dynamic Application Settings* section.

- ***Relationships***. Dynamic Applications can be configured to automatically create relationships between devices. For example, the Dynamic Applications in the VMware vSphere and NetApp PowerPacks are configured to create relationships between VMware Datastore component devices and their associated NetApp Volume component devices. Relationships created by Dynamic Applications are used and visualized by Skylar One in the same manner as relationships created by topology collection, Dynamic Component Mapping, and manually in the user interface. For more information about ***Relationships***, see the *Dynamic Application Relationships* section.
    - **Database Dynamic Applications**. The settings for configuring the creation of relationships in a Database Dynamic Application with Configuration archetype are the same as the relationship settings for other Dynamic Application protocols.
    - **Snippet Dynamic Applications**. The settings that allow you to create relationships in a Snippet Dynamic Application with Configuration archetype are the same as the relationship settings for other Dynamic Application protocols.
    - **XML/SOAP/XSLT Dynamic Applications**. The settings for configuring the creation of relationships in an XML, SOAP, and XSLT Dynamic Application with Configuration archetype are the same as the relationship settings for other Dynamic Application protocols.
- ***Collection Labels***. When multiple Dynamic Applications collect the same type of performance data from different devices, you can use collection labels to view data from those different Dynamic Applications at the same time. For example, when Skylar One displays the CPU utilization, Memory utilization, or Swap utilization for a device, the utilization values come from a presentation object in a Dynamic Application aligned with that device. For more information about ***Collection Labels***, see the *Presentation Objects* and *Collection Labels* sections.
- ***Custom Attributes***. Dynamic Applications of archetype *configuration* can collect configuration data and use that data to create and/or populate the value of a custom attribute. There are two ways Dynamic Applications can interact with custom attributes. A Dynamic Application can populate the value of an existing custom attribute, or a Dynamic Application can both create the custom attribute and populate its value. For general information on custom attribute, see the section on *custom attributes*. For details on using a Dynamic Application to populate or create custom attributes, see the *Collection Objects* section.
- ***Asset Linking***. Skylar One can use data collected by configuration Dynamic Applications to automatically populate fields in asset records. For more information about ***Asset Linking***, see the *Collection Objects* section.
- ***Inventory Linking***. Skylar One can use data collected using configuration Dynamic Applications to automatically populate inventories of hardware components and installed software. For more information about ***Inventory Linking***, see the *Collection Objects* section.
- ***Standard Deviation Alerting***. Skylar One can calculate standard deviation data for values collected by performance Dynamic Applications. The standard deviation data can generate alerts when values deviate by a specified amount. For more information about enabling standard deviation calculations for collected data, see the *Collection Objects* section. For more information about using standard deviation in alerts, see the *Alerts and Thresholds* section.

Managing Dynamic Applications

This section describes the operational and administrative tasks required to manage Dynamic Applications after they have been created or deployed. These tasks include viewing Dynamic Applications and searching by various criteria, manually aligning Dynamic Applications with devices, managing collection status and credentials, configuring thresholds and poll frequencies on a per-device basis, and coordinating Dynamic Application discovery and alignment. This section also covers how to test collection for debugging purposes, reset statistical and session data when needed, and manage the lifecycle of collected data including data removal and Dynamic Application un-alignment.

Viewing the List of Dynamic Applications

The **Dynamic Applications Manager** page (System > Manage > Dynamic Applications) displays a list of all existing Dynamic Applications. For each Dynamic Application, the page displays the following fields:

TIP: To sort the list, click on a column heading. The list will be sorted by the column value, in ascending order. To sort the list by descending order, click the column heading again. You can also filter the items on this inventory page by typing filter text or selecting filter options in one or more of the filters found above the columns on the page. For more information, see *Filtering Inventory Pages* in the ***Introduction to Skylar One*** manual.

- ***Dynamic Application Name***. Name of the Dynamic Application, as defined in the **Dynamic Applications Properties Editor** page.
- ***Poll Rate***. Frequency, in minutes, at which Skylar One will poll all devices that use this Dynamic Application. The ***Poll Rate*** column displays the default poll frequency for the Dynamic Application, as defined in the **Dynamic Applications Properties Editor** page. You can define a ***custom*** poll frequency for one or more devices in a device template. The poll frequency defined in the device template overrides the poll frequency defined for the Dynamic Application. Devices to which the device template is applied will use the poll frequency defined in the device template.
- ***Type***. Type of Dynamic Application. The choices are:
    - *Bulk Snippet Configuration*. A *single instance of the Dynamic Application* uses custom-written Python code to collect static configuration data from *multiple devices*. This is useful for systems that include a large number of component devices. For details on creating bulk snippet Dynamic Applications, see the ***Snippet Dynamic Application Development*** manual.
    - *Bulk Snippet Performance*. A *single instance of the Dynamic Application* uses custom-written Python code to collect trendable performance data from *multiple devices*. This is useful for systems that include a large number of component devices. For details on creating bulk snippet Dynamic Applications, see the ***Snippet Dynamic Application Development*** chapter of this manual.

- *Database Configuration*. The Dynamic Application retrieves configuration data from a database application. The Dynamic Application uses SQL queries. The queried device returns table data. For details on creating database Dynamic Applications, see the ***DatabaseDynamic Application Development*** chapter of this manual.
- *Database Performance*. The Dynamic Application retrieves trendable performance data from a database application. The Dynamic Application uses SQL queries. The queried device returns table data. For details on creating database Dynamic Applications, see the ***DatabaseDynamic Application Development*** chapter of this manual.
- *Internal Collection Inventory*. The Internal Collection Inventory Dynamic Application (ICDA) retrieves configuration data about filesystems and interface. For filesystem, an ICDA Inventory can retrieve data such as storage size, filesystem type, and storage used. These ICDAs can also collect configuration data about interfaces, such as physical address, operational status, and IP addresses. For details on creating ICDAs, see the ***Internal Collection Dynamic Application Development*** section of this manual.
- *Internal Collection Performance*. The Internal Collection Performance Dynamic Application (ICDA) retrieves data about availability and latency, device information (system description, system uptime, system locale), filesystem performance, and interface performance. For details on creating ICDAs, see the ***Internal Collection Dynamic Application Development*** section of this manual.
- *IT Service*. A special type of Dynamic Application that Skylar One uses to monitor IT Services. When you create and edit an IT Service in the **IT Service Editor** page, Skylar One will automatically create and maintain a Dynamic Application for that IT Service. Dynamic Applications for IT Services will appear in the **Dynamic Applications Manager** page. However, if you want to edit the settings for an IT Service, you should not edit the Dynamic Application for that IT Service. Instead, use the **IT Service Editor** page to edit IT Services. For details on creating IT Service policies, see the manual ***IT Services***.
- *PowerShell Configuration*. The Dynamic Application uses PowerShell commands to collect static configuration data from a Windows device. For details on creating PowerShell Dynamic Applications, see the chapter ***Dynamic Application Development - WMI and PowerShell***. For information on configuring Skylar One and external systems to use PowerShell Dynamic Applications, see the manual ***Monitoring Windows Systems with PowerShell*** and ***Monitoring Windows Systems with WMI***.
- *PowerShell Performance*. The Dynamic Application uses PowerShell commands to collect trendable performance data from a Windows device. For details on creating PowerShell Dynamic Applications, see the chapter ***Dynamic Application Development - WMI and PowerShell***. For information on configuring Skylar One and external systems to use PowerShell Dynamic Applications, see the manual ***Monitoring Windows Systems with PowerShell*** and ***Monitoring Windows Systems with WMI***.
- *Snippet Configuration*. The Dynamic Application uses custom-written Python code to collect configuration data from a device. For details on creating snippet Dynamic Applications, see the ***Snippet Dynamic Application Development*** chapter of this manual.
- *Snippet Journal*. The Dynamic Application uses custom-written Python code to collect data formatted as log entries from a device. For details on creating snippet Dynamic Applications, see the ***Snippet Dynamic Application Development*** chapter of this manual.

- *Snippet Performance*. The Dynamic Application uses custom-written Python code to collect trendable performance data from a device. For details on creating snippet Dynamic Applications, see the ***Snippet Dynamic Application Development*** chapter of this manual.
- *SNMP Configuration*. The Dynamic Application uses SNMP to retrieve static, configuration data from devices or applications. For details on creating SNMP Dynamic Applications, see the ***SNMP Dynamic Application Development*** chapter of this manual.
- *SNMP Performance*. The Dynamic Application uses SNMP to retrieve trendable performance data from devices or applications. For details on creating SNMP Dynamic Applications, see the ***SNMP Dynamic Application Development*** chapter of this manual.
- *SOAP Configuration*. The Dynamic Application uses XML and SOAP to retrieve static configuration data from a SOAP server. The queried device returns XML data. For details on creating SOAP Dynamic Applications, see the ***XML, SOAP, and XSLT Dynamic Application Development*** chapter of this manual.
- *SOAP Performance*. The Dynamic Application uses XML and SOAP to retrieve trendable performance data from a SOAP server. The queried device returns XML data. For details on creating SOAP Dynamic Applications, see the ***XML, SOAP, and XSLT Dynamic Application Development*** chapter of this manual.
- *WMI Configuration*. The Dynamic Application retrieves configuration information from either WMI or WBEM running on a managed device. WMI Dynamic Applications use a query format to request data from a managed device. WBEM Dynamic Applications use wbemcli and HTTP to request data from a managed device. For details on creating WMI Dynamic Applications, see the chapter ***Dynamic Application Development - WMI and PowerShell***. For information on configuring Skylar One and external systems to use PowerShell Dynamic Applications, see the manual ***Monitoring Windows Systems with PowerShell*** and ***Monitoring Windows Systems with WMI***.
- *WMI Performance*. The Dynamic Application retrieves trendable performance data from either WMI or WBEM running on a managed device. WMI Dynamic Applications use a query format to request data from a managed device. WBEM Dynamic Applications use wbemcli and HTTP to request data from a managed device.
- *XML Configuration*. The Dynamic Application uses HTTP GET queries. The queried device returns static configuration data in XML format. For details on creating SOAP Dynamic Applications, see the ***XML, SOAP, and XSLT Dynamic Application Development*** chapter.
- *XML Performance*. The Dynamic Application uses HTTP GET queries. The queried device returns trendable performance data in XML format. For details on creating SOAP Dynamic Applications, see the ***XML, SOAP, and XSLT Dynamic Application Development*** chapter.
- *XSLT Configuration*. The Dynamic Application uses XML and SOAP to retrieve static configuration data from a SOAP server. The requests used to retrieve data are generated by performing an XSLT transformation on an XML document that contains data already collected by the Dynamic Application. The queried device returns XML data, which must be changed to a specific format by performing a second XSLT transformation. For details on creating SOAP Dynamic Applications, see the ***XML, SOAP, and XSLT Dynamic Application Development*** chapter.

    - *XSLT Performance*. The Dynamic Application uses XML and SOAP to retrieve trendable performance data from a SOAP server. The requests used to retrieve data are generated by performing an XSLT transformation on an XML document that contains data already collected by the Dynamic Application. The queried device returns XML data, which must be changed to a specific format by performing a second XSLT transformation. For details on creating SOAP Dynamic Applications, see the ***XML, SOAP, and XSLT Dynamic Application Development*** chapter.
- ***State***. Specifies whether the Dynamic Application is *Enabled* or *Disabled*.
- ***Version***. Version number to assign to the Dynamic Application. You can customize this value and increment it according to your change management policies.
- ***ID***. Unique application ID, assigned by Skylar One.
- ***Subscribers***. Number of devices on which the Dynamic Application is enabled to collect data. Clicking on the icon leads to the **Application Subscribers** modal, where you can view the list of devices and access other pages for each subscriber device. You can also access this page by selecting the wrench icon ( ) for a Dynamic Application and selecting the **[Subscribers]** tab.
- ***PowerPack***. Specifies whether or not the Dynamic Application is included in a PowerPack.
- ***Environment***. The execution environment to which the Dynamic Application is aligned, if it is a snippet or internal collection Dynamic Application. If it is not a snippet or internal collection Dynamic Application, then this column displays "n/a".
- ***Collects***. Number of objects included in the Dynamic Application. Clicking on the icon ( ) leads to the **Collection Objects** page, where you can view the list of collection objects and edit their properties.
- ***Alerts***. Number of custom alerts defined for the Dynamic Application. Clicking on the icon ( ) leads to the **Alert Objects** page, where you can view and edit each alert defined for the Dynamic Application.
- ***Events***. Number of events associated with the Dynamic Application. Clicking on the icon ( ) leads to the **Event Policy Manager** page, where you can view information about each event definition associated with the Dynamic Application definition and edit each event definition.
- ***Thresh***. Number of threshold objects defined for the Dynamic Application. Clicking on the icon ( ) leads to the **Threshold Objects** page, where you can view and edit information about each threshold object defined for the Dynamic Application.
- ***Edited By***. Username of the person who created or last edited the Dynamic Application.
- ***Last Edit***. Date that the Dynamic Application was created or last edited.

Performing Other Tasks in the Dynamic Application Manager Page

From the **Dynamic Applications Manager** page, you can also perform the following tasks:

- ***Manually perform dynamic discovery for a Dynamic Application***. In the **Dynamic Applications Manager** page, find the Dynamic Application you want to use for network-wide discovery and select its lightning bolt icon ( ).

- ***Filter the list by application***. Selecting the Filter on this Application icon (▼) sets the **Dynamic Applications Manager** page to display only Dynamic Applications with the same name as the selected Dynamic Application.
- ***Change an application's type***. To change the type of one or more Dynamic Applications, select the checkbox of each Dynamic Application you want to change. In the ***Select Action*** drop-down menu in the bottom right, scroll to the *Change Type* section and select a new type. Select the **[Go]** button. Each selected Dynamic Application will now be assigned the new type.
- ***Change polling interval***. To change the polling interval for one or more Dynamic Application, select the checkbox of each Dynamic Application you want to change. In the ***Select Action*** drop-down menu in the bottom right, select a new polling interval for the Dynamic Application. Select the **[Go]** button. Each selected Dynamic Application will now be assigned the new polling interval.
- ***Delete an Application***. To delete one or more Dynamic Applications, select the checkbox of each Dynamic Application you want to change. In the ***Select Action*** drop-down menu in the bottom right, select *DELETE Application*, and then select the **[Go]** button. A confirmation dialog will appear. Review the list of Dynamic Applications you selected for deletion, then click **[Confirm]** to proceed. Each selected Dynamic Application will be deleted.
- ***Clear Application Data***. To clear all historical data, including logs and reports, associated with one or more Dynamic Application, select the checkbox for each Dynamic Application you want to change. In the ***Select Action*** drop-down menu in the bottom right, select *CLEAR Application*, and then select the **[Go]** button. A confirmation dialog will appear. Review the list of Dynamic Applications, then click **[Confirm]** to proceed. The historical data for each selected Dynamic Application will be deleted.
- ***Enable an Application***. Enables the selected Dynamic Applications so that Skylar One can collect the data for those applications. In the ***Select Action*** drop-down menu in the bottom right, select *ENABLE Application*, and then select the **[Go]** button.
- ***Disable an Application***. Disables the selected Dynamic Applications. In the ***Select Action*** drop-down menu in the bottom right, select *DISABLE Application*, and then select the **[Go]** button.
- ***Discover Applications using the Select Action drop-down menu***. To start the Dynamic Application discovery process on all managed devices, select the checkbox for each Dynamic Application you want to use in the discovery process. In the ***Select Action*** drop-down menu in the bottom right, select DISCOVER Applications. Select the **[Go]** button. Skylar Onestarts the Dynamic Application discovery process on all devices in Skylar One using the selected Dynamic Applications.
- ***Validate and Repair Applications***. For each selected Dynamic Application, ensures that the Dynamic Application includes the objects, thresholds, and alerts that are referenced in the Dynamic Application and in other parts of Skylar One. Also ensures that if an event that is associated with the Dynamic Application has an auto-clear event, that auto-clear event is also associated with the Dynamic Application. This tool will troubleshoot and then fix the database tables that reference the Dynamic Application's objects, presentations, thresholds and alerts.
- ***Align a Device Dashboard***. To align one or more Dynamic Applications with a Device Dashboard, select the checkbox for each Dynamic Application you want to align with the Device Dashboard. In the ***Select Action*** drop-down menu in the bottom right, scroll to the *Change Device Dashboard* section and select a Device Dashboard. Select the **[Go]** button. For each device that subscribes to the selected Dynamic Application, the selected device dashboard will appear as an entry in the ***Device Dashboard*** field in the **Device Summary** page.

- ***Align an Execution Environment***. To align one or more Dynamic Applications with an execution environment, select the checkbox for each Dynamic Application you want to align with the environment. In the ***Select Action*** drop-down menu in the bottom right, scroll to the ***Align Collection Environment*** section and select an execution environment, and then click the **[Go]** button.

Viewing the Dynamic Applications Associated with a Device

You can view the Dynamic Applications associated with a device from the **Dynamic Application Collections** page. The **Dynamic Application Collections** page also allows you to *associate another Dynamic Application with the device*, *assign a credential to the Dynamic Application*, or *perform other administrative tasks for an aligned Dynamic Application*.

To view the Dynamic Applications associated with a device:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the device for which you want to view Dynamic Applications. Select its wrench icon ( ).
3. In the **Device Administration** panel, select the **[Collections]** tab.
4. The **Dynamic Application Collections** page displays a list of all Dynamic Applications aligned with the current device. For each Dynamic Application, the **Dynamic Application Collections** page displays the following read-only information:
    - ***Plus Sign*** ( ). Clicking on this icon displays a list of all Presentation Objects included in Dynamic Applications of type "Performance" and "Journal" or a list of all Collection Objects included in Dynamic Applications of type "Configuration". You can click on the plus sign next to each Presentation Object to see all the Collection Objects included in the Presentation Object.
    - ***Minus Sign*** ( ). Collapses a Dynamic Application and hides the display of Presentation Objects and Collection Objects.
    - ***Dynamic Application***. Name of the Dynamic Application.
    - ***ID***. Numeric ID for the Dynamic Application.
    - ***Poll Frequency***. Frequency at which Skylar One will query the device to retrieve the data specified in the Dynamic Application. Each Dynamic Application includes a default frequency. From this page (**Dynamic Application Collections**), you can change the poll frequency for a Dynamic Application on the current device. This edited poll frequency will override the default frequency for the Dynamic Application and the poll frequency defined for a Dynamic Application in one or more device templates.
    - ***Type***. The protocol used by the Dynamic Application (Database [SQL], Internal Collection Inventory or Internal Collection Performance (ICDA), Snippet [Python], SNMP, SOAP, WMI, XML, or XSLT) and the type of data collected by the Dynamic Application (Configuration, Performance, or Journal).
    - ***Credential***. Name of the credential that Skylar One uses to access the device and retrieve the data specified in the Dynamic Application.

NOTE: Cache-consuming Dynamic Applications do not require a credential. If you aligned a cache-consuming Dynamic Application in the **Dynamic Application Alignment** modal page, the ***Credential*** field displays *N/A* and is grayed out. You do not have to select a credential in the **Dynamic Application Alignment** modal page.

- ***Collector***. Name of the specific Data Collector used to collect data from the Dynamic Application.

NOTE: Based on the Dynamic Application's ***Collector Affinity*** settings, the Dynamic Application might be assigned to a different Data Collector than the Data Collector that is assigned to the device in the Device Properties page (Devices > Classic Devices > wrench icon). In the **Dynamic Application Collections** page, hover your mouse over the ***Collector*** name for any of the Collection Objects to view a tooltip that explains why the Dynamic Application is assigned to its particular Data Collector.

- ***Run Dynamic Application*** (⚡). Performs a test run of data collection for the selected Dynamic Application on the current device.

NOTE: If a device is currently unavailable, the lightning-bolt icon (⚡) will be grayed out for each Dynamic Application aligned with the device.

- ***Checkbox*** (☐). Apply an action from the ***Select Action*** field to this instance of the Dynamic Application.

5. From this page, you can *associate another Dynamic Application with the device*, *assign a credential to the Dynamic Application*, or *perform other administrative tasks for an aligned Dynamic Application*.

Viewing the Status of a Dynamic Application

For each device, Skylar One maintains the collection status for each collection object in each Dynamic Application aligned with that device. The **Dynamic Application Collections** page displays the status of each collection object for a device as represented by two values: ***Found*** and ***Collect***. The **Dynamic Application Collections** page also displays the ***Found*** and ***Collect*** values for each presentation object, which are derived from the status of each collection object used by the presentation object.

Found

The ***Found*** status for a collection object has two possible values:

- *Yes*. Data has been successfully collected from this device for this object. ***Found*** is set to *Yes* the first time data is successfully collected from this device for this object.
- *No*. Data has never been successfully collected from this device for this object. *No* is the initial value of ***Found*** for every object when a Dynamic Application is initially aligned with a device.

The ***Found*** status for a presentation object also has two possible values (*Yes* and *No*).

- ***If the presentation object uses only one collection object***, the presentation object always has the same default ***Found*** and default ***Collect*** values as that collection object.
- ***If a presentation object uses multiple collection objects***, the default ***Found*** value for the presentation object will be *Yes* only if all the collection objects used by the presentation object have a ***Found*** value of *Yes*.

After ***Found*** is set to *Yes* for an object, Skylar One will never automatically change the value of ***Found*** for this object.

The value of ***Found*** is used by Skylar One to determine whether icons, tabs, and Navbar links that lead to the **[Performance]** or **[Configs]** page where the collection object is used should be active.

Collect

The ***Collect*** status for a collection object has two possible values:

- *Yes*. Skylar One will attempt to collect data for this object when collection for this Dynamic Application occurs. *Yes* is the initial value for ***Collect*** for every object when a Dynamic Application is initially aligned with a device.
- *No*. Skylar One will not attempt to collect data for this object when collection for this Dynamic Application occurs. Skylar One might set ***Collect*** to *No* automatically if no data has been collected.
- If a collection object has a ***Collect*** value of *No*, all presentation objects that use that collection object will also have a ***Collect*** value of *No*.

The ***Collect*** status for a presentation object also has two possible values (*Yes* and *No*).

- ***If the presentation object uses only one collection object***, the presentation object always has the same default ***Found*** and default ***Collect*** values as that collection object.
- ***If a presentation object uses multiple collection objects***, the default ***Collect*** value for the presentation object will be *Yes* only if ***all*** the collection objects used by the presentation object have a ***Collect*** value of *Yes*. If one or more collection objects used by the presentation object have a ***Collect*** value of *No*, the presentation object will also have a default ***Collect*** value of *No*.
- The ***Collect*** status for a presentation object has no effect upon its collection objects. If you manually change the ***Collect*** status for a presentation object, the ***Collect*** status for the collection objects used by the presentation object will not change.

NOTE: Before determining which collection objects defined in a Dynamic Application will be collected, Skylar One determines whether the Dynamic Application itself should be collected. Dynamic Applications are not collected for devices that are unavailable (because of a failed availability check) or have collection disabled (either manually by a user or because of maintenance scheduled in Skylar One) regardless of the ***Collect*** value of the objects.

How Skylar One Manages Collect Status

Stopping Collection

One of the ScienceLogic hourly maintenance tasks checks the last collection time for every collection object being collected from every device. If the last collection time for an object on a device is more than 48 hours ago, collection is stopped for that collection object on that device. Skylar One will set the ***Collect*** status of that object to *No*.

NOTE: If a device is in maintenance mode, is unavailable, or has been manually disabled by a user, Skylar One will not automatically set the ***Collect*** status of objects to *No*. Skylar One will automatically set the ***Collect*** status of objects to *No* only if the device is up and running, but Skylar One still cannot collect the object.

When Skylar One sets the ***Collect*** status of that object to *No*, Skylar One generates an event. The event will include the name of the device, the name of the Dynamic Application, the name of the collection object, and the collection object IDs. By default, this event is of severity "notice".

NOTE: For Dynamic Applications that have the ***Component Mapping*** checkbox selected in the **Dynamic Applications Properties Editor** page, Skylar One will never automatically set the ***Collect*** status to *No* for any of the collection objects in the Dynamic Application.

NOTE: For Dynamic Applications that have the ***Caching*** fields set to either *Cache Results* or *Consume cached results* in the **Dynamic Applications Properties Editor** page, Skylar One will never automatically set the **Collect** status to *No* for any of the collection objects in the Dynamic Application.

Starting Collection

For each object that has the ***Collect*** status of *No*, Skylar One will attempt to re-collect the object once a day. If re-collection is successful, Skylar One will automatically set the ***Collect*** value for that object to *Yes*.

NOTE: If a user manually sets the ***Collect*** status of a collection object or presentation object to *No*, Skylar One will **not** attempt to re-collect the object once a day and will **not** set the ***Collect*** status to *Yes*.

Collection Objects that are Excluded from Maintenance

The ***Collect*** status of the following collection objects is never changed automatically:

- Collection objects in Dynamic Applications that have the ***Component Mapping*** checkbox checked in the **Dynamic Applications Properties Editor** page.
- Collection objects in Dynamic Applications that have the ***Caching*** fields set to either *Cache Results* or *Consume cached results*, in the **Dynamic Applications Properties Editor** page.
- Collection objects that have the ***Disable Object Maintenance*** setting enabled.
- Collection objects that have a ***Collect*** status defined by a user, i.e. collection objects that were manually enabled or disabled by a user.

Status of Objects for Deviation

Skylar One allows you to examine the value of an object and trigger an alert if that value falls outside the range of "normal" values for that object at the hour of the day on that day of the week. The deviation function allows you to define such alerts.

To use the deviation function, you must configure Skylar One to store and calculate the mean values and standard deviation for an object. You do this by selecting the ***Enable Deviation Alerting*** field in the **Collection Objects** page. You then specify the minimum and maximum number of weeks to collect deviation data for the object. Skylar One must have already collected at least the minimum number of weeks' worth of values for an object before Skylar One can evaluate alert formulas that use the deviation function. To use the deviation function, you must specify a minimum value of at least two weeks.

If a Dynamic Application in the **Dynamic Application Collections** page contains one or more alerts that use the deviation function, the **Dynamic Application Collections** page displays the status of the collection objects.

For example, suppose an alert in a Dynamic Application will apply the deviation function to object "o_123". Suppose that you specified that Skylar One must collect at least two weeks' worth of deviation data for this object. Suppose that Skylar One contains only one weeks' worth of values for object "o_123". In this case, the **Dynamic Application Collections** page will display the following message:

```
Note: object 123 not ready for deviation alerting.
```

When Skylar One contains at least two weeks worth of values for object "o_123", the **Dynamic Application Collections** page will display the following message:

```
All objects ready for deviation alerting.
```

Manually Associating a Dynamic Application with a Device

From the **Dynamic Application Collections** page, you can manually associate a new Dynamic Application with a device.

To manually associate a Dynamic Application with a device:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the device you want to associate with a Dynamic Application. Click its wrench icon (🔧).
3. In the **Device Administration** panel, click the **[Collections]** tab.
4. In the **Dynamic Application Collections** page, click the **[Actions]** menu and select *Add Dynamic Application*.
5. The **Dynamic Application Alignment** modal page appears. To align a Dynamic Application with a device in this page:
    - Select the Dynamic Application you want to align with the device in the ***Dynamic Applications*** field. You can filter the list of Dynamic Applications using the search field above the ***Dynamic Applications*** field.
    - After selecting a Dynamic Application, you must select a credential. Select a credential in the ***Credentials*** field. You can filter the list of credentials using the search field above the ***Credentials*** field.

    NOTE: Your organization membership(s) might affect the list of credentials you can see in the ***Credentials*** field.

    NOTE: Cache-consuming Dynamic Applications **do not** require a credential. If you selected a cache-consuming Dynamic Application in the **Dynamic Application Alignment** modal page, the ***Credential*** field displays *N/A* and is grayed out. You do not have to select a credential in the **Dynamic Application Alignment** modal page.

6. Click the **[Save]** button in the **Dynamic Application Alignment** modal page to align the Dynamic Application and the credential to the device.
7. Skylar One will associate the Dynamic Application with the device and immediately attempt to collect the data specified in the Dynamic Application using the selected credential.
8. After the first, immediate collection, Skylar One will collect the data at the frequency defined in the ***Polling Frequency*** field in the **Application Configuration Editor** page for the Dynamic Application.

Editing the Credential Associated with a Dynamic Application

From the **Dynamic Application Collections** page, you can change the credential associated with a Dynamic Application. This credential will be used by Skylar One for this specific Dynamic Application associated with this specific device . For all other devices, Skylar One will use the default credential associated with the device, or will use the credential defined in the **Dynamic Application Collections** page for each device.

> **NOTE**: Cache-consuming Dynamic Applications do not require a credential. If you aligned a cache-consuming Dynamic Application with this device (you do this in the **Dynamic Application Alignment** modal page), the ***Credential*** field displays *N/A* and is grayed out.

To change the credential associated with a Dynamic Application for a device:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the device for which you want to define a credential. Select its wrench icon ( ).
3. In the **Device Administration** panel, select the **[Collections]** tab.
4. In the **Dynamic Application Collections** page, find the Dynamic Application for which you want to change the credential. Select its checkbox. To apply a credential to multiple Dynamic Applications, select the checkbox for each Dynamic Application.
5. From the ***Select Action*** drop-down list, select the credential from the list of all credentials that you are allowed to use, and then select the **[Go]** button.

> **NOTE:** Your organization membership(s) might affect the list of credentials you can see in the ***Select Action*** drop-down list.

> **NOTE:** If this Dynamic Application has already been aligned with a credential to which you do not have access, the ***Credential*** column will display the value *Restricted Credential*. If you align the device with a different credential, you will not be able to re-align the device with the *Restricted Credential*.

6. You should see your change reflected in the ***Credential*** column in the **Dynamic Application Collections** page.

Performing Other Administrative Tasks for an Aligned Dynamic Application

You can perform the following other administrative tasks for an aligned Dynamic Application in the **Dynamic Application Collections** page:

- Enable or disable one or more collection objects or presentation objects.
- Stop data collection for the whole Dynamic Application.
- Reset the statistical data that has been stored for standard deviation alerting.
- Reset persistent session objects that have been collected and stored for a Dynamic Application.
- Test collection for a Dynamic Application.
- Remove all data collected using the Dynamic Application and optionally unalign the Dynamic Application from the device.

To perform one of these tasks:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the device for which you want to perform an administrative task. Select its wrench icon ( ).
3. In the **Device Administration** panel, select the **[Collections]** tab.
4. In the **Dynamic Application Collections** page, find the Dynamic Application for which you want to perform an administrative task. The following sections describe how to perform each task.

Enabling or Disabling Objects

From the **Dynamic Application Collections** page, you can customize the collection performed by the Dynamic Application for the current device. This customization will be used by Skylar One only for this specific device. For all other devices, Skylar One will use the default list of objects from the Dynamic Application's definition or will use the list of objects defined in the **Dynamic Application Collections** page for that device.

NOTE: If a collection object has a ***Collect*** value of *No*, all presentation objects that use that collection object will also have a ***Collect*** value of *No*.

To enable or disable collection for one or more objects in a Dynamic Application:

- ***To disable collection for one or more collection objects***, unselect the checkbox for each object for which you want to disable collection.
- For each unselected object, the ***Collect*** column should now display *No*.
- ***To enable collection for one or more collection objects***, select the checkbox for each object for which you want to enable collection.
- For each selected object, the ***Collect*** column should now display *Yes*.
- Select the **[Save]** button.

> **NOTE**: If a user **manually** sets the ***Collect*** status of a collection object or presentation object to *No*, Skylar One will **not** attempt to re-collect the object once a day and will **not** automatically set the ***Collect*** status to *Yes*.

Restarting Automatic Maintenance of Collection Objects

If a user **manually** sets the ***Collect*** status of a collection object or presentation object, Skylar One will **not** automatically change the ***Collect*** status of that object as described in the *How Skylar One Manages Collect Status* section.

If you want Skylar One to restart automatic maintenance of the objects in a Dynamic Application, perform the following steps:

1. In the **Dynamic Application Collections** page, select the checkbox for the Dynamic Application for which you want to restart automatic collection maintenance. To restart automatic collection maintenance for multiple Dynamic Applications, select the checkbox for each Dynamic Application.
2. From the ***Select Action*** drop-down list, select *Restore System Control of Collection State* and then select the **[Go]** button.
3. Automatic collection maintenance for all objects in the Dynamic Application will now occur. The ***Collect*** status of the objects in the Dynamic Application will not change immediately.

Editing the Poll Frequency for a Dynamic Application on the Current Device

Poll Frequency is the frequency at which Skylar One will query the device to retrieve the data specified in the Dynamic Application. Each Dynamic Application includes a default frequency.

From the **Dynamic Application Collections** page, you can change the poll frequency for a Dynamic Application on the current device. For the current device, the edited poll frequency will override:

- the default frequency for the Dynamic Application.
- the poll frequency defined for a Dynamic Application in one or more device templates.

To edit the poll frequency for a Dynamic Application on the current device:

1. In the **Dynamic Application Collections** page, select the checkbox for the Dynamic Application for which you want to change the poll frequency. To change the poll frequency for multiple Dynamic Applications, select the checkbox for each Dynamic Application.
2. From the ***Select Action*** drop-down list, select *Poll Frequency* from the list of poll frequencies and then select the **[Go]** button.
3. You should see your change reflected in the ***Poll Frequency*** column in the **Dynamic Application Collections** page.

Stopping Data Collection for a Dynamic Application

You can stop data collection for a Dynamic Application on the current device. This will affect collection only for this specific device. For all other subscriber devices, Skylar One will continue to use this Dynamic Application to collect data.

To stop data collection for a Dynamic Application on this device:

1. Select the checkbox of each Dynamic Application for which you want to stop data collection.
2. From the ***Select Action*** drop-down list, select the following:
    - ***Disable All Collection Objects***. For all collection objects in the selected Dynamic Application (s), the ***Collect*** value will be set to *No*.
3. Select the **[Go]** button.

> **NOTE**: If a user manually sets the ***Collect*** status of a collection object or presentation object to *No*, Skylar One will not attempt to re-collect the object once a day and will not set the ***Collect*** status to *Yes*.

Resetting Statistical Data for a Dynamic Application

Skylar One allows you to examine the value of an object and trigger an alert if that value falls outside the range of "normal" values for that object at that hour of the day on that day of the week. The deviation function allows you to define such alerts.

To use the deviation function, you must configure Skylar One to store and calculate the mean values and standard deviation for an object. You do this by selecting the ***Enable Deviation Alerting*** field in the **Collection Objects** page. You then specify the minimum and maximum number of weeks to collect deviation data for the object. Skylar One must have already collected at least the minimum number of weeks' worth of values for an object before Skylar One will evaluate alert formulas that use the deviation function. To use the deviation function, you must specify a minimum value of at least two weeks.

In some cases, you might want to delete all the collected statistics for an object and start over. This is useful if known circumstances change the value of an object, and you no longer want to use the old data to calculate the "normal" ranges. You can do this by "resetting" the statistical data for an object.

For example, suppose you were monitoring bandwidth usage with a standard deviation alert. Suppose your company previously ran on a 09:00 to 17:00 work schedule. Suppose your company has recently

added a nightshift to the schedule. In this circumstance, you might want to reset the statistical data to determine the new "normal" usage patterns.

When you reset the statistical data for an object, you are telling Skylar One to ignore all previously collected values and to use only values from today onward. When you reset the statistical data for an object, the **Dynamic Application Collections** page will again display a message like:

```
Note: object 123 not ready for deviation alerting.
```

until enough data has been collected to again calculate standard deviation for the object. Skylar One will again start collecting the minimum number of weeks of data for the object (as specified in the ***Enable Deviation Alerting*** field in the **Collection Objects** page) and calculating the "normal" ranges for those objects for each hour at each day of the week.

To delete all current statistical data for an object:

1. In the Dynamic Application, find the object for which you want to reset data.
2. In that Dynamic Application, find the object for which you want to reset data. Select its checkbox.
3. From the ***Select Action*** drop-down list, select the following option:
    - *Reset Statistical Data*. Removes all previously collected statistical data for the selected object. Skylar One will again start collecting the minimum number of weeks of data for the object (as specified in the ***Enable Deviation Alerting*** field in the **Collection Objects** page) and calculating the "normal" ranges for those objects for each hour at each day of the week.
4. Select the **[Go]** button.
5. The **Dynamic Application Collections** page will display a message like:

    ```
    Note: object 123 not ready for deviation alerting.
    ```

Resetting Persistent Session Objects for a Dynamic Application

SOAP or XSLT Dynamic Applications can contain a collection object that stores a Session ID. The value for this collection object can be defined as a persistent value. If Skylar One has already retrieved and stored a value in the collection object for the Session ID, Skylar One will not collect a new value for the collection object until a SOAP fault occurs. You can force Skylar One to re-collect a Session ID collection object by deleting the current persistent value.

To delete the current persistent value for a session object:

1. In the Dynamic Application, find the object for which you want to reset data. Select its checkbox.
2. From the ***Select Action*** drop-down list, select *Reset Persistent Session Objects*. Removes the stored value for collection objects of type ***SOAP/XSLT Session ID***. ***SOAP/XSLT Session ID*** objects are persistent across collection periods; Skylar One does not collect a ***SOAP/XSLT Session ID*** object if a collected value is available from a previous poll. After selecting this option, Skylar One will delete the existing value for the object and collect a new value during the next collection.
3. Select the **[Go]** button.

Testing Data Collection for a Dynamic Application

On a single device, you can perform a test-run of collection with a single Dynamic Application. During this test run, Skylar One displays details of each step of the collection process. This information can be very helpful for troubleshooting and debugging.

> **NOTE:** During a test run of a collection with a Dynamic Application, Skylar One does not store the collected data or generate alerts. Skylar One will continue to collect data and generate alerts using the selected Dynamic Application at the frequency defined in the Dynamic Application.

To execute a test run of collection with a single Dynamic Application:

1. Locate the device on which you want to test the Dynamic Application and click its wrench icon ( ).
2. Click the **Collections** tab.
3. Find the Dynamic Application for which you want to test collection and click its lightning bolt icon ( ).

   > **NOTE:** If a device is currently unavailable, the lightning bolt icon ( ) will be grayed out for each Dynamic Application aligned with the device.

4. Skylar One displays a **Session Logs** modal that includes details about each step of the collection process and diagnostic details about alerts in the Dynamic Application. This information can be helpful during troubleshooting.

Removing Data Collected by a Dynamic Application

You can remove the data retrieved with a Dynamic Application from the current device. You have two options for removing Dynamic Application data associated with a device:

- Remove all previously collected data, but continue to collect data at the specified polling frequency.
- Remove all normalized data, but retain all raw collected data and continue to collect data at the specified polling frequency.
- Remove all previously collected data and stop collecting data with this Dynamic Application. This unaligns the device from the Dynamic Application. The device will no longer be a subscriber to the Dynamic Application.

To remove Dynamic Application data associated with a device:

1. In the **Dynamic Application Collections** page, select the checkbox of the Dynamic Application for which you want to remove data. To remove data for multiple Dynamic Applications, select the checkbox for each Dynamic Application.
2. From the ***Select Action*** drop-down list, select one of the following options:

- ***Remove Data***. Removes all previously collected data, but data will continue to collect at the specified polling frequency.
- ***Remove Normalized Data***. Removes all normalized data, but all raw collected data is retained and data will continue to collect at the specified polling frequency.
- ***Stop Collection and Remove Data***. Removes all previously collected data and stops collection of data with this Dynamic Application. This "unaligns" the device from the Dynamic Application. The device is no longer considered a subscriber to the Dynamic Application. If you perform this option and later want to subscribe to this Dynamic Application again, you must re-align the device with the Dynamic Application.

3. Select the **[Go]** button.

Bulk Un-aligning Dynamic Applications

The **Application Subscribers** page contains a drop-down field in the lower right called ***Select Action***. This field allows you to un-align a Dynamic Application from one or more subscriber-devices.

To un-align a Dynamic Application from one or more devices:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. In the **Dynamic Applications Manager** page, find an application with a subscriber icon ( ) in the ***Subscribers*** column. Click the icon.
3. The **Application Subscribers** page appears.
4. In the **Application Subscribers** page, select the checkbox for each device you want to apply the action to. To select all checkboxes for all devices, select the checkbox at the top of the page.
5. In the ***Select Action*** drop-down list, select *Unalign Device and Remove Collection Data*. This option un-aligns the device from the Dynamic Application and deletes all data collected by the Dynamic Application from the device. The device is no longer considered a subscriber to the Dynamic Application. If you perform this option and later want to subscribe to this Dynamic Application again, you must re-align the device with the Dynamic Application.
6. Click the **[Go]** button to apply the action to all selected devices.

Setting Thresholds for Dynamic Applications

If a Dynamic Application includes one or more ***thresholds***, you can change the threshold value on a per-device basis. To change a Dynamic Application threshold for a device:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the device for which you want to define a threshold. Select its wrench icon ( ).
3. In the **Device Administration** panel, select the **[Thresholds]** tab.
4. The **Device Thresholds** page displays a list of thresholds defined for each Dynamic Application that is aligned to the device. To change a threshold, move the slider for that threshold or enter a value in

the number field for that threshold.

5. After changing one or more thresholds, select the **[Save]** button to save your changes.

NOTE: Changing a threshold in the **Device Thresholds** page affects only the current device. The threshold values defined in the Dynamic Application remain unchanged.

Dynamic Applications and Discovery

Discovery is the ScienceLogic tool that automatically discovers devices in your network. You supply the discovery tool with a range or list of IP addresses, and the discovery tool determines if a device exists at each IP address. The discovery tool also determines which (if any) Dynamic Applications to align with the device. If the discovery tool finds Dynamic Applications to align with the device, the discovery tool triggers collection for each aligned Dynamic Application.

To learn more about discovery, see the ***Discovery and Credentials*** manual.

How Does Skylar One Align Dynamic Applications During Discovery?

Most Dynamic Applications include a discovery object. A discovery object enables Skylar One to determine which devices to align with a Dynamic Application.

During discovery, Skylar One:

1. Searches the list of Dynamic Applications.
2. If a Dynamic Application includes a discovery object, Skylar One adds that Dynamic Application to the list of Dynamic Applications to try to align during discovery.
3. For each Dynamic Application that includes a discovery object, Skylar One checks the current discovery session for an appropriate credential. For example, for each database Dynamic Application, Skylar One would look for one or more database credentials that have been selected for the discovery session.
4. For each discovered device, both those that support SNMP and those that don't, discovery tries to determine which Dynamic Applications to align. For each discovered device, Skylar One tries to align each Dynamic Application in the list of Dynamic Applications to try during discovery. For each Dynamic Application in the list, Skylar One tries to connect to each device with each of the appropriate credentials (until Skylar One finds a working credential) and then tries to find the discovery object. If Skylar One is able to connect to a device with one of the credentials and can then retrieve the discovery object, Skylar One will align the Dynamic Application with the device.

NOTE: Skylar One also includes more sophisticated logic that allows you to define multiple discovery objects, validate the value of the discovery object, and to align the Dynamic Application if a discovery object is not available. However, the most common use of a discovery object is as described above (discovery object exists).

5. If discovery aligns a Dynamic Application with a device, immediately after discovery completes Skylar One will start the first collection from that device using the aligned Dynamic Application. This step is not performed for Dynamic Applications that meet all of the following three criteria:
    - Has a collection frequency of 1 minute, 2 minutes, 3 minutes or 5 minutes.
    - Does not have component mapping enabled (does not discover component devices).
    - Is aligned with a component device.

NOTE: During discovery, Skylar One tries each SNMP credential specified in the discovery session on each discovered device, to determine if Skylar One can collect SNMP details from the device. Later in the discovery session, during alignment of Dynamic Applications, discovery again tries each SNMP credential specified in the discovery session. If one of the SNMP credentials times out three times ***without any response***, discovery will stop trying to use that SNMP credential to align SNMP Dynamic Applications. Note that "no response" means that a device did not respond at all. Note that if a device reports that "no OID was found" or "the end of the OID tree was reached", these are considered a legitimate response and would not cause Skylar One to abandon the credential.

Queuing Discovery from the Dynamic Applications Manager Page

From the **Dynamic Applications Manager** page, you can manually run the Dynamic Application alignment portion of discovery for all devices in the system using one or more selected Dynamic Applications.

To manually queue discovery from the **Dynamic Applications Manager** page:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. In the **Dynamic Applications Manager** page, select the checkbox for each Dynamic Application you want to use for discovery.
3. In the ***Select Action*** drop-down list, select *Discover Applications*. Select the **[Go]** button.
4. You can also run the Dynamic Application alignment portion of discovery for all devices in the system using a single Dynamic Application. To do this, select the lightning bolt icon ( ) for that Dynamic Application.

System Settings That Affect Dynamic Application Discovery

Some of the parameters in the **Behavior Settings** page affect how Skylar One aligns Dynamic Applications during discovery (discovery, auto-discovery, and re-discovery).

To define or edit the settings that affect discovery in the **Behavior Settings** page:

1. Go to the **Behavior Settings** page (System > Settings > Behavior).
2. In the **Behavior Settings** page, edit the values in one or more of the following fields:
    - ***Initial Discovery Scan Level***. Specifies the data to be gathered during the initial discovery session. You can override this setting for a single discovery session in the **Discovery Session Editor** modal page. The options are:

- *0. Model Device Only*. Discovery tool will discover if device is up and running and if so, collect the make and model of the device. Skylar One will then generate a device ID for the device, so it can be managed by Skylar One.
- *1. Initial Population of Apps*. Discovery tool will search for Dynamic Applications to associate with the device. Discovery tool will attempt to collect data for the aligned Dynamic Applications. Discovery will also perform *0. Model Device Only* discovery.
- *2. Discover SSL Certificates*. Discovery tool will search for SSL certificates and retrieve SSL data. Discovery tool will also perform *1. Initial Population of Apps* and *0. Model Device Only*.
- *3. Discover Open Ports*. Discovery tool will search for open ports. Discovery tool will also perform *2. Discover SSL Certificates*, *1. Initial Population of Apps*, and *0. Model Device Only*. If your system includes a firewall and you select *3. Discover Open Ports*, discovery might be blocked and/or might be taxing to your network.
- *4. Advanced Port Discovery*. Discovery tool will search for open ports, using a faster TCP/IP connection method. Discovery tool will also perform *2. Discover SSL Certificates*, *1. Initial Population of Apps*, and *0. Model Device Only*. If your system includes a firewall and you select *4. Advanced Port Discovery*, some auto-discovered devices might remain in a pending state (purple icon) for some time after discovery. These devices will achieve a healthy status, but this might take several hours.
- *5. Deep Discovery*. Discovery tool will use nmap to retrieve operating system name and version. Discovery will also scan for services running on each open port and can use this information to match devices to device classes. Discovery tool will search for open ports, using a faster TCP/IP connection method. Discovery tool will also perform *2. Discover SSL Certificates*, *1. Initial Population of Apps*, and *0. Model Device Only*. For devices that don't support SNMP, option *5. Deep Discovery* allows you to discover devices that don't support SNMP and then align those devices with a device class other than "pingable".

CAUTION: Option *5. Deep Discovery* is compute-intensive and might significantly tax your network if used as the default setting. ScienceLogic recommends that you use this option on a per-discovery basis by selecting it in the **Discovery Session Editor** page.

- ***Rediscovery Scan Level (Nightly)***. Specifies the data to be gathered/updated each night during auto-discovery. The auto-discovery process will find any changes to previously discovered devices and will also find any new devices added to the network. The options are the same as for ***Initial Discovery Scan Level***.

TIP: ScienceLogic recommends that you delete all unused PowerPacks from your Skylar One system to improve the performance of the nightly auto-discovery process.

- ***Discovery Scan Throttle***. Specifies the amount of time a discovery process should pause between each IP address or hostname in a discovery session. (You specify the list of IP addresses or hostnames for a discovery session in the ***IP Address/Hostname Discovery List*** field in the **Discovery Session Editor** page.) Pausing discovery processes between IP addresses or hostnames spreads the amount of network traffic generated by discovery over a longer period of time. The choices are:
    - *Disabled*. Discovery processes will not pause.
    - *1000 Msec to 10000 Msec*. A discovery process will pause for a random amount of time between half the selected value and the selected value.
- ***Port Scan All IPs***. Specifies whether Skylar One should scan all IP addresses on a device for open ports. You can override this setting for a single discovery session in the **Discovery Session Editor** modal page. The choices are:
    - *0. Disabled*. Skylar One will scan only the primary IP address (the one used to communicate with Skylar One) for open ports.
    - *1. Enabled*. Skylar One will scan all discovered IP addresses for open ports.
- ***Port Scan Timeout***. Length of time, in milliseconds, after which Skylar One should stop trying to scan an IP address for open ports and begin scanning the next IP address (if applicable). You can override this setting for a single discovery session in the **Discovery Session Editor** modal page. Choices are between 60,000 and 1,800,000 milliseconds.
- ***Restart Windows Services (Agent required)***. Specifies whether Skylar One should automatically restart failed Windows services that have been defined on the device with a startup type of "automatic". To use this feature, the managed device must be running the agent SNMP Informant, WMI Edition. For assistance or information on purchasing and installing this agent, please contact ScienceLogic. Users must also supply a value in the ***SNMP Write*** field in the **Device Properties** page for the device. The choices are:
    - *0. Disabled*. Skylar One will not automatically restart failed Windows services that have been defined on the device with a startup type of "automatic".
    - *1. Enabled*. Skylar One will automatically restart failed Windows services that have been defined on the device with a startup type of "automatic".
- ***Hostname Precedence***. Specifies which name Skylar One will use for each discovered device. Choices are:
    - *SNMP System Name*. Use the device name specified in the device's SNMP System MIB. If *SNMP System Name* is selected and Skylar One cannot find an SNMP name for the device, Skylar One will assign the name returned by the DNS Reverse Lookup. If Skylar One cannot find a DNS Reverse Lookup name for the device, Skylar One will use the device's Admin Primary IP address as the device name in Skylar One.
    - *DNS Reverse Lookup*. Use the device name specified in the device's reverse-lookup record.

- ***Event Interface Name Format***. Specifies the format of the network interface name that you want to appear in events. If you selected *Interface Alias* for the deprecated ***Interface Name Precedence*** field in a previous release of Skylar One, the format for existing interfaces is set to {alias}. If you selected *Interface Name* for the deprecated ***Interface Name Precedence*** field in a previous release of Skylar One, the format for existing interfaces is set to {name}. The default format is {name}. You can use a combination of string text and the following tokens to define the interface name format for events, such as string_{name}, string_{alias}, {name}{alias}, or {ifdesc}:
    - {alias}
    - {name}
    - {state}
    - {ifdescr}
    - {if_id}
    - {did}
    - {ifindex}
    - {ifphysaddress}
    - {iftype}
    - {ifspeed}
    - {ifhighspeed}
    - {ifoperstatus}
    - {ifadminstatus}
- ***DNS Hostnames***. If Skylar One will use the DNS Reverse Lookup name as the device name (see the description of the field ***Hostname Precedence***), this field specifies whether Skylar One will use the fully-qualified domain name or only the hostname for each discovered device. Choices are:
    - *Strip Device Name (Hostname)*. Skylar One will use only the device name as the DNS hostname for each device.
    - *Use Full Domain Name (FQDN)*. Skylar One will use the fully-qualified domain name as the device name for each device.

Viewing Dynamic Application Data and Alerts

You can view the data collected from a device by Dynamic Applications in the following places:

- The **Device Performance** page allows you to view detailed reports for the selected device, including the graphs generated for each presentation object in each performance Dynamic Application that is aligned to the device.

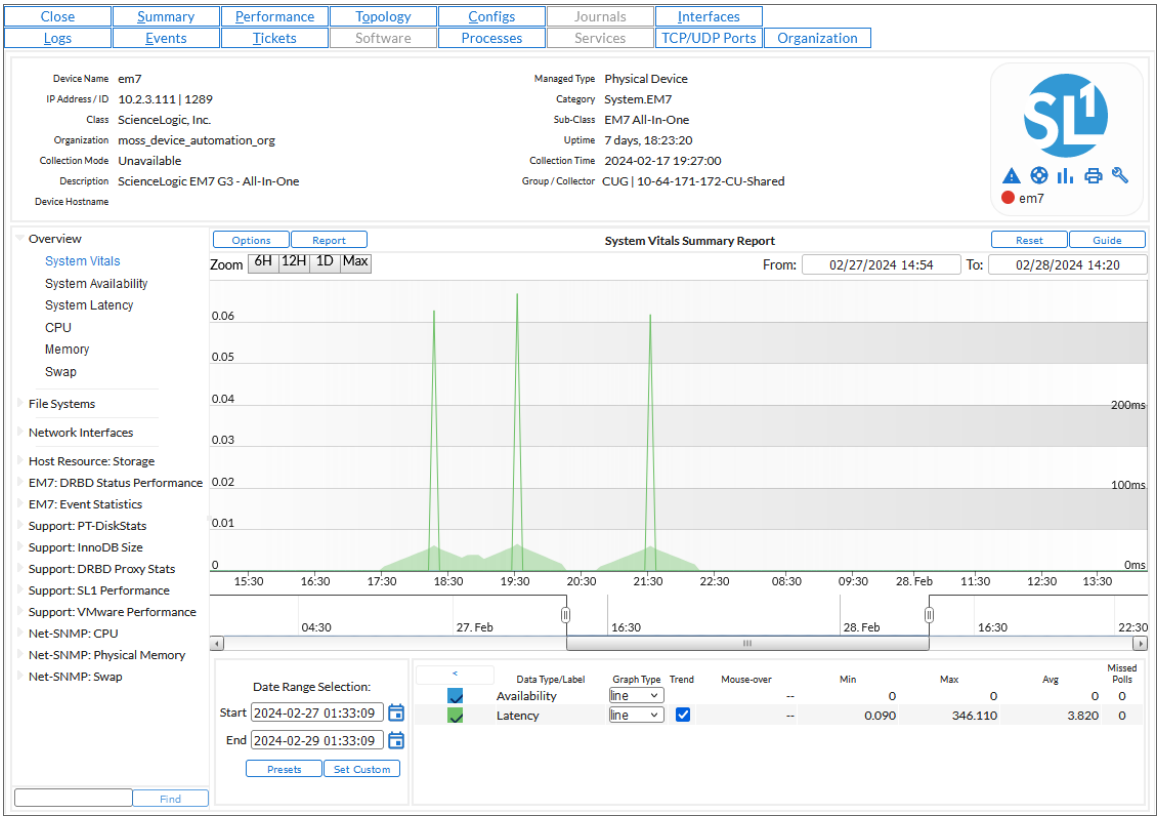
- The **Configuration Report** page displays data collected by all configuration Dynamic Applications aligned to a device.
- The **Journal Report** page displays data collected by all journal Dynamic Applications aligned to a device.

This section will describe how to view each type of Dynamic Application and how to view alerts generated by Dynamic Applications:

Performance Dynamic Applications

The **Device Performance** page displays performance reports for a device. To view the **Device Performance** page:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Find the device for which you want to view Dynamic Application data. Select its bar graph icon ( ).
3. Click the **[Performance]** tab. The **Device Performance** page is displayed.



The **Device Performance** page displays all performance graphs that are available for the device. These graphs include:

- Graphs defined by the Dynamic Applications associated with the device.
- Graphs generated using data collected by other processes, such as availability data, file system usage, interface usage, and data from monitoring policies.

Viewing Performance Dynamic Application Data

To view a performance graph defined in a Dynamic Application:

1. The bottom of the left NavBar of the **Device Performance** page lists all performance Dynamic Applications that include performance graphs. Select the Dynamic Application for which you want to view data.
2. Selecting a performance Dynamic Application will expand the NavBar item and display a list of all presentation objects in that Dynamic Application. Select a presentation object to view the graph for that presentation object.

For each graph associated with a performance Dynamic Application:

- The x-axis of a Performance Dynamic Application graph displays time.
- The y-axis displays the collected value(s).
- Mousing over the graph will display the exact value for each data series at that point on the graph.
- The exact values are displayed in the "Mouse-over" column in the table below the graph.

Viewing Details with the Data Table

At the bottom of each graph is a table that allows you to view details about each data series and overview information about the entire graph.

The data table includes the following:

- ***Data Type/Label***. For graphs that include multiple data series on a single graph (when the collection objects used to generate the graph have collected a list of values instead of a single value), each data series has its own row in this table. This column displays the name of the data series and how it is color-coded in the report. Clicking on the check-mark toggles the display of the data series in the graph.
- ***Trend***. Allows you to toggle the trendline on and off. The trendline shows average values on each side of the current point. This checkbox is checked (trendline is displayed) by default.
- ***Mouse-over***. When you mouse-over the graph, this column displays the exact value for each data series at that time point on the graph.
- ***Min***. The column displays the minimum value for the data series in the graph.
- ***Max***. This column displays the maximum value for the data series in the graph.
- ***Avg***. This column displays the average value for the data series in the graph.
- ***Missed Polls***. This column displays the number of times Skylar One was unable to collect the data within the time span of the graph.

Configuring the Data for the Report

You can use the **[Options]** menu to determine how you want to configure the data that is displayed in the graph. Your choices are:

- ***Default***. The initial graph that is displayed is not normalized and displays every collected value.
- ***Normalized***. In Skylar One, normalized data does not include polling sessions that were missed or skipped. So for normalized data, null values are not included when calculating maximum values, minimum values, or average values. When you select this option, Skylar One normalizes all the data collected in each 24 hour period and displays a single value for each day.
- ***Percentile***. Displays percent on the y-axis. This can be applied to normalized or non-normalized graphs.
- ***Kiosk***. Displays the report in full-page mode. This is helpful for NOC personnel who need to display graphs on large screens.
- ***Detach***. Spawns the graph in a new window.
- ***Series Selection***. Each performance graph is limited to displaying eight data series at a time. When you select this option, the **Graph Index Selection** page is displayed, where you can select up to eight data series to display.
- ***Edit Current Presentation***. Selecting this option allows you to edit the Presentation Object that was used to generate the graph.

Generating a Report from a Performance Graph

You can use the **[Report]** button to save the report to a file on your local computer. You can select the format to save the file in. Your choices are:

- ***HTML with Images***. Saves the graph and a table of all the data in the report, in HTML format.
- ***HTML Text Only***. Saves the report as a table of data, in HTML format.
- ***HTML Text Only all indexes***. Saves the report as a table of data, in HTML format. In the **Device Performance** page, the report can include up to eight data series (indexes); when you select this option, the HTML report will include all indexes collected by Skylar One, even if the number of indexes is greater than eight.
- ***CSV***. Saves the data from the report (usually date, time, and value) as comma-separated values.
- ***CSV all indexes***. Saves the data from the report (usually date, time, and value) as comma-separated values. In the **Device Performance** page, the report can include up to eight data series (indexes); when you select this option, the CSV report will include all indexes collected by Skylar One, even if the number of indexes is greater than eight.
- ***Graph Image Only***. Saves only the graph from the report as a .png file.
- ***ODS w/Chart Img***. Saves the graph and a table of all the data in the report, in ODS format.
- ***ODS Plain***. Saves the table of all the data in the report, in ODS format.
- ***XLSX w/ Chart Img***. Saves the graph and a table of all the data in the report, in XLSX (Excel) format.
- ***XLSX Plain***. Saves the table of all the data in the report, in XLSX (Excel) format.
- ***PDF w/chart Img***. Saves the graph and a table of all the data in the report, in PDF format.
- ***PDF Plain***. Saves the table of all the data in the report, in PDF format.

Defining the Date Range for a Report

The fields in the **Date Range Selection** pane allow you to define the time span that will be displayed in the report. When you define a Date Range, only data from the specified time span will be included in the report.

You can use the following fields, menus, and buttons to define the dates and associated data that will be displayed in the report:

- ***Start***. Allows you to define the Start date for the report. Data from this date until the specified End date will be included in the report.
- ***End***. Allows you to define the end date for the report. Data from the specified Start date until this End date will be included in the report.
- **[Presets]**. Allows you to select some pre-defined time spans for the report.
- **[Set Custom]**. Applies the values from the ***Start*** and ***End*** fields to the report.
- **[Range buttons]**. Allows you to zoom in or out for a specific time span. Choices are dependent upon the values selected in the **Date Range Selection** pane and can range from 6 hours to 90 days.
- You can drag the Date Range slider to the left or right, and the Performance graph will display the information for the selected time range.

Configuration Dynamic Applications

The **Configuration Report** page displays data collected from the device by configuration Dynamic Applications. Usually, configuration data contains static information about hardware and configuration settings, such as serial numbers, version numbers, and hardware status.

> **NOTE:** If you select the ***Hide Object*** checkbox for an object in the **Collection Objects** page (System > Manage > Dynamic Applications > Create/Edit), the object will not be included in the **Configuration Report** page.

You can define the layout of the **Configuration Report** page in the **Collection Objects** page for the Dynamic Application. In the **Collection Objects** page, you can use the ***Group*** field and the ***Table Alignment*** fields to define which objects will be grouped together, and which table each object will appear in.

You can enable change detection for an object in the in the **Collection Objects** page for the Dynamic Application, in the ***Change Alerting*** field. If an object's value has changed, it will be highlighted in red in the **Configuration Report** page. You can then click on the object's value in the **Configuration Report** page and view a list of historical values for the object.

For more information about configuring the table layout and change detection for a configuration Dynamic Application, see *Collection Objects*.

For objects of type "enum," you can mouseover the object and view all the possible values for the object.

NOTE: The **Configuration Report** page does not display Dynamic Applications that have *Cache Results* selected in the ***Caching*** field in the **Dynamic Applications Properties Editor** page. Dynamic Applications that cache results are designed to collect data only for other Dynamic Applications and cannot be used to display data.

To view Configuration Dynamic Application information:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Find the device for which you want to view configuration Dynamic Application data and select its bar graph icon ( ).
3. In the **Device Administration** panel, select the **[Configs]** tab. The **Device Configuration** page appears.

Selecting Data to View

If one or more Dynamic Applications of type "configuration" are associated with the device, the **Configuration Report** page will display that list of Dynamic Applications in the left NavBar.

NOTE: The left navigation bar does not display Dynamic Applications that have *Cache Results* selected in the ***Caching*** field in the **Dynamic Applications Properties Editor** page. Dynamic Applications that cache results are designed to collect data only for other Dynamic Applications and cannot be used to display data.

Viewing Data

When you select a Dynamic Application in the left NavBar, the right pane displays data collected from the device by the Dynamic Application.

- Some objects may appear in a list at the top of the right pane. These are objects that are not grouped into a table. For each of these values, no values were specified in the ***Group*** field and the ***Table Alignment*** field, in the **Collection Objects** page. These are usually objects for which there is only one, non-changing value (like model number, for example).
- Some objects may appear in tables. Tables work best for objects with multiple values, like RAM location. Each row represents one value from each collection object in the group, which all have the same index.
    - Each column heading is the name of an object. Mousing over the column heading displays a description of the object. To edit the description, click on the column heading. The **Collection Objects** page appears, populated with values from the appropriate object. You can edit the value in the *Description* field, and that value will appear when you mouseover the column heading in the **Configuration Report** page.
- Mousing over a value can display the following:

- If the object is of type "enum", the mouseover text displays the list of all possible values for the object. For example, "0 unknown, 1 disabled, 2 enabled".
- If change detection has not been enabled, displays the text "Change detection is disabled. No history available".
- If change detection has been enabled, displays "Click to view change history". If you click, Skylar One displays the **Change History** modal, where you can view all the values collected from the device for the selected object.

Generating a Report of the Data

You can generate a report about the data in the **Configuration Report** page. To do so:

1. In the **Configuration Report** page, in the Navigation Bar (left pane), select the Dynamic Application you want to generate a report from.
2. In the **Configuration Report** page, select the **[Actions]** menu. Select *Print a Report*.
3. Skylar One generates an HTML report that contains all the data from the **Configuration Report** page. You can view, print, or save the report.

Viewing Historical Data

By default, the **Configuration Report** page displays data from the latest polling session. However, you can use the **Snap-Shot Selector** page to display data from a previous polling session in the **Configuration Report** page.

The **Snap-Shot Selector** page displays a list of polling sessions where a change was discovered in the configuration data. If none of the data in a Dynamic Application changes from one polling session to the next, then Skylar One does not include an entry in the **Snap-Shot Selector** page.

To display data from a previous polling session in the **Configuration Report** page:

1. In the **Configuration Report** page, in the Navigation Bar (left pane), select the Dynamic Application for which you want to view historical data.
2. When the data is displayed in the right pane, select the **[Snap-Shots]** button.
3. The **Snap-Shot Selector** modal page appears. This page displays a calendar interface, in which you can select a date for which you want to view a list of Snap-Shots.
4. To select a date for a Snap-Shot, scroll through the calendar until you find the month that you are interested in. Click on the date you are interested in.
5. The pane to the right will display a list of all available Snap-Shots for the selected date. Each Snap-Shot is labeled with a date and time stamp and specifies how many objects had changed values. To select a Snap-Shot, click on it and select the **[View Snapshot]** button.

> **NOTE**: If the pane to the right does not display one or more available Snap-Shots, this means that Skylar One did not detect any changes to the objects on the selected date.

6. The data from the selected Snap-Shot is loaded and displayed in the **Configuration Report** page.

Editing the Application

From the **Configuration Report** page, you can edit the properties of a Dynamic Application. When you do so, you change the behavior of the Dynamic Application for all subscriber devices, not just the current device.

To edit a Dynamic Application from the **Configuration Report** page:

1. In the **Configuration Report** page, in the Navigation Bar (left pane), select the Dynamic Application you want to view and edit.
2. When the data from the Dynamic Application is displayed in the right pane, select the **[Actions]** menu and choose *Edit This Application*.
3. The **Collection Objects** page appears. In this page, you can edit how Skylar One retrieves values for an object and how those values are displayed in the **Configuration Report** page. You can also access all the other tabs in the Dynamic Applications panel for the Dynamic Application.

For more information about editing collection objects, see the *Collection Objects* section.

Journal Dynamic Applications

The **Journal View** page displays journal entry information collected from the device by Dynamic Applications. All information from Dynamic Applications of type journal is included in the **Journal View** page. Journal Dynamic Applications store information in log format; for example, telephone call records or access logs.

> **TIP:** To sort the list, click on a column heading. The list will be sorted by the column value, in ascending order. To sort the list by descending order, click the column heading again. You can also filter the items on this inventory page by typing filter text or selecting filter options in one or more of the filters found above the columns on the page. For more information, see *Filtering the Items on a Classic Page* in the ***Introduction to Skylar One*** manual.

To view journal Dynamic Application information:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Find the device for which you want to view journal Dynamic Application data and select its bar graph icon ( ).
3. In the **Device Reports** panel, select the **[Journals]** tab. The **Journal View** page appears.

Selecting Data to View

If one or more journal Dynamic Applications are associated with the device, the **[Journals]** tab of the **Device Investigator** (or the **Journal View** page in the classic Skylar One user interface) will display that list of Dynamic Applications on the left side of the page.

When you select a Dynamic Application on the left side of the page, the right pane displays data collected from the device by the selected Dynamic Application.

Viewing Data

The **[Journals]** tab of the **Device Investigator** (or the **Journal View** page in the classic Skylar One user interface) arranges collected journal entries in tabular format.

- The table contains a row for each journal entry.
- The table contains a column for each presentation object, plus the ***State*** and ***Collected On*** columns. Presentation objects define the text to display in each row in the column, including which collection values will be displayed. Presentation objects are defined in the **Presentation Objects** page for the Dynamic Application.

The **[Journals]** tab of the **Device Investigator** (or the **Journal View** page in the classic Skylar One user interface) displays the following about each journal entry:

> **TIP:** To sort by descending order, click the column heading again. To sort a column that contains presentation objects, sorting must be enabled in the **Presentation Objects** page (System > Manage > Dynamic Applications > Create/Edit). Date and time column sorts by descending order on the first click; to sort by ascending order, click the column heading again.

- ***Presentation Objects***. One or more columns in the table of journal entries will be presentation objects defined in the Dynamic Application. The values in this column can be based on one or more collection objects, and can be a text string, a number, or a time and date value.
- ***State***. Specifies the current state of the journal entry. Journal entries can have one of the following states:
    - Open
    - Closed
    - Abandoned
    - Error
    - Reopened
- ***Collected On***. Specifies the last time the journal entry was updated.

Generating a Report of the Data

You can generate a report about the data in the **[Journals]** tab of the **Device Investigator** (or the **Journal View** page in the classic Skylar One user interface).

To generate a report about the a device's journal data:

1. Go to the **[Journals]** tab of the **Device Investigator** (or the **Journal View** page in the classic Skylar One user interface).

2. In the left NavBar, select the Dynamic Application from which you want to generate a report.
3. You can filter the journal entries to include in the report. Using the search filters at the top of the table of journal entries, filter the list of journal entries so that only the journal entries you want to include on the report are displayed.
4. Click the **[Actions]** button and then select *Generate Report*.
5. The **Export current view as a report** page displays. Select the output format for the report, optionally select if Skylar One must force the browser to save the file to disk, and then click **[Generate]**.

Editing the Application

From the **[Journals]** tab of the **Device Investigator** (or the **Journal View** page in the classic Skylar One user interface), you can edit the properties of a Dynamic Application. When you do so, you change the behavior of the Dynamic Application for all subscriber devices, not just the current device.

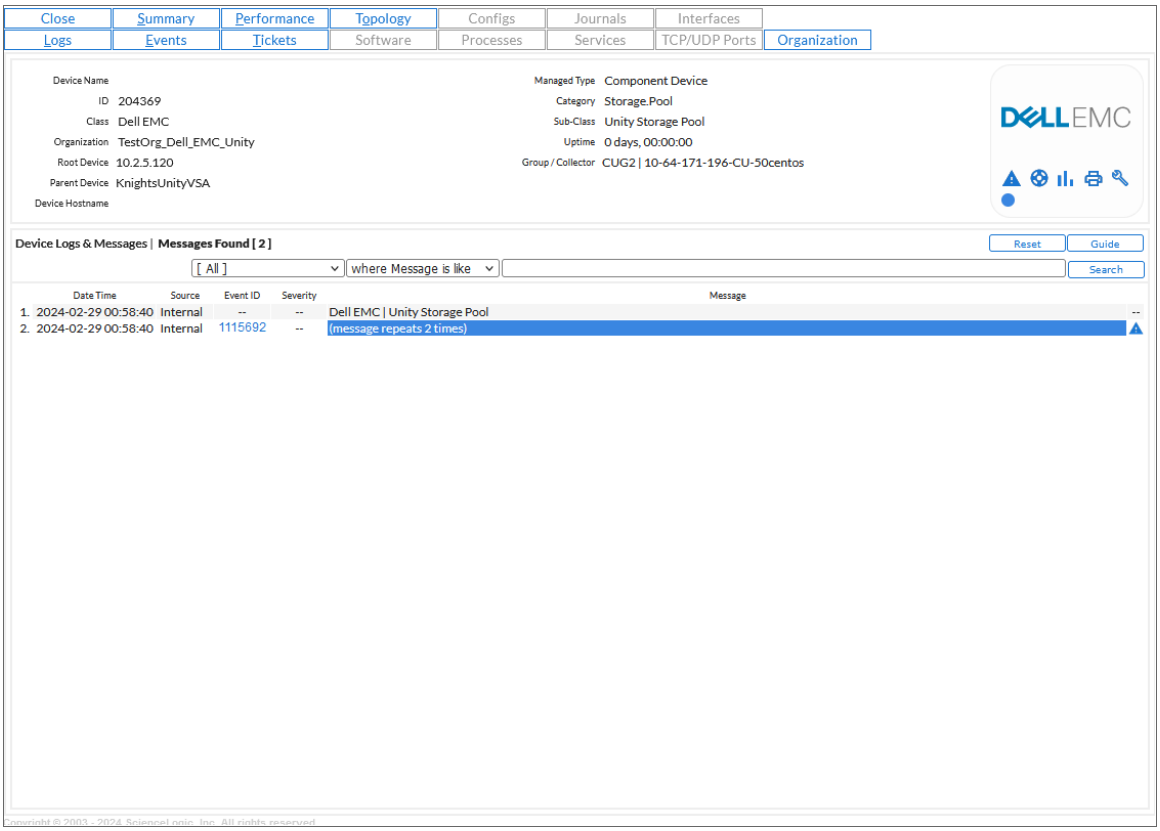
To edit a journal Dynamic Application:

1. Go to the **[Journals]** tab of the **Device Investigator** (or the **Journal View** page in the classic Skylar One user interface).
2. In the left NavBar, select the Dynamic Application you want to view and edit.
3. When the data from the Dynamic Application is displayed in the right pane, click the **[Actions]** button and then select *Edit This Application*.
4. The **Collection Objects** page appears. In this page, you can edit how Skylar One retrieves values for an object. You can also access all the other tabs in the Dynamic Applications panel for the Dynamic Application.

Viewing Alerts Generated by Dynamic Applications

You can view alerts generated by a Dynamic Application in the **Device Logs & Messages** page. To view alerts on a device generated by a Dynamic Application:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the device for which you want to view alerts generated by a Dynamic Application. Select its bar graph icon ( ).
3. Click the **[Logs]** tab. The **Device Logs & Messages** page displays the following for each log message:



- ***Date Time***. The date and time the entry was made in the log.
- ***Source***. The entity or process that generated the message. If generated by a Dynamic Application, the source will be *Dynamic*.
- ***Event ID***. If an event was created, a unique event ID, generated by Skylar One. If the log entry is not associated with an event, no ID appears in this column.
- ***Priority***. This column is not applicable to alerts generated by Dynamic Applications.
- ***Message***. Text of the log entry, color-coded to match event severity (if applicable).

Searching the List of Log Entries for a Dynamic Application

The search fields at the top of the **Device Logs & Messages** pane allows you to search log entries for messages generated by a Dynamic Application associated with a device. To search the list of log entries for a Dynamic Application:

1. In the **Device Logs & Messages** pane, select the **[Search All Messages]** drop-down list.
2. Select *[Search Dynamic]*.
3. In the **drop-down list** search bar, you can optionally define the parameters for the search. Choices are:
    - *Where Message is like*. Search the message portion of the log entry.
    - *Where Date received is like*. Search the date portion of the log entry.

- *Where Priority is like*. This selection is not applicable to alerts generated by Dynamic Applications.

4. In the regular expression field, you manually enter the text to search for. You can use the following special characters in this field:
   - * Match zero or more characters preceding the asterisk. For example:

     "dell*" would match "dell", "dell2650", "dell7250" and "dell1700N".

     "*dell*" would match "mydell", "dell", "dell2650", "dell7250" and "dell1700N".

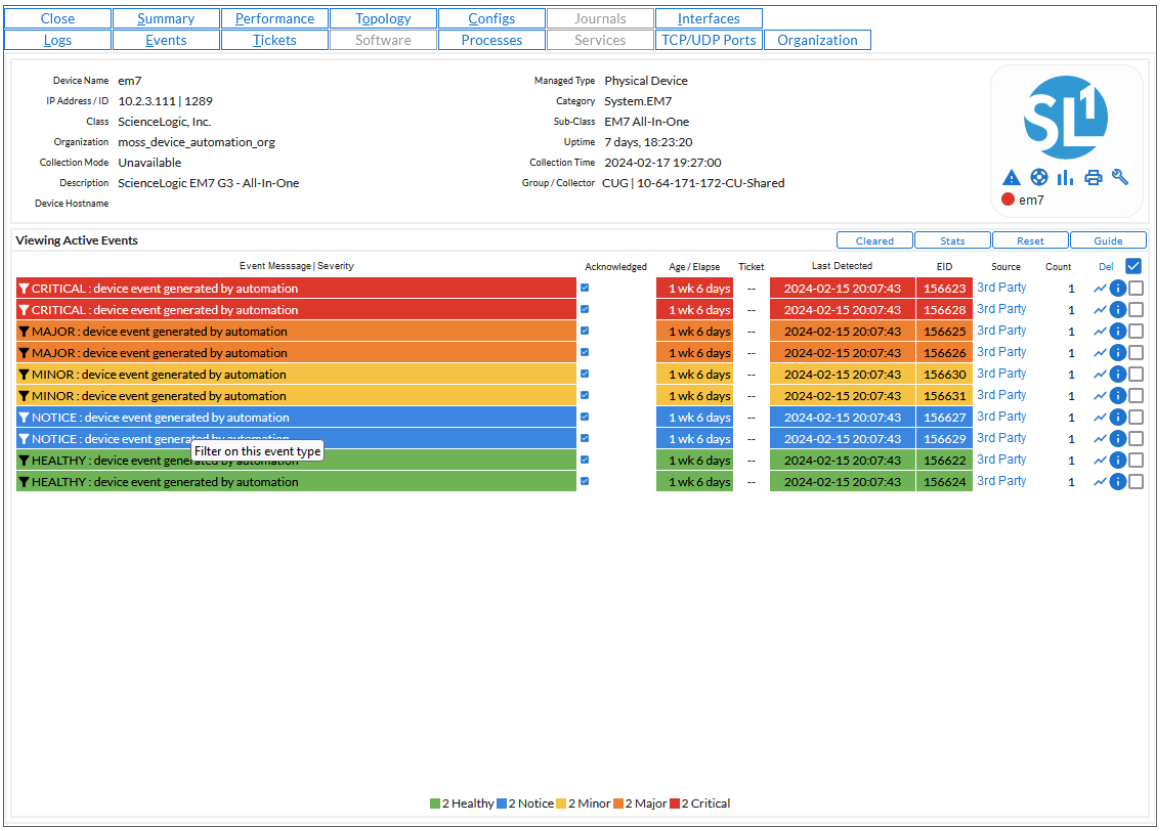
   - % Match zero or more characters preceding the percent symbol. This special character behaves in the same way as the asterisk.
5. When you select the **[Search]** button, the **Device Logs & Messages** page will be refreshed and will display only log entries that match the search parameters.

Viewing Active Events for a Dynamic Application

Events are messages that are triggered when a specific condition is met. For example, an event can signal that a server has gone down, that a device's hard drives are getting too full, or simply display the status of a device. You can view events generated by Dynamic Applications in the **Viewing Events** page. You can view active events, as well as events that have been cleared.

To view active events generated by a Dynamic Application:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the device for which you want to view events generated by a Dynamic Application. Select its bar graph icon ( ).
3. In the **Device Reports Panel**, select the **[Events]** tab. The **Viewing Events** page displays the following for each event message:



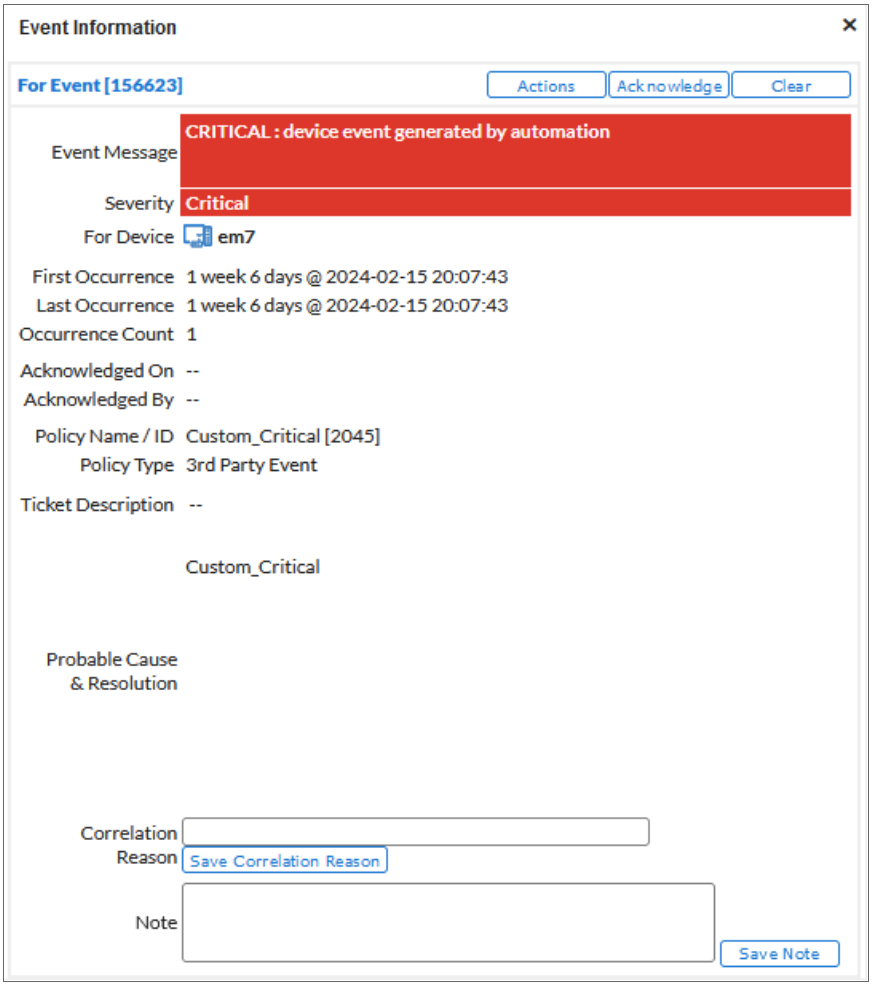
- ***Event Message | Severity***. Message generated by event, as defined in the **Event Policy Editor** page. The message is color-coded for severity.
    - Critical. Critical events are those that require immediate attention.
    - Major. Major events are those that require immediate investigation.
    - Minor. Minor events are those that need to be investigated before problems become severe.
    - Notice. Notice events are those that require attention but are not problem-related.
    - Healthy. Healthy events are those that are not urgent.
- ***Acknowledged***. Specifies whether anyone has acknowledged this event.
    - *Red check*. Event has not been acknowledged.
    - *Gray check with name*. Event has been acknowledged.
- ***Age / Elapse***. Number of days, hours, and minutes since the last occurrence of the event.
- ***Ticket***. Ticket ID associated with this event, if applicable.
- ***Last Detected***. Date and time of last occurrence of the event.
- ***EID***. Unique ID for the event, generated by Skylar One. By default, this column appears only if specified in your Account Settings or if you select the **[Advanced]** button.

- ***Source***. Source of the log message that triggers the event, as defined in the **Event Policy Editor** page. If the event was generated by a Dynamic Application, the ***Source*** will be *Dynamic*.
- ***Count***. Number of times this event has occurred.
- ***External Ticket***. The numeric ID associated with a ticket from an external ticketing system (that is, a ticket that was not created in Skylar One). If this field displays a value, you can click on that value to spawn a new window and view the external ticket.

Viewing Details About an Active Event

You can view details about an active event in the **Event Information** page. The **Event Information** page displays more detailed data about a single active event. To view details about an active event:

1. In the **Viewing Active Events** pane, find the event for which you want to view detailed information.
2. Select its information icon (ℹ).
3. The **Event Information** page displays the following for each active event:



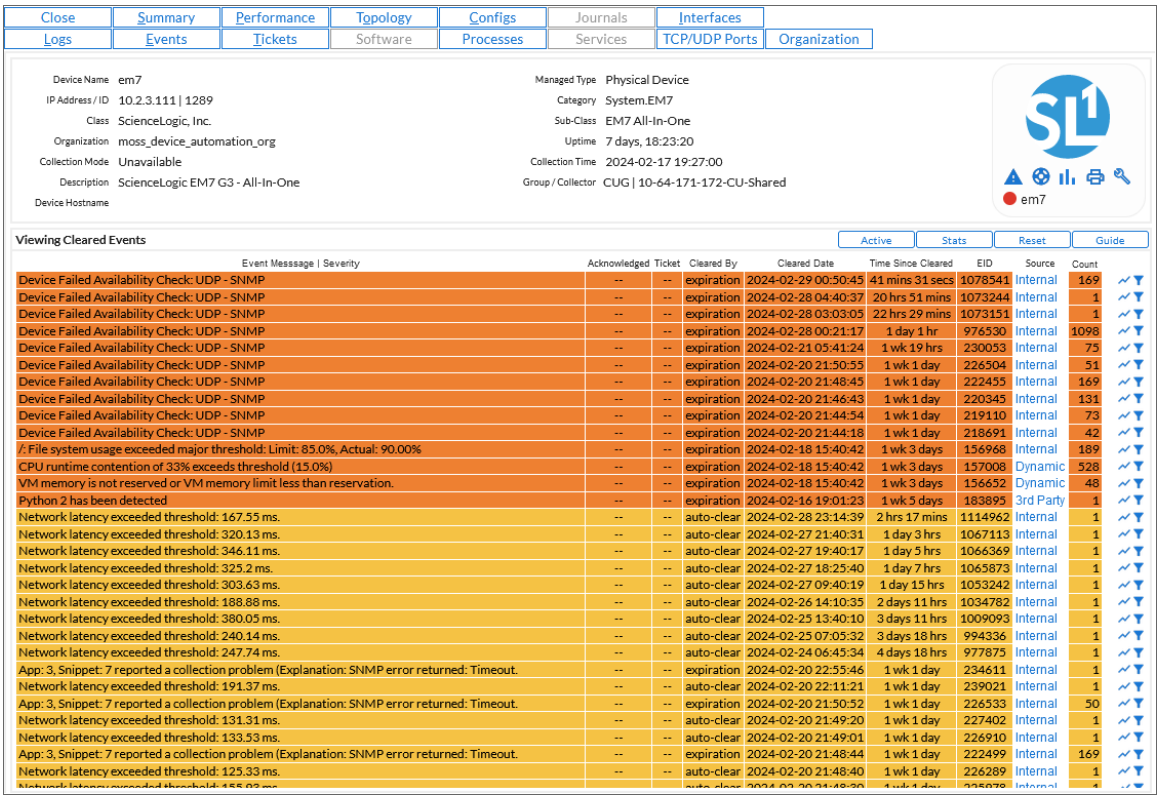
- ***Event ID***. Unique ID for the event, generated by Skylar One.
- ***Event Message***. Message generated by event, as defined in the **Event Policy Editor** page.
- ***Severity***. Severity of the event. Choices are:
    - *Critical*
    - *Major*
    - *Minor*
    - *Notice*
    - *Healthy*
- ***For Element***. Name of the element associated with the event.
- ***First Occurrence***. Number of days and hours since the first occurrence of the event. Date and time of first occurrence of the event.
- ***Last Occurrence***. Number of days and hours since the last occurrence of the event. Date and time of last occurrence of the event.
- ***Occurrence Count***. Number of times this event has occurred on this entity.
- ***Acknowledged On***. Date and time the event was acknowledged.
- ***Acknowledged By***. Username of user who acknowledged the event.
- ***Policy Name/ID***. Name of the event policy, as defined in the **Event Policy Editor** page and policy ID.
- ***Policy Type***. Source of the log message that triggers the event, as defined in the **Event Policy Editor** page. If the event was generated by a Dynamic Application, the ***Policy Type*** will be *Dynamic*.
- ***Ticket Description***. Description field from the associated ticket, if applicable.
- ***Probable Cause & Resolution Text***. This pane displays additional information about the event, as defined in the **Event Policy Editor** page.

Viewing Cleared Events from a Dynamic Application

You can view cleared events from a Dynamic Application from the **Viewing Cleared Events** page. A cleared event is an event that is no longer occurring, and has been manually cleared by a user, or has expired.

To view cleared events from a Dynamic Application:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the device for which you want to view events generated by a Dynamic Application. Select its bar graph icon ( ).
3. Click the **[Events]** tab.
4. In the **Viewing Events** page, click the **[Cleared]** button. The **Viewing Cleared Events** page displays the following information:



- ***Event Message | Severity***. Message generated by event, as defined in the **Event Policy Editor** page. The message is color-coded for severity.
    - **Critical**. Critical events are those that require immediate attention.
    - **Major**. Major events are those that require immediate investigation.
    - **Minor**. Minor events are those that need to be investigated before problems become severe.
    - **Notice**. Notice events are those that require attention but are not problem-related.
    - **Healthy**. Healthy events are those that are not urgent.
- ***Acknowledged***. Specifies whether anyone has acknowledged this event.
    - *Gray Dash*. Event has not been acknowledged.
    - *Gray check with name*. Event has been acknowledged.
- ***Ticket***. Ticket ID associated with this event, if applicable.
- ***Cleared By***. Username of the person who cleared the ticket.
- ***Cleared Date***. Date and time on which event was cleared.
- ***Time Since Cleared***. Number of days, hours, and minutes since the event was cleared.
- ***EID***. Unique ID for the event, generated by Skylar One. By default, this column appears only if specified in your Account Settings or if you select the **[Advanced]** button.

- ***Source***. Source of the log message that triggers the event, as defined in the **Event Policy Editor** page. If the event was generated by a Dynamic Application, the ***Source*** will be *Dynamic*.
- ***Count***. Number of times this event has occurred.

Dynamic Application Settings

Every Dynamic Application has a set of properties specific to that Dynamic Application. This section will describe how to view and edit the properties for a Dynamic Application.

There are also some global settings that affect the operation of Dynamic Applications; these global settings are described in *Managing Dynamic Applications*.

Dynamic Application Properties

Every Dynamic Application has a set of properties that define the general settings for that Dynamic Application. To view and edit the settings for a Dynamic Application:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to view or edit the properties for. Select its wrench icon ( ). The **Dynamic Applications Properties Editor** page is displayed.
3. A Dynamic Application has the following properties:
    - ***Application Name***. The name of the Dynamic Application.
    - ***Application Type***. Specifies the Dynamic Application type, which comprises a *protocol* and an *archetype*. For a description of protocols and archetypes, see the *Types of Dynamic Application* section. The following options are available in the ***Application Type*** field:
        - *Bulk Snippet Configuration*
        - *Bulk Snippet Performance*
        - *Database Configuration*
        - *Database Performance*
        - *IT Service*
        - *PowerShell Configuration*
        - *PowerShell Performance*
        - *SNMP Configuration*
        - *SNMP Performance*
        - *SOAP Configuration*
        - *SOAP Performance*
        - *Snippet Configuration*

- *Snippet Journal*
- *Snippet Performance*
- *WMI Configuration*
- *WMI Performance*
- *XML Configuration*
- *XML Performance*
- *XSLT Configuration*
- *XSLT Performance*

NOTE: The ***Application Type*** field cannot be changed after the Dynamic Application is created. When editing an existing Dynamic Application, the ***Application Type*** field is view-only.

- ***Execution Environment***. The execution environment to which the Dynamic Application is aligned. This field appears only for snippet and internal collection Dynamic Applications. An execution environment contains the supporting modules, code, scripts, directories, and files (packaged in ScienceLogic Libraries) for the snippet. An execution environment includes its own installation directories, doesn't share libraries with other environments, and allows granular control of dependencies and versions. The default execution environment is *System*. For more information, see the ***ScienceLogic Libraries and Execution Environments*** manual.
- ***Caching***. This field is disabled for Dynamic Applications that use Bulk Snippet, Database, or Internal Collection (ICDA) type. When Skylar One requests information from a device during Dynamic Application collection, Skylar One can optionally ***cache*** the response from the device. If a response is cached, other Dynamic Applications that use the same protocol can retrieve collection object values from the cached response. When you cache responses, you reduce the number of requests performed by Skylar One and speed up collection. Typically, Dynamic Applications aligned to component devices retrieve values from a cached repository. The cached repository is created with a Dynamic Application aligned with the root device. Choices are:

  NOTE: When Skylar One performs Dynamic Application collection, the Dynamic Application API uses the ***Verify SSL for Collections by Default*** setting (System > Settings > Collection Settings) as the default value for SSL verification during collection, instead of a hardcoded value. This change does not affect credential types that do not support SSL verification, such as classic PowerShell or SSH credentials, and does not affect SNMP Dynamic Applications.

  - *No caching*. Specifies that this Dynamic Application does not use the caching feature. The requests that are used to populate the collection objects in this Dynamic Application are defined within this application and the responses to those requests are not available to other Dynamic Applications.

- *Cache Results*. Specifies that Skylar One should cache the responses from the requests defined in this Dynamic Application. Other Dynamic Applications that use the same protocol or Snippet Dynamic Applications can retrieve data from the cached responses associated with this Dynamic Application. When this option is selected, the collection objects defined in this Dynamic Application are no longer collected at a regular frequency. If you select this option, you must not define collection objects that are used to generate configuration tables or performance graphs.
- *Consume Cached Results*. Specifies that the collection objects defined in this Dynamic Application are populated with values from other Dynamic Applications that use the same protocol and have *Cache Results* enabled.
  - In a SOAP or WMI Dynamic Application that uses the *Consume Cached Results* option, you cannot define requests.
  - For XSLT Dynamic Applications that use the *Consume Cached Results* option, you must still define the XSLT Parser Code that will be used to transform the cached responses.
  - For Snippet Dynamic Applications that use the *Consume Cached Results* option, you must still define snippet code that processes the cached responses; however, Snippet Dynamic Applications that use the *Consume Cached Results* option can consume cached responses from any Dynamic Application type that can cache responses.
  - Dynamic Applications that use the *Consume Cached Results* option can use cached responses from multiple different Dynamic Applications.
  - If a cached response is unavailable when Skylar One performs collection for a "Consume Cached Results" Dynamic Application, Skylar One will automatically attempt to execute the appropriate request to re-populate the cache.

NOTE: If a Dynamic Application includes *Consume Cached Results* and is aligned with a device that is ***not*** a component device, the Dynamic Application that provides the results to be consumed (includes *Cache Results*) must be aligned with the same device. If a Dynamic Application includes *Consume Cached Results* and is aligned with device that is a ***component device***, the related Dynamic Application that provides the results to be consumed (includes *Cache Results*) must be aligned with the root device for that component.

- ***Device Dashboard***. Select a device dashboard from a list of all device dashboards in Skylar One. The selected device dashboard will be associated with all devices that subscribe to this Dynamic Application and will appear as an option in the **Device Summary** page. For more information about device dashboards, see the ***Classic Dashboards*** manual.
- ***Version Number***. Specifies the current revision level of the Dynamic Application as specified by the developer of the Dynamic Application.

- ***Operational State***. Specifies whether Skylar One will collect data from devices using the Dynamic Application. The ***Operational State*** also specifies whether Skylar One will automatically align the Dynamic Application to devices during discovery, re-discovery, and nightly auto-discovery. The following options are available:
    - *Enabled*. Skylar One will collect data from devices using the Dynamic Application and will automatically align this application to devices during discovery, re-discovery, and nightly auto-discovery.
    - *Disabled*. Skylar One will not collect data from devices using the Dynamic Application and will not align this application to devices during discovery, re-discovery, and nightly auto-discovery. Selecting *Disabled* does not remove existing device alignments for the Dynamic Application and does not prevent a user from manually aligning the Dynamic Application with a device.
- ***Collector Affinity***. Specifies which Data Collectors are allowed to run collection jobs for this Dynamic Application. Choices are:
    - *Default*. If the Dynamic Application is auto-aligned to a component device during discovery, then the Data Collector assigned to the root device will collect data for this Dynamic Application as well. For devices that are not component devices, the Data Collector assigned to the device running the Dynamic Application will collect data for the Dynamic Application.
    - *Root Device Collector*. The Data Collector assigned to the root device will collect data for the Dynamic Application. This guarantees that Dynamic Applications for an entire Device Component Map (DCM) tree will be collected by a single Data Collector. You might select this option if:
        - The Dynamic Application has a cache dependency with one or more other Dynamic Applications.
        - You are unable to collect data for devices and Dynamic Applications within the same Device Component Map on multiple Data Collectors in a collector group.
        - The Dynamic Application will consume cache produced by a Dynamic Application aligned to a non-root device (for instance, a cluster device).
        - The Dynamic Application includes snippet code with a ***root_device*** tuple.
    - *Assigned Collector*. The Dynamic Application will use the Data Collector assigned to the device running the Dynamic Application. This allows Dynamic Applications that are auto-aligned to component devices during discovery to run on multiple Data Collectors. You might select this option if:
        - The Dynamic Application has no cache dependencies with any other Dynamic Applications.
        - You want the Dynamic Application to be able to make parallel data requests across multiple Data Collectors in a collector group.

- The Dynamic Application can be aligned using mechanisms other than auto-alignment during discovery (for instance, manual alignment or alignment via Device Class Templates or Run Book Actions).

- ***Poll Frequency***. Specifies how often Skylar One will use this Dynamic Application to collect data from the device(s) the Dynamic Application is aligned with. Choices range from *Every 1 Minute* to *Every 24 Hours*.

NOTE: You can define a custom poll frequency for a Dynamic Application aligned with one or more devices in a device template. The poll frequency defined in the device template overrides the poll frequency defined for the Dynamic Application. Devices to which the device template has been applied will use the poll frequency defined in the device template. You can override both the ***Poll Frequency*** field and on a per-device basis from the **Dynamic Application Collections** page.

- ***Maximum Devices***. Appears only for Dynamic Applications of type Bulk Snippet Configuration and Bulk Snippet Performance. Specifies the maximum number of devices that a single instance of the Dynamic Application can collect data from. If the number of discovered devices exceeds the value in this field, Skylar One runs additional instances of the Dynamic Application. Enter a number between 1 and 500.
- ***Abandon Collection***. Specifies how many collection objects must timeout in one poll period before Skylar One stops trying to collect data for the Dynamic Application from a specific device. For more information about how this field works, see the *Abandon Collection Property* section.
- ***Context***. Optional field that is used for Dynamic Applications that use the SNMP protocol. For SNMPv3, specifies the snmp ID of the device from which you want to retrieve data. This helps differentiate between multiple instances of MIBs and OIDs.
- ***Disable Rollup of Data***. If you select this checkbox, Skylar One does not roll up the collected data from this Dynamic Application. Selecting this checkbox decreases the workload on the Skylar One System. Skylar One displays rolled-up data from Dynamic Applications in multiple pages throughout the user interface, including, but not limited to, dashboard widgets, reports, and device summary/performance pages. If you select this checkbox, those pages that depend on rolled-up data will no longer display data for this Dynamic Application.
- ***Description***. Optional field that describes the purpose and function of the Dynamic Application.
- ***Release Notes & Change Log***. Optional field that specifies any additional information the developer wants to provide with the Dynamic Application.

For Dynamic Applications that use the Snippet, SOAP, WMI, XML, or XSLT protocols, the following additional fields appear:

- ***Caching***. For information about this field, see *Caching*. If you do not want to use the caching feature, select *No Caching* in this field.

NOTE: The ***Caching*** field does not appear for Bulk Snippet Dynamic Applications.

- ***Component Mapping***. If you select this checkbox, Skylar One will use data collected by the Dynamic Application to create Component Devices. For more information about this checkbox, see *Dynamic Component Mapping* section.

For Dynamic Applications of type "configuration, the following additional fields appear:

- ***Null Row Option***. This field is active only for Dynamic Applications of type "configuration". If this Dynamic Application does not collect any values for a row in a table of configuration data that previously returned data, you can specify how Skylar One will display the row in the **Configuration Report** page. Choices are:
    - *--values*. Skylar One will display the "--" (dash dash) characters to signify that no data was collected for the current snapshot.
    - *Last collected*. Skylar One will display the last successfully collected values for the row. Skylar One will not indicate that it has collected no values for the row.
    - *Last collected with indicator*.
    - *Hide Row*. Skylar One will not display the row that contains no values.
- ***Null Column Option***. This field is active only for Dynamic Applications of type "configuration". If this Dynamic Application does not collect a value for a table cell for which data was previously collected, you can specify what Skylar One will display in the table cell in the **Configuration Report** page. Choices are:
    - *--values*. Skylar One will display the "--" (dash dash) characters to signify that no data was collected for the current snapshot.
    - *Last collected*. Skylar One will display the last successfully collected value for the table cell. Skylar One will not indicate that it has collected no value for the table cell.
    - *Last collected with indicator*. Skylar One will display the last successfully collected value for the table cell. The value in the table cell is highlighted, to indicate that this is not the latest value, but only the latest successfully collected value.

4. To edit one or more of the properties for a Dynamic Application, change the value for that property, and then select the **[Save]** button.

> **NOTE:** If you modify a Dynamic Application that is included in an imported PowerPack, for example, Dynamic Applications provided by ScienceLogic, your changes will be overwritten if the PowerPack is reimported and reinstalled.

The Abandon Collection Property

The ***Abandon Collection*** field in the **Dynamic Applications Properties Editor** specifies how many collection objects must timeout before Skylar One stops trying to collect data for the Dynamic Application from a specific device. At the next scheduled collection interval, Skylar One will try again to collect data for this Dynamic Application from the device.

The following options are available in the ***Abandon Collection*** field:

- ***Default***. Specifies a threshold of two collection objects.
- ***After 1 Timed-Out Objects*** to ***After X Timed-Out Objects***. Specifies how many collection objects must time out before this collection session is abandoned on the device. Options will be displayed for each number between 1 and the number of collection objects defined in the Dynamic Application. For example, if three collection objects are defined in a Dynamic Application, the following options are available:

1. *Default*
2. *After 1 Timed-Out Objects*
3. *After 2 Timed-Out Objects*
4. *After 3 Timed-Out Objects*

During a polling session on a device, the frequency of which is controlled by the ***Poll Frequency*** field, the Dynamic Application collection process connects to the aligned device using the aligned credential and then requests each collection object. If Skylar One cannot retrieve a value for a collection object before the timeout limit (defined in the credential), the collection object is considered timed-out. In a single polling session on a single device, when the number of collection objects that are timed-out reaches the number specified in the ***Abandon Collection*** field, Skylar One abandons collection of the Dynamic Application on the device for the current polling session. When collection is abandoned:

- Skylar One will not attempt to collect any of the other collection objects in the Dynamic Application from that device in this polling session.
- Skylar One can generate an event with the message "Dynamic Application stopped collection - abandon threshold crossed". The event will include the Dynamic Application ID, device ID, and the collection object IDs that timed out.

> **NOTE:** The event "Dynamic Application Stopped collection - abandon threshold crossed" is disabled by default. To enable this event, go to the **Event Category Manager** page and edit the definition for this event.

- Subsequent polling sessions are not affected. During the next polling session, the collection process for the Dynamic Application will attempt to collect all the collection objects
- If a device is unavailable or consistently abandons collection for more than two days, Skylar One will set the ***Collect*** status of all objects for all the Dynamic Applications aligned with the device to *No*.

> **NOTE:** For all objects except those retrieved from a database, the timeout limit is specified in the credential. For database objects, the timeout limit is specified internally by Skylar One.

Creating a Dynamic Application

This chapter describes how to create Dynamic Application for Skylar One.

NOTE: Skylar One creates an audit log message whenever you create, edit, or delete a Dynamic Application. For more information, see the "Daily Health Tasks" chapter of the ***System Administration*** manual.

To create a new Dynamic Application, you must perform the following general steps:

1. Determine what data you want to collect from a monitored device that would help you manage that device.
2. Determine what management interfaces are available on the monitored device and how the data that you need is made available - through an SNMP agent, WMI agent, web server, database, or another method. You can then choose the appropriate protocol type for your Dynamic Application. The same information might be available through multiple management interfaces on the monitored device; you will need to determine which management interface you will use based on ease of implementation, security, and other factors.
3. Determine which data points you want to collect, and how you want that data displayed in Skylar One. You can then:
    - Choose the archetype of your Dynamic Application.
    - List the requests, collection objects, and presentation objects you will need to create.
4. Determine the other features that you will use for this Dynamic Application. In addition to collecting and displaying data, Skylar One can:
    - Automatically align your Dynamic Application to devices during discovery.
    - Generate alerts when the collected data meets specified thresholds.
    - Create component devices using the collected data.
    - Cache the results of requests defined in Snippet, SOAP, WMI, XML, and XSLT Dynamic Applications for use by other Dynamic Applications.

      NOTE: If you configure your Dynamic Application to cache the results, you cannot include collection objects (other than a discovery object) or presentation objects in the same Dynamic Application. You must create a second Dynamic Application to use the cached results to display data in Skylar One. If you configure your Dynamic Application to cache the results, skip steps six and seven of these instructions.

    - Use the performance graphs defined in a Dynamic Application to represent CPU, Memory, or Swap utilization for subscriber devices.
    - Use the collected data to automatically populate asset records.
    - Use the collected data to automatically populate inventory lists for hardware components and installed software.
    - Use the collected data to create and/or populate custom attributes.

5. Create the container for the Dynamic Application. During this step, you will use the protocol and archetype you chose in the previous steps to define the Dynamic Application type. You will also define the basic properties for your Dynamic Application, such as how often Skylar One will perform collection. There are three methods for creating a container for your Dynamic Application:
    - Creating a new Dynamic Application from scratch.
    - Creating a new Dynamic Application by duplicating an existing Dynamic Application using the **[Save As]** button. This option is useful if you are creating multiple, similar Dynamic Applications.
    - If you are creating an SNMP Dynamic Application, use the SNMP Walker Tool to create the Dynamic Application and populate the collection objects.
6. If you are creating a Snippet, Bulk Snippet, SOAP, WMI, or XSLT Dynamic Application, define the requests or snippets that will be used to populate the collection objects in your Dynamic Application. If one or more other Dynamic Applications are already configured to cache all the data you need to populate the collection objects in this Dynamic Application, you can configure your Dynamic Application to consume cached requests.
7. Create the collection objects for the Dynamic Application.
8. If you are creating a performance or journal Dynamic Application, create the presentation objects for the Dynamic Application.
9. At this stage of development, you might want to test the basic functionality of your Dynamic Application before adding additional features such as discovery objects or alerts. To test your Dynamic Application:
    - If required, create the appropriate credential for your Dynamic Application.
    - Discover a device from which the Dynamic Application can collect data.
    - In the **Dynamic Application Collections** page for the device, manually align the Dynamic Application.

    TIP: To speed up your testing, you might want to change the ***Poll Frequency*** setting in the **Dynamic Applications Properties Editor** to *Every 1 minute*. When your Dynamic Application is complete, you can change this setting back to the desired value.

10. Implement the other features you want to use for your Dynamic Application. After implementing each feature, you might want to re-test your Dynamic Application. To implement additional features:
    - To automatically align your Dynamic Application to devices during discovery, create one or more *discovery objects*.
    - To generate alerts when the collected data meets specified thresholds, define *thresholds and alerts*.
    - To create component devices using the collected data, enable *Dynamic Component Mapping*.
    - To cache the results of requests for use by other Dynamic Applications, enable *caching*.
    - To use the performance graphs defined in your Dynamic Application to represent CPU, Memory, or Swap utilization for subscriber devices, define a vitals link for one or more *presentation objects*.

- To use the data collected by your Dynamic Application to automatically populate asset records, define an asset link for one or more *collection objects*.
- To use the data collected by your Dynamic Application to automatically populate inventory lists for hardware components and installed software, define an inventory link for one or more *collection objects*.
- To use data collected by your Dynamic Application to automatically create and/or populate custom attributes, align a custom attribute with one or more *collection objects*.

Creating the Container for a Dynamic Application

If you are creating a new Dynamic Application from scratch, you must create a container for your Dynamic Application. To do this:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Select the **[Actions]** button, and then select *Create New Dynamic Application*. The **Dynamic Applications Create New Application** page appears in a new window.
3. Enter a value in each field in the **Dynamic Applications Create New Application** page. For a description of each field, see the *Dynamic Application Settings* section.
4. Select the **[Save]** button to create your new Dynamic Application.

Duplicating an Existing Dynamic Application

If you are creating a new Dynamic Application that is similar to an existing Dynamic Application, you can duplicate the existing Dynamic Application. To do this:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to duplicate. Select its wrench icon (🔧). The **Dynamic Applications Properties Editor** page is displayed.
3. Enter a new name for your Dynamic Application in the ***Application Name*** field. You can also change the values in the other fields on this page. For a description of each field on this page, see *Dynamic Application Settings*.
4. Select the **[Save As]** button to duplicate the Dynamic Application:

When a Dynamic Application is duplicated:

- A new Dynamic Application is created using the properties defined in this page.
- New instances of the collection objects, presentation objects, thresholds, and alerts in the existing Dynamic Application are created and associated with the new Dynamic Application.
- New instances of the event policies associated with the existing Dynamic Application are created and associated with the alerts in the new Dynamic Application.
- For Dynamic Applications that have the ***Component Mapping*** checkbox enabled, a new instance of the associated component device class is created and associated with the new Dynamic Application.
- For Dynamic Applications that have the ***Component Mapping*** checkbox enabled, the Dynamic Applications aligned in the **Dynamic App Component Alignment** page will also be aligned with the new Dynamic Application. Skylar One does not create new instances of the Dynamic Applications aligned in the **Dynamic App Component Alignment** page.
- The new Dynamic Application is not automatically included in a PowerPack.

Creating a Dynamic Application Using the SNMP Walker Tool

The easiest way to create an SNMP Dynamic Application is to use the SNMP Walker Tool to determine what information is available from the SNMP agent on a device. You can then select the SNMP OIDS directly from the SNMP Walker Tool and add them to a new or existing Dynamic Application. For information on how to use the SNMP Walker Tool to create an SNMP Dynamic Application, see the ***SNMP Dynamic Application Development*** chapter.

Collection Objects

This section describes how Skylar One uses Collection Objects.

What is a Collection Object?

Each ***Collection Object*** in a Dynamic Application specifies a data point that Skylar One will attempt to collect. A Dynamic Application must have at least one Collection Object. Each Dynamic Application protocol uses a different method to specify how Skylar One should collect the data point associated with a Collection Object.

How Does a Collection Object Work?

Dynamic Applications that use the Snippet, SOAP, WMI, XSLT, or PowerShell protocols include one or more ***requests*** that define how Skylar One should request data from a device. For Snippet Dynamic Applications, the requests are referred to as ***snippets***. Each request specifies an operation that will gather a response from the device.

Each collection object in a Snippet, SOAP, WMI, XSLT, or PowerShell Dynamic Application is associated with a request. The collection object specifies how to parse a data point from the response. A single request can be used to populate multiple collection objects.

For example, the default "Windows Interface" WMI Dynamic Application includes one request and will return one response. The request includes a WMI query that retrieves six properties. The Dynamic Application includes six collection objects, one collection object for each property included in the response.

Collection objects in Dynamic Applications that use the Database, SNMP, XML, or PowerShell protocols do not use separate requests. For Database, SNMP, XML, or PowerShell Dynamic Applications, the collection object itself defines the method of collection:

- A ***Database*** collection object specifies a database query that Skylar One must execute to populate the collection object.
- An ***SNMP*** collection object specifies the SNMP OID that Skylar One must retrieve to populate the collection object.
- An ***XML*** collection object specifies how to parse a value from an XML document to populate the collection object. The XML document to parse is specified in the SOAP/XML credential aligned with the Dynamic Application.
- A ***PowerShell*** collection object specifies the PowerShell command that Skylar One must execute to populate the collection object.

For information on how to define requests and define the collection object fields specific to each Dynamic Application protocol, see the supplementary manual for each Dynamic Application protocol type.

How Collection Objects are Used by Skylar One

The data collected for each Collection Object is displayed differently for each Dynamic Application archetype:

- For the ***Configuration*** archetype, each collection object represents a column in a table of collected data. The value(s) collected by Skylar One for the collection object are displayed in that column. For example, the default "Host Resource: Software" Dynamic Application includes two collection objects called "Install Date" and "Software Title".

  NOTE: A Dynamic Application can define multiple tables.

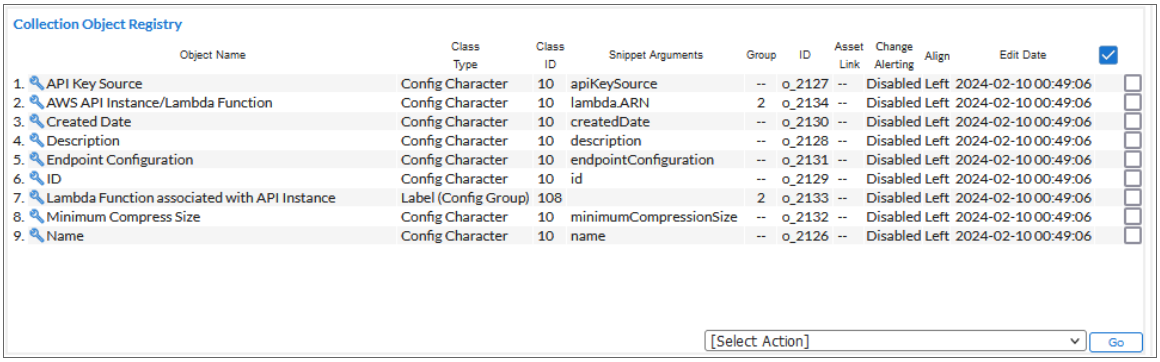
- For the ***Performance*** archetype, collection objects are used in ***Presentation Objects***. Each presentation object specifies a formula that must include one or more collection objects. The formula in a presentation object is used to generate a time series graph of data. For more information on presentation objects for Dynamic Applications with an archetype of Performance, see the *Presentation Objects*.
- For the ***Journal*** archetype, collection objects are attributes of a ***Journal Entry***. The collection objects are used in ***Presentation Objects*** to define how the collection objects will be formatted in the log of journal entries. For information on collection objects in Dynamic Applications with an archetype of Journal, see the ***Snippet Dynamic Application Development*** chapter.

Collection Objects are also used to define alerts in Dynamic Applications. Each alert specifies a formula that must include one or more collection objects. Skylar One examines alert formulas to evaluate whether an alert should be generated based on the collected data. For more information on alerts for Dynamic Applications, see *Alerts and Thresholds*.

Viewing the Collection Objects in a Dynamic Application

To view the collection objects in a Dynamic Application:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to view the collection objects for. Select its wrench icon (🔧). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page is displayed. The **Collection Object Registry** pane at the bottom of the page displays the following information about each collection object:


Collection Object Registry

	Object Name	Class Type	Class ID	Snippet Arguments	Group	ID	Asset Link	Change Alerting	Align	Edit Date
1.	API Key Source	Config Character	10	apiKeySource	--	o_2127	--	Disabled	Left	2024-02-10 00:49:06
2.	AWS API Instance/Lambda Function	Config Character	10	lambda.ARN	2	o_2134	--	Disabled	Left	2024-02-10 00:49:06
3.	Created Date	Config Character	10	createdDate	--	o_2130	--	Disabled	Left	2024-02-10 00:49:06
4.	Description	Config Character	10	description	--	o_2128	--	Disabled	Left	2024-02-10 00:49:06
5.	Endpoint Configuration	Config Character	10	endpointConfiguration	--	o_2131	--	Disabled	Left	2024-02-10 00:49:06
6.	ID	Config Character	10	id	--	o_2129	--	Disabled	Left	2024-02-10 00:49:06
7.	Lambda Function associated with API Instance	Label (Config Group)	108		2	o_2133	--	Disabled	Left	2024-02-10 00:49:06
8.	Minimum Compress Size	Config Character	10	minimumCompressionSize	--	o_2132	--	Disabled	Left	2024-02-10 00:49:06
9.	Name	Config Character	10	name	--	o_2126	--	Disabled	Left	2024-02-10 00:49:06

[Select Action] Go

- ***Object Name***. The name of the collection object.
- ***Class Type*** and ***Class ID***. The data type of the collection object. See the *Creating a Collection Object* section for a description of the possible class types.
- ***Protocol Specific Collection Method***. The label displayed for this column is different for each Dynamic Application protocol. The values displayed in this column specify how Skylar One will request the collection object or parse the collection object from a separately defined request.
- ***Group***. The group number for the collection object.
- ***ID***. The unique ID assigned to the collection object by Skylar One. You can use this ID to reference the collection object in presentation formulas and alert formulas. The unique ID will always start with "o_" (lowercase "oh", underscore).
- ***Edit Date***. The last time a user edited the collection object.

For collection objects in configuration Dynamic Applications, the following additional information is displayed:

- ***Asset Link***. The asset record field that is automatically populated with values collected for the collection object.
- ***Change Alerting***. Specifies whether change alerting is enabled for the collection object. Possible values are:
    - *Enabled*. Specifies that if a collected value for the collection object changes, Skylar One will trigger an event.

- *Disabled*. Specifies that if a collected value for the collection object changes, Skylar One will not trigger an event.

- ***Align***. Specifies how the collected values for the collection object will be justified in the table of configuration data. Possible values are:
  - *Left*
  - *Center*
  - *Right*

To edit a collection object, select its wrench icon ( ).

For SNMP Dynamic Applications, an information icon ( ) is displayed for each collection object. Select this icon to view information about the SNMP OID defined for a collection object. The information displayed is defined in the SNMP MIB for that SNMP OID.

Creating a Collection Object

> **NOTE:** This section does not apply to Dynamic Applications with an archetype of Journal. For information on collection objects in Dynamic Applications with an archetype of Journal, see the ***Snippet Dynamic Application Development*** chapter.

To add a Collection Object to a Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to add a collection object to. Select its wrench icon ( ). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page is displayed. The fields that appear on this page are different for each Dynamic Application type.
4. Enter values in the fields on this page. This chapter describes the fields on this page in separate sections. Depending on the type, purpose, and settings of your Dynamic Application, you will need to define values listed in different sections. The sections are:
   - *Basic Settings*. This section describes the fields that specify the name, description, and data type of the collection object. You must supply values in these fields for all collection objects.
   - *Collection Settings*. This section describes the fields that specify how Skylar One should collect values for the collection object. These fields are different for each Dynamic Application protocol. You must supply values in these fields for all collection objects.
   - *Grouping & Indexing Settings*. This section describes the fields that specify how Skylar One should group collection objects. These fields are optional; however, collection objects in most Dynamic Applications have values specified in these fields. The ***Usage Type*** field also defines how the data collected by a Dynamic Application is used to build device relationships.

- *Configuration Dynamic Application Settings*. This section describes the fields that specify how Skylar One should use and display the collected data for collection objects in configuration Dynamic Applications. These fields appear only for configuration Dynamic Applications. These fields are optional.
- *Standard Deviation Settings*. This section describes the fields that specify how Skylar One should calculate standard deviation data for a collection object. These fields appear only for performance Dynamic Applications. These fields are optional. Supply values in these fields only if you are using this collection object in the standard deviation function in an alert formula.
- *Dynamic Component Mapping Settings*. This section describes the fields that specify how Skylar One should create component devices using the collection object. These fields appear only for Dynamic Applications that have the ***Component Mapping*** checkbox checked in the **Dynamic Applications Properties Editor** page. These fields are optional. Supply values in these fields only if you are using this collection object to create a component device.

5. Select the **[Save]** button to save the collection object.

Basic Settings

To define the basic settings for a collection object, supply a value in the following fields:

- ***Object Name***. The name of the collection object. This name is used to label the collection object in the following places in Skylar One:
    - In the **Collection Object Registry** pane at the bottom of this page.
    - In the list of collection objects in the **Dynamic Applications Presentation Objects** page.
    - In the list of collection objects in the **Dynamic Applications Alert Objects** page.
    - For configuration Dynamic Applications, in the **Dynamic Application Collections** page (the **[Collections]** tab in the **Device Administration** panel). This page displays the collection status of each collection object.
    - For configuration Dynamic Applications, in the **Configuration Report** page (the **[Configs]** tab in the **Device Reports** panel). The ***Object Name*** appears as the column heading for the collection object.
- ***Class Type***. The type of data that will be collected for this collection object. The data type defines how Skylar One will process and display the collected values. For configuration Dynamic Applications, the choices are:
    - *[10] Config Character*. Collection object will be populated with strings that Skylar One should display with no text formatting applied.
    - *[11] Config Enum*. Collection object will be populated with numeric values that map to a numeric label. You must define the label value for each of the possible numeric values in the ***Enum Values*** field.
    - *[12] Config Timeticks*. Collection object will be populated with numeric values that represent timeticks.

- *[14] Config Gauge*. Collection object will be populated with numeric values that can increase and decrease with each polling session.
- *[15] Config Numeric*. Collection object will be populated with numeric values.
- *[16] Config Time Stamp*. Collection object will be populated with timestamps.
- *[17] Config MAC Address*. Collection object will be populated with MAC addresses.
- *[18] Config IP Address*. The values collected by this collection object are IP addresses. If the collected value is a hexadecimal string, Skylar One will convert the string to the standard dotted decimal format and store that dotted decimal value for the collection object.
- *[19] Config Date & Time*. Collection object will be populated with date and time values in the format "Year-Month-Day,Hour:Minute:Second" or in Hex notation. Skylar One will convert the collected value to a human-readable date & time string before storing the value.
- *[100] Discovery*. For information about this class type, see the *Creating a Discovery Object* section.
- *[101] Label (Hourly Polled)*. Collection object will be populated with strings that will be collected only once an hour and for which only the newest collected value will be stored.
- *[104] Label (Always Polled)*. Collection object will be populated with strings for which only the newest collected value will be stored.
- *[105] Index*. Collection object will be populated with the indexes associated with the list of values, not the values themselves.
- *[108] Label (Config Group)*. Collection objects of this type are not collected. The ***Object Name*** defined for this collection object will be used to label the table of data for the group selected in the ***Group*** field for this collection object.

For performance Dynamic Applications, the choices are:

- *[1] Performance Counter*. Collection object will be populated with counter values. When Skylar One evaluates a presentation formula or alert formula that contains a collection object of this type, the value for the collection object is calculated by subtracting the previous collected value for the collection object from the current collected value.
- *[2] Performance Percent*. Collection object will be populated with percentage values (i.e. values from 0 - 100).
- *[3] Performance Kilobyte*. Collection object will be populated with values that represent size in kilobytes.
- *[4] Performance Gauge*. Collection object will be populated with numeric values that can increase or decrease with each polling session. The values collected are the actual values that should be used when Skylar One evaluates presentation formulas and alert formulas.
- *[100] Discovery*. For information about this class type, see the *Creating a Discovery Object* section.

- *[101] Label (Hourly Polled)*. Collection object will be populated with strings that will be collected only once an hour. Collection objects of this type are used to label the lines on a performance graph.
- *[104] Label (Always Polled)*. Collection object will be populated with strings. Collection objects of this type are used to label the lines on a performance graph.
- *[105] Index*. Collection object will be populated with the indexes associated with the list of values, not the values themselves.
- *[110] Label IP (Hourly Polled)*. The object value for the label (IP address) will be retrieved only hourly, not at the ***Poll Frequency*** defined for the Dynamic Application. In the ***String Type*** field, specify the format of the collected string, to allow Skylar One to convert the string to dotted decimal notation.
- *[111] Label IP (Always Polled)*. The object value for the label (IP address) will be at the ***Poll Frequency*** defined for the Dynamic Application. In the ***String Type*** field, specify the format of the collected string, to allow Skylar One to convert the string to dotted decimal notation.

- ***String Type***. Specifies whether a collected string is in hex notation or standard text. This field helps Skylar One convert collected strings to the appropriate format. Choices are:
  - *Standard*. Collected string is a standard text string.
  - *Hex*. Collected string is in hex notation.
- ***Description***. A description of the collection object. This field is optional.
- ***Formula***. If you require Skylar One to examine and manipulate the object's value before storing the value in Skylar One, you can enter a formula in this field. Instead of storing the collected values for this collection object, Skylar One stores the result of this formula. The formula field can include any arithmetic Python statement and any Python time statement that returns a single value. That returned, single value will be stored instead of the collected value. You can use the following variables in the formula field:
  - **%V**. Collected value for the object.
  - **%P**. Collected value for the object from the last successful poll.
  - **%L**. Date and time of the last successful poll of the object, in DateTime format, including seconds.
  - **%C**. Date and time of the current poll of the object, in DateTime format, including seconds.
  - **%E**. Number of seconds, in integer format, that have elapsed since last poll of object.
- ***Disable Object Maintenance***. When you select this checkbox, Skylar One will never automatically disable collection of this object on an aligned device. For details on how Skylar One manages collection status, see *Managing Dynamic Applications*.

Collection Settings

The fields that define how Skylar One should collect data for a collection object are different for each Dynamic Application protocol.

In general, one field specifies how Skylar One should request the collection object or parse the collection object from a separately defined request. For example, for the SNMP protocol, this field is called ***SNMP OID*** and specifies the SNMP OID that Skylar One must request to obtain a value for this collection object. This field has one of the following labels:

- *SNMP OID*
- *SQL Query*
- *SOAP Tags*
- *Snippet Arguments*
- *WMI Request Arguments*
- *PowerShell Arguments*
- *XML Tags*
- *XSLT Tags*

For SOAP, Snippet, WMI, and XSLT Dynamic Applications, a second field specifies the request or snippet that will return a value for the collection object. This field has one of the following labels:

- *SOAP Request*
- *Snippet*
- *WMI Request*
- *PowerShell Request*
- *XSLT Request*

For more information about these fields, see the supplementary chapter for the protocol you are working with.

Grouping & Indexing Settings

The collection objects in a Dynamic Application can be divided into *groups*. When collection objects are in the same group:

- For configuration Dynamic Applications, the collection objects will appear in the same table of configuration data.
- The collected values for the collection objects will share the same ***indexes***. When a list of values is collected for a collection object, indexes are used to maintain an association between values collected at different times for the same collection object and maintain associations between collection objects in the same Dynamic Application. For more information, see the *Indexing* section. Typically, you should group collection objects together if:
    - The collection objects will be used in the same presentation object formula.
    - The collection objects will be used in the same alert formula.

The following fields define the grouping and indexing settings:

- ***Group Number***. The group for this collection object. The options are *No Group* or a group number from 1 to 50.
- ***Usage Type***. If *Group Index* is selected in this field, Skylar One will use the collected values for this collection object to assign indexes for this ***Group Number***. To use the default indexing method for the collection objects in this group, select *Standard* in this field. See the *Indexing* section for information on how to use this field. The ***Usage Type*** field also defines how the data collected by a Dynamic Application is used to build device relationships. For more information about how to use the ***Usage Type*** field to define relationships, see the *Dynamic Application Relationships* section.

Configuration Dynamic Application Settings

For collection objects in configuration Dynamic Applications, the following additional fields are displayed:

- ***Custom Attribute***. This field appears only for collection objects in configuration Dynamic Applications. For details, see the section on *Custom Attribute Settings*.
- ***Asset/Form Link***. Skylar One can automatically populate the asset record for a device using data collected by configuration Dynamic Applications. In these two drop-down lists, you can select a standard asset field that appears in the **Asset Properties** page or the other pages in the **Asset Administration** panel. The selected asset field will be automatically populated with the value collected for this collection object. To populate a standard asset field, select the name of the field from the first drop-down list. Choices are:
    - *Make*
    - *Model*
    - *Serial Number*
    - *Operating System*
    - *Host ID / SID*
    - *OS System Name*
    - *CPU Count*
    - *CPU Type/Make*
    - *CPU Speed*
    - *Firmware / BIOS Name*
    - *Installed Memory*
    - *Asset Tag*
    - *Disk Array Size*
    - *Disk Count*
    - *Disk Size*

To change the priority value of an asset field, select the priority value in the second drop-down list. Increasing the value from 50 will cause the asset to have a higher precedence.

- ***Form Link***. To populate a custom asset field, select the name of a field in the second drop-down list. The choices are all fields included in the Application Form for "Embedded Asset Form Fields". To view or configure this form, go to System > Customize > Form Fields.
- ***Inventory Link***. Skylar One can automatically populate inventory lists of hardware components and installed software using data collected by configuration Dynamic Applications. In the first drop-down list, you can select the type of inventory record to populate. In the second drop-down list, select the field in that record to populate. If you select *Disabled* in the first drop-down list, the second drop-down list is not displayed. Choices are:
    - *CPU*. The collection object will populate a hardware inventory record for a CPU. If you select this option, the following field choices appear in the second drop-down list:
        - *CPU Index ID*
        - *CPU Description*
    - *Disk*. The collection object will populate a hardware inventory record for a physical disk. If you select this option, the following field choices appear in the second drop-down list:
        - *Disk Index ID*
        - *Disk Name*
        - *Disk Capacity Kb*
        - *Disk Read/Write Mode*
        - *Media Description*
    - *Memory*. The collection object will populate a hardware inventory record for installed memory. If you select this option, the following field choices appear in the second drop-down list:
        - *Memory Index ID*
        - *Memory Name*
        - *Total Physical Memory Kb*
        - *Physical Memory in Use Kb*
        - *Physical Memory Percent in Use*
    - *Virtual Memory*. The collection object will populate a hardware inventory record for virtual memory. If you select this option, the following field choices appear in the second drop-down list:
        - *Swap Index ID*
        - *Swap Description*
        - *Total Swap Memory Kb*

        - *Swap Memory in Use Kb*
        - *Swap Memory Percent in Use*
    - *Hardware Components*. The collection object will populate a hardware inventory record for any other type of hardware component. If you select this option, the following field choices appear in the second drop-down list:
        - *Component Index ID*
        - *Component Description*
        - *Component Type*
    - *Software Titles*. The collection object will populate a software inventory record. If you select this option, the following field choices appear in the second drop-down list:
        - *Install Date*
        - *Software Title*
- ***Change Alerting***. This field enables or disables change detection. If the collected value of the collection object changes or is removed, Skylar One can automatically trigger an event. A change alert is created when a collected value is different from the previously collected value. When a new value is added for the first time, a change alert is not created. You can view the change history for the collection object in the **Configuration Report** page for each subscriber device. The choices for this field are:
    - *Enabled*. Specifies that if the value of this collection object changes, Skylar One will trigger an event.
    - *Disabled*. Specifies that if the value of this collection object changes, Skylar One will not trigger an event.
- ***Table Alignment***. These two drop-down lists define how the values collected for this collection object will be displayed in the table of configuration data in the **Configuration Report** page. The first drop-down list defines how the value will be justified in the table cell. The choices in the first drop-down list are:
    - *Left*
    - *Center*
    - *Right*
- ***Hide Object***. If you select this checkbox, the object will not appear in the **Configuration Report** page for each subscriber device.
- ***Enum Values***. Applies only to collection objects with a class type *[11] Config Enum*. In this field you can map each possible collected value to a descriptive string that will be displayed in Skylar One. Use the following format:

```
#number:value#number:value#number:value
```

Where **number** is the collected value and **value** is the descriptive string. You must include a "#" as the first character in this field and as a separator between number:value pairs.

For example, suppose a *[11] Config Enum* object returns an integer that represents a status condition. The possible returned values and the conditions they represent might be:

3 = healthy4 = minor5 = major6 = critical

You would supply the following string in the ***Enum Values*** field:

```
#3:healthy#4:minor# 5:major#6:critical
```

Custom Attribute Settings

In Dynamic Applications of archetype *configuration*, you can:

- Use a collection object to populate the value of an existing custom attribute.
- Use a pair of collection objects to create a custom attribute and provide a value for that custom attribute. You must define a collection object to define the name of the custom attribute; this causes the Skylar One system to create a custom attribute with the name from the collection object. You must also define a second collection object to populate the value of the custom attribute.

NOTE: For details on creating and managing custom attributes, see the manual ***Using the ScienceLogic API***.

NOTE: For component devices, only assigned Collector Affinity is supported when using a Dynamic Application to populate custom attributes. If a custom attribute value is manually edited or removed for a device that is using a different collector than its root device, the Dynamic Application will not be able to update the value on subsequent polls unless the Dynamic Application is using assigned Collector Affinity. For more information, see the section on *Collector Affinity*.

The following fields in the **Collection Objects** page allow you to use one or more collection objects to define and/or populate a custom attribute:

- ***Align to Custom Attribute***. Specify the custom attribute to associate with this collection object. The custom attribute will be populated with a value from a collection object. Choices are:
    - *None*. This collection object is not associated with a custom attribute.
    - *Static*. This collection object is associated with a specific custom attribute.
    - ***Static Name***. If you selected *Static* in the ***Custom Attribute*** field, the *Static Name* field appears. In this field, specify the name of the custom attribute that you want to populate with the value of the collection object. You can select from a list of existing custom attributes.
        - If the list does not include the custom attribute you want to align with the collection, select the plus-sign icon (+). The icon clears the field and allows you to manually enter a value.

- If you manually specify a custom attribute, Skylar One will search for a custom attribute with a matching name and populate the custom attribute with the value of this collection object. If Skylar One does not find a custom attribute with a matching name and therefore creates the custom attribute, the new custom attribute will be an extended custom attribute, for devices. The data type will be integer (for numeric values) or string (for all other value types).

◦ *Dynamic Name*. You can use a pair of collection objects to populate the name and value of a custom attribute. You must define each collection object separately. When you select *Dynamic Name* in the ***Custom Attribute*** field, the name of the custom attribute is populated with the value of the collection object. If Skylar One does not find a custom attribute with a matching name, Skylar One will create the custom attribute. If Skylar One does not find a custom attribute with a matching name and therefore creates the custom attribute, the new custom attribute will be an extended custom attribute, for devices. The data type will be integer (for numeric values) or string (for all other value types).

NOTE: If you select *Dynamic Name* in the ***Custom Attribute*** field, you must create a second collection object that will populate the value of the custom attribute.

NOTE: Names for custom attributes must conform to XML naming standards. The attribute name can contain any combination of alphanumeric characters, a period, a dash, a combining character or an extending character. If a collected value for an attribute name does not conform to XML standards, Skylar One will replace non-valid characters with an underscore + the hexadecimal value of the illegal character + an underscore. So "serial number" would be replaced with "serial_X20_number". The attribute label will use the original, non-converted value ("serial number").

◦ ***Dynamic Value***. The value of the custom attribute selected in the *Dynamic Name* field is populated with the value of the collection object.

◦ ***Dynamic Name***. If you selected *Dynamic Value* in the ***Custom Attribute*** field, the *Dynamic Name* field appears. Select from the list of collection objects that have a ***Custom Attribute*** value of *Dynamic Name*.

NOTE: The collection object assigned to the *Dynamic Value* is added to the same ***Group*** as the collection object assigned to the associated *Dynamic Name*. If the collection object for *Dynamic Name* is not assigned to a ***Group***, you will be prompted to select a ***Group*** for the both the collection object for *Dynamic Name* and the collection object for *Dynamic Value*.

NOTE: Each group can contain only one collection object that is assigned to a *Dynamic Value* and only one collection object that is assigned to a *Dynamic Name*. The group can contain other collection objects, but should not contain more than one collection object assigned to a *Dynamic Value* and not more than one collection object assigned to a *Dynamic Name*.

Standard Deviation Settings

To use a collection object with the standard deviation function (deviation) in an alert formula, you must enable standard deviation alerting for the collection object. To enable standard deviation alerting for a collection object, supply a value in the following fields:

> **NOTE:** These fields appear only for collection objects in performance Dynamic Applications that collect numeric values.

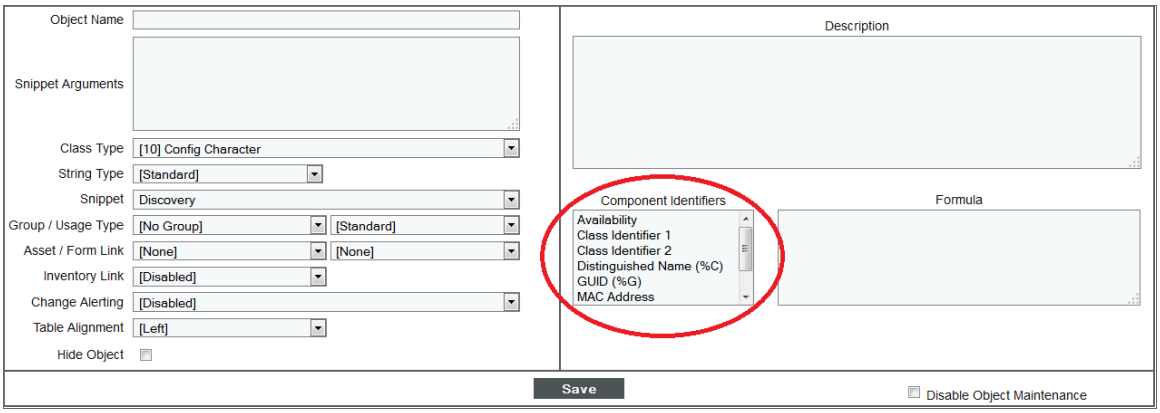
- ***Enable Deviation Alerting***. If selected, Skylar One will calculate and store standard deviation data for the collection object. To use a collection object with the standard deviation function in an alert formula, this checkbox must be checked.
- ***Max Weeks Data***. If you selected ***Enable Deviation Alerting***, you must specify a value in this field. This field specifies the maximum number of weeks of data that Skylar One should use when evaluating a standard deviation function that uses this collection object. The default value is "4". This means that when Skylar One evaluates an alert formula that contains a standard deviation function that uses this collection object, Skylar One will use the last four weeks of data for the collection object to calculate the standard deviation. Any data older than the last 4 weeks is not included in calculations for standard deviation.
- ***Min Weeks Data***. If you selected ***Enable Deviation Alerting***, you must specify a value in this field. This field specifies the minimum number of weeks of data required for Skylar One to evaluate a standard deviation function that uses this collection object. The default value is "2". This means that Skylar One requires at least the data from the last two weeks from today to calculate standard deviation for this collection object.

> **NOTE:** If you enable deviation alerting for an existing collection object and if Skylar One has already collected data for that collection object, Skylar One will use the already-collected data to "backfill" and calculate mean and standard deviation using the historical data. For example, suppose a Dynamic Application has already been collecting data for 12 weeks. Suppose during the twelfth week of collection, you enable deviation alerting for a collection object for the first time. Suppose you set ***Max Weeks Data*** to "6" and ***Min Weeks Data*** to "4". The first time Skylar One performs calculations for standard deviation for this collection object (once every hour), Skylar One would then "backfill" and include the last six weeks of historical data when calculating standard deviation for the collection object. If Skylar One has already collected at least the Min Weeks Data, Skylar One will use the historical data to calculate standard deviation.

NOTE: The backfill functionality works only during the very first time Skylar One performs calculations for standard deviation for a collection object. If you change the settings for ***Min Weeks Data*** and ***Max Weeks Data*** so that the new value of ***Min Weeks Data*** exceeds the previous value for ***Max Weeks Data***, Skylar One will require you to collect additional data to meet the new requirement. For example, suppose you initially set ***Min Weeks Data*** to "2" (two weeks) and ***Max Weeks Data*** to "4" (four weeks). Suppose that 9 weeks later, you edit ***Min Weeks Data*** to "5" (five weeks) and ***Max Weeks Data*** to "7". Because of the previous ***Max Weeks Data*** of "4", Skylar One will have retained only four weeks of calculated standard deviation data. After editing the values, you will have to wait one additional week before Skylar One will have enough data (five weeks, as specified by ***Min Weeks Data***) to generate standard deviation alerts. Skylar One will not use historical data to backfill the additional week.

Dynamic Component Mapping Settings

For Dynamic Applications that have the ***Component Mapping*** checkbox selected in the **Dynamic Applications Properties Editor** page, the ***Component Identifiers*** field is displayed in the **Collection Objects** page:



In this field, select one or more identifiers that Skylar One can use to create component devices. The values collected for this collection object will populate the selected identifier for the component devices. Collection objects can define the following component identifiers:

- *Availability*. The availability of the component devices. The values collected using this collection object will be used to generate availability reports and graphs for each component device.
- *Class Identifier 1*. The class type of the component devices, typically the vendor.
- *Class Identifier 2*. The class sub-type of the component devices, typically the model.
- *Device Name (%N)*. The name of the component devices in Skylar One.
- *Distinguished Name (%C)*. The name that the system that monitors the component devices (the root device) uses to identify the component devices.

- *GUID (%G)*. The globally unique identifier Skylar One will use for the component devices. This value must be unique for all component devices in Skylar One. This value is used to determine when a component device should be associated with a different root device.
- *MAC Address*. The MAC Address of the component device.
- *Organization*. During discovery, if the value of the collection object exactly matches an organization name in Skylar One, Skylar One will align the component device with that organization. During subsequent collections, if the value of this collection object changes and exactly matches an organization name in Skylar One, Skylar One aligns the new organization with the component device. If the value of this collection object does not exactly match any organizations in Skylar One, Skylar One will create an event and align the event with the device aligned with the Dynamic Application.
- *Unique Identifier (%U)*. The unique identifier Skylar One will use for the component devices. This value must be unique for all component devices associated with the same root device.
- *UUID*. The universally unique identifier for the component devices.
- *GUID*. Globally unique identifier. This value will be unique within Skylar One. No two component devices can have the same GUID. This component identifier protects the integrity of a component device even if it is aligned with a different root device (for example, with a vmotion event).

NOTE: Each of these options can be defined by only one collection object in a Dynamic Application. If a component identifier is already defined by a collection object in this Dynamic Application, that component identifier will not be displayed in the list.

If the Dynamic Application is aligned with a component device class, Skylar One will create a component device when the collection objects that map to the *Device Name* and *Unique Identifier* of a component device are successfully collected from a device.

For more information about configuring a Dynamic Application to use Dynamic Component Mapping, see the *Dynamic Component Mapping* section.

Creating a Discovery Object

A ***discovery object*** is a type of collection object that allows Skylar One to automatically align a Dynamic Application to a device during discovery.

Skylar One attempts to automatically align Dynamic Applications to devices under the following circumstances:

- When a discovery session that has an ***Initial Scan Level*** of *1. Initial Population of Apps* or higher is executed, either manually or on a schedule, Skylar One will attempt to automatically align Dynamic Applications using the credentials specified in the discovery session.
- For devices that have the ***Auto-Update*** and ***Dynamic Discovery*** checkboxes enabled on the **Device Properties** page, Skylar One will attempt to automatically align Dynamic Applications once a day using the credentials aligned with the device.
- When a user selects the lightning bolt icon ( ) for a Dynamic Application in the **Dynamic Applications Manager** page, Skylar One will attempt to automatically align that single Dynamic Application with all devices using the credentials aligned with each device.

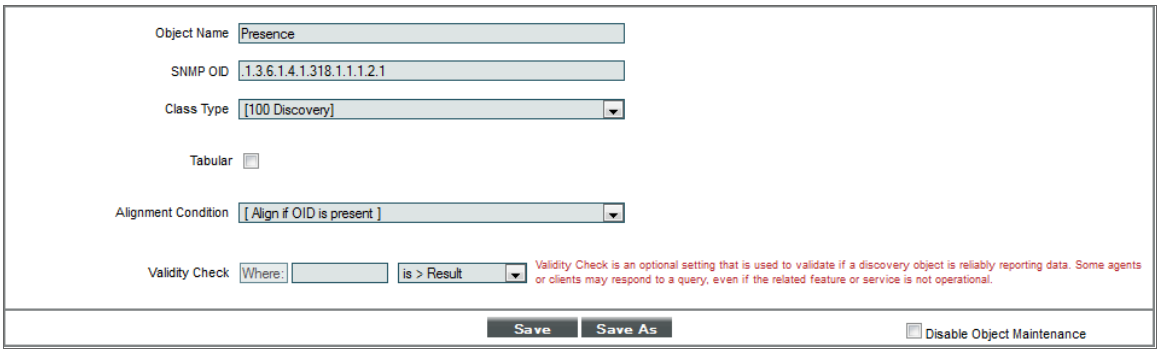
Every discovery object specifies a condition that can evaluate to *true* or *false*. When Skylar One attempts to automatically align a Dynamic Application to a device, every available credential is used to try and collect every discovery object in the Dynamic Application. If ***all*** the discovery objects in a Dynamic Application evaluate to *true* for a device, Skylar One will align the Dynamic Application with the device.

A discovery object can specify one of the following conditions:

- If the device returns a value for the discovery object, evaluate to *true*. Otherwise, evaluate to *false*.
- If the device returns a value for the discovery object and that value matches specific criteria, evaluate to *true*. Otherwise, evaluate to *false*.
- If the device ***does not*** return a value for the discovery object, evaluate to *true*. Otherwise, evaluate to *false*.

To add a discovery object to a Dynamic Application:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to add a collection object to. Select its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page is displayed. Enter values in the following fields:
    - ***Object Name***. The name of the discovery object. If you are defining only one discovery object for this Dynamic Application, best practice is to enter "Discovery" in this field.
    - ***Protocol Specific Collection Method***. The label displayed for these fields is different for each Dynamic Application protocol. Specify how Skylar One should collect the discovery object.
    - ***Class Type***. Select *[100] Discovery* in this from the drop-down list.
4. Select the **[Save]** button to save the collection object. Because this collection object has been defined with a ***Class Type*** of *[100] Discovery*, additional fields that are specific to discovery objects are displayed:



5. Enter values in the following fields:
    - ***Tabular***. If the value specified in the ***Protocol Specific Collection Method*** fields will return tabular data (a list of values) and you want to match the returned values to specific criteria, select this checkbox. You must then specify a value in the ***Validity Check*** field. Skylar One will iterate through the returned tabular data and search for the value specified in the ***Validity Check*** field.

- ***Alignment Condition***. Specifies how this discovery object should be evaluated. Choices are:
    - *Align if OID is present*. Specifies that if the device returns a value for the discovery object, evaluate to *true*. Optionally, in the ***Validity Check*** field you can specify the criteria that the returned value must meet.
    - *Align if OID is NOT present*. Specifies that if the device does not return a value for the discovery object, evaluate to *true*.
- ***Validity Check***. If you selected *Align if OID is present* in the ***Alignment Condition*** field, this field is displayed. In this field, you can specify criteria that the returned value must meet for the discovery object to evaluate to *true*. To use this field, enter a value or regular expression in the text area, then select from the drop-down list of operators to specify how that value should be compared to the returned value. The choices for the comparison operator are:
    - *is > Result*. Specifies that the discovery object evaluates to *true* if the specified value is greater than the returned value.
    - *is == Result*. Specifies that the discovery object evaluates to *true* if the specified value is equal to the returned value.
    - *is < Result*. Specifies that the discovery object evaluates to *true* if the specified value is less than the returned value.
    - *is != Result*. Specifies that the discovery object evaluates to *true* if the specified value is not equal to the returned value.
    - *is LIKE Result*. Specifies that the discovery object evaluates to *true* if the returned value contains the specified value.
    - *matches REGEX*. Specifies that the discovery object evaluates to *true* if the returned value matches the specified regular expression.

6. Click **[Save]** to save the discovery object.

Editing a Collection Object

To edit a Collection Object in a Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to edit. Select its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page is displayed.
4. In the **Collection Object Registry** pane at the bottom of the page, locate the collection object you want to edit. Select the wrench icon () for the collection object you want to edit.
5. The fields in the top pane will be populated with the saved values for the selected collection object. Edit the values in one or more fields. For a description of each field, see the *Creating a Collection Object* section.
6. Select the **[Save]** button to save your changes to the collection object. Select the **[Save As]** button to save the changes as a new collection object.

Performing Bulk Actions on Collection Objects

You can perform the following bulk actions on the collection objects in a Dynamic Application:

- Delete collection objects from Skylar One.
- Change the ***group*** value.
- Enable or disable ***Change Alerting*** for collection objects in configuration Dynamic Applications.

To perform a bulk action on the collection objects in a Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to edit. Select its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page is displayed.
4. In the **Collection Object Registry** pane at the bottom of the page, locate the collection object(s) you want to perform the action on. Select the checkbox for each collection object you want to perform the action on.
5. In the ***Select Action*** drop-down list, select the action you want to perform, and then select the **[Go]** button. The selected collection objects will have the selected action applied to them.

Caching

This section describes ***caching***, which is when Skylar One requests information from a device during Dynamic Application collection, and Skylar One saves or ***caches*** the response from the device. If a response is cached, other Dynamic Applications that use the same protocol can use the cached response to populate collection objects.

What is Caching?

When Skylar One requests information from a device during Dynamic Application collection, Skylar One can ***cache*** the response from the device. If a response is cached, other Dynamic Applications that use the same protocol can use the cached response to populate collection objects. Caching responses reduces the number of requests performed by Skylar One and speeds up collection. Typically, Dynamic Applications aligned to component devices must use cached responses from Dynamic Applications aligned with the root device (the device that "manages" the component device).

Caching is configured in the ***Caching*** drop-down list in the general properties for a Dynamic Application. The following options are available:

- *No caching*. Specifies that this Dynamic Application does not use the caching feature. The requests that populate the collection objects in this Dynamic Application are defined within this Dynamic Application. The responses to those requests are not available to other Dynamic Applications.
- *Cache Results*. Skylar One will cache all responses retrieved by the Dynamic Application. See the *Caching Responses* section for more information.

- *Consume Cached Results*. Skylar One will use the cached responses from other Dynamic Applications to populate the collection objects defined in this Dynamic Application. See the *Consuming Cached Responses* section for more information.

NOTE: All types of Dynamic Applications except Bulk Snippet, Database, and ICDA can both cache results and consume cached results.

Caching Responses

Dynamic Applications that use the following protocols can cache responses:

- SNMP
- Snippet
- SOAP
- WMI
- PowerShell
- XML
- XSLT

NOTE: Bulk Snippet Dynamic Applications cannot cache responses.

For XSLT Dynamic Applications, the untransformed response is cached, not the transformed response.

For Dynamic Applications that cache responses, the collection objects defined in this Dynamic Application are no longer collected at a regular frequency. You must not define collection objects that are used to generate configuration tables or performance graphs in a Dynamic Application that caches responses. Typically, Dynamic Applications that cache results will contain only a discovery object.

For information about caching responses in Snippet Dynamic Applications, see the ***Snippet Dynamic Application Development*** chapter.

Consuming Cached Responses

Dynamic Applications that use the following protocols can consume cached responses:

- SNMP
- Snippet
- SOAP
- WMI
- PowerShell
- XML
- XSLT

NOTE: Bulk Snippet Dynamic Applications cannot consume cached responses.

Snippet Dynamic Applications can consume cached results from any Dynamic Application that caches responses. For more information about using cached responses in snippet code, see the ***Snippet Dynamic Application Development*** chapter. All other Dynamic Applications that consume cached responses can do so only from Dynamic Applications that use the same protocol. For example, SOAP Dynamic Applications can consume cached responses only from other SOAP Dynamic Applications. A SOAP Dynamic Application cannot consume a cached response from an XSLT Dynamic Application.

For Dynamic Applications that consume cached responses:

- All collection objects in the Dynamic Application must be populated using cached responses from other Dynamic Applications. If a Dynamic Application consumes cached responses, you cannot define additional requests in that Dynamic Application.
- If the Dynamic Application is aligned to a non-component device, all the other Dynamic Applications that cache requests used by the Dynamic Application must be aligned to the same device.
- If the Dynamic Application is aligned to a component device, all the other Dynamic Applications that cache requests used by the Dynamic Application must be aligned to the same device or the root device.
- If a cached response is unavailable when Skylar One performs collection for a Dynamic Application that consumes cached responses, Skylar One will automatically attempt to execute the appropriate request to re-populate the cache.
- For XSLT Dynamic Applications that use the *Consume Cached Results* option, XSLT Parser Code must still be defined in the Dynamic Application. The XSLT Parser Code will be used to transform the cached responses.

Configuring Collection Objects in Cache-Consuming Dynamic Applications

The following sections describe how to configure collection objects to use responses that are cached by other Dynamic Applications in SOAP, WMI, XML, and XSLT Dynamic Applications.

SOAP Dynamic Applications

When a SOAP Dynamic Application is configured to *Consume Cached Results*:

- The **[Requests]** tab will not be displayed.
- In the **Dynamic Applications | Collections Objects** page, the ***SOAP Request*** field will display a list of all requests defined in SOAP Dynamic Applications that cache responses. Select the cached response that Skylar One should use to populate the collection object.

WMI Dynamic Applications

When a WMI Dynamic Application is configured to *Consume Cached Results*:

- The **[Requests]** tab will not be displayed.
- In the **Dynamic Applications | Collections Objects** page, the ***WMI Request*** field will display a list of all requests defined in WMI Dynamic Applications that cache responses. Select the cached response that Skylar One should use to populate the collection object.

PowerShell Applications

When a PowerShell Dynamic Application is configured to *Consume Cached Results*:

- The **[PowerShell]** tab will not be displayed.
- In the **Dynamic Applications | Collections Objects** page, the ***PowerShell Request*** field will display a list of all requests defined in PowerShell Dynamic Applications that cache responses. Select the cached response that Skylar One should use to populate the collection object.

XML Dynamic Applications

When an XML Dynamic Application is configured to *Consume Cached Results*, an additional field is displayed in the **Dynamic Applications | Collections Objects** page:

- ***XML Document***. This field lists all XML Dynamic Applications that cache responses. Select the XML Dynamic Application that requests the XML Document that contains this collection object.

> **NOTE:** For the XML Dynamic Application that caches responses, the URL of the requested XML Document is defined in the credential that is aligned to that Dynamic Application.

XSLT Dynamic Applications

When an XSLT Dynamic Application is configured to *Consume Cached Results*, you must still define XSLT Requests for the Dynamic Application in the **Dynamic Applications Request Editor and Registry** page (the **[Requests]** tab). To define an XSLT Request for an XSLT Dynamic Application that is configured to *Consume Cached Results*:

1. Go to the **Dynamic Applications Request Editor and Registry** page (the **[Requests]** tab) for the Dynamic Application.
2. Supply values in the ***Request Name***, ***Execution Sequence***, and ***Active State*** fields.
3. The ***XSLT Request Code*** field in the **Dynamic Applications Request Editor and Registry** page (the **[Requests]** tab) is renamed ***Cached XSLT Request***. The ***Cached XSLT Request*** field displays a list of all requests defined in XSLT Dynamic Applications that cache responses. Select the cached response that Skylar One should transform using the ***XSLT Parser Code*** defined in this request.
4. Define the XSLT document that Skylar One should use to transform the cached response selected in the ***Cached XSLT Request*** field. The output of the transformation should be in the standard format from which Skylar One parses collection objects.

For an XSLT Dynamic Application that is configured to *Consume Cached Results*, the ***XSLT Request*** field in the **Dynamic Applications | Collections Objects** page will contain only the XSLT Requests defined in the same Dynamic Application.

Collector Affinity for Dynamic Applications That Use Caching

Collector Affinity specifies which Data Collectors are allowed to run collection jobs for Dynamic Applications aligned to component devices. You can define Collector Affinity for each Dynamic Application.

For Dynamic Applications that create cache or consume cache, you should select the following option for ***Collector Affinity***:

- ***Root Device Collector***. The Data Collector assigned to the root device will collect data for the Dynamic Application. This guarantees that Dynamic Applications for an entire DCM tree will be collected by a single Data Collector. This option ensure that caching Dynamic Applications can access the required cache.

Dynamic Component Mapping

This section describes ***Dynamic Application Mapping***, which allows Skylar One to collect data from a single management system, such as a VMware ESX server, and then use that data to create multiple device records for the entities managed by that single management system.

What is Dynamic Component Mapping?

Dynamic Component Mapping allows Skylar One to collect data from a single management system, such as a VMware ESX server, and then use that data to create multiple device records for the entities managed by that single management system. For example, the managed entities for a VMware ESX server would be the Guest VMs hosted by that ESX server.

Skylar One uses Dynamic Applications to retrieve data from the management device and discover each entity managed by that management device. Skylar One then uses that retrieved data to create a device for each managed entity. In some cases, the managed entities are nested.

- In Skylar One, a managed entity is called a ***component device***. A component device is an entity that runs under the control of a management device.
- In Skylar One, the ***root device*** is the physical device that manages one or more component devices.
- In Skylar One, a ***parent device*** is a device that has associated entities modeled as component devices. A parent device can be either a root device or another component device.
- In Skylar One, a ***component identifier*** is a collection object that Skylar One uses to populate information about a component device, such as the device name.

For example, in a Cisco UCS system, Skylar One might discover a physical server that hosts the UCS manager. Skylar One might discover a chassis as a component device. The chassis is a child device to the physical server; the physical server is the root device. Skylar One might also discover a blade as a component device that is part of the chassis. The blade is a child device to the chassis. The chassis is the parent device.

Where in Skylar One are Component Devices Displayed?

You can view root devices and their associated component devices in the following places in Skylar One:

- The **Device Components** page (Devices > Device Components) displays a list of all root devices and component devices discovered by Skylar One. The **Device Components** page displays all root devices and component devices in an indented view, so you can easily view the hierarchy and relationships between child devices, parent devices, and root devices.
- The **Component Map** page (Classic Maps > Device Maps > Components) allows you to view devices by root node and view the relationships between root nodes, parent components, and child components in a map. This makes it easy to visualize and manage root nodes and their components. Skylar One automatically updates the **Component Map** as new component devices are discovered. Skylar One also updates each map with the latest status and event information.

Configuring a Dynamic Application to Create Component Devices

Dynamic Applications that use the following protocols can be configured to create component devices:

- SNMP
- Snippet
- SOAP
- WMI
- PowerShell
- XML
- XSLT

For Skylar One to create component devices using your Dynamic Application, you must perform the following steps to configure Dynamic Component Mapping:

1. In the **Dynamic Applications Properties Editor** page, select a configuration ***Application Type***.
2. In the **Dynamic Applications Properties Editor** page, check the ***Component Mapping*** checkbox.
3. In the **Dynamic Applications | Collections Objects** page, configure a collection object that maps to the *Device Name* component identifier.
4. In the **Dynamic Applications | Collections Objects** page, configure a collection object that maps to the *Unique Identifier* component identifier.
5. Create a Device Class with a ***Device Type*** of *Component* to align with the Dynamic Application.

NOTE: Performance Dynamic Applications cannot be used to create component devices.

If a Dynamic Application is configured to create component devices, Skylar One will create a component device when the collection objects that map to the *Device Name* and *Unique Identifier* of a component device are successfully collected from a device.

For information on configuring device classes for component devices, see the section on *aligning device classes with component devices*.

You can also perform the following additional configuration steps for Dynamic Applications that create component devices:

- In the **Dynamic Applications | Collections Objects** page, configure collection objects that map to the following component identifiers:
    - The availability of the component devices. The values collected using this collection object will be used to generate availability reports and graphs for each component device.
    - The manufacturer of the component devices.
    - The model of the component devices.
    - The names the management system uses to refer to the devices, known as the *Distinguished Name*.
    - The globally unique identifier of the component devices. This value is used to determine when a component device should be associated with a different root device.
    - The MAC Address of the component devices.
    - The UUID of the component devices.
- In the **Dynamic Application Component Alignment** page, define a list of Dynamic Applications Skylar One should automatically align to component devices created using this Dynamic Application.

Configuring Collection Objects as Component Identifiers

To configure a collection object to map to a component identifier:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Find the Dynamic Application that you want to configure. Select its wrench icon (🔧). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page is displayed.
4. Select the wrench icon (🔧) for the collection object you want to edit.
5. For Dynamic Applications that have the ***Component Mapping*** checkbox selected in the **Dynamic Applications Properties Editor** page, the ***Component Identifiers*** field is displayed. In this field, select one or more identifiers that Skylar One can use to create and monitor component devices. The values collected for this collection object will populate the selected identifier for the component devices. Collection objects can define the following component identifiers:
    - *Availability*. The availability of the component devices. Skylar One uses *Availability* collection objects to generate availability reports and graphs for each component device. For information about how Skylar One calculates availability using this component identifier, see the *Availability for Component Devices* section.

- *Class Identifier 1*. The class type of the component devices, typically the vendor. This field can be used to align a discovered component device with an existing device class.
- *Class Identifier 2*. The class sub-type of the component devices, typically the model. This field can be used to align a discovered component device with an existing device class.
- *Device Name (%N)*. The name of the component devices in Skylar One.
- *Distinguished Name (%C)*. The name that the system that monitors the component devices (the root device) uses to identify the component devices. Using this component identifier is optional; this name is not used as a unique identifier by Skylar One. This component identifier is useful if an additional substitution character is required in your Dynamic Application.
- *GUID (%G)*. The globally unique identifier Skylar One will use for the component devices. This value must be unique for all component devices in Skylar One. This value is used to determine when a component device should be associated with a different root device. For information about how Skylar One moves component devices using this component identifier, see the *Moving Component Devices Between Root Devices* section.
- *IP Address*. The IP address of the component device.
- *MAC Address*. The MAC Address of the component device. This value is used to match component devices to devices that are discovered using the discovery tool. If a component device is matched to a physical device, Skylar One discovers a component device directly using the discovery tool. For more information, see the *Merging Component Devices with Physical Devices* section.
- *Organization*. During discovery, if the value of the collection object exactly matches an organization name in Skylar One, Skylar One will align the component device with that organization. During subsequent collections, if the value of this collection object changes and exactly matches an organization name in Skylar One, Skylar One aligns the new organization with the component device. If the value of this collection object does not exactly match any organizations in Skylar One, Skylar One will create an event and align the event with the device aligned with the Dynamic Application.
- *Unique Identifier (%U)*. The unique identifier for the component devices. This value must be unique for all component devices associated with the same root device.
- *UUID*. The universally unique identifier for the component devices.
- *GUID*. Globally unique identifier. This value will be unique within Skylar One. No two component devices can have the same GUID. This component identifier protects the integrity of a component device even if it is aligned with a different root device (for example, with a vmotion event).

NOTE: Each of these options can be defined by only one collection object in a Dynamic Application. If a component identifier is already defined by a collection object in this Dynamic Application, that component identifier will not be displayed in the list. For example, if you have already assigned a collection object to be the device name, the *Device Name (%N)* entry will not appear in the ***Component Identifiers*** list for other collection objects in the Dynamic Application.

6. Select the **[Save]** button to save your changes.

Duplicate MAC Addresses for Component Devices

Skylar One handles duplicate MACs for component devices differently than duplicate MACs for physical devices. When a component device is assigned a MAC address, Skylar One does not enforce uniqueness and will allow a component device to be created with the same MAC address as existing physical devices and/or existing component devices.

Unlike how Skylar One discovers physical devices, Skylar One uses Dynamic Applications to retrieve data from a management device and "discover" each entity managed by that management device as a component device. Skylar One then uses that retrieved data to create a device for each managed entity. In some cases, the managed entities are nested. In Skylar One, physical devices are identified by IP address and MAC address. In Skylar One, component devices are identified by a device name, a unique identifier, and a device class. A Dynamic Application that creates component devices can assign a MAC address to each component device, but is not required to.

- In Skylar One, a managed entity is called a ***component device***. A component device is an entity that runs under the control of a physical management device.
- In Skylar One, the ***root device*** is the physical device that manages one or more component devices.
- In Skylar One, a ***parent device*** is a device that has associated entities modeled as component devices. A parent device can be either a root device or another component device.

For example, in a Cisco UCS system, Skylar One might discover a physical server that hosts the UCS manager. This physical server is the ***root device***. Skylar One might discover a chassis on the root device. The chassis is a ***component device***. The chassis is a child device to the physical server. Skylar One might also discover a blade as a component device that is part of the chassis. The blade is a child device to the chassis. The chassis is the ***parent device***.

Skylar One does not automatically combine new component devices with any existing device record using the MAC address of the new component device. A component device can be combined with an existing device record under the following conditions:

- Dynamic Applications that create component devices can assign a globally unique identifier (GUID) to each component device. When Skylar One performs collection for a Dynamic Application, and the Dynamic Application includes a collection object with a GUID component identifier, Skylar One compares the collected values for that collection object with all GUID values for all component devices discovered in the system. If a newly collected value matches a GUID value for an existing component device, the device from which Skylar One collected the new value will become the parent of the existing component device. The existing component device will no longer be associated with its previous parent device. No new component device will be created.
- You can merge a physical device and a component device. You can do this in the **[Actions]** menu in the **Device Properties** page (Devices > Classic Devices > wrench icon) for either the physical device or the component device. When you merge a physical device and a component device, the device record for the component device is no longer displayed in the user interface; the device record for the physical device is displayed in user interface pages that previously displayed the component device. For example, the physical device is displayed in lieu of the component device in the **Device Components** page and the **Component Map** page. All existing and future data for both devices will be associated with the physical device. You can unmerge a component device from a physical device in the **[Actions]** menu in the **Device Properties** page for the physical device (Devices > Classic Devices > wrench icon).

Component Devices, Collector Groups, and Load Balancing

To perform initial discovery, Skylar One uses a single, selected Data Collector from the collector group. This allows you to troubleshoot discovery if there are any problems.

After each discovered device is modeled (that is, after Skylar One assigns a device ID and creates the device in the database), Skylar One distributes devices among the Data Collectors in the collector group. The newest device is assigned to the Data Collector currently managing the lightest load.

This process is known as ***Collector load balancing***, and it ensures that the work performed by the Dynamic Applications aligned to the devices is evenly distributed across the Data Collectors in the collector group.

Skylar One performs Collector load balancing in the following circumstances:

- A new Data Collector is added to a collector group
- New devices are discovered
- Failover or failback occurs within a collector group (if failover is enabled)
- A user clicks the lightning bolt icon (⚡) for a collector group to manually force redistribution
- Devices in DCM or DCM-R trees will be loaded on the Data Collector currently assigned to the DCM or DCM-R tree rather than being distributed across the collector group. DCM or DCM-R trees will be rebalanced as an aggregate when rebalancing occurs to an available Data Collector with sufficient capacity to sustain the load.

NOTE: Whenever a device is load-balanced from one Data Collector to another, whether due to failover or regular load balancing, the device state information is not transferred to the new Data Collector.

NOTE: The lightning bolt icon (⚡) appears only for collector groups that contain more than one Data Collector. For collector groups with only one Data Collector, this icon is grayed out. This icon does not appear for All-In-One Appliances.

When all of the devices in a collector group are redistributed, Skylar One will assign the devices to Data Collectors so that all Data Collectors in the collector group will spend approximately the same amount of time collecting data from devices.

Collector load balancing uses two metrics:

- ***Device Rating***. A device's rating is the total elapsed time consumed by either 1) all of the Dynamic Applications aligned to the device, or 2) collecting metrics from the device's interfaces, whichever is greater. A Collector's load is the sum of the ratings of the devices assigned to the Collector. The balancer tries to evenly divide the work performed by Collectors by assigning devices to Collectors using the device ratings and Collector loads.
- ***Collector Load***. The sum of the device ratings for all of the devices assigned to a collector.

Skylar One performs the following steps during Collector load balancing:

1. Searches for all devices that are not yet assigned to a collector group.
2. Determines the load on each Data Collector by calculating the device rating for each device on a Data Collector and then summing the device ratings.
3. Determines the number of new devices (less than one day old) and old devices on each Data Collector.
4. On each Data Collector, calculates the average device rating for old devices (sum of the device ratings for all old devices divided by the number of old devices). If there are no old devices, sets the average device rating to "1" (one).
5. On each Data Collector, assigns the average device rating to all new devices (devices less than one day old).
6. Assigns each unassigned device (either devices that are not yet assigned or devices on a failed Data Collector) to the Data Collector with the lightest load. Add each newly assigned device rating to the total load for the Data Collector.

Collector Affinity

Collector Affinity specifies the Data Collectors that are allowed to run collection for Dynamic Applications aligned to component devices. You can define Collector Affinity for each Dynamic Application. Choices are:

- ***Default***. If the Dynamic Application is auto-aligned to a component device during discovery, then the Data Collector assigned to the root device will collect data for this Dynamic Application as well. For devices that are not component devices, the Data Collector assigned to the device running the Dynamic Application will collect data for the Dynamic Application.

> **NOTE:** *Default* might appear as the ***Collector Affinity*** setting for older Dynamic Applications. However, it is no longer an option when creating or editing a Dynamic Application.

- ***Root Device Collector***. The Data Collector assigned to the root device will collect data for the Dynamic Application. This guarantees that Dynamic Applications for an entire DCM tree will be collected by a single Data Collector. You might select this option if:
  - The Dynamic Application has a cache dependency with one or more other Dynamic Applications.
  - You are unable to collect data for devices and Dynamic Applications within the same Device Component Map on multiple Data Collectors in a collector group.
  - The Dynamic Application will consume cache produced by a Dynamic Application aligned to a non-root device (for instance, a cluster device).
  - The Dynamic Application includes snippet code with a ***root_device*** tuple.
- ***Assigned Collector***. The Dynamic Application will use the Data Collector assigned to the device running the Dynamic Application. This allows Dynamic Applications that are auto-aligned to component devices during discovery to run on multiple Data Collectors. This is the default setting. You might select this option if:

- The Dynamic Application has no cache dependencies with any other Dynamic Applications.
- You want the Dynamic Application to be able to make parallel data requests across multiple Data Collectors in a collector group.
- The Dynamic Application can be aligned using mechanisms other than auto-alignment during discovery (for instance, manual alignment or alignment via Device Class Templates or Run Book Actions).

Failover for Collector Groups for Component Devices

If you specified ***Default*** or ***Root Device Collector*** for Dynamic Applications, and the single Data Collector in the collector group for component devices fails, users must create a new collector group with a single Data Collector and manually move the devices from the failed collector group to the new collector group. For details on manually moving devices to a new collector group, see the section on *Changing the Collector Group for One or More Devices*.

Merging Component Devices

Skylar One does not automatically combine new component devices with any existing device record. A component device can be combined with an existing device record under the following conditions:

- Dynamic Applications that create component devices can assign a globally unique identifier (GUID) to each component device. When Skylar One performs collection for a Dynamic Application, and the Dynamic Application includes a collection object with a GUID component identifier, Skylar One compares the collected values for that collection object with all GUID values for all component devices discovered in the system. If a newly collected value matches a GUID value for an existing component device, the device from which Skylar One collected the new value will become the parent of the existing component device. The existing component device will no longer be associated with its previous parent device. No new component device will be created.
- You can merge a physical device and a component device. You can do this two ways:
  - From the **[Actions]** menu in the **Device Properties** page (Devices > Classic Devices > wrench icon) for either the physical device or the component device.
  - From the **[Actions]** menu in the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface), select *Merge Devices* to merge devices in bulk.

When you merge a physical device and a component device, the device record for the component device is no longer displayed in the user interface; the device record for the physical device is displayed in user interface pages that previously displayed the component device. For example, the physical device is displayed in lieu of the component device in the **Device Components** page and the **Component Map** page. All existing and future data for both devices will be associated with the physical device.

If you manually merge a component device with a physical device, Skylar One allows data for the merged component device and data from the physical device to be collected on different Data Collectors. Data that was aligned with the component device can be collected on the Data Collector for its root device. If necessary, data aligned with the physical device can be collected on a different Data Collector.

NOTE: You can merge a component device with only one physical device.

- You can unmerge a component device from a physical device. You can do this in two ways:
    - From the **[Actions]** menu in the **Device Properties** page (Devices > Classic Devices > wrench icon) for either the physical device or the component device.
    - From the **[Actions]** menu in the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface), select *Unmerge Devices* to unmerge devices in bulk.

Availability for Component Devices

For non-component devices, Skylar One checks availability every 5 minutes by performing a ping request, SNMP request, or TCP connection to the admin primary IP address for the device. The result of the availability check is either *available* or *unavailable*. Skylar One uses availability data to generate events when a device is unavailable and to generate availability reports and graphs.

In the **Dynamic Applications | Collections Objects** page, you can configure a collection object to represent the availability of the component devices created by a Dynamic Application. Skylar One processes and stores the availability values returned by the availability collection object in the same way that Skylar One processes and stores the values collected by the availability check for physical devices.

When you configure a collection object to represent the availability of component devices:

- When Skylar One performs collection for this Dynamic Application, if Skylar One can collect a value for a component device using the collection object and the value is not 0 (zero) or "false", the component device is considered "available".
- When Skylar One performs collection for this Dynamic Application, if Skylar One cannot collect a value for a component device using the collection object or Skylar One collects a value that is 0 (zero) or "false", the component device is considered "unavailable".
- If the collection objects aligned with the *Device Name* and *Unique Identifier* component identifiers return lists of values, Skylar One will create multiple component devices. Each component device will be associated with an index, i.e. a location in the list of values. To monitor the availability of all the component devices in the list, the collection object aligned with the *Availability* component identifier should return a list of values with a value at each index associated with a component device. Skylar One will consider a component device "unavailable" when the list of values returned by the collection object aligned with the *Availability* component identifier does not include a value, or returns a value of 0 (zero) or false, at the index for that component device. For more information about Dynamic Application indexing, see the *Indexing* section.
- If you align a collection object with the *Availability* component identifier, Skylar One will create a system availability graph in the **Device Performance** page of each component device created using the Dynamic Application.
- If you align a collection object with the *Availability* component identifier and Skylar One determines that a component device is unavailable, Skylar One will generate an event and supply the value "Unavailable" in the ***Collection State*** column in the **Device Components** page.

The following rules apply to the availability state for component devices:

- Component devices can use a Component Identifier to monitor availability. However, in a tree of component devices, some component devices might have a component identifier for availability and others might not. For example, suppose a component device has a component identifier for availability, and Skylar One considers that component device "unavailable". All the descendents of that component device that do not have their own component identifier for availability will be considered unavailable. As soon as Skylar One finds a descendent with its own component identifier for availability, Skylar One stops checking that descendent and its descendents for availability. Component devices without their own component identifier for availability inherit their availability from their nearest ancestor that has a component identifier for availability.
- For trees that include merged devices, i.e. those that include both hardware devices and component devices, Skylar One skips over the hardware devices and allows them to use a network-based protocol to determine availability. For example, suppose you have a tree like this:
    - Grandparent device is a component device with a component identifier for availability. Skylar One has determined that the grandparent device is unavailable.
    - Child device is a hardware device that uses ICMP and ping to determine availability. When Skylar One evaluates the grandparent's component identifier, Skylar One skips over this device. ICMP and ping determine the availability of this device.
    - Grandchild device is a component device that does not have its own component identifier for availability. When Skylar One evaluates the grandparent's component identifier, Skylar One assigns the grandparent's availability state to this grandchild device.
- If all the hosts in a cluster are powered off or unavailable, both the hardware-based hosts and the associated component devices will have an availability value of "Unavailable". When at least one host in the cluster becomes available, some or all of the associated component devices will also become available.

For instructions on how to align a collection object to a component identifier, see the *Configuring Collection Objects as Component Identifiers* section.

Moving Component Devices Between Root Devices

Depending on the type of management system that you want to monitor using Dynamic Component Mapping, it might be possible for a component device to move from one root device to another root device. For example, if a Dynamic Application with Dynamic Component Mapping configured is aligned with multiple VMware ESX servers that are root devices with guest VMs discovered as component devices, a vMotion event might move a guest VM from one VMware ESX server to another VMware ESX server.

If your Dynamic Applications will create component devices that can move from one root device to another root device, you can configure a collection object that maps to a globally unique identifier (GUID) for each component device. The value of the GUID component identifier must be unique for every component device in Skylar One.

When Skylar One performs collection for a Dynamic Application, and the Dynamic Application includes a collection object with a GUID component identifier, Skylar One compares the collected values for that collection object with all GUID values for all component devices discovered in the system. If a newly collected value matches a GUID value for an existing component device, the device from which Skylar One collected the new value will become the parent of the existing component device. The existing component device will no longer be associated with its previous parent device.

Automatically Aligning Dynamic Applications with Component Devices

In a Dynamic Application that has Dynamic Component Mapping enabled, you can define a list of Dynamic Applications that Skylar One will automatically align to the component devices that are discovered by the Dynamic Application. The Dynamic Applications in the list are automatically aligned to a component device when the component device is created. Every time collection is performed for the Dynamic Application that created a component device, any new Dynamic Applications that have been added to the list of Dynamic Applications are aligned to the component device.

This feature is useful when you want to nest component devices, that is, discover additional component devices that have a component device as the parent. To nest component devices, you must specify that Skylar One should immediately align Dynamic Applications that discover additional component devices upon creation of a component device.

To define a list of Dynamic Applications that you want Skylar One to automatically align to component devices:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application that has Dynamic Component Mapping enabled. Select its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Component]** tab. The **Dynamic Application Component Alignment** page is displayed.

> **NOTE:** The **[Component]** tab and the **Dynamic Application Component Alignment** page do not appear for Dynamic Applications that do not have component mapping enabled in the **Dynamic Applications Properties Editor** page.

4. The **Dynamic Application Component Alignment** page displays two lists of Dynamic Applications:
   - ***Unaligned Dynamic Apps***. Displays a list of all Dynamic Applications in Skylar One that are not automatically aligned to the component devices discovered by this Dynamic Application.
   - ***Aligned Dynamic Apps***. Displays a list of Dynamic Applications that Skylar One will automatically align to the component devices discovered by this Dynamic Application.

   To move a Dynamic Application from the ***Unaligned Dynamic Apps*** list to the ***Aligned Dynamic Apps*** list, select the Dynamic Application and then select the right-arrow button.

   To move a Dynamic Application from the ***Aligned Dynamic Apps*** list to the ***Unaligned Dynamic Apps*** list, select the Dynamic Application and then select the left-arrow button.
5. Select the **[Save]** button to save your changes.

Automatically Assigning a Device Class to Each Component Device

Skylar One can automatically align a device class with each discovered component device. If you do not want to automatically assign all component devices to the same device class, you can use Component Identifiers (variables) to align component devices with device classes. The value of two collection objects will determine the device class for each component device.

To customize the alignment between device classes and component devices, perform the steps in the following two section.

Configuring a Device Class to Match Collected Values

1. Go to the **Device Class Editor** page (System > Customize > Device Classes).
2. When you create or edit a device class for component devices, you must define these three values:
    - ***Device Type***. Select "Component".
    - ***Class Identifier 1***. Enter a value collected by one of the collection objects in the Dynamic Application that creates the component devices. Typically, this value is the manufacturer of the device. For a component device to be assigned to a device class based on component identifiers, the value entered in this field must match the value(s) collected by the ***Class Identifier 1*** component identifier in the collection object.
    - ***Class Identifier 2***. Enter a value collected by one of the collection objects in the Dynamic Application that creates the component devices. Typically, this value is the model of the device. If you want to assign this device class based only on the ***Class Identifier 1***–that is, if you are creating a "fall back" device class for components that do not match specific ***Class Identifier 1*** and ***Class Identifier 2*** combinations–do not enter a value in this field.

> **NOTE**: The combination of the ***Class Identifier 1*** and ***Class Identifier 2*** values should be unique to devices that subscribe to the device class.

Configuring Component Identifiers for the Collection Objects

When you create a Dynamic Application, you can *map the value of a collection object to a component identifier*. The value collected for the collection object (from each device) will populate the ***Component Identifier*** field(for that device).

Two of these component identifiers help Skylar One automatically align a device class with each component device.

- ***Class Identifier 1***. For a component device to be assigned to a device class based on component identifiers, the value(s) collected by this collection object must match the value you entered in the ***Class Identifier 1*** field for that device class.

- ***Class Identifier 2***. For a component device to be assigned to a device class based on component identifiers, either the value(s) collected by this collection object must match the value you entered in the ***Class Identifier 2*** field for that device class or the device class must not include a value for the ***Class Identifier 2*** field.

How Does Skylar One Use Component Identifiers to Automatically Assign a Device Class to Each Component Device?

Skylar One uses the following procedure to automatically assign a device class to each new component device:

1. If the values for component identifiers ***Class Identifier 1*** and ***Class Identifier 2*** exactly match the values specified in the device class fields ***Class Identifier 1*** and ***Class Identifier 2*** in an existing component device class, Skylar One will align the component device with that device class. During subsequent collections, if the value of the one or both of the collection objects changes and exactly matches a device class name and device class description, Skylar One will change the device class to match the values of ***Class Identifier 1*** and ***Class Identifier 2.***
2. If the value for component identifier ***Class Identifier 1*** exactly matches the device class field ***Class Identifier 1*** in a component device class definition, but the component identifier ***Class Identifier 2*** either does not match a device class description or is not defined, Skylar One will assign the component device to the device class with the matching value in the field ***Class Identifier 1*** and no value defined for ***Class Identifier 2***.

> **NOTE**: ScienceLogic recommends that you create such a device class as a "fallback" for the specified ***Class Identifier 1***. Skylar One can then use this device class for any component devices that have an unrecognized ***Class Identifier 2***.

3. If the Dynamic Application does not include component identifiers ***Class Identifier 1*** and ***Class Identifier 2*** **OR** the values for ***Class Identifier 1*** and ***Class Identifier 2*** do not match the fields in an existing component device class, Skylar One aligns the component device with the device class associated with the Dynamic Application. You can define this device class in the **Device Class Editor** (System > Customize > Device Classes > edit a device class > Dynamic App Alignment field) or in the Dynamic Application **Editor** window, in the **Components** tab (System > Manage > Applications > edit a Dynamic Application > Components tab).
4. If the Dynamic Application does not include component identifiers ***Class Identifier 1*** and ***Class Identifier 2*** **OR** the values for ***Class Identifier 1*** and ***Class Identifier 2*** do not match the fields in an existing device class, **AND** the Dynamic Application is not aligned with a device class (either in the **Device Class Editor** page or in the **Components** tab), Skylar One automatically assigns the component device to the device class "Generic: Component".

Aligning a Default Device Class with a Dynamic Application

You can align a device class with a Dynamic Application that creates component devices. If you choose not to use the component identifiers ***Class Identifier 1*** and ***Class Identifier 2*** in your Dynamic Application to match a device class, Skylar One will automatically align each newly discovered component device with the default device class that you aligned with the Dynamic Application.

There are two ways to align a default device class with a Dynamic Application:

- In the **Device Class Editor** page (System > Customize > Device Classes), in the ***Dynamic app Alignment*** field.
- In the **Dynamic Application Editor** window, in the **Components** tab (System > Manage > Applications > Edit a Dynamic Application > Components tab).

Defining a Default Device Class in the Device Class Editor

To define a default device class for a Dynamic Application from the **Device Class Editor** page:

1. Go to the **Device Class Editor** page (System > Customize > Device Classes).
2. Create or edit the device class you want to align with the Dynamic Application.
3. In the field ***Dynamic App Alignment***, select a Dynamic Application.
4. Select the **[Save]** button.

Defining a Default Device Class in the Dynamic Application

To define a default device class for a Dynamic Application from the **Dynamic Application Editor** window:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application that you want to associate a device class with. Select its wrench icon ( ). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Component]** tab. The **Dynamic Application Component Alignment** page is displayed.

> **NOTE**: The **[Component]** tab does not appear for Dynamic Applications that do not have dynamic component mapping enabled.

4. In the ***Device Class Alignment*** field, select a device class.
5. Select the **[Save]** button.

Variables in Dynamic Applications for Component Devices

If Dynamic Applications are aligned to component devices, Skylar One uses the following rules when performing collection:

- If the %D substitution is used in the credential aligned with the Dynamic Application, Skylar One substitutes the IP address of the root device.
- If the Dynamic Application uses cached responses from other Dynamic Applications, the Dynamic Application that caches the response can be aligned to either the component device or the root device.

In SOAP and XSLT Dynamic Applications that are aligned to component devices, you can include the following substitution characters in the ***SOAP Request Code***, ***XSLT Request Code***, and ***XSLT Parser Code*** fields:

- ***%N***. Skylar One will substitute the value of the *Device Name* component identifier for the component device for which collection is being performed.
- ***%U***. Skylar One will substitute the value of the *Unique Identifier* component identifier for the component device for which collection is being performed.
- ***%C***. Skylar One will substitute the value of the *Distinguished Name* component identifier for the component device for which collection is being performed.
- ***%G***. Skylar One will substitute the value of the *GUID* component identifier for the component device for which collection is being performed.

Caching and Component Mapping

Although the caching and dynamic component mapping features can be used separately, the caching feature is useful to collect performance and configuration data for component devices. For example, the following set of Dynamic Applications might be used to discover and monitor component devices:

- A Dynamic Application that requests all the information needed for all other Dynamic Applications in this set. This Dynamic Application caches the responses and is aligned to the root device.
- A Dynamic Application that creates component devices. This Dynamic Application consumes cache and is aligned to the root device.
- One or more Dynamic Applications that collect performance and/or configuration data for each component device. These Dynamic Applications consume the cached responses from the first Dynamic Application and are aligned to each component device.

To associate performance or configuration data with only the appropriate component device (that is, the component device that generated the performance or configuration data), you would use substitution characters in your requests. When the request is executed, Skylar One replaces the substitution character with a component identifier for the component device for which collection is currently being performed.

Dynamic Application Relationships

Dynamic Applications can be configured to ***automatically create relationships between devices***. For example, the Dynamic Applications in the VMware vSphere and NetApp PowerPacks are configured to create relationships between VMware Datastore component devices and their associated NetApp Volume component devices. Relationships created by Dynamic Applications are used and visualized by Skylar One in the same manner as relationships created by topology collection, Dynamic Component Mapping, and manually in the user interface.

There are three methods for configuring Dynamic Applications to automatically create relationships:

- A single configuration Dynamic Application that collects the unique identifier (UID) for a component device. When Skylar One polls a subscriber device and collects a UID value, and that UID value matches the UID for an existing component device in the same component tree as the subscriber device, Skylar One creates a relationship between the subscriber device and the component device.

- A single configuration Dynamic Application that collects the globally unique identifier (GUID) for a component device, typically a component device that is in a different component tree than the subscriber device. When Skylar One polls a subscriber device and collects a GUID value, and that GUID matches the GUID of an existing component device that is in a different component tree than the subscriber device, Skylar One creates a relationship between the subscriber device and the component device.
- A pair of configuration Dynamic Applications collect the same data. One of these Dynamic Applications is configured to define a unique identity for subscriber devices and also references the relationship. The other Dynamic Application defines the relationship and also references the unique identity. If each Dynamic Application collects the same value for unique identity and the same value for relationship, a relationship is created between the two subscriber devices. This type of relationship is described in the section *Configuring Identity-based Relationships*.

Configuring Collection Objects for Relationships

To configure a collection object for the creation of relationships:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the configuration Dynamic Application that you want to configure. Select its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page is displayed.
4. Select the wrench icon () for the collection object you want to edit.
5. The ***Usage Type*** field specifies how this collection object is used to build relationships. The options in this field are:
    - *Standard*. The collection object is not used to create relationships or designated as an index for the specified collection object group.
    - *Group Index*. The collection object is configured as the index for the selected collection object ***Group***. For more information about indexing, see the *Indexing* section. Additionally, if other collection objects in the same group are configured with a ***Usage Type*** of *Identity Namespace* or *Relationship Namespace*, Skylar One uses the collection object with a ***Usage Type*** of *Group Index* to define the unique identifier and create relationships between the subscribers of two related Dynamic Applications. See the *Configuring Identity-based Relationships* section for a description of how to implement this type of relationship.
    - *Identity Namespace*. Skylar One uses the collection object to create relationships between the subscribers of two related Dynamic Applications. This collection object defines the identity portion of the relationship. If Skylar One collects this collection object from a device, Skylar One will make that device the child in a parent-child relationship. See the *Configuring Identity-based Relationships* section for a description of how to implement this type of relationship.
    - *Relationship Namespace*. Skylar One uses this collection object to create relationships between the subscribers of two related Dynamic Applications. If Skylar One collects this collection object from a device, Skylar One will make that device the parent in a parent-child relationship. See the *Configuring Identity-based Relationships* section for a description of how to implement this type of relationship.

NOTE: ***Usage Type*** of *Identity Namespace* defines the **child** portion of an identity-based relationships. ***Usage Type*** of *Relationship Namespace* defines the **parent** portion of an identity-based relationships.

- *GUID Relationship*. When Skylar One collects a value for this collection object, and that value matches the GUID for a component device in the system, Skylar One creates a relationship between the subscriber device and the component device. The subscriber device will be the parent device and the component device will be the child device. The relationship will be labeled with the name of this collection object. For information about globally unique identifiers for component devices, see the *Dynamic Component Mapping* section.
- *Unique Identifier Relationship*. When Skylar One collects a value for this collection object, and that value matches the unique identifier for a component device in the same component tree as the subscriber device, Skylar One creates a relationship between the subscriber device and the component device. The subscriber device will be the parent device and the component device will be the child device. The relationship will be labeled with the name of this collection object. For information about unique identifiers for component devices, see the *Dynamic Component Mapping* section.
- *GUID Relationship + Index*. Skylar One uses this collection object to create relationships in the same manner as the GUID Relationship option. When Skylar One collects a value for this collection object, and that value matches the GUID for a component device in the system, Skylar One creates a relationship between the subscriber device and the component device. The subscriber device will be the parent device and the component device will be the child device. The relationship will be labeled with the name of this collection object. Additionally, the collection object is configured as the index for the specified collection object group. For more information about indexing, see the *Indexing* section.
- *Unique Identifier + Index*. Skylar One uses this collection object to create relationships in the same manner as the Unique Identifier Relationship option. When Skylar One collects a value for this collection object, and that value matches the unique identifier for a component device in the same component tree as the subscriber device, Skylar One creates a relationship between the subscriber device and the component device. The subscriber device will be the parent device and the component device will be the child device. The relationship will be labeled with the name of this collection object. Additionally, the collection object is configured as the index for the specified collection object group. For more information about indexing, see the *Indexing* section.

4. Click **[Save]**.

Configuring Identity-Based Relationships

An identity-based relationship is created using a pair of Dynamic Applications:

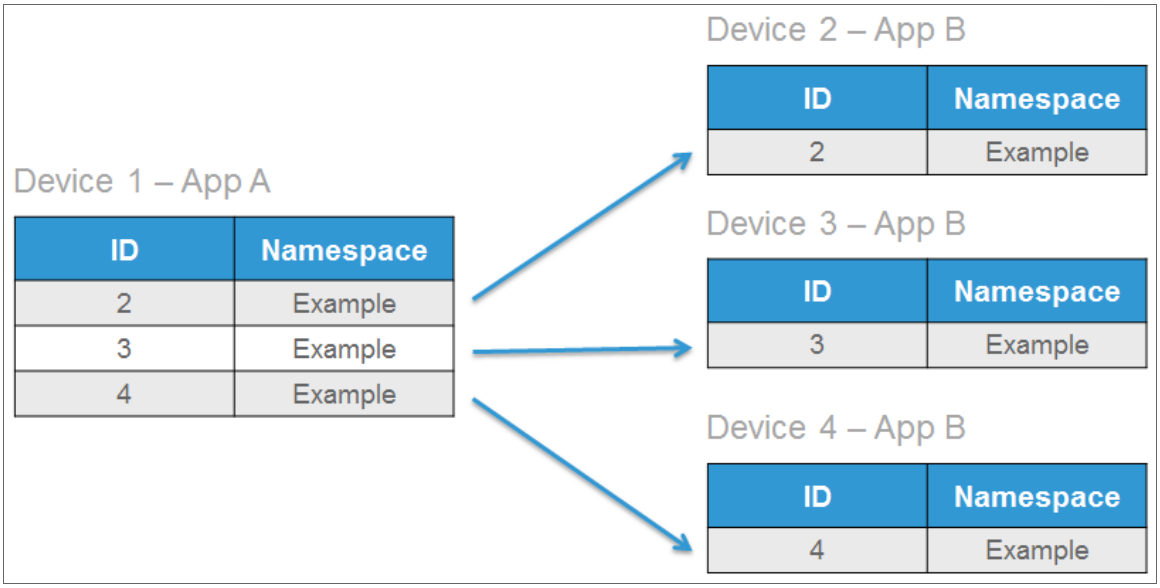
- A Dynamic Application that defines a unique identity for subscriber devices. This Dynamic Application defines:
    - a unique ID for the device, using a collection object that usually has a ***Usage Type*** of *Group Index*.
    - the identity side of the relationship, using a collection object that usually has a ***Usage Type*** of *Identity Namespace*. The devices that subscribe to this Dynamic Application will be children in the parent-child relationship that is created using the unique identity.

> NOTE: This identifier must be unique to this relationship, meaning no two subscribers of the same Dynamic Applicationshould collect the same identifier for the same relationship. However, both Dynamic Applicationsin the pair must collect the same identifier value for a relationship to be created.

- A Dynamic Application that defines the relationship for subscriber devices. This Dynamic Application defines:
    - a unique ID for the device, using a collection object that usually has a ***Usage Type*** of *Group Index*.
    - a relationship, using a collection object that usually has a ***Usage Type*** of *Relationship Namespace*. The devices that subscribe to this Dynamic Application will be parents in the parent-child relationship that is created using the unique identity.

A relationship is created when:

- both Dynamic Applications in a pair collect identical values for the unique identifier (the collection object with a ***Usage Type*** of *Group Index*.
- the value for the collection object that has a ***Usage Type*** of *Identity Namespace* is identical to the value for the collection object that has a ***Usage Type*** of *Relationship Namespace*.



To configure a pair of Dynamic Applications to create identity-based relationships, go to the Dynamic Applications Collection Objects page:

- In both Dynamic Applications, select the collection object that collects the identifier for the relationship. In the ***Usage Type*** field, select *Group Index*. Select a ***Group*** for the collection object. The Group should be the same in both Dynamic Applications.
- In the Dynamic Application that defines a unique identity for subscriber devices, select the collection object that collects the namespace for the relationship. In the ***Usage Type*** field, select *Identity Namespace*. For this collection object, select the same ***Group*** as the *Group Index* collection object.
- In the Dynamic Application that defines the relationship, select the collection object that collects the namespace for the relationship. In the ***Usage Type*** field, select one of the values that includes the text "Relationship". For this collection object, select the same ***Group*** as the *Group Index* collection object.

NOTE: The *Identity Namespace* and *Relationship Namespace* objects must return a value for each of the indexes returned by the *Group Index* object. Typically, the values at each index returned by the *Identity Namespace* and *Relationship Namespace* objects are equal.

NOTE: For an example of a Dynamic Application with an Identity-Based relationship, see the ***Example of a Dynamic Application with an Identity-Based Relationship*** section in the ***Database Dynamic Application Development*** chapter.

Viewing a Map of Component Devices

The **Component Map** page allows you to view devices by root node and also view the relationships between root nodes, parent components, and child components. This map makes it easy to visualize and manage root nodes and their components.

NOTE: User accounts of type "user" can view only root nodes and device components that belong to their organization(s).

Skylar One uses Dynamic Applications to retrieve data from a management device and discover each entity managed by that management device. Skylar One then uses that retrieved data to create a device for each managed entity. In some cases, the managed entities are nested.

- In Skylar One, a managed entity is called a ***component device***. A component device is an entity that runs under the control of a physical management device.
- In Skylar One, the ***root device*** is the physical device that manages one or more component devices.
- In Skylar One, a ***parent device*** is a device that has associated entities modeled as component devices. A parent device can be either a root device or another component device.

For example, in a Cisco UCS system, Skylar One might discover a physical server that hosts the UCS manager. Skylar One might discover a chassis as a component device. The chassis is a child device to the physical server; the physical server is the root device. Skylar One might also discover a blade as a component device that is part of the chassis. The blade is a child device to the chassis. The chassis is the parent device.

Skylar One automatically updates the Component Map as new component devices are discovered. Skylar One also updates each map with the latest status and event information.

NOTE: For a user of type *User*, you can view only root nodes and device components that belong to your organization(s).

Viewing a Component Map

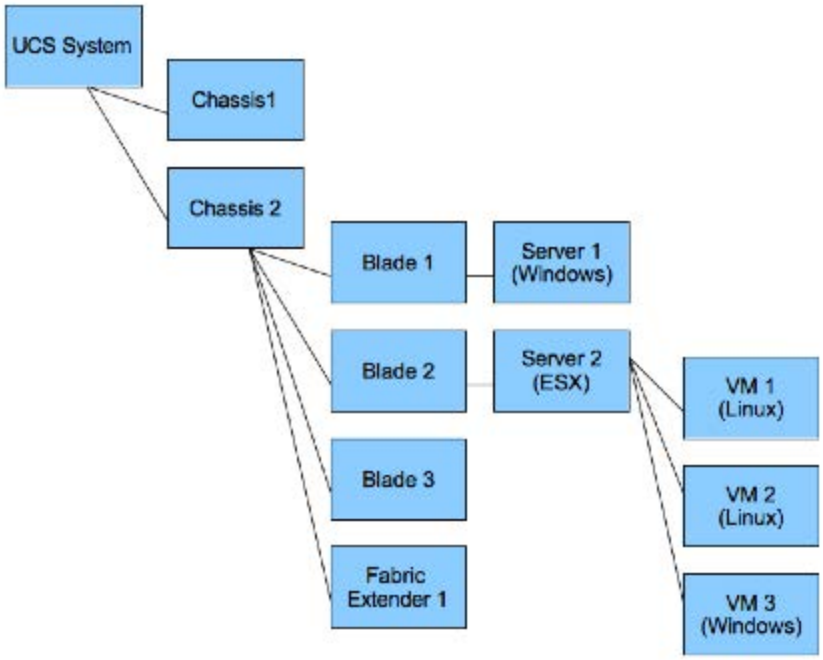
You can view a component map from the **Component Map** page. To view a Component Map:

1. Go to the **Classic Maps** page (Maps > Classic Maps).
2. Expand the links for Device Maps > Components and then select a root node. The **Device Component Map** page appears and displays the map of the root node and its component devices.

Adding Devices from Another Component Tree to the Component Map

You can add a device from another component tree to the map of the current component device.

For example, suppose you have a UCS system. Suppose you are running an ESX server and VMs on one of the blades of the UCS system:



You can add the device where the ESX server resides to the map of the UCS system. To do this:

1. Go to the **Component Map** page (Maps > Classic Maps > Device Maps > Components).
2. Expand the list of maps. Find the component map to which you want to add a device.
3. Select the **[Nodes]** button. Select the **Add Dev** icon (in the upper left).
4. In the **Device Browser** modal page, select the root node you want to add to the component map. Select the **[Add To Map]** button.

> **NOTE**: When you complete these steps, the Skylar One system will automatically add all the child devices for the newly added node to the map.

5. Select the **[Links]** button. Click on the parent device and then click on the newly added child device.
6. The link you created is an ***Event Correlation Override*** link. If you want to manually define a parent device and child device for two devices that do not share a Layer-2 link, a Layer-3 link, a CDP link, an LLDP link, or a VMware relationship, you can create an ***Event Correlation Override*** link between the devices. Additionally, you can then define an event hierarchy for these devices.
7. Select the **[Save]** button.
8. Select the **[Reset]** button to view the changes to the map.

Viewing Relationships Maps in the Organization View

The **Organizational Map** page allows you to view devices by organization. This makes it easy to visualize and manage devices in organizations. If devices in the organization include relationships created by Dynamic Applications, these relationships are displayed in the map. If devices in this organization include CDP, LLDP, layer-2, or layer-3 relationships, they are included in the map.

All elements, policies, events, tickets, and users in Skylar One are associated with an organization. An organization is a group for managing elements and user accounts. You can define organizations by geographic area, departments, types of devices, or any structure that works best for your needs. The minimum characteristics of an organization consist of:

- A unique name
- Users who are members of the organization
- Elements, such as devices, associated with the organization

Skylar One automatically updates the Organizational Maps as new devices are added to organizations and as new organizations are defined. Skylar One also updates each map with the latest status and event information.

To select an Organizational Map, go to the **Classic Maps** page (Maps > Classic Maps), expand the links for Device Maps > Organizational, and then select an organization. The **Organizational Map** page appears and displays a map of the selected organization and its devices.

Viewing an Organizational Map

You can view an organizational map from the **Organizational Map** page. To view an organizational map:

1. Go to the **Classic Maps** page (Maps > Classic Maps).
2. Expand the sections for Device Maps > Organizational, and then select an organization. The **Organizational Map** page appears and displays a map of the organization and its member devices:

- The search field at the top allows you to find one or more devices in the map. You can enter a string, and Skylar One will highlight only the devices that have a device name that matches the string.
- Each member device appears as an icon in the map.
- The Organizational Map page for an organization also displays the relationships between devices in the organization. This includes:
    - Layer-2 devices and their clients
    - Layer-3 devices and layer-2 devices
    - Component devices and their parents, e.g. virtual machines and their hypervisors
    - Network devices that use CDP (Cisco Discovery Protocol) and devices that are specified as neighbors in the CDP tables
    - Network devices that use LLDP (Link Layer Discovery Protocol) and devices that are specified as neighbors in the CDP tables
    - Device relationships created with Dynamic Applications
    - Manually created parent-child relationships that affect event correlation

3. When the map appears, you can view and reposition the components. The map can be edited and rearranged using drag-and-drop features. Devices can be repositioned for easier reading, if necessary.
4. Mousing over a device displays its name, IP address, device type, and device category.

Viewing Relationships in Customized Maps

A customized map allows you to view the devices and links that are most important to you.

When you create a customized map, you are also creating a new device group (which appears in the **Device Groups** page). You can add devices and other sub-device groups to the new map, just as you would to a standard device group.

If Skylar One has information about the relationships between the devices in a customized map, Skylar One automatically includes the appropriate links in the customized map. Customized maps display every type of relationship data, which includes:

- Layer-2 devices and their clients
- Layer-3 devices and layer-2 devices
- Component devices and their parents, e.g. virtual machines and their hypervisors
- Network devices that use CDP (Cisco Discovery Protocol) and devices that are specified as neighbors in the CDP tables
- Network devices that use LLDP (Link Layer Discovery Protocol) and devices that are specified as neighbors in the LLDP tables
- Device relationships created with Dynamic Applications
- Manually created parent-child relationships that affect event correlation

Customized maps appear in the following sections:

- ***My Customized Maps***. Personalized maps that you create.
- ***User Customized Maps***. If you are a user of type "administrator", you can navigate to the maps in this section to view and edit all customized maps in Skylar One, even if the device group associated with the map was defined with the field ***Shared (visible to all users)*** set to *no*.
- ***Shared Customized Maps***. If a customized map or device group is defined as "shared", you can view the maps in this section. The maps in **Shared Customized Maps** require the same Access Hooks and Access Keys as device groups. Depending upon the Access Keys assigned to your account, you might be able to edit **Shared Customized Maps** created by other users. To learn more about Access Hooks and Access Keys, see the manual ***Access Permissions***.

> **NOTE**: If you create a device group from the **Device Groups** page and set the ***Visibility*** field to include *Maps/Views*, the device group will appear as a map in the **Custom Device Group Map** page (Maps > Classic Maps > My Customized Maps). If you set the ***Shared*** field to *yes*, the device group will appear for view by other users as a map in the **Shared Customized Maps** page (Maps > Classic Maps > Shared Customized Maps).

To learn more about Customize Maps, see the manual ***Views***.

Viewing Relationships for a Single Device

You can view all links for a single device on the **Relationships** tab of the **Device Investigator** (or on the **Device Relationships** page in the **Device Properties** panel in the classic Skylar One user interface).

To view all links for a single device:

1. Go to the **Relationships** tab of the **Device Investigator**. (Alternatively, go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface), click the wrench icon for a device (🔧) and then click the **[Relationships]** tab.) The **Device Relationships** page appears.
2. The left pane of the **Device Relationships**page displays links to parent devices. The right pane of the **Device Relationships** page displays links to child devices. For each relationship, the **Device Relationships** page displays the following information:
    - ***Type of relationship***. Possible values are:
        - *ARP*. Network devices that use ARP (Address Resolution Protocol) and devices that are specified as neighbors in ARP tables.
        - *CDP*. Network devices that use CDP (Cisco Discovery Protocol) and devices that are specified as neighbors in CDP tables.
        - *Component Mapping*. Relationships defined using Dynamic Applications.
        - *Event Correlation*. Relationships defined manually by users through the user interface.
        - *Layer 2*. Layer-2 devices and their clients.
        - *Layer 3*. Layer-3 devices and layer-2 devices.

    - *LLDP*. Network devices that use LLDP (Link Layer Discovery Protocol) and devices that are specified as neighbors in LLDP tables, including those devices that respond only to LLDP v2.
    - *VMware*. Hypervisors and their virtual machines.
- ***Parent Device***. The name of the parent device and a link to the **Device Properties** page for the parent device.
- ***Parent Interface***. The name of the interface through which the parent device communicates with the child device and a link to the **Interfaces Found** page for the parent interface.
- ***Child Device***. The name of the child device and a link to the **Device Properties** page for the child device.
- ***Child Interface***. The name of the interface through which the child device communicates with the parent device and a link to the **Interfaces Found** page for the child interface.

NOTE: Clicking on a device reloads the **Device Relationships** page and makes the selected device the primary device.

The **Device View** page appears when a user clicks the **Topology** tab in the Device Reports panel. The **Device View** page displays a map of the device and all of the devices with which the device has relationships.

These relationships include:

- Layer-2 devices and their clients
- Layer-3 devices and Layer-2 devices
- Component devices and their parent devices. For example, virtual machines and their hypervisors and their virtual machines.
- Network devices that use CDP (Cisco Delivery Protocol) and devices that are specified as neighbors in CDP tables
- Links between network devices that use CDP (Cisco Discovery Protocol) and devices that are specified as neighbors in CDP tables
- Network devices that use LLDP (Link Layer Delivery Protocol) and devices that are specified as neighbors in LLDP tables
- Links between network devices that use LLDP (Link Layer Discovery Protocol) and devices that are specified as neighbors in LLDP tables
- Device relationships between root devices, parent devices, and component devices (Component Mapping)
- Device relationships created with Dynamic Applications
- Manually created parent-child relationships that affect event correlation

> **NOTE**: Double-clicking on a device reloads the **Device View** page and makes the selected device the primary device.

For details on the toolbars that appear in this page, see the ***Views*** manual.

Presentation Objects for Performance Dynamic Applications

This section describes ***Presentation Objects***, which define how Skylar One should present data for Dynamic Applications with an archetype of **Performance** or **Journal**.

What is a Presentation Object?

Presentation Objects define how Skylar One should present data for Dynamic Applications with an archetype of ***Performance*** or ***Journal***. For performance Dynamic Applications, a presentation object defines how Skylar One uses the collected values for the collection objects to generate graphs.

For example, a Dynamic Application that is designed to collect information about file system usage on a device might define:

- Two collection objects, one that collects the total size of each file system and one that collects the amount of space used on each file system.
- A presentation object that defines a graph. This graph displays the percentage of space used for each file system. The presentation object would include a formula that uses both collection objects.

> **NOTE:** This section describes presentation objects in performance Dynamic Applications. This section does not apply to journal Dynamic Applications with an archetype of Journal. For information on presentation objects in journal Dynamic Applications, see the ***Snippet Dynamic Application Development*** chapter.

Viewing the Presentation Objects in a Dynamic Application

To view the presentation objects in a Dynamic Application:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to view the presentation objects for. Click its wrench icon (🔧). The **Dynamic Applications Properties Editor** page is displayed.
3. Click the **[Presentations]** tab. The **Dynamic Applications Presentation Objects** page is displayed. The **Presentation Object Registry** pane at the bottom of the page displays a list of defined reports.

4. For each report in a Dynamic Application of type Performance, the **Presentation Object Registry** displays the following:
    - ***Report Name***. The name of the presentation object.
    - ***State***. The state of the presentation object. Possible values are:
        - *Enabled*. Skylar One will generate graphs for this presentation object.
        - *Disabled*. Skylar One will not generate graphs for this presentation object.
    - ***Abbreviation Suffix***. Abbreviation for the unit of measure to display in the report.
    - ***Label Group***. Name of the Collection Group aligned with the presentation object. For more information on Collection Labels, see the section on *Collection Labels*.
    - ***Label***. Name of the Collection Label aligned with the presentation object. The ***Vitals*** group is available as a Collection Label and preserves the previous behavior of ***Vitals Link***.For more information on Collection Labels, see the section on *Collection Labels*.
    - ***Precedence***. Weight assigned to this presentation object. For more information on Collection Labels and Precedence, see the section on *Precedence*.
    - ***Show as Percent***. Specifies whether the resulting graph displays percent values. If so, the y-axis of the graph will range from 0 - 100 percent.
        - *Enabled*. The graph displays percent values.
        - *Disabled*. The graph does not display percent values.
    - ***ID***. The unique ID assigned to the report. Skylar One automatically generates and assigns this identifier.
    - ***Date Edit***. Date and time the report definition was last edited.

Creating a Presentation Object

To add a presentation object to a performance Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the performance Dynamic Application you want to add a presentation object to. Click its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.
3. Click the **[Presentations]** tab. The **Dynamic Applications Presentation Objects** page is displayed.
4. In the **Dynamic Applications Presentation Objects** page, click the **[Reset]** button to clear values from the fields in the top pane, the **Formula Editor** pane, and the **Guide Text** pane.
5. Supply values in the following fields:
    - ***Report Name***. Name of the report. Can be any combination of alpha-numeric characters. This name will appear as the name of the report.
    - ***Summarization State***. Specifies whether Skylar One should include hourly and daily summary statistics for the presentation object in the graph. Choices are:
        - *Enabled*. Skylar One will include summary statistics for the presentation object in the graph.

    - *Disabled*. Skylar One will not include summary statistics for the presentation object in the graph.
- ***Data Unit***. Units used in the graph. Can be any combination of alphanumeric characters, up to 25 characters in length.
- ***Abbreviation/Suffix***. Abbreviation for units used in the graph. Can be any combination of alphanumeric characters, up to 16 characters in length.
- ***Show as Percent***. Specifies whether the resulting graph displays percent values. If so, the y-axis of the graph will range from 0 – 100 percent.
    - *Enabled*. The graph displays percent values.
    - *Disabled*. The graph does not display percent values.
- ***Precedence***. Weight to assign to this presentation object. For more information on Collection Labels and Precedence, see the section on *Precedence*.
- ***Label Group***. Name of the Collection Group to align with the presentation object. The ***Vitals*** group is available as a Collection Label and preserves the previous behavior of ***Vitals Link***. You can click the plus-sign (+) to create a new Collection Group. For more information on Collection Labels, see the section on *Collection Labels*.
- ***Label***. Name of the Collection Label to align with the presentation object. You can click the plus-sign (+) to create a new Collection Label.
- ***Formula Editor***. The formula Skylar One will use to generate graphs for the presentation object. For a full description of this field, see the *Presentation Object Formulas* section.
- ***Guide Text***. Enter optional guide text for the presentation object.

6. Click the **[Save]** button to save the presentation object.

Presentation Object Formulas

The ***Formula Editor*** allows users to define which object value(s) that will be used to generate the graph for this presentation object and the mathematical manipulations that must be performed on those object values. The result of a defined formula must be a single numeric value.

A presentation object formula can include:

- The ID for one or more collection objects in the same Dynamic Application. All collection object IDs start with "o_" (lowercase "oh", underscore).
- The four main arithmetic operators (+, -, *, /).
- Parentheses, which set precedence for arithmetic operators.
- Braces ({ and }), which can be used to specify that an object is optional, and the formula should still be evaluated even if a value for the optional object has not been collected.
- Any PHP math function that evaluates to a numeric value. For information about PHP math functions, see http://www.php.net/math.

The operators and PHP math functions may be used in any combination and with any number of collection object IDs, provided that the result of the expression always equates to a single numeric value.

The following examples are all valid presentation formulas:

```
o_123
```

```
abs(o_123)
```

```
o_123 / o_124 * 100
```

```
(o_123 + o_124) * o_125
```

```
o_123 + {o_124}
```

The scrolling list below the ***Formula Editor*** contains a list of all objects in the Dynamic Application. To insert the ID for a collection object in to the formula, select the collection object from the list, and then select the **[Add]** button.

The ***Guide Text*** pane allows users to include descriptive text with each graph. When a user views the graph in the **Device Performance** page, the **[Guide]** button will display the text from this pane.

Vitals Linking

When ***Vitals Linking*** is enabled for a presentation object in a Dynamic Application, Skylar One will use the presentation object to calculate and display CPU, Memory, or Swap utilization values for devices that subscribe to the Dynamic Application.

For example, suppose that a device subscribes to the default "Net-SNMP: CPU" Dynamic Application, which collects CPU data from Net-SNMP agents. The "Overall CPU" presentation object in this Dynamic Application has *CPU* selected in the ***Vitals Link*** field. When a user selects the **[Performance]** tab in the **Device Summary** window for this device:

- The **System Vitals Summary Report** (Overview > System Vitals) will include the graph generated by the "Overall CPU" presentation object.
- The **Overall CPU Utilization Report** (Overview > CPU Utilization) will show the graph generated by the "Overall CPU" presentation object.

Additionally, Skylar One will calculate hourly CPU usage for that device using the "Overall CPU" presentation formula. The presentation formula uses collected values from the device, specified in the collection objects. The hourly CPU data is stored separately from the other Dynamic Application data.

Editing a Presentation Object

To edit a presentation object in a Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to edit. Click its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.
3. Click the **[Presentations]** tab. The **Dynamic Applications Presentation Objects** page is displayed.
4. In the **Presentation Object Registry** pane at the bottom of the page, locate the presentation object you want to edit. Click the wrench icon () for the presentation object you want to edit.

5. The fields in the top pane will be populated with the saved values for the selected presentation object. Edit the values in one or more fields. For a description of each field, see the *Creating a Presentation Object* section.
6. Click the **[Save]** button to save your changes to the presentation object. Click the **[Save As]** button to save the changes as a new presentation object.

Deleting a Presentation Object

To delete a presentation object from a Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to edit. Click its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.
3. Click the **[Presentations]** tab. The **Dynamic Applications Presentation Objects** page is displayed.
4. In the **Presentation Object Registry** pane at the bottom of the page, locate the presentation object you want to delete. Click its delete icon (). The presentation object is deleted from Skylar One. Any collected data that the presentation object used will remain in Skylar One. However, you will no longer be able to view any graphs generated using the presentation object.

Alerts and Thresholds

This section describes what an alert and threshold is and how to use them in a Dynamic Application.

What is an Alert?

An ***alert*** defines a formula that Skylar One will evaluate each time data is collected. Each formula includes ***collection objects***. When Skylar One evaluates the alert formula, Skylar One will substitute in the most recently collected values for each collection object. If the formula evaluates to *true*, Skylar One generates an alert. When Skylar One generates an alert, Skylar One makes an entry in the device log. Additionally, you can define an event policy that will trigger an event if a specific alert is generated.

What is a Threshold?

A ***threshold*** defines a variable that can be used in one or more ***alerts***. Each threshold defines a range of values for the variable and a default value. Users can change the value of the threshold on a per-device basis without having to modify the Dynamic Application.

Viewing the Thresholds in a Dynamic Application

To view the thresholds in a Dynamic Application:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to view the thresholds for. Select its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.

3. Select the **[Thresholds]** tab. The **Dynamic Applications Threshold Objects** page is displayed. The **Thresholds Object Registry** pane at the bottom of the page displays the following information about each threshold:
    - ***Name***. The name of the threshold. This name will be used to label the threshold in the **[Thresholds]** tab in the **Device Administration** panel.
    - ***Override***. Defines whether this threshold can be overridden on a per-device basis. Possible values are:
        - *Enabled*. When the Dynamic Application that contains this threshold is aligned with a device, a user will be able to set a new value for this threshold in the **[Thresholds]** tab in the **Device Administration** panel.
        - *Disabled*. Users cannot set a new value for this threshold on a per-device basis.
    - ***Type***. Specifies whether the threshold will be a *Percentage*, *Integer*, or *Decimal*.
    - ***Numeric Range High***. When a user overrides this threshold for a device, this is the maximum value the threshold can be set to.
    - ***Numeric Range Low***. When a user overrides this threshold for a device, this is the minimum value the threshold can be set to.
    - ***Threshold Unit***. Specifies the unit of measure for the threshold value.
    - ***Threshold Value***. The default value for the threshold. If a user does not define an override for a device, Skylar One will use this threshold value for that device.
    - ***ID***. The unique ID assigned to the threshold by Skylar One. This ID is used to reference the threshold in alert formulas. The unique ID will always start with "t_".
    - ***Date Edit***. The last time a user edited this threshold.

Creating a Threshold

To add a threshold to a Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to add a threshold to. Select its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Thresholds]** tab. The **Dynamic Applications Threshold Objects** page is displayed.
4. Supply values in the following fields:
    - ***Threshold Name***. The name of the threshold. This name will be used to label the threshold in the **[Thresholds]** tab in the **Device Administration** panel.
    - ***Override Threshold Value***. Defines whether this threshold can be overridden on a per-device basis. When disabled, this threshold does not appear on the **Device Thresholds** page for each subscriber device and cannot be edited for each subscriber device. Choices are:
        - *Enabled*. Threshold appears on the **Device Thresholds** page for each subscriber device and can be edited for each subscriber device.

    - *Disabled*. Threshold does not appear on the **Device Thresholds** page for each subscriber device.
- ***Numeric Range High***. Specifies the high end of the possible values. In the **Device Thresholds** page for each subscriber device, this number will appear at the high end of the slider. When a user overrides this threshold for a device (in the **Device Thresholds** page), this is the maximum value the threshold can be set to.
- ***Numeric Range Low***. Specifies the low end of the possible values. In the **Device Thresholds** page for each subscriber device, this number will appear at the low end of the slider. When a user overrides this threshold for a device (in the **Device Thresholds** page), this is the minimum value the threshold can be set to.
- ***Threshold Type***. Specifies whether the threshold will be a *Percentage*, *Integer*, or *Decimal*.
- ***Threshold Unit***. Specifies the unit of measure for the threshold value.
- ***Threshold Value***. Assigns a default value to the threshold object. If a user does not override the threshold in the **Device Thresholds** page for a subscriber device, Skylar One will use this threshold value for the subscriber device.

5. Select the **[Save]** button to save the threshold.

Editing a Threshold

To edit an already-defined threshold:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. In the **Dynamic Applications Manager** page, find the Dynamic Application for which you want to edit a threshold. Select its wrench icon ( ).
3. Select the **[Thresholds]** tab for the Dynamic Application.
4. In the **Threshold Objects** page, find the threshold in the **Threshold Object Registry** pane. Select its wrench icon ( ).
5. The fields in the top pane are populated with values from the selected threshold. You can edit the value of one or more fields. For a description of each field, see the *Creating a Threshold* section.
6. Select the **[Save]** button to save your changes to the threshold.

Deleting a Threshold

You can delete a threshold from a Dynamic Application. To do this:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. In the **Dynamic Applications Manager** page, find the Dynamic Application for which you want to delete a threshold. Select its wrench icon ( ).
3. Select the **[Thresholds]** tab for the Dynamic Application.
4. In the **Threshold Objects** page, find the threshold in the **Threshold Object Registry** pane. Select its delete icon ( ). The threshold will be deleted from Skylar One. You must manually remove the threshold from all alert formulas in which that threshold appears.

Viewing the Alerts in a Dynamic Application

To view the alerts in a Dynamic Application:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to view the alerts for. Select its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Alerts]** tab. The **Dynamic Applications Alert Objects** page is displayed. The **Alert Object Registry** pane at the bottom of the page displays the following information about each alert:
    - ***Policy Name***. The name of the alert.
    - ***Formula***. The formula that Skylar One will evaluate to determine whether to generate the alert. An alert formula can contain references to collection objects ("o_" followed by a numeric value), threshold objects ("t_" followed by a numeric value), and/or other alerts ("a_", followed by a numeric value).
    - ***State***. Specifies whether Skylar One will evaluate the alert when new data is collected for the Dynamic Application. Possible values are:
        - *Enabled*. Skylar One will evaluate the alert when new data is collected.
        - *Disabled*. Skylar One will not evaluate the alert when new data is collected.
    - ***Maintain***. Specifies whether Skylar One will track the status of this alert (active or inactive) for use as a condition in other alert formulas. Possible values are:
        - *Yes*. Skylar One tracks the status of this alert. The status of this alert can be used as a condition in other alert formulas.
        - *No*. Skylar One does not track the status of this alert. The status of this alert cannot be used as a condition in other alert formulas.
    - ***Events***. Specifies whether an event policy has been created for this alert. Possible values are:
        - *Yes*. An event policy has been created for this alert. Selecting the event icon () will open the **Event Policy Editor**, where you can edit the associated event.
        - *No*. An event policy has not been created for this alert. Selecting the gray event icon ( ) will open the **Event Policy Editor**, where you can define an event policy for this alert.
    - ***ID***. The unique ID assigned to the alert by Skylar One. This ID is used to reference the alert in other alert formulas. The unique ID will always start with "a_".
    - ***Date Edit***. The last time a user edited this alert.

Creating an Alert

To add an alert to a Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to add an alert to. Select its wrench icon (🔧). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Alerts]** tab. The **Dynamic Applications Alert Objects** page is displayed.
4. Supply values in the following fields:
    - ***Policy Name***. Enter a name for the alert.
    - ***Active State***. Specifies whether Skylar One will evaluate the alert when new data is collected for the Dynamic Application. Possible values are:
        - *Enabled*. Skylar One will evaluate the alert when new data is collected.
        - *Disabled*. Skylar One will not evaluate the alert when new data is collected.
    - ***Log Message***. Enter the log message associated with the alert. When Skylar One generates this alert, this message will appear in the device logs. You can use the following substitution characters in this field. The substitution characters are populated by including certain functions in the ***Formula Editor*** for the alert:
        - *%V*. The value returned by the result() function.
        - *%L*. The value returned by the "label" argument in the result() function.
        - *%T*. The value returned by the threshold() function.
    - ***Maintain State***. Specifies whether Skylar One will track the status of this alert (active or inactive) for use as a condition in other alert formulas. Possible values are:
        - *Yes*. Skylar One tracks the status of this alert. Select this option if you want to use the status of this alert as a condition in other alert formulas.
        - *No*. Skylar One does not track the status of this alert. Select this option if you do not want to use the status of this alert as a condition in other alert formulas.
    - ***Trigger Alert***. This is a deprecated field. Leave this field set to the default value.
    - ***Formula Editor***. Enter the formula that Skylar One will evaluate to determine whether to generate the alert. For a full description of this field, see the *Alert Formulas* section.
5. Select the **[Save]** button to save the alert.

Alert Formulas

The ***Formula Editor*** field allows you to define the formula that Skylar One will evaluate for an alert. The result of a defined alert formula must be either *True* or *False*. If the formula evaluates to *True*, Skylar One generates an alert.

An alert formula can include:

- The ID for one or more collection objects in the same Dynamic Application. All collection object IDs start with "o_" (lowercase "oh", underscore). However, do not use a collection object of type *Label: Hourly Polled* in an alert formula.
- The ID for one or more thresholds in the same Dynamic Application. All threshold IDs start with "t_".

- Arithmetic operators (+, -, *, /), comparison operators (>, <, !=, ==), and/or boolean operators ("and", "or", "not").

> **NOTE:** The "==" operator is used to test equality. The single "=" assignment operator is not supported, except when used inside some functions).

- The result() and/or threshold() functions. The results of these functions can be included in the log message for the alert.
- One or more ScienceLogic specific functions described in this section.
- One or more ScienceLogic specific date & time variables or the tab_idx variable described in this section.
- Any standard Python function.

The operators, functions, and variables may be used in any combination and with any number of collection object IDs and threshold IDs, provided that the result of the expression always equates to *True* or *False*.

The scrolling list below the ***Formula Editor*** contains a list of all collection objects and thresholds in the Dynamic Application. To insert the ID for a collection object or threshold in to the formula, select the collection object or threshold from the list, then select the **[Add]** button.

Evaluate

To evaluate the value of a collection object, enter an expression in the ***Formula Editor*** that compares the value of the object to something else, where "something else" can be another collection object, a threshold object, a simple number, a string, the result of a function, or a combination of objects and numbers.

For example, suppose you want to generate an alert based on the value of collection object "o_15160", which contains the current temperature of a hardware component in a Dell server. You could select object "o_15160: Temp. Reading" from the scrolling list. When the object appears in the ***Formula Editor***, you could enter the following formula:

```
o_15160 > 50
```

The formula evaluates to *True* if the value returned by object o_15160 is greater than 50.

> **NOTE:** Skylar One assigns each object an object ID. This is different than the object number assigned to the object in the MIB. Skylar One requires that you use the object ID when referring to an object.

If one or more of the collection objects used in an alert formula collect a list of values, Skylar One evaluates the alert formula for each entry in the list. An alert can be generated for each entry in the list. For example, suppose the following alert formula is evaluated:

```
o_123 / o_124 * 100 > 90
```

Suppose that for each poll period for a device, the collection objects (o_123 and o_124) return a list of two values. Skylar One will evaluate the alert formula twice for this device. For the first evaluation, the first value from the list of values collected for o_123 at that poll period and the first value from the list of values collected for o_124 at that poll period will be substituted into the alert formula. For the second evaluation, the second values in the list of collected values will be substituted in to the alert formula.

If you compare a collection object to a string or an integer, be careful to use quotation marks consistently. The collection object and the comparison string or comparison integer must use the same quotation marks. Either both the collection object and the comparison value must be surrounded by single quotation marks or neither the collection object nor the comparison value should be surrounded by single quotation marks. For example:

- o_1346 == '5' is incorrect syntax (notice no single quotation marks around o_1346 but single quotation marks around 5)
- 'o_1346' == '5' is correct
- o_1346 == 5 is correct

For more details, see the section on *Using Quotes in an Alert*.

NOTE: You can use `== '--'` in your alert formula to alert when the Dynamic Application reports "NO DATA". For example:

```
o_100 == '--' or o_101 != 100
```

Operators

The ***Formula Editor*** accepts only arithmetic operators (+, -, *, /), comparison operators (>, <. !=, ==) and Boolean operators (and, or, not). Parentheses are used to group and set precedence for operators.

The comparison operators are those commonly used in many modern programming languages. It is important to note that the '==' operator is used to test equality.

The single '=' assignment operator is not supported (except when passing optional arguments to the result () function, see below).

The operators may be used in any combination and with any number of object IDs, provided that the result of the expression always equates to TRUE or FALSE.

The following examples are all valid formulas:

o_123 / 100 > 3

o_123 > 3 and o_123 < 7

o_123 == 5

o_123 == 2 or o_123 == 4

Valid examples using multiple objects:

o_123 == o_124

o_123 / o_124 > 2.5

(o_123 + o_124) > (o_125 + 6)

Dividing Integers

When you divide two integers, sometimes the result is a number that includes a decimal point. For example, if you divide 3/2, the result is 1.5. A value with a decimal point is called a floating-point number.

In Skylar One, ***only if the divisor is a float can the resulting quotient be a float***. If an integer is divided by another integer, the result is an integer. For example, if "3/2" is evaluated in an alert formula, the result will be 1, not 1.5. If you are dividing two numbers in Skylar One, you can use the python float() function to ensure that the divisor is converted to a floating-point number. This will ensure that the resulting quotient is also a floating-point number

For example, suppose your Dynamic Application includes two objects, o_123 and o_456. Suppose you want to divide the value of o_123 by the value of o_456. You should use the float() function like this:

```
o_123/float(o_456)
```

Because the divisor is a floating-point number, the resulting quotient can be a floating-point number.

Using Quotes in an Alert

Some alert functions accept either a string or a number as an argument. Some alert functions also require quotes around the arguments. When using quotes with alert functions, follow these guidelines:

- If you want to perform string operations on the object (such as find()), put the collection object ID in quotes.
- If you want to perform numeric operations on the object, do not put the collection object ID in quotes.
- You can use single quotes or double quotes – either work, but they must be paired (you cannot start with ' and end with ").
- For consistency, ScienceLogic recommends that you always use single quotes around strings in formulas.
- Take care if you are copying and pasting text from a text editor – the Python interpreter that evaluates alert formulas requires specific quote characters. These are valid quote characters: ' ", these are not: ‘ ’ “ ”.

The result() Function

The ***result()*** function passes a value for use in the alert message or event message associated with the alert. The passed value is available in the %V substitution. The expression defined in the result() function can be any combination of object IDs, numeric values, and strings allowed in the normal alert formula logic. The syntax is:

```
result(expression)
```

For example, if temperature is provided via object ID o_126, the following formula would return the temperature value for use in the alert message or the event message associated with the alert:

```
result(o_126) > 35
```

If the temperature is returned in Celsius, and you want it displayed in the alert message or event message in Fahrenheit, you could use the following formula:

```
result((o_126 * 1.8) + 32) > 120
```

The expression passed to the result() function can also refer to multiple objects. For example, suppose you wanted to determine the total number of IP DHCP addresses in a subnet. You could add together the following example objects:

- o_5678 = number of addresses in use.
- o_8910 = number of addresses free.
- o_1112 = number of addresses pending.

The result() function would look like this:

```
result(o_5678 + o_8910 + o_1112)
```

There are two optional arguments to the result function: enums and label. These provide for further descriptive text to be substituted into the alert message or event message associated with the alert. The syntax is:

```
result((expression), enums={number:'value',…}, label='label object')
```

The ***enums*** argument specifies a list of substitution characters that Skylar One should use to translate the value passed by the result() function. In general, this argument is used only if the expression refers to a collection object that is of type *Config Enum*. This argument must be contained within curly braces {} and is formatted with the value and substitution characters separated by a colon, and each element separated by commas.

For example, suppose you want to monitor a status object, o_987. Suppose this status object is of type *Config Enum* and can return integer values 3, 4, 5, or 6. These integer values map to the following states:

- 3 = OK
- 4 = nonCritical
- 5 = critical
- 6 = nonRecoverable

If collection object o_987 is passed using the result() function, the integer value of o_987 is stored in the %V variable. By using the enums argument, you can pass the string value instead:

```
result(o_987, enums = {3:'OK',4:'nonCritical', 5:'critical',
6:'nonRecoverable'})
```

The ***label*** argument passes an additional value for use in the alert message or event message associated with the alert. The passed label value is available in the %L substitution variable.

For example, suppose you have multiple temperature probes and you want to capture which temperature probe is generating the alert. If collection object o_5678 contains the name of the temperature probe, you could supply the following value in the label argument (again using a conversion from Celsius to Fahrenheit):

```
result(((1.8 * o_1234)+32), label='o_5678')
```

The value of the label argument can now be used in an event definition using the %L substitution.

NOTE: The value of the label argument must be in single quotes.

The threshold() Function

Like the ***result()*** function, the ***threshold()*** function passes a value for use in the alert message or event message associated with the alert. The passed value is available in the %T substitution variable. The syntax is:

```
threshold(expression)
```

The threshold() function is intended to make the value of a threshold that triggered an alert visible in the corresponding log message and/or event.

As with the result() function, the threshold() function can contain a combination of object IDs and numeric values.

For example, suppose you are monitoring temperature. The object o_1234 contains temperature in Celsius. The threshold t_99 contains the maximum allowable temperature. You could enter the following formula to convert the temperature values to Fahrenheit and generate an alert when the temperature reading exceeded the allowable maximum:

```
result((1.8 * o_1234)+32) > threshold((1.8 * o_t_99)+32)
```

If an alert used the result() function with the label argument and the threshold() function, the alert message might look like this:

```
%L reporting temperature of %V F, threshold is %T F
```

The values substituted in to replace %L, %V and %T are the actual values captured in the alert formula definition by the result() and threshold() functions. If the alert is generated, the alert message might look like this:

Backplane-probe reporting temperature of **130** F, threshold is **120** F

The items in bold are the substituted values.

The active() Function

The ***active()*** function checks the state of another alert in the same Dynamic Application. If the alert is active, the active() function returns the value TRUE. The active() function can be used in combination with other expressions. When used in combination with other expressions, if the entire expression resolves to TRUE, a new alert is generated. The syntax is:

```
active(alert_ID)
```

To use the active() formula to check the state of an alert, the alert being checked must have the ***Maintain State*** set to *Yes*.

For example, suppose you are monitoring a circuit for failed authentications. You could define two alerts:

The first alert is assigned ID a_175 by Skylar One. In the first alert, the object o_16060 contains the number of failed authentications. The object o_16056 contains the name of the circuit being monitored for failed authentication.

```
result(o_16060, label='o_16056') > threshold(13)
```

In this expression, a high-end threshold is defined. In this expression, if more than 13 failed authentications occur on the circuit specified in o_16056, an alert is triggered. This alert defines the problem. For this alert, the ***Maintain State*** drop-down list is set to *Yes*.

The second alert is assigned ID a_176 by Skylar One. In the second alert, we define a low-end threshold for failed authentications.

```
result(o_16060, label='o_16056') < threshold(7) and active(a_175)
```

In this expression, if less than seven failed authentications occur on the circuit specified in o_16056, AND alert a_175 is still active, a new alert is triggered. The second alert defines the "healthy" criteria.

Suppose we had defined a critical event policy based on alert a_175. This event would warn that too many failed authentications occurred on circuit o_16056.

Now suppose we wanted to clear that critical event when failed authentications fall back to an acceptable number. We could create a healthy event policy based on alert a_176. We could define this event to auto-clear the previous critical event. The new event would inform you that failed authentications were back within acceptable levels.

The avg() Function

The ***avg()*** function calculates and returns the average of all values returned by a collection object when that collection object returns a list of values. When a collection object returns a single value, the avg() function will return that single value. The syntax is:

```
avg(object_ID)
```

For example, suppose a device has three network interfaces. Now suppose the object o_1234 contains the value for "octets in".

On our example device, we would have three separate instances of object o_1234. We could use the avg () function to calculate and return the average of ALL instances of object o_1234. If that average is greater than 1,000,000 octets, an alert is triggered:

```
avg(o_1234) > 1000000
```

The deviation() Function

The ***deviation()*** function allows you to define alerts that are triggered when an object has a value that is outside the "normal" range for the current hour on the current day of the week. The syntax is:

```
deviation(object_ID)
```

The deviation() function allows you to compare the current value of a collection object to a specified number of standard deviations outside the "normal" value for the object. You can specify that you want to trigger an alert if the current value falls outside the specified number of standard deviations from "normal".

To use the deviation() function, you must configure Skylar One to store and calculate the mean values and standard deviations for a collection object. You do this by selecting the ***Enable Deviation Alerting*** field on the **Collection Objects** page. You then specify the minimum and maximum number of weeks to collect deviation data for the object.

Skylar One must have already collected at least the minimum number of weeks' worth of data for an object before Skylar One will evaluate alerts that use the deviation() function.

IMPORTANT: **To use the deviation() function, you must specify a minimum value of at least two weeks.** This value is not configured by default, and in some cases a PowerPack upgrade can overwrite this value. You are responsible for making sure that the min/max week settings are properly configured on the **Collection Objects** page with your preferred defaults.

NOTE: Make sure that the raw data retention settings for your Dynamic Application performance data being is correct, as the deviation() function is dependent on this data. The system default in most cases is 7 days, and the deviation alerting feature will require you to either increase this at the system level or go in manually to each Dynamic Application for a device and set the window to something large. For example, 30 days is ideal in cases with 4 weeks in the max window settings. ScienceLogic recommends that you do not set your maximum week value too high' in most cases, a minimum of 2 weeks and maximum of 4 weeks will be sufficient.

Skylar One evaluates the deviation() function against the aggregated values collected for multiple weeks. The number of weeks used during the evaluation depends on the amount of data available and is at least the minimum number of weeks specified for the collection object.

After Skylar One has collected the minimum number of weeks' worth of data for an object, Skylar One calculates the mean value for that object at every hour of every day and then calculates the standard deviations from that mean value.

The deviation function uses the following formula to examine the value of an object:

```
(current value of a collection object -mean value for current hour on
current day of week) / (standard deviation for current hour at current day
of week)
```

The deviation() function converts the value of (*current value of a collection object -mean value for current hour on current day of week*) to a positive value. Therefore, you should always compare the results of the deviation function to zero ("0") or to a positive number.

The actual substitution will come out as "False and 0" in cases where:

- There is not enough data, such as when you only have 7 days of data and you need 2 weeks (14 days).
- Nothing deviated.

"False and 0" might cause some confusion, but it was a strategic way to get an entire deviation formula to return false."

The deviation() function sends data to the collector in the form of upserts to calculate the standard deviation. In cases where a collector failover might happen, or if a device moves collector groups, the standard deviation data is re-upserted on the new collector.

Also, when enabling deviation alerting on a collector, if you set the raw data retention threshold and the standard deviation minimum weeks alerting window, any time that the deviation alerting window is higher than the raw data retention threshold, deviation alerting will not be available after a collector failover. Deviation alerting will be available only when Skylar One has gathered enough new raw data to meet the minimum weeks specified by standard deviation.

Some possible uses for the deviation function are:

- Determining if an application is functioning properly. For example, if a log file for an application begins to grow at a rate outside the "normal" range, you can trigger an alert to determine if there is a problem with the application.
- Monitoring security. For example, if bandwidth usage exceeds the normal activity, you can trigger an alert that indicates that your network might have been compromised.

Example 1

For example:

- Suppose Skylar One has already collected four weeks' worth of data for an object (o_123).
- Suppose that for the last four weeks, every Monday between 09:00 and 10:00, the mean value for o_123 is "50".
- Suppose that 68% of all values on Monday between 09:00 and 10:00 fall within +/- "10" of the mean value (that is, between 40 and 60).
- For o_123 on Mondays between 09:00 and 10:00, a standard deviation of "1" is +/- "10".

Suppose we specify the following alert formula:

```
deviation(o_123) > 1
```

This alert formula specifies that if o_123 collected a value outside "1" standard deviation of the "normal" value, Skylar One should trigger an alert.

If on Monday at 09:30, the value of o_123 is "45", the deviation function performs the following calculation:

```
(45-50)/10 > 1
```

This evaluates to FALSE, therefore Skylar One would not trigger an alert.

Example 2

For another example:

- Suppose Skylar One has already collected four weeks' worth of data for a collection object (o_789).
- Suppose that on Monday, between 09:00 and 10:00, the value of o_789 is always "50" and has not varied ***at all during the entire four weeks***.
- The mean value of o_789 on Mondays, between 09:00 and 10:00 is "50".
- 68% of all values on Mondays between 09:00 and 10:00 will fall within +/- "0" of the mean.
- For o_789 on Monday between 09:00 and 10:00, a standard deviation of "1" is +/- "0".

Suppose on Monday at 09:15, the value of o_789 is "50". If this is the expected behavior, we could specify:

```
deviation(o_789) >1
```

The deviation function performs the following calculation:

```
(50-50)/0 > 1
```

This evaluates to FALSE, therefore Skylar One would not trigger an alert.

Now suppose that on Monday at 09:30, the value of o_789 is "45", the deviation function performs the following calculation:

```
(45-50)/0 > 1
```

In this case, the deviation function would return "infinity". The formula will evaluate to TRUE, and trigger an alert. If we did not expect to repeatedly collect the same value every Monday between 09:00 and 10:00, this formula might be appropriate to monitor object o_789.

NOTE: In the alert editor, you can specify infinity as float("inf")

If you do expect the first standard deviation to be "+/- 0", that is, if you expect the value of an object to never fluctuate, you can test for this with the following:

```
deviation(o_789) == float("inf")
```

Because (45-50)/0 does equal infinity, this will evaluate to TRUE and trigger an alert. We can define this alert to provide details about a standard deviation of “0”.

The 68-95-99.7 rule

Statistically, 99.7% of all values lie within three standard deviations from the mean.

In theory, if an alert formula uses a standard deviation of "1" as the threshold, such as the one in Example 1, the alert will trigger for 32% of the collected values.

Using the 68-95-99.7 rule, the following list indicates the percentage of collected values that will trigger an alert for different standard deviation thresholds:

- ***1 (one) standard deviation.*** **32%** of collected values will fall outside this range. Therefore, an alert that compares an object's value to a value of :1 (one) standard deviation will be triggered 32% of the time.

  For example:

  ```
  deviation(o_123)> 1
  ```

  will trigger an alert on 32% of the values of o_123.

- ***2 (two) standard deviations***. **5%** of collected values will fall outside this range.
- ***3 (three) standard deviations***. **0.3%** of collected values will fall outside this range.

The find() Function

The ***find()*** function searches a collection object for a specific alpha-numeric string. The syntax is:

```
object_ID.find('alpha-numeric string')
```

If the collection object contains the sting, the find() function returns the location of the string in the collection object, where 0 is the first character of the collection object. If the collection object does not contain the string, the find() function returns -1 (negative one).

For example, suppose you want to trigger an alert if inbound bandwidth-usage drops below a certain level. Suppose collection object o_4567 collects "inbound octets". Suppose collection object o_4568 returns the type of interface. Now suppose you don't want to trigger an alert if this condition occurs on a loopback interface. You could write an alert like this:

```
result(o_4567) < 500 and o_4568.find('loopback') = -1
```

This logic says "if object o_4557 is less than 500 and object o_4558 does not contain the text "loopback" (that is, the find function cannot find the string "loopback", so returns "-1"), trigger an alert.

The global() Function

The ***global( )*** function returns the collected value (or, for list objects that are not collected at each collection, the last collected value) of a collection object regardless of the group or index that is currently being evaluated for the alert formula.

The syntax is:

```
global(object_ID)
```

When you include multiple collection objects in an alert, those collection objects must be in the same group. If a list or table of values is returned by a collection object, the platform evaluates the alert for each index in that group.

You can use the global( ) function to include a collection object that returns a single value in an alert formula that also includes collection objects that return lists.

For example:

Suppose that you have a group, ***Group 1***, that contains the following two collection objects:

- ***o_123***. Contains file system utilization in percent.
- ***o_124***. Contains name of file system.

Suppose that you have an additional collection object that is not aligned with any group:

- ***o_125***. Contains a single value - the file system that is used for data storage for the primary application.

Suppose you want to trigger an event when a file system is over 80% utilized:

- Suppose that you want to trigger a Major event for all file systems except the one used for data storage for the primary application.
- Suppose you want to trigger a Critical event for the file system used for data storage for the primary application.

To do this, you can create two alerts:

- An alert for the file system for data storage for the primary application
- An alert for all other file systems.

In the alert, the object ***o_125*** will be compared to ***o_124*** to determine whether this is the file system for data storage for the primary application.

Because ***o_125*** is a single value in a different group/index than the other collection objects, we use the **global()** function to return the value for ***o_125***.

The two alerts would look like this:

```
o_123 > 80 and 'o_124'.find('global(o_125)') > -1
```

```
o_123 > 80 and 'o_124'.find('global(o_125)') = -1
```

The first alert will trigger only for the file system for data storage (file system name is the same name as contained in o_125).

The second alert will trigger for all other file systems (file system name is not the same as in o_125).

NOTE: If the collection object specified in the **global()** function is a string, the Skylar One system will automatically substitute the value surrounded by single quote characters.

The log() Function

The ***log()*** function calculates and returns the logarithm of a specified number to the specified base. If base is not specified, the log function uses base *e* to return the natural logarithm. The syntax is:

```
log(number, base)
```

For example:

```
log(o_123, 10)
```

This example would calculate the base-10 logarithm of the value of collection object o_123.

The prior() Function

The ***prior()*** function returns the previous value of a collection object (from the previous polling session). The syntax is:

```
prior(object_ID)
```

For example:

```
o_123 > prior(o_123)
```

This example evaluates to true if the value of object o_123 is now greater than it was during the last polling session.

Suppose you have defined a polling frequency of five minutes for a Dynamic Application. Every five minutes, Skylar One will retrieve the value from object o_123. The example above could be used to trigger an alert if the value of object o_123 is now greater than it was five minutes ago.

The round() Function

The ***round()*** function rounds the value of a collection object to a specified number of digits. The ***round()*** function returns a floating point value.

The syntax is

```
round(object ID, number_of_digits)
```

- The ***round()*** function rounds values to the closest multiple of 10 to the negative n.
    - round (0.5) returns 1.0
    - round (-0.5) returns -1.0

- If you omit the optional *number_of_digits* argument, the function defaults to a whole number and ".0:, like "2.0"

For example, suppose object ***o_567*** contains the value "4.688".

```
round(o_567, 2)
```

would return the value

```
4.67
```

The sum() Function

The ***sum()*** function calculates and returns the sum of all values returned by a collection object when that collection object returns a list of values. When a collection object returns a single value, the sum() function will return that single value. The syntax is:

```
sum(object_ID)
```

For example, suppose a device has three network interfaces. Now suppose the object o_1234 contains the value for "octets in".

On our example device, we would have three separate instances of object o_1234. We could use the sum () function to calculate and return the sum of ALL instances of object o_1234. If that sum is greater than 1,000,000 octets, an alert is triggered:

```
sum(o_1234) > 1000000
```

Date & Time Variables

You can use the time and date variables to define alerts that occur only during specified dates or times.

Values starting with the word “local” refer to the local date and time, specified in ***System Timezone*** field in the **Behavior Settings** page (System > Settings Behavior).

For every “local” value there is a “utc” equivalent that uses values based on UTC.

The following is a list of time and date variables that can be used in alerts:

- ***localyear***. The year, in local date and time, as specified in the ***System Timezone*** field in the **Behavior Settings** page. Values can be any four-digit year. For example, 2007.
- ***localmonth***. The month, in local date and time, as specified in the ***System Timezone*** field in the **Behavior Settings** page. Values can be 01 – 12, with 01 being January and 12 being December.
- ***localmonthday***. The day of the month, in local date and time, as specified in the ***System Timezone*** field in the **Behavior Settings** page. Value can be 01 – 31.
- ***localhour***. The hour, in local date and time, as specified in the ***System Timezone*** field in the **Behavior Settings** page. Values can be 0 – 23, with 0 being midnight and 23 being 11PM.
- ***localminute***. The minutes, in local date and time, as specified in the ***System Timezone*** field in the **Behavior Settings** page. Values can be 0 -59, with 0 being 0 minutes after the hour and 59 being 59 minutes after the hour.

- ***localsecond***. The seconds, in local date and time, as specified in the ***System Timezone*** field in the **Behavior Settings** page. Values can be 0 – 61. 60 is used for leap seconds; 61 is used for double leap-seconds.
- ***localweekday***. Day of the week, in local date and time, as specified in the ***System Timezone*** field in the **Behavior Settings** page. Values can be 0 – 6, with 0 being Monday and 6 being Sunday.
- ***localyearday***. Day of the year, in local date and time, as specified in the ***System Timezone*** field in the B**ehavior Settings** page. Values can be 1 – 366.
- ***utcyear***. The year, in UTC date and time. Values can be any four-digit year. For example, 2007.
- ***utcmonth***. The month, in UTC date and time. Values can be 01 – 12, with 01 being January and 12 being December.
- ***utcmonthday***. The day of the month, in UTC date and time. Value can be 01 – 31.
- ***utchour***. The hour, in UTC date and time. Values can be 0 – 23, with 0 being midnight and 23 being 11 PM.
- ***utcminute***. The minutes, in UTC date and time. Values can be 0 -59, with 0 being 0 minutes after the hour and 59 being 59 minutes after the hour.
- ***utcsecond***. The seconds, in UTC date and time. Values can be 0 – 61. 60 is used for leap seconds; 61 is used for double leap-seconds.
- ***utcweekday***. Day of the week, in UTC date and time. Values can be 0 – 6, with 0 being Monday and 6 being Sunday.
- ***utcyearday***. Day of the year, in UTC date and time. Values can be 1 – 366.

Example 1

For example, you could define an alert that is triggered only when both the following are TRUE:

- the value of o_999 falls to zero
- the condition occurs during business hours (between 8AM and 6 PM)

To define the alert, you could use the localhour variable:

```
result(o_999) == 0 and localhour >= 8 and localhour <= 18
```

Example 2

You could also define an alert that excludes weekend days. The alert would be triggered only when both the following are TRUE:

- the value of o_999 falls to zero
- the condition occurs during weekdays

To define the alert, you could use the localweekday variable. The days of the week are numbered from 0 to 6, with Saturday being 5 and Sunday being 6. To exclude weekends you could define the alert to trigger only when the “localweekday” number is less than 5

```
result(o_999) == 0 and localweekday < 5
```

Example 3

You could also define an alert that is triggered only on the first day of the month. The alert would be triggered only when both the following are TRUE:

- the value of o_999 falls to zero
- the condition occurs when the day of the month is “1”

To define the alert, you could use the localmonthday variable:

```
result(o_999) == 0 and localmonthday == 1
```

The tab_idx Variable

The ***tab_idx*** variable allows you to apply an alert only to specific indexes when a collection object returns a list of values. For SNMP Dynamic Applications, the ***tab_idx*** variable returns the SNMP index that corresponds to the collection object values that are currently being evaluated. For other Dynamic Applications, the ***tab_idx*** variable returns the location in the list of values that corresponds to the collection object values that are currently being evaluated.

For example:

- Suppose that in a list of values representing device state, there may be six potential indexes in the list of values, but only 1, 3, 4, and 6 are used.
- Suppose that the object o_999 represents the device state, with 0 being healthy.

You could define the alert to check when object o_999 does not equal zero (is not healthy). But you could limit the alert to check only indexes 1, 3, 4, and 6. For an SNMP Configuration or SNMP Performance Dynamic Application, the alert formula would be:

```
o_999 != 0 and tab_idx in ['.1','.3','.4','.6']
```

For all other types of Dynamic Applications, the alert formula would be:

```
o_999 != 0 and tab_idx in ['1','3','4','6']
```

Notice the syntax for the tab_idx variable. You must use the syntax as it appears in the example:

- Include the text "in" after the tab_idx variable
- Surround the list of index numbers with square brackets
- Surround each index with single-quotation marks (')
- For SNMP Configuration and SNMP Performance Dynamic Applications, precede each index with a period (.)
- Separate the list of index numbers with commas (,)

Creating an Event Policy for an Alert

When Skylar One generates an alert for a device, the log message associated with that alert is inserted into the device logs. Optionally, you can associate an alert with an event policy. When Skylar One generates an alert that is associated with an event policy and all the conditions defined in the event policy are met, Skylar One will generate an event. In addition to appearing in the device logs, the event will appear:

- On the **Events** page, as well as on the **Event Console** page (Events > Classic Events, or the Events tab in the classic user interface).
- On the **Events** panel of the **Device Investigator**.
- On the **[Events]** tab of the **Device Investigator**.
- In the **[Summary]** tab in the **Device Reports** panel.
- In the **[Events]** tab in the **Device Reports** panel.

To create an event policy for an alert:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the Dynamic Application you want to add an event policy for. Click its wrench icon ( ). The **Dynamic Applications Properties Editor** page is displayed.
3. Click the **[Alerts]** tab. The **Dynamic Applications Alert Objects** page is displayed.
4. In the **Dynamic Applications Alert Objects** page, find the alert in the **Alert Object Registry** pane. Click its event icon ( ).
5. The **Event Policy Editor** page is displayed. The following fields in the **Event Policy Editor** are populated with information about the alert
    - ***Event Source***. This field is set to *Dynamic*, which tells Skylar One that this event policy matches alerts generated using Dynamic Applications.
    - ***Policy Name***. This field is populated with the value you specified in the ***Policy Name*** field for the alert.
    - ***Link-Alert***. This field appears on the **[Advanced]** tab in the **Event Policy Editor**. This field is populated with a link to the alert, which tells Skylar One that the alert you selected can trigger this event policy.
6. Before you save the event policy, you must supply a value in the following fields:
    - ***Operational State***. Specifies whether Skylar One should trigger this event if the associated alert is generated and all the conditions defined in the event policy are met. Choices are:
        - *Enabled*. Skylar One will trigger this event if the associated alert is generated and all the conditions defined in the event policy are met.
        - *Disabled*. Skylar One will never trigger this event. Selecting this option does not stop Skylar One from generating the associated alert.

- ***Event Severity***. The severity value to associate with events created with this event policy. Choices are:
    - *Healthy*
    - *Notice*
    - *Minor*
    - *Major*
    - *Critical*
- ***Event Message***. The event message that Skylar One will associate with instances of this event. If you want to use the message generated by the alert that triggers this event, enter "%M" in this field.

7. You can optionally specify values in the additional fields, including the ***Policy Description*** field and the fields that appear in the **[Advanced]** tab. For more information about the **Event Policy Editor**, see the ***Events*** manual. If you choose to make further changes to your event policy, you must not change the values in the following fields:
    - The ***Event Source*** field in the **[Policy]** tab.
    - The ***Link-Alert*** field in the **[Advanced]** tab.
8. Click the **[Save]** button to save the event policy.

Editing an Alert

To edit an already-defined alert:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. In the **Dynamic Applications Manager** page, find the Dynamic Application for which you want to edit an alert. Click its wrench icon ( ).
3. Click the **[Alerts]** tab for the Dynamic Application.
4. In the **Dynamic Applications Alert Objects** page, find the alert in the **Alert Object Registry** pane. Click its wrench icon ( ).
5. The fields in the top pane are populated with values from the selected alert. You can edit the value of one or more fields. For a description of each field, see the *Creating an Alert* section.
6. Click the **[Save]** button to save your changes to the alert.

Deleting an Alert

You can delete an alert from a Dynamic Application. To do this:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. In the **Dynamic Applications Manager** page, find the Dynamic Application for which you want to delete an alert. Click its wrench icon ( ).
3. Click the **[Alerts]** tab for the Dynamic Application.

4. In the **Dynamic Applications Alert Objects** page, find the alert in the **Alert Object Registry** pane. Click its delete icon (). The alert will be deleted from Skylar One. If an event policy was associated with the alert, you must manually delete the event in the **Event Policy Manager** page (Registry > Events > Event Manager).

Validating an Alert

After you have created an alert formula, you can use a command line interface (CLI) tool to validate that the alert works as intended. This tool enables Dynamic Application developers to validate alerts by directly inserting data into the Alerting module without using a full Skylar One System.

Running the Alert Validator Command Line Tool

The ***alert_validator.py*** Python CLI tool enables Dynamic Application developers to validate the alert formulas that are included in Dynamic Applications using shell sessions at the command line. This tool is located in the following folder:

```
/opt/em7/backend/alert_validator.py
```

NOTE: The alert_validator.py tool can be run only from a Data Collector or All-In-One Appliance.

The alert_validator.py tool checks an alert formula to validate that the following are all true:

- All collection object and device threshold object references are structured correctly and are valid for the specified Dynamic Application.
- Any collection objects that are referenced outside of global(), sum(), and avg() all belong to the same group of metrics.
- The following functions are structured correctly and contain a valid collection object reference:
    - sum
    - avg
    - global
    - prior
    - changed
    - deviation
- The active() function alert reference is structured correctly and is valid for the specified Dynamic Application, with an alert formula that belongs to the same group as the alert being tested.
- The result() and threshold() functions are used correctly (including optional result arguments).
- The alert formula contains no syntax errors.

To run the alert_validator.py tool, you must first run a command to create JSON files that capture data about the Dynamic Application. This command uses the following structure, and ***must*** always end with the device ID and Dynamic Application ID:

```
sudo -u s-em7-core python /opt/em7/backend/alert_validator.py --capture-data [Device ID Dynamic Application ID]
```

NOTE: Replace `[Device ID Dynamic Application ID]` with the Device ID and Dynamic Application ID, respectively. Do not include the brackets. The Device ID and Dynamic Application ID should be separated by a space and no other characters.

When you run this command, a folder containing all of the generated JSON files is created in the /tmp/ folder. The folder name includes the Dynamic Application ID.

After the folder with the JSON files has been created, you can then run a separate command to evaluate the alert formulas. This command uses the following structure:

```
sudo -u s-em7-core python /opt/em7/backend/alert_validator.py -j AlertInternalState.json -j DynamicAppAlert_[Alert ID].json -j DynamicAppThreshold_[Threshold ID].json -j DynamicAppObjects_[Object ID].json -j DynamicAppObjectSchema_[Dynamic Application ID].json
```

NOTE: Replace `[Alert ID]`, `[Threshold ID]`, `[Object ID]`, and `Dynamic Application ID` with the Alert ID, Threshold ID, Object ID, and Dynamic Application ID, respectively. Do not include the brackets.

TIP: ScienceLogic recommends running the command from the folder in which the JSON files are stored so you do not need to include the path for each JSON file.

If the alert validation is successful, a detailed report will display indicating that the specified alerts were successfully triggered.

If the alert validation fails, a message will display that specifies the reason why it failed.

Example

The following example creates JSON files to capture data for device ID 3, Dynamic Application ID 1429:

```
sudo -u s-em7-core python /opt/em7/backend/alert_validator.py --capture-data 3 1429
```

After the JSON files have been created, you can then use them to evaluate the alert formulas. For example:

```
sudo -u s-em7-core python /opt/em7/backend/alert_validator.py -j
AlertInternalState.json -j DynamicAppObjects_3.json -j
DynamicAppThreshold_651.json -j DynamicAppAlert_2111.json -j
DynamicAppObjectSchema_1429.json
```

Indexing

This section describes describes how Dynamic Applications handle and organize data when a single collection object collects multiple instances or values.Dynamic Application indexing determines how Skylar One creates unique identifiers for each instance, how those instances are displayed in configuration tables and performance graphs, and how they are referenced in alert formulas.

This section covers the default indexing methods, explains how instance value creation differs across protocols (SNMP, Snippet, and others), describes how to designate a custom index, and explains the system settings that affect indexing behavior.

What is Dynamic Application Indexing?

When Skylar One performs collection for a collection object, the response might contain a list of values for that collection object instead of a single value.

For example, suppose a collection object collects the amount of space used in a file system. For that single collection object, a device might respond with a list of values. Each value in the list is the amount of space used for each of the file systems on that device.

When a collection object returns multiple values, Skylar One uses ***Indexing*** to track each of the values. Each of the values in the list is given an ***index***, which is a numerical value that allows Skylar One to:

- ***Maintain associations between collection objects in the same Dynamic Application.*** For example, suppose a collection object collects the amount of space used in a file system and a device responds with multiple values. Suppose a second collection object returns the names of those file systems. The lists of values for each collection object should be assigned the same index, so that the values for space used are associated with the correct file system name in graphs and reports. The first value in the list of amount of space used values is associated with the first value in the list of names, etc.
- ***Maintain an association between values collected at different times for the same collection object***. Every time Skylar One collects a list of values for a collection object, each value that is associated with a value in the previously collected list should have the same index value as the previous value. For example, suppose a collection object collects the amount of space used in a file system and a device responds with multiple values. Suppose that during the first collection, the amount of space used for the file system called "C:\" is assigned index "1". The next time Skylar One collects the list of values, the amount of space used for the file system called "C:\" should again be associated with index "1". When Skylar One generates a graph of data, each line on the graph is drawn using values with the same index.

Index values are unique and maintain association only for each group of collection objects in each Dynamic Application for each device. For example:

- In a single Dynamic Application, a collection object in group 1 can have the same index values as a collection object in group 2, but the collected values for these collection objects are not associated with each other.
- In a single Dynamic Application, for each group of collection objects, Skylar One maintains separate sets of indexes for each aligned device. A collection object in group 1 might have different index values for device A and for device B.

Skylar One has a default method to assign index values. For some Dynamic Applications, the default method is not sufficient and might create incorrect associations between values. Typically, this is caused when the list of values for a collection object changes order from one collection to the next. If the default indexing method is not sufficient, you can designate a collection object that will control the indexing behavior.

NOTE: Index values are internal to Skylar One and are typically not displayed to users. Skylar One displays index values as the labels for performance graphs only when collected labels are not available.

How Indexing Affects Configuration Tables

For each group of collection objects defined in a configuration Dynamic Application, Skylar One displays a table in the configuration report for that Dynamic Application:

- Each collection object is assigned its own column in the table.
- Each index is assigned its own row in the table.

If a given collection object does not have a collected value for an index, an empty field will be displayed for that row and column.

How Indexing Affects Performance Graphs

For each presentation object defined in a Dynamic Application, Skylar One generates a graph that includes one line for each index.

For example, suppose a presentation formula in a Dynamic Application includes a single collection object:

```
o_123
```

Suppose that when this Dynamic Application is aligned to a device, Skylar One collects a list of three values for o_123.

- The graph for this presentation object will display three lines, one for each index.
- When new data is collected, the new poll time will be included on the x-axis of the graph.
- For each of the three lines, the data point (y-axis) for the new poll time will be the collected value for that index.

For example, suppose Skylar One assigns indexes 0, 1, and 2 to the lists of collected values for o_123. Suppose Skylar One then performs three polls at 00:00, 00:05, and 00:10. Suppose the indexed collected values look like this:

Index	00:00	00:05	00:10
0	6	8	7
1	15	30	20
2	50	60	55

When Skylar One generates the performance graph:

- One graph line will include the values 6, 8, and 7.
- One graph line will include the values 15, 30, and 20.
- One graph line will include the values 50, 60, and 55.

Now suppose that during the 00:15 poll, a list of three values is returned for the collection object, but the values are assigned the indexes 1, 2, and 3:

Index	00:00	00:05	00:10	00:15
0	6	8	7	-
1	15	30	20	9
2	50	60	55	25
3	-	-	-	70

The graph will now include four lines:

- One graph line will include the values 6, 8, and 7. This line will not include a data point for the 00:15 poll.
- One graph line will include the values 15, 30, 20, and 9.
- One graph line will include the values 50, 60, 55, and 25.
- One graph line will not include data points for the 00:00, 00:05, and 00:10 polls, and will include the value 70 for the 00:15 poll.

For presentation formulas that contain multiple collection objects, Skylar One evaluates the formula for the values at each index. A graph line is drawn for each index for which there are enough collected values to calculate a result. For example, suppose a presentation object has the following formula:

```
o_123 + o_321
```

If Skylar One collects the following lists of values for o_123 and o_321 and assigns the following index values:

Index	o_123	o_321
0	100	-
1	1000	750
2	100	30
3	-	80

Skylar One will evaluate the presentation formula for each of the four indexes. However, there are only enough collected values to calculate a result for two indexes (1 and 2). Therefore, two graph lines will appear on the graph.

Suppose the presentation formula was changed to make o_123 optional:

```
o_123 + {o_321}
```

For the same set of collected values, the graph would include three graph lines, which would correspond to indexes 0, 1, and 2. Index 3 would not be included, because object o_123 is not optional and does not include a value at index 3.

The indexes assigned to collected values are also used to label the lines in the graph key that appears in the lower right of the **Performance** page. When a graph line is drawn for an index, Skylar One looks for a collected value that matches the following criteria:

- Was collected for a collection object in the Dynamic Application that has a ***Class Type*** of *101 Label (Hourly Polled)* or *104 Label (Always Polled)*.
- Has the same group value as the collection objects that were used to draw the graph line.
- Has the same index value as the values used to draw the graph line.

If Skylar One finds a collected value that matches the criteria, that value is displayed in the graph key for that graph line. If Skylar One does not find a collected value that matches the criteria, the graph line is labeled "Index X", where X is the assigned index for the values used to generate the graph line.

NOTE: The poll time for the collected value is not considered when Skylar One chooses a label for a graph line. For collection objects that have a ***Class Type*** of *101 Label (Hourly Polled)* or *104 Label (Always Polled)*, Skylar One stores only the last collected value.

How Indexing Affects Alerts

When Skylar One evaluates an alert, the formula is evaluated separately for each index assigned to the collected values.

For example, suppose an alert formula in a Dynamic Application uses one collection object:

```
o_123 > 0
```

Suppose that when this Dynamic Application is aligned to a device, Skylar One collects a list of three values for o_123. Skylar One will evaluate the alert formula three times, once for each collected value.

Suppose another alert formula uses two collection objects:

```
(o_123 / 100) * o_321 > 90
```

Suppose Skylar One collects the following lists of values for o_123 and o_321 and assigns the following index values:

Index	o_123	o_321
0	100	80
1	1000	750
2	100	30

During each poll period, Skylar One will evaluate the alert formula three times, once for each pair of values for each index.

Now suppose that for a different device, Skylar One collects a list of three values for each collection object, but the indexes assigned to the collected values do not match:

Index	o_123	o_321
0	200	-
1	300	160
2	100	30
3	-	80

For this set of collected values, the alert formula will be evaluated four times, once for each index assigned to the collected values. However, the evaluation for indexes 0 and 3 will fail because not all the collection objects have a value for that index.

The Default Indexing Method

For each group of collection objects in each Dynamic Application associated with a single device, Skylar One maintains a list of index values. When a list of values is collected for a collection object, Skylar One uses the following information about each value in the list to determine the index for that value:

- The ID for the device the data was collected from.
- The ID for the Dynamic Application the data was collected using.

- The ID for the group the collection object is included in.
- Where in the list of values the value appears, called the ***instance***.

Each index in the list of indexes maintained by Skylar One has the same four attributes (device ID, Dynamic Application ID, group ID, and instance). During collection, Skylar One attempts to match each value in the list of collected values with an index in the list of indexes. If a match is found, i.e. the device ID, Dynamic Application ID, group ID, and instance are all identical, Skylar One associates the matching index with the collected value. If a match is not found, Skylar One creates a new record in the list of indexes for that device ID, Dynamic Application ID, group ID, and instance and assigns a new index value.

Instance Values and Index Creation for SNMP Dynamic Applications

For collection objects in Dynamic Applications that use the SNMP protocol, the ***instance*** of a value in a list of collected values is the SNMP index associated with the value. For example, suppose the SNMP OID used in a collection object is ".1.3.6.1.2.1.25.2.3.1.6". This OID represents the amount of storage space used (as reported by the Host Resources MIB). Suppose a device responds to this OID in the following way:

```
HOST-RESOURCES-MIB::hrStorageUsed.1 = INTEGER: 6097420

HOST-RESOURCES-MIB::hrStorageUsed.3 = INTEGER: 6097420

HOST-RESOURCES-MIB::hrStorageUsed.6 = INTEGER: 472112

HOST-RESOURCES-MIB::hrStorageUsed.7 = INTEGER: 1984116
```

The SNMP indexes for the returned values are ".1", ".3", ".6", and ".7". These SNMP indexes are used as the instance for each value when Skylar One assigns an index.

If Skylar One needs to create a new index record for an SNMP collection object, that is, the device ID, Dynamic Application ID, group ID, and instance do not match an existing index record, Skylar One will first attempt to use the numeric equivalent of the instance as the index. In the example above, Skylar One will attempt to create indexes 1, 3, 6, and 7.

If Skylar One tries to add an index record and determines that the index already exists for the same device ID, Dynamic Application ID, and group ID, Skylar One will create the new record with an index equal to one more than the current highest index value for that device ID, Dynamic Application ID, and group ID.

In the example above, suppose that Skylar One has already created index records for an instance other than ".1", ".3", ".6", and ".7" for the same device ID, Dynamic Application ID, and group ID. Suppose that it is the only instance that has an existing index record. Suppose that instance has been assigned index 7. When Skylar One attempts to create index records for the four instances in the example above, index records for instances ".1", ".3", and ".6" are successfully created with indexes 1, 3, and 6. Because index 7 is already taken for this device ID, Dynamic Application ID, and group ID, the index record for instance ".7" is created with index 8, which is one more than the current highest index value (7).

Instance Values and Index Creation for Snippet Dynamic Applications

For collection objects in Snippet Dynamic Applications, the ***instance*** of a value in a list of collected values is defined by the snippet code that collects that collection object. Collection objects are passed back to

Skylar One by snippet code in a tuple. The first value in the tuple is the ***instance***, the second value in the tuple is the collected value.

Skylar One creates index records for Snippet Dynamic Application collection objects with a method similar to that used for SNMP Dynamic Applications:

- If Skylar One needs to create a new index record for a snippet collection object, Skylar One will first attempt to use the value of the instance as the value of the index.
- If Skylar One tries to add a record for an index value that already exists for the same device ID, Dynamic Application ID, and group ID, Skylar One will create the new record with an index equal to one more than the current highest index value for that device ID, Dynamic Application ID, and group ID.

Instance Values and Index Creation for Other Dynamic Applications

For collection objects in Dynamic Applications that use a protocol of Database, SOAP, WMI, XML, and XSLT, the ***instance*** of a value in a list of collected values is the location at which that value appears in the response. The first value in the list is instance "0", the second value in the list is instance "1", and so on.

For example, suppose a collection object for a Database Dynamic Application runs a query that selects the column "speed" from a database table. Suppose the following list of data is returned:

speed
1000
500
1200
700

The instance values would be:

speed	instance
1000	0
500	1
1200	2
700	3

To create a new index record for a Database, SOAP, WMI, XML, or XSLT collection object, Skylar One will create the new record with an index equal to one more than the current highest index value for that device ID, Dynamic Application ID, and group ID. The first created index for a device ID, Dynamic Application ID, and group ID is 0.

Designating an Index

In cases where the order or size of the list of values for a collection object might change, the default method of assigning indexes is not sufficient to maintain the associations between collected values.

For example, consider the initial instance values that were assigned to the list of "speed" values in the previous example:

speed	instance
1000	0
500	1
1200	2
700	3

Now suppose that the list of values changes order during the next poll:

speed
1000
500
700
1200

The "700" and "1200" values are now matched to different instance values:

speed	instance
1000	0
500	1
700	2
1200	3

This means that when Skylar One stores the new values "700" and "1200", they are associated with different index values than the previously stored data. If these values are used in a performance graph, the new value of "700" will appear in the graph line for index 2 (which has "1200" as the previous data points), and the new value of "1200" will appear in the graph line for index 3 (which has "700" as the previous data points). If these values are used in a configuration table, Skylar One will detect a change from "700" to "1200" for index 2 and a change from "1200" to "700" for index 3.

When the default method of assigning indexes is not sufficient, you can designate a collection object in a group of collection objects as the index. When you designate a collection object as an index, Skylar One still uses the same information about each value in the list to determine the index for that value:

- The ID for the device the data was collected from.
- The ID for the Dynamic Application the data was collected using.
- The ID for the group the collection object is included in.
- The ***instance*** associated with the value.

When you designate a collection object as an index, Skylar One uses the value and location of the "index" collection object to define the ***instance***. The collected values for the "index" collection object are recorded

as the instances. For the collected values for the other collection objects in the same group, the instance is the collected value for the "index" collection object that appears at the same location in the list for that poll period.

For example, suppose that a group contains two collection objects:

- A collection object that collects the storage space used as reported by the Host Resources MIB (SNMP OID .1.3.6.1.2.1.25.2.3.1.6)
- A collection object that collects the storage description as reported by the Host Resources MIB (SNMP OID .1.3.6.1.2.1.25.2.3.1.3).

Suppose that the collection object that collects the storage description is designated as the index for the group. Suppose the first time this Dynamic Application collects data from "Device A", the device responds to the storage description OID in the following way:

```
HOST-RESOURCES-MIB::hrStorageDescr.1 = STRING: C:\

HOST-RESOURCES-MIB::hrStorageDescr.3 = STRING: D:\

HOST-RESOURCES-MIB::hrStorageDescr.6 = STRING: Virtual Memory

HOST-RESOURCES-MIB::hrStorageDescr.7 = STRING: Physical Memory
```

For this group of collection objects in this Dynamic Application for this device, Skylar One will create the following record of instances to internal indexes:

instance	index
C:\	1
D:\	3
Virtual Memory	6
Physical Memory	7

NOTE: Skylar One will still attempt to use the numeric equivalent of the SNMP index as the index.

For the same first poll of the device, suppose the device responds to the storage used OID in the following way:

```
HOST-RESOURCES-MIB::hrStorageUsed.1 = INTEGER: 6097420

HOST-RESOURCES-MIB::hrStorageUsed.3 = INTEGER: 0

HOST-RESOURCES-MIB::hrStorageUsed.6 = INTEGER: 24863

HOST-RESOURCES-MIB::hrStorageUsed.7 = INTEGER: 27257
```

To determine the internal index to associate with the value at SNMP index ".1", 6097420:

1. Skylar One determines the value for the designated index (the storage description) at the same location (SNMP index ".1"). This value is "C:\".
2. Skylar One uses "C:\" as the instance value to match against the record of internal indexes for this group, Dynamic Application, and device.
3. The instance value of "C:\" matches to internal index 1. When Skylar One stores the value "6097420", Skylar One associates the value with index 1.

Skylar One repeats this process for the other values returned for the storage used OID. For the first poll of the device, the following table shows the values returned for the storage used OID, the instance value Skylar One used, and the internal index that is associated with the stored values:

collected value	instance	index
6097420	C:\	1
0	D:\	3
24863	Virtual Memory	6
27257	Physical Memory	7

Every time Skylar One uses this Dynamic Application to collect data from the device, Skylar One determines the instance values using this method. Suppose that during the next poll, the device responds to the storage description OID in the following way:

```
HOST-RESOURCES-MIB::hrStorageDescr.1 = STRING: C:\

HOST-RESOURCES-MIB::hrStorageDescr.6 = STRING: Virtual Memory

HOST-RESOURCES-MIB::hrStorageDescr.7 = STRING: Physical Memory

HOST-RESOURCES-MIB::hrStorageDescr.8 = STRING: D:\
```

And responds to the storage used OID in the following way:

```
HOST-RESOURCES-MIB::hrStorageUsed.1 = INTEGER: 6097420

HOST-RESOURCES-MIB::hrStorageUsed.6 = INTEGER: 24863

HOST-RESOURCES-MIB::hrStorageUsed.7 = INTEGER: 27257

HOST-RESOURCES-MIB::hrStorageUsed.8 = INTEGER: 0
```

Notice that the values for the "D:\" drive have moved from SNMP index ".3" to SNMP index ".8".

To determine the internal index to associate with the values at SNMP index ".8":

1. Skylar One determines the value for the designated index (the storage description) at SNMP index ".8". This value is "D:\".
2. Skylar One uses "D:\" as the instance value to match against the record of internal indexes for this group, Dynamic Application, and device.
3. The instance value of "D:\" matches to internal index 3. When Skylar One stores the value "D:\" and "0", Skylar One associates the values with index 3.

Therefore, the storage description and storage used values for the "D:\" drive are associated with the same internal index as was associated with the previous values for the storage description and storage used.

System Settings that Affect Indexing

By default, Skylar One throttles surges of multi-index Dynamic Application alerts on individual devices.

To this end, it ignores Dynamic Application alert messages when those messages spike beyond 25 alerts per second. This is done to prevent potential system outages in scenarios where a very large number of alerts are occurring more rapidly than the Data Collector can process them.

The "Skylar One Default Internal Events" PowerPack that is included in Skylar One contains an event policy, "System: Dynamic App Alert Surge," that notifies the system administrator when alerts for a device exceed that 25-per-second threshold and discards the current batch of messages.

You can adjust this default behavior in the ***Dynamic Alert per-device*** field on the **System Threshold Defaults** page (System > Settings > Thresholds > System).

To edit the threshold for Dynamic Application alert throttling:

1. Go to the **System Threshold Defaults** page (System > Settings > Thresholds > System).
2. In the ***Dynamic Alert per-device*** field, drag the slider or enter a new value in the text box to edit the threshold for incoming alerts for a Dynamic Application on a given device.
3. Click **[Save]**.

Grouping Dynamic Application Data Using Collection Labels

This section describes how to use collection labels to view and manage collected data when multiple Dynamic Applications collect the same type of performance data from different devices or sources.

Collection labels enable you to consolidate similar data across Dynamic Applications into unified views, reports, and dashboards.

This section covers creating collection groups and labels, viewing and filtering existing labels, managing data normalization and handling duplicate values, understanding precedence rules for label alignment, and aligning presentation objects with collection labels so that data from different Dynamic Applications can be displayed and reported together.

What are Collection Labels and Collection Groups?

Collection Labels and ***Collection Groups*** allow you to group and view data from multiple performance Dynamic Applications in a single dashboard widget.

For example:

- Suppose you monitor phone systems from multiple vendors.
- Suppose you want to create a dashboard that displays the ten phone systems that drop the most calls.
- You could create a Collection Group called "Dropped Calls".
- You could create two Collection Labels: "Average Dropped Calls", and "Raw Dropped Calls".
- For each vendor, you could edit the appropriate performance Dynamic Application and align a collected value with "Average Dropped Calls" and align another collected value with "Raw Dropped Calls".
- You could then create a dashboard that displays the ten phone systems with the highest values for "Raw Dropped Calls" and also displays the ten phone systems with the highest values for "Average Dropped Calls".

Viewing the List of Collection Labels

The **Collection Labels** page (System > Manage > Collection Labels) displays a list of all the existing Collection Labels. By Default, Skylar One includes the following Collection Groups:

- ***Vitals***. Includes the Collection Labels "CPU", "Memory", and "Swap".
- ***Video Performance***. Includes Collection Labels for common performance metrics associated with video endpoint devices.

The **Collection Labels** page displays the following about each existing Collection Label:

- ***Label Name***. Name of the Collection Label.
- ***Label Description***. Description of the Collection Label. This field is optional.
- ***Group Name***. Collection Group that contains this Collection Label.
- ***Frequent Data***. Specifies whether frequently rolled up data is calculated for the Collection Label.
- ***Aligned Presentations***. Presentation Objects aligned with this Collection Label.
- ***Aligned Devices***. Devices that currently populate the Collection Label.
- ***Duplicates***. Number of devices for which two or more Presentation Objects are aligned with the same Collection Label.

TIP: To sort the list, click on a column heading. The list will be sorted by the column value, in ascending order. To sort the list by descending order, click the column heading again. You can

also filter the items on this inventory page by typing filter text or selecting filter options in one or more of the filters found above the columns on the page. For more information, see *Filtering Inventory Pages* in the ***Introduction to Skylar One*** manual.

Creating a Collection Group

You cannot create a Collection Group separately from creating a Collection Label. When you *create a Collection Label*, you can specify a new Collection Group or specify an existing Collection Group. If you specify a new Collection Group, Skylar One saves the new Collection Group when it saves the new Collection Label.

Creating a Collection Label

You can create a new Collection Label from the **Collection Labels** page (System > Manage > Collection Labels). To do so:

1. Go to the **Collection Labels** page (System > Manage > Collection Labels).
2. Click the plus-sign in the lower left of the page.
3. Enter values in the following columns:
    - ***Label Name***. Name of the Collection Label. This field is required.
    - ***Label Description***. Description of the Collection Label. This field is optional.
    - ***Group Name***. Collection Group to align with the Collection Label. You can select from a list of existing Collection Groups or enter the name of a new Collection Group. This field is required.
    - ***Frequent Data***. Specifies whether frequently rolled up data is calculated for the Collection Label. If the Collection Label will include data that is collected every five minutes or more frequently, and you require that dashboard data be updated every 15 minutes or 20 minutes, select *Yes* in this field. This data is available immediately for use in a collection label.
    - ***Save icon*** (). Select this icon to save your new Collection Label.
4. The new Collection Label appears in the page.

What is Normalization?

Normalization and roll-up are the processes by which Skylar One manages collected performance data for display and storage.

- ***Raw data*** is the data exactly as it was collected from a device or application.
- ***Normalized*** and ***rolled up*** data is data for which Skylar One has performed calculations, usually averaging raw data over a period of time.

Dynamic Applications can collect raw performance data from a device at the following intervals:

- 1 minute

- 2 minutes
- 3 minutes
- 5 minutes
- 10 minutes
- 15 minutes
- 30 minutes
- 1 hour
- 2 hours
- 6 hours
- 12 hours
- 24 hours

For performance Dynamic Applications, you specify this interval in the ***Poll Frequency*** field, in the **Properties Editor** page (System > Manage > Dynamic Applications).

Skylar One ***rolls up*** data so that reports with a larger timespan do not become difficult to view and to save storage space on the ScienceLogic database. When Skylar One rolls up data, Skylar One groups data into larger sets and calculates the average value for the larger set.

There are two types of roll up:

- ***Hourly***. Way to group and average data that is collected at intervals of less than or equal to 60 minutes. Skylar One rolls up data and calculates an average hourly value for each metric. Hourly samples include samples from the top of the hour to the end of the hour. For example, for an hourly rollup of data collected at 1-minute intervals between 1 am and 2 am, the first data point would be the one collected at 01:00:00 and ending at 01:59:00.
- ***Daily***. Way to group and average all data. Skylar One rolls up data and calculates an average daily value for each metric. Daily samples include samples from the beginning of the day until the end of the day. For example, for a daily roll-up of data collected at 1-minute intervals, the first data point is collected at 00:00:00 and the last data point is collected at 23:59:00.

Skylar One rolls up raw performance data as follows:

Frequency of Raw Collection	Roll-up
Every 1 minute	60 minutes, 24 hours
Every 2 minutes	60 minutes, 24 hours
Every 3 minutes	60 minutes, 24 hours
Every 5 minutes	60 minutes, 24 hours

Frequency of Raw Collection	Roll-up
Every 10 minutes	60 minutes, 24 hours
Every 15 minutes	60 minutes, 24 hours
Every 30 minutes	60 minutes, 24 hours
Every 60	60 minutes, 24 hours
Every 120 minutes or longer	24 hours

Before Skylar One normalizes data, Skylar One ***transforms*** the data. To transform data, Skylar One:

- For bandwidth data and data from Dynamic Applications of type "Performance", Skylar One derives rates from counter metrics.
- The rate from counter metrics are expressed in units-per-polling_interval. For example, rates for 5-minute collections are expressed as units-per-5-minutes.
- For data from Dynamic Applications of type "Performance", Skylar One evaluates presentation formulas. Counter metrics are first transformed into rates before evaluation.

NOTE: During the data transform steps, Skylar One does not directly roll up the raw data in the database tables.

When Skylar One rolls up data, Skylar One must ***normalize*** that data. To normalize data, Skylar One:

- groups and orders the data
- determines the sample size
- calculates count
- determines the maximum value
- determines the minimum value
- calulcates the mean value
- calculates the average value
- calculates the sum
- determines the standard deviation

NOTE: In Skylar One, normalized data does not include polling sessions that were missed or skipped. So for normalized data, null values are not included when calculating sample size, maximum values, minimum values, or average values.

Example

For example, suppose that ***every five minutes***, Skylar One collects data about file system usage on the device named ***my_device***. When Skylar One normalizes and rolls up the collected data for file system usage for ***my_device***, Skylar One will:

1. Apply any necessary data transforms (mentioned above).
2. Repeat the following step for both hourly normalization and daily normalization:
3. If this is the first data point for an hourly normalization or a daily normalization, insert summary statistics for that one data point:
    - Sample size = 1
    - Average = value of new data point
    - Max = value of new data point
    - Min = value of new data point
    - Sum = value of new data point
    - Standard Deviation = 0
4. For all subsequent data points for an hourly normalization or a daily normalization, Skylar One will update the summary statistics for the already existing data points in the data set (either hourly data set or daily data set).
5. If there are no gaps in collection, the summary statistics for hourly normalization will represent 12 data points, and the summary statistics for daily normalization will represent 288 data points.

What are Duplicates and How Does Skylar One Manage Them?

Multiple presentation objects can be aligned with a single Collection Label. For example, suppose that a Dynamic Application includes a presentation object for "memory used", and another Dynamic Application includes a presentation object for "memory usage". Suppose that both of these presentation objects are aligned with the Collection Label named "Memory".

Suppose that one of the devices monitored by Skylar One subscribes to both of those Dynamic Applications (for example, a Dynamic Application that monitors OEM hardware and a Dynamic Application that monitors the operating system). For that device, Skylar One will collect values for both presentation objects that are aligned with the Collection Label named "Memory".

When this situation arises, Skylar One uses precedence and some internal rules to assign a single presentation object to the Collection Label for that device. However, you can manually assign a different presentation object to the Collection Label after discovery.

If a device has a duplicate, Skylar One uses the following rules to determine which presentation object to use for that Collection Label for that device:

- If a manually defined Collection Label-presentation object pair exists, use that pair.
- If Skylar One cannot find a manually defined Collection Label-presentation object pair, use the pair with the lowest *precedence* value.
- If Skylar One finds more than one Collection Label-presentation object pair with the same precedence value, Skylar One will create a pair using the presentation object with the lowest presentation ID.

What is Precedence?

Skylar One performs discovery (during initial discovery and during nightly updates) and aligns Dynamic Applications with devices. During discovery, Skylar One will also align Collection Labels with devices. For devices with duplicates, Skylar One evaluates ***precedence*** to automatically align a single presentation object with each Collection Label. For devices with duplicates, Skylar One assigns the Collection Label-presentation object pair with the lowest precedence value.

Skylar One evaluates precedence:

- During nightly update discovery.

> **NOTE**: If you have manually defined a Collection Label-presentation object pair for one or more devices, nightly update discovery will not change the Collection Label-presentation object pair.

- When a Dynamic Application is manually aligned with a device in the **Dynamic Application Collections** page
- When devices are manually merged.

Aligning a Presentation Object with a Collection Label

You can align one or more presentation objects with a collection label. This allows Skylar One to compare and display reports on data from multiple performance Dynamic Applications.

To align a presentation object with a collection label:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the performance Dynamic Application that contains the presentation object you are interested in. Select the wrench icon (🔧) for that Dynamic Application.
3. In the Dynamic Application panel, select the **Presentations** tab.
4. In the **Presentation Objects** page, go to the **Presentation Object Registry** pane and find the presentation object you want to align with a Collection Label. Select the wrench icon (🔧) for that presentation object.
5. The top pane is populated with values from the selected presentation object. Select values for the following fields:

- ***Precedence***. Set the global precedence for this Collection Label-presentation object pair. For more information, see the section on *Precedence*.
- ***Label Group***. Select from a list of existing Collection Groups or click on the plus-sign icon (+) and enter the value for a new Collection Group. The current presentation object will be a member of the specified Collection Group.
- ***Label***. Select from a list of existing Collection Labels or click on the plus-sign icon (+) and enter the value for a new Collection Label. The current presentation object will be aligned with the specified Collection Label.

6. When you generate reports on the selected Collection Label, this presentation object will be included in the report.

Viewing and Managing the List of Presentation Objects Aligned with a Collection Label

From the **Collection Labels** page, you can view information about each Collection Label. For each Collection Label, you can view a list of presentation objects aligned with that Collection Label. To view this information:

1. Go to the **Collection Labels** page (System > Manage > Collection Labels).
2. Find the Collection Label you are interested in. In the ***Aligned Presentations*** column, select the pencil icon ( ). The **Aligned Presentations** modal page appears.
3. In the **Aligned Presentations** modal page, you can view information about the presentation objects aligned with the current Collection Label and perform actions to manage those presentation objects. You can also *unalign a presentation object* from a Collection Label and *change the precedence* for one or more Collection Label-presentation object pairs.

To globally unalign a presentation object from a Collection Label:

1. In the **Aligned Presentations** modal page, find the presentation object that you want to unalign from the Collection Label and select its checkbox.
2. From the ***Select Action*** field in the lower right, select *Unalign from Label*. Select the **[Go]** button.
3. The selected presentation object will no longer be associated with the Collection Label.

For each Collection Label-presentation object pair, you can define precedence. For example, suppose that both the "Cisco: CPU" Dynamic Application and the "Host Resource: CPU" include a presentation object that is aligned with the ***CPU*** Collection Label. You can define precedence to specify priority for each presentation object associated with a Collection Label.

Collection Group / Collection Label	Presentation Object	Dynamic Application
Vitals / CPU	CPU Average	Host Resource: CPU
Vitals / CPI	CPU 5 minutes average percent	Cisco: CPU

To set the precedence for the Collection Label (in our example, "CPU"):

1. The **Aligned Presentations** modal page displays all the presentation objects associated with the selected Collection Label. By default, each presentation object has a precedence of 50.
2. In the **Aligned Presentations** modal page, you can edit precedence in two ways:
    - In the ***Precedence*** column, use the up arrow and down arrow to change the value for a single presentation object. Repeat for each presentation object for which you want to edit precedence.
    - Select the checkbox of one or more presentation objects. In the ***Select Action*** field, select *Change Precedence* and a value. Select the **[Go]** button. Each selected presentation object will be assigned the new (and identical) precedence value.
3. Repeat step 2 for each Presentation Object for which you want to edit the precedence value.

> **NOTE**: The precedence values you define in the **Aligned Presentations** modal page override the precedence value you set per presentation object in the **Presentation Objects** page.

Viewing and Editing Duplicate Presentation Objects by Collection Label

You can view a list of devices where duplicates occur, view how Skylar One assigned the Collection Label-presentation object pair, and edit the Collection Label-presentation object pair for one or more devices. When you manually define a Collection Label-presentation object pair for a device, Skylar One will not edit or change that pair.

1. Go to the **Collection Labels** page (System > Manage > Collection Labels).
2. Find the Collection Label you are interested in. In the ***Duplicates*** column, select the pencil icon ( ). The **Duplicates** modal page appears.
3. In the **Duplicates** modal page, you can view a list of devices for which there are multiple possible Collection Label-presentation object pairs. You can view which pair is currently assigned to the device.
4. To change the pair for a device, click on the pair's radio button.
5. Repeat step #4 for each device on which you want to edit the duplicate.
6. In the ***Select Action*** field (in the lower right), select *Align Presentation for Device*. Select the **[Go]** button.
7. Each edited device will now use the selected Collection Label-presentation object pair.

Viewing and Managing the List of Devices Aligned with a Collection Label

From the **Collection Labels** page, you can view information about each Collection Label. For each Collection Label, you can view a list of devices from which Skylar One is collecting values. To view this information:

1. Go to the **Collection Labels** page (System > Manage > Collection Labels).
2. Find the Collection Label you are interested in. In the ***Aligned Devices*** column, select the pencil icon ( ).
3. In the **Aligned Devices** modal, you can view information about the devices that are aligned with the current Collection Label and perform actions to manage those devices.

For devices that include duplicates, you can reset the presentation object for one or more devices. When you manually define a Collection Label-presentation object pair for a device, Skylar One will not edit or change that pair.

1. In the **Aligned Devices** modal, select the checkbox for one or more devices for which you want to change the Collection Label-presentation object pair.
2. In the menus in the lower right, select ***Set Collection Presentation*** and then select the presentation object. Select the **[Go]** button.

For devices that include duplicates, you can clear all current settings, including manual settings. Skylar One will then automatically evaluate the precedence for each possible presentation object and assign the Collection Label-presentation object pair with the lowest precedence.

To clear the current Collection Label-presentation object pair for one or more devices:

1. In the **Aligned Devices** modal page, select the checkbox for one or more devices for which you want to clear the aligned presentation object.
2. In the menus in the lower right, select ***Recalculate Presentation Alignment***. Select the **[Go]** button.
3. Skylar One will evaluate the precedence of each possible presentation object and assign the presentation object with the lowest precedence.

Editing Duplicate Presentation Objects by Device

You can view a list of devices where duplicates occur, view how Skylar One assigned the Collection Label-presentation object pair, and edit the Collection Label-presentation object pair for one or more selected devices. When you manually define a Collection Label-presentation object pair for a device, Skylar One will not edit or change that pair:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Select the checkbox for each device you are interested in.
3. If you want to view a list of duplicates for all possible devices, select the checkbox at the top of the page to select all devices.
4. In the ***Select Action*** field (lower right), select **FIND Collection Label Duplicates**. Select the **[Go]** button.
5. The **Current Duplicates** page is displayed. For each device, you can edit the presentation object that is aligned with a Collection Label.
    - To select a Collection Label, use the drop-down list in the upper left.
    - To change the aligned presentation object for one or more devices:
        - Click on the radio button for the desired presentation object for the device.

- For each additional device you want to edit, click on the radio button for the desired presentation object.
- In the ***Select Action*** menu (lower right), select *Align Presentation for Device*. Select the **[Go]** button.

Editing Duplicate Presentation Objects for a Single Device

You can edit the Collection Label-presentation object pair for a single device. If a single device includes duplicate Collection Label-presentation object pairs, you can specify which one Skylar One should use for that device.

To edit the Collection Label-presentation object pairs for a single device:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Find the device you want to edit. Select its wrench icon ( ).
3. Select the **[Collections]** tab. In the **Dynamic Application Collections** page, click on the plus signs ( ) to expand each Dynamic Application.
4. You will notice that some presentation objects include the chart icon in the ***Label*** column. These presentation objects are duplicates that are not currently aligned with a Collection Label. If you want to align one of these presentation objects with the Collection Label (instead of the current alignment), click on the chart icon.
5. You will be prompted before Skylar One aligns the presentation object with the Collection Label. After approving, you will notice that a new presentation object now displays a chart icon in its ***Label*** column. This is because this presentation object is no longer associated with a Collection Label.

Editing a Collection Label

You can edit a Collection Label from the **Collection Labels** page (System > Manage > Collection Labels). To do so:

1. Go to the **Collection Labels** page (System > Manage > Collection Labels).
2. Find the Collection Label you want to edit. Select its wrench icon ( ).
3. You can edit one or more of the following:
   - ***Label Name***. Name of the Collection Label. This field is required.
   - ***Label Description***. Description of the Collection Label. This field is optional.
   - ***Group Name***. Collection Group to align with the Collection Label. You can select from a list of existing Collection Groups or enter the name of a new Collection Group. This field is required.
   - ***Frequent Data***. Specifies whether frequently rolled up data is calculated for the Collection Label. If the Collection Label will include data that is collected every five minutes or more frequently, and you require that dashboard data be updated every 15 minutes or 20 minutes, select *Yes* in this field. This data is available immediately for use in a collection label.
   - ***Save icon*** ( ). Select this icon to save your changes.

Deleting a Collection Label

You can delete a Collection Label from the **Collection Labels** page (System > Manage > Collection Labels) only if the Collection Label has no ***Aligned Presentations***. To delete a Collection Label:

> **NOTE**: You can delete a Collection Label only if no presentation objects are aligned with that label.

1. Go to the **Collection Labels** page (System > Manage > Collection Labels).
2. Find the Collection Label you want to delete.
3. Select its delete icon ( ). The Collection Label will be deleted from Skylar One.

Viewing Reports About Collection Labels on a Single Device

For each device in Skylar One, the **Device Performance** page displays time-series graphs about the data collected from that device.

If a device subscribes to a Dynamic Application that includes Collection Labels, Skylar One will display the Collection Group in the left pane of the **Device Performance** page. You can expand the Collection Group and select a Collection Label.

The graph for a Collection Label displays collected values on the Y-axis and time on the X-axis.

Viewing Dashboards About Collection Labels

You can use the following dashboard widgets to include data associated with Collection Labels in a dashboard:

- Multi-Series Performance Widget
- Leaderboard / Top-N Widget
- Gauge / Meter

For details on each widget, see the ***Dashboards*** manual.

Chapter 2

Snippet Dynamic Application Development

Overview

This chapter describes how to define collection objects and snippet code for Snippet Dynamic Applications. This chapter does not cover elements of Dynamic Application development that are common to all Dynamic Application types.

You should be familiar with the common elements and concepts of Dynamic Applications before reading this chapter. Snippet code is written using the Python programming language. You must be familiar with the syntax, programming techniques, and data structures of the Python language before developing Snippet Dynamic Applications.

This chapter covers the following topics:

Performance and Configuration Snippets *177*

Developing a Journal Snippet Dynamic Application *189*

Caching and Cache-consuming Snippets *206*

Excluded Modules and Additional Functions *208*

Example Snippet Dynamic Applications *216*

Using Snippets to Collect Database Data *237*

What is a Snippet Dynamic Application?

A Snippet Dynamic Application is a Dynamic Application that performs collection by executing one or more discrete blocks of Python code, called Snippets. Snippets provide the logic for the collection and manipulation of data. Skylar One passes credential and other configuration information to the snippet, and at the end of execution, the snippet must pass collected data back to Skylar One. The Dynamic Application developer defines snippet code that collects data and processes data. Snippet Dynamic Applications allow for data collection that cannot be implemented using other Dynamic Application types. For example, you could use a Snippet Dynamic Application if you need Skylar One to collect data from a proprietary system that does not support the other types of Dynamic Applications or if you need Skylar One to perform complex data manipulation before collected values are stored.

There are five types of Snippet Dynamic Applications:

- ***Snippet Configuration***. Snippet Configuration Dynamic Applications use custom-written Python code to collect configuration data from a device. If a Snippet Configuration Dynamic Application is aligned to multiple devices, the snippet code is executed separately for each device.
- ***Snippet Journal***. Snippet Journal Dynamic Applications use custom-written Python code to collect log data from a device.
- ***Snippet Performance***. Snippet Performance Dynamic Applications use custom-written Python code to collect performance data from a device. If a Snippet Performance Dynamic Application is aligned to multiple devices, the snippet code is executed separately for each device.
- ***Bulk Snippet Configuration***. Bulk Snippet Configuration Dynamic Applications use custom-written Python code to collect configuration data from multiple component devices. The configuration of a Bulk Snippet Configuration Dynamic Application specifies the maximum number of devices for which data can be collected during a single execution of the snippet code. If a Snippet Configuration Dynamic Application is aligned to multiple devices, Skylar One will automatically calculate the number of times the snippet code must be executed (based on the maximum number of devices) and will automatically assign multiple devices to each execution.
- ***Bulk Snippet Performance***. Bulk Snippet Performance Dynamic Applications use custom-written Python code to collect performance data from multiple component devices. The configuration of a Bulk Snippet Performance Dynamic Application specifies the maximum number of devices for which data can be collected during a single execution of the snippet code. If a Snippet Performance Dynamic Application is aligned to multiple devices, Skylar One will automatically calculate the number of times the snippet code must be executed (based on the maximum number of devices) and will automatically assign multiple devices to each execution.

Relationship Between the Dynamic Application Builder and Snippet Dynamic Applications

The Dynamic Applications Builder supports multiple snippet-based Dynamic Applications types.

Snippet Framework Dynamic Applications define collection as an ordered sequence of framework steps supplied by toolkits (for example, Syntax, Requestor, and Processor steps). These applications do not require custom Python snippet code and are configured primarily through snippet arguments. This manual describes how to configure both application types using the Builder. It does not replace developer documentation for Python-based snippet applications.

For more information, see the **Snippet Framework** manual.

Common Dynamic Application Elements

For comprehensive information on elements that apply to Snippet Dynamic Applications, see the **Dynamic Application Development** chapter.

Execution Environments

One way in which Snippet Dynamic Applications are unique from most other types of Dynamic Applications is that they can be aligned with a specific execution environment.

In the Skylar One user interface, an ***execution environment*** is a list of one or more ScienceLogic libraries. When a Dynamic Application snippet, Run Book Automation snippet, or credential test is executed, the execution environment defines a virtual Python environment that is deployed on-demand and includes all of the ScienceLogic libraries aligned with it for use during snippet execution.

When you create a Snippet Dynamic Application, you must select an execution environment to align with that Dynamic Application. All of the snippets contained in the Dynamic Application will then use that execution environment whenever the Dynamic Application runs. If you do not specify an execution environment, Skylar One will align the Dynamic Application with the default *System* environment. For more information about selecting an execution environment for a Dynamic Application, see the ***Dynamic Application Development*** chapter.

You can create and manage execution environments on the **Environment Manager** page (System > Customize > ScienceLogic Libraries > Actions > Execution Environments). For more information about creating and managing execution environments, see the ***ScienceLogic Libraries*** manual.

Prerequisites

This chapter describes how to define collection objects and snippet code for Snippet Dynamic Applications. This chapter does not cover elements of Dynamic Application development that are common to all Dynamic Application types.

You should be familiar with the common elements and concepts of Dynamic Applications before reading this chapter. Snippet code is written using the Python programming language. You must be familiar with the syntax, programming techniques, and data structures of the Python language before developing Snippet Dynamic Applications.

About this Chapter

This chapter includes the following sections:

- Performance & Configuration Snippets. Describes how to configure collection objects and how to create snippets in performance and configuration Snippet Dynamic Applications and bulk performance and bulk configuration Snippet Dynamic Applications.
- Journal Snippets. Describes how to configure collection objects in journal Snippet Dynamic Applications; how to create Snippets in journal Snippet Dynamic Applications; and how to create presentation objects in journal Snippet Dynamic Applications.

- Caching and Cache Consuming Snippets. Describes how to write a Snippet that caches results and how to write a Snippet that consumes cached results.
- Excluded Modules and Platform Wrappers. Describes which python modules cannot be used in Snippets and includes information about specific functions that can be used in Snippets.

This chapter also includes four additional chapters that walk through example Snippet Dynamic Applications.

Performance and Configuration Snippets

This chapter describes how to define collection objects and snippet code for the following types of Snippet Dynamic Applications:

- Snippet Configuration
- Snippet Performance
- Bulk Snippet Configuration
- Bulk Snippet Performance

All other elements of these Dynamic Applications, such as presentation objects and alerts, behave in the same manner as other Dynamic Application types.

Collection Objects

Similar to other Dynamic Application types, Snippet Dynamic Applications contain collection objects. Collection objects for Snippet Dynamic Applications have characteristics common to collection objects for other Dynamic Application types, such as naming, data typing, and grouping.

Collection objects for Snippet Dynamic Applications have the following unique elements:

- ***Snippet Arguments***. When a snippet is executed by Skylar One, the snippet is passed information about each collection object defined in the Dynamic Application, including the name and class type for the collection object. The ***Snippet Arguments*** field is used to provide additional parameters to indicate to the snippet code how to perform collection for that collection object. For example, if the snippet code is using the SNMP protocol to collect data from the device, the ***Snippet Arguments*** might define the OID to walk for each collection object.
- ***Snippet***. Because a Snippet Dynamic Application can contain multiple snippets, each collection object in a Snippet Dynamic Application must be associated with a specific snippet that is responsible for collecting that collection object.

Snippet Code

Snippet Dynamic Applications must have one or more snippets defined under the **[Snippets]** tab in the Dynamic Application pane. Snippets have the following properties:

- ***Snippet Name***. The name of the snippet.
- ***Active State***. Specifies whether the snippet should be executed by Skylar One when performing collection for the Dynamic Application. Choices are:

- *Enabled*. Skylar One will execute this snippet when performing collection for the Dynamic Application.
- *Disabled*. Skylar One will not execute this snippet when performing collection for the Dynamic Application.

- ***Required***. Specifies whether this snippet is required for successful collection of all other snippet requests. Choices are:
  - *Required - Stop Collection*. If this snippet request fails, the platform will not attempt to execute any other snippet requests in this Dynamic Application. Dynamic Applications that consume the cache of this Dynamic Application will halt collection.
  - *Not Required - Continue Collection*. If this snippet request fails, the platform will continue executing all remaining snippet requests in this Dynamic Application. Dynamic Applications that consume the cache of this Dynamic Application will continue collection.
- ***Snippet Code***. The python code that will be executed when Skylar One performs collection for this Dynamic Application.

The snippet code must be a block of valid python code that performs collection and passes collected values to Skylar One using the dictionary called ***result_handler***. This section describes:

- Using the result_handler dictionary.
- The variables available to snippet code.
- How to use aligned credentials.
- How to log collection problems in snippet code.

Populating Collection Objects

To pass collected values back to Skylar One, snippet code must populate the ***result_handler*** dictionary.

NOTE: The legacy self.oids dictionary is no longer supported but will continue to work for older Dynamic Applications. ScienceLogic recommends using result_handler for all Dynamic Applications.

The snippet code must populate the result_handler dictionary. Before each snippet is executed, Skylar One populates result_handler with information about all the collection objects in the current snippet. Some elements in the dictionary are left undefined and must be populated by your snippet code.

For Snippet Performance and Snippet Configuration Dynamic Applications, result_handler has the following structure:

```
oid: {'prime': , 'error_msg': , 'enum': , 'name': ,

'oid': , 'string_type': , 'class': , 'wm_walk_length': , 'oid_time': ,
'result': , 'oid_type': , 'factor': '', 'trend_col': '', 'monitor_config':
},

}
```

```
.

.

oid: {'prime': , 'error_msg': , 'enum': , 'name': ,

'oid': , 'string_type': , 'class': , 'wm_walk_length': , 'oid_time': ,
'result': , 'oid_type': , 'factor': '', 'trend_col': '', 'monitor_config':
},

}
```

For Bulk Snippet Performance and Bulk Snippet Configuration Dynamic Applications, result_handler has the following structure:

```
(device id, oid): {'prime': , 'error_msg': , 'enum': , 'name': ,

'oid': , 'string_type': , 'class': , 'wm_walk_length': , 'oid_time': ,
'result': , 'oid_type': , 'factor': '', 'trend_col': '', 'monitor_config':
},

}

.

.

(device id, oid): {'prime': , 'error_msg': , 'enum': , 'name': ,

'oid': , 'string_type': , 'class': , 'wm_walk_length': , 'oid_time': ,
'result': , 'oid_type': , 'factor': '', 'trend_col': '', 'monitor_config':
},

}
```

For Snippet Performance and Snippet Configuration Dynamic Applications, result_handler uses the oid value (Snippet Argument) as a key for each collection object. For Bulk Snippet Performance and Bulk Snippet Configuration Dynamic Applications, result_handler uses a tuple of the device ID and the oid value (Snippet Argument) as a key for each collection object, i.e. result_handler includes an entry for every collection object for every device that the snippet code has been assigned.

Each entry in result_handler has a dictionary that contains the following keys:

- ***oid***. The value defined in the ***Snippet Arguments*** field in the **Collection Objects** page for this collection object.
- ***prime***. The value defined for the ***Index*** checkbox in the **Collection Objects** page for this collection object.

- ***error_msg***. An optional error message associated with this collection object. Snippets should set this value to an error message string if this object cannot be collected.
- ***enum***. The value defined in the ***Enums*** field in the **Collection Objects** page for this collection object.
- ***name***. The value defined in the ***Object Name*** field in the **Collection Objects** page for this collection object.
- ***oid***. The value defined in the ***Snippet Arguments*** field in the **Collection Objects** page for this collection object.
- ***string_type***. This value refers to the internal operation of Skylar One's collection process. Do not modify this value.
- ***class***. The value defined in the ***Class Type*** field in the **Collection Objects** page for this collection object. ***class*** is set to the integer value that appears at the beginning of each ***Class Type*** option.
- ***wm_walk_length***. This value refers to the internal operation of Skylar One's collection process. Do not modify this value.
- ***oid_time***. Your snippet can optionally set this value to the amount of time in seconds it took to collect the object.
- ***result***. The collected value or values for this collection object. Your snippet code must populate ***result*** with a list of tuples. Each result tuple must contain two values: an index followed by a result string. The index can be any scalar type, but no two tuples can have the same index value.
- ***oid_type***. The ID number of the snippet responsible for collecting this collection object. Your snippet should use this value to determine whether the snippet should collect this object, i.e. compare this value to the value in the *snippet_id* variable.
- ***factor***. This value refers to the internal operation of Skylar One's collection process. Do not modify this value.
- ***trend_col***. This value refers to the internal operation of Skylar One's collection process. Do not modify this value.
- ***monitor_config***. This value refers to the internal operation of Skylar One's collection process. Do not modify this value.

To update the ***result*** value for a collection object using the ***result_handler*** dictionary, assign a value directly to the key for that collection object. Remember that the keys in result_handler are::

- For Snippet Performance and Snippet Configuration Dynamic Applications, the value in the ***Snippet Arguments*** field.
- For Bulk Snippet Performance and Bulk Snippet Configuration Dynamic Applications, a tuple that includes a device ID and the value in the ***Snippet Arguments*** field.

For example:

```
result_handler['cpu'] = [(0,0),]
```

- ***cpu***. This is the Snippet Argument, used to uniquely identify the collection object
- ***0,0***. At index zero, set the value to zero.

You can update the result value for all the collection objects at the same time by using the update() function of result_handler. For example:

```
result_handler.update(results)
```

The results parameter must be a dictionary that has the same keys as ***result_handler***. Each key in the supplied dictionary must reference that value to be set as the result value for that collection object.

Available Variables

The following local variables are available to use in your snippet code for Snippet Performance and Snippet Configuration Dynamic Applications:

- ***snippet_id***. The ID number for this snippet.
- ***this_device***. A named tuple that contains information about the device for which collection is being performed.
- ***parent_device***. This variable is available for component devices and merged devices. A named tuple that contains information about the direct parent of the component device for which collection is being performed.
- ***root_device***. This variable is available only for component devices. A named tuple that contains information about the root device of the component device for which collection is being performed.

> **NOTE:** If a Dynamic Application includes snippet code with a ***root_device*** tuple, its ***Collector Affinity*** setting should be set to *Root Device Collector*. For more information, see the section on "Dynamic Application Settings" in the ***Dynamic Application Development*** chapter.

- ***self.cred_details***. A dictionary that contains information about the credential aligned with the Dynamic Application. For information about the structure of credential dictionaries, see the *section Using Credential Dictionaries*.
- ***self.device_ip***.The IP address of the target device for which collection is being performed. This variable is available for component or merged devices.
- ***self.device_hostname***. The hostname of the target device for which collection is being performed. This variable is available for component or merged devices.
- ***self.device_pdu_packing***. A boolean value indicating if PDU Packing is enabled on the target device for which collection is being performed. This variable is available for component or merged devices. PDU packing enables quicker collection of voluminous SNMP data. For more information, see the "Additional Configuration for Concurrent Network Interface Collection" topic in the ***Monitoring Device Infrastructure Health*** manual.
- ***self.root_ip***. The IP address of the root device of the component device for which collection is being performed. This variable is available for component or merged devices.
- ***self.root_hostname***. The hostname of the root device of the component device for which collection is being performed. This variable is available for component or merged devices.

- ***self.root_pdu_packing***. A boolean value indicating if PDU Packing is enabled on the root device of the component device for which collection is being performed. This variable is available for component or merged devices. PDU packing enables quicker collection of voluminous SNMP data. For more information, see the "Additional Configuration for Concurrent Network Interface Collection" topic in the ***Monitoring Device Infrastructure Health*** manual.

The ***this_device***, ***parent_device***, and ***root_device*** named tuples include the following values:

- ***did***. The device ID for the device.
- ***name***. The name of the device.
- ***ip***. The primary management IP address of the device.
- ***snmp_cred_id***. The ID of the SNMP read credential assigned to the device.
- ***root_did***. For component devices, the device ID of the root device associated with the device.
- ***parent_did***. For component devices, the device ID of the direct parent for the device.
- ***unique_id***. For component devices, the unique identifier of the component device. This variable is equivalent to the %U component identifier substitution variable for WMI, XSLT, and SOAP Dynamic Applications.
- ***guid***. For component devices, the globally unique identifier of the component device. This variable is equivalent to the %G component identifier substitution variable for WMI, XSLT, and SOAP Dynamic Applications.

For example, to use the name of the root device of the component device for which collection is being performed, you would use the variable ***root_device.name***.

Available Variables for Bulk Performance and Bulk Configuration Snippets

The following local variables are available to use in your snippet code for Bulk Snippet Performance and Bulk Snippet Configuration Dynamic Applications:

- ***snippet_id***. The ID number for this snippet.
- ***self.cred_details***. A dictionary that contains information about the credential aligned with the Dynamic Application. For information about the structure of credential dictionaries, see the *Using Credential* section.
- ***devices***. A dictionary that contains information about the devices for which collection is being performed. During each collection, the platform will automatically assign one or more devices to each execution of your snippet code. The keys in the devices dictionary are the device IDs of the devices assigned to the snippet. The values in the devices dictionary are named tuples with the following values:
    - *device*. A named tuple that contains information about the component device for which collection is being performed.
    - *parent_device*. Contains the device ID of the direct parent of the component device for which collection is being performed.

The ***device*** named tuples in the devices dictionary include the following values:

- ***did***. The device ID for the device.
- ***name***. The name of the device.
- ***ip***. The primary management IP address of the device.
- ***snmp_cred_id***. The ID of the SNMP read credential assigned to the device.
- ***root_did***. For component devices, the device ID of the root device associated with the device.
- ***parent_did***. For component devices, the device ID of the direct parent for the device.
- ***unique_id***. For component devices, the unique identifier of the component device. This variable is equivalent to the %U component identifier substitution variable for WMI, XSLT, and SOAP Dynamic Applications.
- ***guid***. For component devices, the globally unique identifier of the component device. This variable is equivalent to the %G component identifier substitution variable for WMI, XSLT, and SOAP Dynamic Applications.

For example, to use the name of the device with ID 16, you would use the variable *devices [16].device.name*.

Using Credentials

When information about a credential is passed by Skylar One to snippet code in a variable, the credential information is stored in a dictionary. This section describes the structure of credential dictionaries. Snippet Dynamic Applications can use any type of credential.

> TIP: If your snippet code requires that a specific credential type be aligned with the Dynamic Application, you can examine the ***cred_type*** field and generate an appropriate error message if an incorrect credential is aligned. This check will ensure that your snippet code does not reference an entity in the credential dictionary that does not exist in the aligned credential.

Several elements in the credential dictionary are common to all credential types, and each credential type (other than Basic/Snippet) has unique elements that appear only in the credential dictionary for that credential type. The following elements are common to every type of credential dictionary:

- ***cred_id***. Integer. Unique credential ID.
- ***cred_type***. Integer. Type of credential .
    - 1 SNMP. *SNMP credentials allow Skylar One to access SNMP data on a managed device.*
    - 2 DB. *Database credentials allow Skylar One to access data on a database on a managed device.*
    - 3 SOAP/XML. *SOAP/XML credentials allow Skylar One to access a web server on a managed device.*
    - 4 LDAP. *LDAP or Active Directory credentials allow Skylar One to access data on an LDAP server or Active Directory server.*

- 5 Snippet. *Basic/Snippet credentials define standard authentication parameters, but are not tied to a specific authentication protocol.*
- 6 SSH. *SSH/Key credentials specifies the hostname or IP address of the system you want to monitor, the port number used to access that system, and the private key used for authentication.*
- 7 PowerShell. *PowerShell credentials allow Dynamic Applications to retrieve data from Microsoft devices.*

> **NOTE**: For more information, see the ***Credential Management*** chapter in the ***Discovery and Credentials*** manual.

- ***cred_host***. String. Host name or IP address (%D substitution string).
- ***cred_port***. Integer. TCP/IP port for connections.
- ***cred_pwd***. String. Password (encrypted in the database, stored as clear text in the dictionary).
- ***cred_user***. String. Username.
- ***cred_timeout***. Integer. Timeout in milliseconds.

The following elements are unique for SNMP credentials:

- ***snmp_version***. Integer. SNMP version, values 1, 2, 3.
- ***snmp_ro_community***. String. Read-only community string.
- ***snmp_rw_community***. String. Read/Write community string.
- ***snmp_retries***. Integer. Number of retries.
- ***snmpv3_auth_proto***. String. V3 auth. protocol,. Can be either MD5 or SHA.
- ***snmpv3_sec_level***. String. V3 security. Can be noAuthNoPriv, AuthNoPriv, or AuthPriv.
- ***snmpv3_priv_proto***. String. V3 privacy protocol. Can be : DES or AES.
- ***snmpv3_priv_pwd***. String. V3 password encrypted in the database and stored as clear text in the dictionary.
- ***snmpv3_context***. String. V3 context.

The following elements are unique for Database credentials:

- ***db_type***. Integer.
    - 1 MySQL
    - 2 MSSQL
    - 3 Oracle
    - 4 Postgress
    - 5 DB2
    - 6 Sybase
    - 7 Informix
    - 8 Ingress).
- ***db_name***. String. Initial database name.
- ***db_sid***. String. Database SID (Oracle only).
- ***db_connect***. String. Database connect string (Oracle only).

The following elements are unique for SOAP/XML credentials:

- ***curl_url***. String. URL.
- ***curl_proxy_ip***. String. Proxy server IP address.
- ***curl_proxy_port***. Integer. Proxy server TCP/IP port.
- ***curl_proxy_acct***. String. Proxy server account.
- ***curl_proxy_passwd***. String. Proxy server password.
- ***curl_encoding***. String. Encoding method (eg text/xml).
- ***curl_post_or_get***. Integer. HTTP method 0 – GET, 1- POST.
- ***curl_http_version***. HTTP version: 10 = 1.0, 11 = 1.1.
- ***curl_request_sub_1***. String. Substitution value to substitute into Snippet code.
- ***curl_request_sub_2***. String. Substitution value to substitute into Snippet code.
- ***curl_request_sub_3***. String. Substitution value to substitute into Snippet code.
- ***curl_request_sub_4***. String. Substitution value to substitute into Snippet code.
- ***curl_headers***. List of Strings. Each string is a HTTP key/value pair.
- ***curl_opts***. Dictionary of Curl options comprising a series of pairs of string key and corresponding string value.

Reporting Collection Status

Your snippet code must report when collection was successful. The following local variables are defined for this purpose:

- ***COLLECTION_PROBLEM***. By default, this variable is automatically instantiated. Its default value is False. When your snippet code completes, you can set this variable to True to indicate that there was a problem with collection (i.e. indicating a missed poll). When your snippet code completes, you can set this variable to False to indicate that collection was successful. This variable is set to True at the start of each collection; you must set this variable to False to report a successful collection.
- ***PROBLEM_STR***. This variable can be used with the COLLECTION_PROBLEM variable, to provide a description of the problem with collection. If you set the ***COLLECTION_PROBLEM*** variable to True (indicating a missed poll), you can additionally specify why the collection failed by assigning a string value to the ***PROBLEM_STR*** variable. Skylar One will use the string value of ***PROBLEM_STR*** to generate an alert.

Generating Alerts

As with other Dynamic Application types, you can use alert formulas and linked event policies to generate events in response to Snippet polling of remote systems. You might also need to generate logs and events about the collection process itself. For example, you might want to generate an alert if an external device does not respond or if the supplied credential is invalid. There are three methods you can use to generate these logs:

1. Generate an internal alert for the problem. The alert will be associated with the device the Dynamic Application is currently collecting data from. Skylar One includes event policies that will trigger if a snippet generates an internal alert.
2. Generate a custom alert for the problem. A custom alert does not have to be associated with the device the Dynamic Application is currently collecting data from; a custom alert can be associated with entity types other than devices, for example, an Organization or an Asset. You can define your own custom event policies for custom alerts.
3. Create one or more collection objects within your Dynamic Application to specifically monitor collection problems.

Generating an Internal Alert

Generating an internal alert is the method used to log collection problems in the examples in this document. During collection, Skylar One maintains a list of internal alerts that are processed and can trigger events after collection is complete.

The data structure for internal alerts is a Python list of tuples. Each tuple has two elements: a message ID and supporting message text. You can use the following values for the message ID element:

- ***257***. Triggers a "General Collection Problem" event, which has a severity of "Major".
- ***519***. Triggers a "Dynamic Snippet Exception" event, which has a severity of "Minor".
- ***518***. Triggers a "Dynamic Snippet Message" event, which has a severity of "Notice".

To generate an event for a collection problem using the built-in method, append a tuple with your message to the self.internal_alerts list. For example:

```
self.internal_alerts.append((257, "Cannot connect to remote device"))
```

Generating a Custom Alert

To generate a custom alert, you can use the following function:

```
em7_snippets.generate_alert(message, xid, xtype, yid, ytype, yname, value,
threshold)
```

You can define an event based on the alert; the event must have a ***Source*** of *API* and use pattern matching to match the alert. The arguments for the function are:

- ***message***. Required argument. The message text for the alert.
- ***xid*** = *value*. Required argument. The entity to associate with the alert. Supply the numeric ID of an entity. For example, if you supply '1' in the ***xtype*** argument, supply a device ID in this argument.
- ***xtype*** = *value*. Required argument. The type of entity for which you specified an ID in the ***xid*** argument. Supply one of the following integer values:
    - *0*. Organization
    - *1*. Device
    - *2*. Asset
    - *4*. Network
    - *5*. Interface
    - *6*. Vendor
    - *7*. User Account
    - *8*. Virtual Interface
    - *9*. Device Group
    - *10*. IT Service
    - *11*. Ticket
- ***yid*** = *value*. The sub-entity to associate with the alert. Supply the numeric ID of a sub-entity. For example, if you supply '3' in the ***ytype*** argument, supply a file system ID in this argument.
- ***ytype*** = *value*. Optional argument. The type of sub-entity for which you specified an ID in the ***yid*** argument. Supply one of the following integer values:
    - *9*. News Feed (if ***xtype*** is 0) or Process (if ***xtype*** is 1).
    - *1*. CPU. Can be specified only if ***xtype*** is 1 (Device).
    - *2*. Disk. Can be specified only if ***xtype*** is 1 (Device).

    - *3*. File System. Can be specified only if ***xtype*** is 1 (Device).
    - *4*. Memory. Can be specified only if ***xtype*** is 1 (Device).
    - *5*. Swap. Can be specified only if ***xtype*** is 1 (Device).
    - *6*. Hardware Component. Can be specified only if ***xtype*** is 1 (Device).
    - *7*. Interface. Can be specified only if ***xtype*** is 1 (Device).
    - *10*. Port. Can be specified only if ***xtype*** is 1 (Device).
    - *11*. Windows Service. Can be specified only if ***xtype*** is 1 (Device).
    - *12*. Web Content. Can be specified only if ***xtype*** is 1 (Device).
    - *13*. Email Monitor. Can be specified only if ***xtype*** is 1 (Device).
- ***yname*** = *value*. Optional argument. The name of the sub-entity for which you specified an ID in the ***yid*** argument.
- ***value***= *string*. Optional argument. A value that will be passed with the alert message. This value is available in the %V substitution character for event policies.
- ***threshold***= *string*. Optional argument. A threshold value that will be passed with the alert message. This threshold value is available in the %T substitution character for event policies.

For example:

```
em7_snippets.generate_alert('Attempted File System Cleanup', '60', '1',
'150', '3')
```

will generate an alert with the message "Attempted File System Cleanup" associated with the file system with ID 150 on the device with ID 60.

Self-Reporting Collection Objects

The self-reporting method uses one or more collection objects (that you define in your Dynamic Application) to report application-specific failures. You can additionally define alerts that evaluate the values returned for these collection objects.

For example, you might define a single un-grouped collection object called "Collection Status" and assign it a value specific to your collection method. If your snippet executes successfully, the snippet would set the value to "OK"; otherwise the snippet would set the value to a failure message, such as "authentication failed" or "connection timeout". Each of these can then be handled separately with appropriate alert definitions and associated event policies.

Writing Debug Information to silo.log

Using the following function, you can configure your snippet to write messages to silo.log:

```
em7_snippets.logger(filename)
```

For example:

```
logger=em7_snippets.logger(filename='/var/log/em7/silo.log')
```

Your snippet code will then write messages to the log file using the following syntax:

```
logger.debug("message")
```

IMPORTANT: Starting with Skylar One version 11.3.0, you can no longer use this process to specify a "filename" for output. Instead Skylar One will write to **/var/log/sl1/snippets.log**.

Debug messages will appear in the specified file on the Data Collector or All-In-One Appliance that runs the snippet. When Skylar One executes snippet code for configuration Snippet and performance Snippet Dynamic Applications, the snippet code writes debug logs to `silo.log` regardless of the ***Debug Mode*** setting for the process "Data Collection: Dynamic App". Therefore, to avoid debug logging on production systems, you must use debug logging only when developing snippet code in a test environment, then remove or comment out debug statements when you have completed development and testing.

Developing a Journal Snippet Dynamic Application

This example describes the development of a Journal Snippet Dynamic Application that collects data from ScienceLogic access logs. Each collected journal entry contains information about a single user session in a Skylar One system. This example demonstrates how to use available variables, create, populate, and update journal entries, and how to report collection problems.

NOTE: This example Dynamic Application is for demonstration purposes only. The data collected by this Dynamic Application is already available in Skylar One in the **Access Sessions** page (System > Monitor > Access Logs).

Requirements

The Dynamic Application developed in this example has the following requirements:

1. The snippet must run once a minute.
2. The snippet must use a MySQL credential to collect data from the following fields in the *master_access.access_log* table in a ScienceLogic database:
    - ***session_id***. The unique key Skylar One uses to identify a user session. The Dynamic Application must use this value as the key for each journal entry.

- ***state***. The state of the user session in the Skylar One system. The Dynamic Application must use this value as the state for each journal entry, using the following mapping:
    - ***0***. Represents the "Logged Out" state in Skylar One. Must map to the journal entry state "Closed".
    - ***2***. Represents the "Logged In" state in Skylar One. Must map to the journal entry state "Open".
    - ***3***. Represents the "Expired" state in Skylar One. Must map to the journal entry state "Abandoned".
- ***user***. The ScienceLogic username for the user session. The Dynamic Application must store this value in a collection object for every journal entry.
- ***login_time***. The time that the user session started. The Dynamic Application must store this value in a collection object for every journal entry.
- ***logout_time***. The time that the user session ended. The Dynamic Application must store this value in a collection object for every abandoned or closed journal entry.

3. For each collection period, the snippet must query the *master_access.access_log* table for all user sessions that have started since the previous collection period. If the time of the previous collection period is unknown, the snippet must query the *master_access.access_log* table for all user sessions that have started in the last 60 seconds. The new entries must be stored in a new journal entry, with information stored for all available collection objects.
4. For each collection period, the snippet must determine which existing journal entries are still open, query the *master_access.access_log* table for those user sessions, and abandon or close the journal entry if the user session is in an expired or logged out state.
5. If a query on the *master_access.access_log* table for an existing journal entry indicates that information about the user session is no longer in the table, the journal entry must be set to the Error state.
6. The snippet must handle the following potential exceptions:
    - If the aligned credential is not a MySQL Database credential, the snippet must report a missed poll and not attempt collection.
    - If Skylar One cannot connect to the database , the snippet must report a missed poll and not attempt collection.
    - If a serialization or deserialization exception occurs in meta data storage/retrieval, the snippet must generate a minor event, but not report a missed poll.
    - If an error occurs when querying for new user sessions and an error occurs when querying for existing user sessions, the snippet must report a missed poll.
    - If an error occurs when querying for new user sessions, but querying for existing user sessions is successful, the snippet must generate a minor event, but not report a missed poll.
    - If querying for new user sessions is successful, but an error occurs when querying for existing user sessions, the snippet must generate a minor event, but not report a missed poll.

Creating the Dynamic Application

To create this example Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Click the **[Actions]** button, then select *Create New Dynamic Application*. The **Create New Application** page is displayed.
3. Supply values in the following fields:
    - ***Application Name***. The name of the Dynamic Application. This example is called "Snippet Journal Example".
    - ***Application Type***. This example is a Snippet Journal. Select *Snippet Journal* in this field.
    - ***Poll Frequency***. To meet *requirement one*, select *Every 1 Minute* in this field.
4. This example does not have specific requirements for the other settings defined in this page. You can leave the remaining fields set to the default values.
5. Click the **[Save]** button.

Because collection objects are assigned a specific snippet, you must create the container for the snippet code before creating the collection objects for this Dynamic Application. To create a container for the snippet code, perform the following steps:

1. Click the **[Snippets]** tab.
2. Supply values in the following fields:
    - ***Snippet Name***. The name of the snippet. The snippet in this example is called "Collect Login Data".
    - ***Active State***. To ensure that the snippet code is run by Skylar One, select *Enabled* in this field.
    - ***Snippet Code***. Leave this field blank. You will add the snippet code later in this example.
3. Click the **[Save]** button.

Creating the Collection Objects

To meet *requirement two*, this Dynamic Application must have three collection objects: username, login time, and logout time. Each journal entry will have a value stored for each of these collection objects. To create the collection objects for this Dynamic Application, perform the following steps:

1. Click the **[Collections]** tab in the **Dynamic Application Editor** pane.
2. Supply values in the following fields to create the username object:
    - ***Object Name***. The name of the collection object. Enter "Username" in this field.
    - ***Variable Name***. The name of the variable the snippet will use to store values for this collection object. Enter "Username" in this field.
    - ***Class Type***. The collected values for the username collection object will be text strings. Select *22 Journal Character* in this field.
    - ***Snippet***. There is only one snippet for this Dynamic Application. Select *Collect Login Data* in this field.
3. Click the **[Save]** button, then click the **[Reset]** button to clear the values you entered.
4. Repeat steps two and three for the login time collection object using the following values:

- ***Object Name***. Enter "Login Time".
- ***Variable Name***. Enter "Login".
- ***Class Type***. Select *30 Journal Date & Time*.
- ***Snippet***. Select *Collect Login Data*.

5. Repeat steps two and three for the logout time collection object using the following values:
    - ***Object Name***. Enter "Logout Time".
    - ***Variable Name***. Enter "Logout".
    - ***Class Type***. Select *30 Journal Date & Time*.
    - ***Snippet***. Select *Collect Login Data*.

Snippet Code

After creating the Dynamic Application and the collection objects, you must write the snippet code. This section walks through all the snippet code used in this example.

The initial section of snippet code imports the MySQLdb module, then initializes variables used to track collection problems:

```
import MySQLdb
```

```
COLLECTION_PROBLEM = False
```

```
create_error = False
```

```
update_error = False
```

The following Boolean values are used to track collection problems:

- ***COLLECTION_PROBLEM***. The variable used to tell Skylar One whether a missed poll has occurred. At the start of execution, this value is "True". The snippet initially changes this value to False (collection was successful), and will later change the value back to True if an exception (as described in *requirement six*) occurs.
- ***create_error***. The snippet will use this variable to track whether there was a query error when collecting new journal entries. The snippet initially sets this variable to "False" (no error). Before the snippet completes, this variable will be used to determine how to report errors (as described in *requirement six*).
- ***update_error***. The snippet will use this variable to track whether there was a query error when collecting existing journal entries. The snippet initially sets this variable to "False" (no error). Before the snippet completes, this variable will be used to determine how to report errors (as described in *requirement six*).

The snippet will use the Dynamic Application metadata functions to store the date and time of the last successful query for new journal entries. The next section of code retrieves the stored metadata in a TRY/EXCEPT block:

```
try:
```

```
    meta = get_app_instance_meta_data()

except DeserializationError as e:

    INTERNAL_ALERTS.append((519, "Could not deserialize stored meta data for
    did:%s, app id:%s, error: %s" % (app_id, did, e)))
```

To meet *requirement six*, if a deserialization error occurs, the snippet generates an internal alert using alert id 519 (snippet exception). Alert ID 519 generates a minor event. To meet *requirement three*, the snippet will continue with collection if a deserialization error occurs, but will query for all new user sessions from the last 60 seconds.

The snippet outputs the credential and collection object information passed in by Skylar One:

```
logger.debug("Credential: %s" % (cred_details,))

logger.debug("Collection objects: %s" % (collection_objects,))
```

If the snippet encounters problems, this information might be useful when diagnosing the issue.

To meet the first item in *requirement six*, the snippet checks the credential passed in by Skylar One:

```
if cred_details['cred_type'] != 2:

    COLLECTION_PROBLEM = True

    PROBLEM_STR = "Database credential not aligned"

else:
```

As described in the Performance and Configuration Snippets section, the 'cred_type' key in the cred_details dictionary contains the type of credential. The snippet checks that the credential is of the correct type, '2' (database). If the credential is not for a database, the snippet sets COLLECTION_PROBLEM to "True" to report a missed poll, and sets the PROBLEM_STR to an appropriate error message. If this condition occurs, Skylar One will generate a minor event that contains the error message in PROBLEM_STR. If the credential is incorrect, the snippet must finish execution without attempting to collect data; therefore, the remainder of the snippet code is inside the ELSE block.

Similar to the test for the credential type, the snippet attempts to create a MySQL connection using the credential and continues execution only if no exception occurs. The remainder of the snippet code is inside the IF block:

```
try:

  conn = MySQLdb.connect(user=cred_details['cred_user'], passwd=cred_
  details['cred_pwd'], host=cred_details['cred_host'], port=cred_details
  ['cred_port'])

  cur = conn.cursor()

except MySQLdb.Error, e:

  COLLECTION_PROBLEM = True

  PROBLEM_STR = "Database connection error: %s: %s" % (e.args[0], e.args
  [1])

if COLLECTION_PROBLEM == False:
```

If an exception occurs, the snippet sets COLLECTION_PROBLEM to "True" to meet *requirement six*. When writing a snippet, you must use TRY/EXCEPT blocks around any calls that might cause an exception; if an exception occurs outside of a TRY/EXCEPT block, the snippet will immediately terminate.

The next section of code executes the database query that will populate new journal entries. The snippet uses an IF/ELSE block to determine whether the time of the last query is available. To meet *requirement three*, if the time of the last query is available, it is used in the WHERE clause of the new query. If the time of the last query is unavailable, a timestamp for 60 seconds ago is used in the WHERE clause:

```
if meta is not None and meta.has_key('time'):

  logger.debug("Have valid meta data, using last query time in WHERE
  clause")

  try:

    cur.execute("""(SELECT session_id, state, user, UNIX_TIMESTAMP
    (login_time) as login, UNIX_TIMESTAMP(logout_time) as logout_time
    FROM master_access.access_log WHERE login_time > FROM_UNIXTIME
    (%s)) UNION (SELECT UNIX_TIMESTAMP(NOW()),UNIX_TIMESTAMP(NOW
    ()),UNIX_TIMESTAMP(NOW()),UNIX_TIMESTAMP(NOW()),UNIX_TIMESTAMP(NOW
    ())) ORDER BY login DESC""", (meta['time'],))

  except MySQLdb.Error, e:

    create_error = True
```

```
        create_problem = "%s: %s" % (e.args[0], e.args[1])
    else:
      logger.debug("No valid meta data, using 60 seconds ago in WHERE
      clause")
      try:
        cur.execute("""(SELECT session_id, state, user, UNIX_TIMESTAMP
        (login_time) as login, UNIX_TIMESTAMP(logout_time) as logout_time
        FROM master_access.access_log WHERE UNIX_TIMESTAMP(login_time) >
        (UNIX_TIMESTAMP() - 60)) UNION (SELECT UNIX_TIMESTAMP(NOW()),UNIX_
        TIMESTAMP(NOW()),UNIX_TIMESTAMP(NOW()),UNIX_TIMESTAMP(NOW()),UNIX_
        TIMESTAMP(NOW())) ORDER BY login DESC""")
      except MySQLdb.Error, e:
        create_error = True
        create_problem = "%s: %s" % (e.args[0], e.args[1])
```

The query selects the fields listed in *requirement two*. The query uses a UNION statement in conjunction with an ORDER BY to make the first row in the result set always return all NOW() timestamps. Again, the snippet uses TRY/EXCEPT blocks to catch any errors. All time values are converted to UNIX timestamps; storing UNIX timestamps for the login time and logout time allow the presentation options to convert to human-readable timestamps based on the user's preferences. To meet *requirement six*, collection of journal entry updates must still be performed if this query fails; therefore, instead of using COLLECTION_PROBLEM and PROBLEM_STR, the snippet uses the variables create_error and create_problem to track errors from this query.

If the query is successful, the first row, which contains the NOW() timestamp, is stored as meta data:

```
    if create_error == False:
      now = cur.fetchone()
      last_query = now[3]
      if last_query:
        meta_data = {'time':last_query}
        logger.debug("Setting Meta Data: %s" % meta_data)
        try:
```

```
        set_app_instance_meta_data(meta_data)

    except SerializationError as e:

        INTERNAL_ALERTS.append((519, "Could not serialize meta data for
        storage for did:%s, app id:%s, error: %s" % (app_id, did, e)))

        logger.debug("SerializationError when setting meta data: %s" %
        e)
```

Similar to the retrieval of this meta data, the set_app_instance_meta_data call is in a TRY/EXCEPT block, and the snippet generates an internal alert if a serialization error occurs.

The snippet now fetches all remaining rows from the result of the query. Each row is used to create a new journal entry. This section of code is inside the "if create_error == False" block:

```
results = cur.fetchall()

for row in results:

    if row[1] == 0:

        entry = create_entry(row[0], JOURNAL_STATE_CLOSED)

        entry_collections = {'Username':row[2], 'Login':row[3],
        'Logout':row[4]}

    elif row[1] == 3:

        entry = create_entry(row[0], JOURNAL_STATE_ABANDONED)

        entry_collections = {'Username':row[2], 'Login':row[3],
        'Logout':row[4]}

    else:

        entry = create_entry(row[0], JOURNAL_STATE_OPEN)

        entry_collections = {'Username':row[2], 'Login':row[3]}

        #logger.debug(entry_collections)

    entry.update_collected_data(entry_collections)

    EM7_RESULT.append(entry)
```

For each row, the snippet calls the create_entry method. The snippet uses the session_id field from the database as the journal key and determines the state of the new journal entry by using the mapping in

requirement two. The snippet creates a dictionary called entry_collections that contains all available collection objects. For journal entries in an open state, the logout time is not included. The entry_collections dictionary uses the variable names used for each collection object as keys. To associate the collected data with the journal entry, the snippet calls update_collected_data using the entry_collections dictionary as the parameter. To pass the journal entry to Skylar One for storage, the snippet appends the new journal entry to EM7_RESULT.

To get a list of journal entries that must be checked for updates, the snippet calls the bulk_get_open_entries function. The snippet iterates through the list of returned journal entries, querying for each user session:

```
open_entries = bulk_get_open_entries()

for entry in open_entries:

  try:

    cur.execute("""SELECT session_id, state, logout_time FROM master_
    access.access_log WHERE session_id = %s""", (entry.entry_key,))

  except MySQLdb.Error, e:

    update_error = True

    update_problem = "%s: %s" % (e.args[0], e.args[1])
```

The snippet uses the key for each journal entry the WHERE clause of the query. Similar to how errors in the query for new entries are recorded, the update_error and update_problem are used to record errors in this query.

If the query is successful, the snippet checks the state of the user session to see if the corresponding journal entry must change state:

```
if update_error == False:

  result=cur.fetchone()

  if result is None:

    entry.set_state(JOURNAL_STATE_ERROR)

    EM7_RESULT.append(entry)

  elif result[1] == 0:

    entry.close()

    entry_collections = {'Logout':result[2]}

    entry.update_collected_data(entry_collections)
```

```
        EM7_RESULT.append(entry)
    elif result[1] == 3:
        entry.abandon()
        entry_collections = {'Logout':result[2]}
        entry.update_collected_data(entry_collections)
        EM7_RESULT.append(entry)
```

To meet *requirement five*, if the query returns no rows (i.e. the user session no longer in the table), the snippet sets the journal entry to the error state. To meet *requirement four*, if the user session is in state 0 or 3 (logged out or expired), the snippet sets the journal entry to the closed or abandoned state, respectively. If the journal entry is set to a closed or abandoned state, the snippet adds the value for the logout time collection object to the journal entry. If the journal entry has been updated, the snippet appends the updated object to the EM7_RESULT list to pass the updated information back to Skylar One.

The snippet has finished collecting new and existing journal entries. To meet *requirement six*, the snippet must check the create_error and update_error variables for the different error combinations:

```
if update_error == True and create_error == True:
    COLLECTION_PROBLEM = True
    PROBLEM_STR = "No database queries successful: %s: %s" % (create_
    problem, update_problem)
elif update_error == True:
    INTERNAL_ALERTS.append((519, "Query failure when updating journal
    entries, but new entry creation successful. did:%s, app id:%s,
    error: %s" % (app_id, did, update_problem)))
elif create_error == True:
    INTERNAL_ALERTS.append((519, "Query failure when creating journal
    entries, but entry update successful. did:%s, app id:%s, error: %s"
    % (app_id, did, create_problem)))
```

If both the query to collect new journal entries and a query to collect an existing journal entry failed, the snippet sets COLLECTION_PROBLEM to "True" to report a missed poll, and populates PROBLEM_STR with an appropriate error message. If either the query to collect new journal entries or the queries to collect an existing journal entry were successful, the snippet does not report a missed poll, but does generate an internal alert. The internal alert uses alert ID 519, which will trigger a minor event.

To add this snippet code to the example Dynamic Application you created, perform the following steps:

1. Click the **[Snippets]** tab in the **Dynamic Application Editor** pane.
2. Click the wrench icon (🔧) for the "Collect Login Data" snippet.
3. Insert the snippet code in the ***Snippet Code*** field. A full listing of the snippet code is included in the last section.
4. Click the **[Save]** button.

Creating the Presentation Objects

To display the collection objects in the **Journal View** page for this Dynamic Application, the Dynamic Application must include one or more presentation objects. Presentation objects define how collected data will be displayed for each journal entry in the **Journal View** page. This example will use three presentation objects, one for each collection object. To create the presentation objects for this Dynamic Application:

1. Click the **[Presentations]** tab in the **Dynamic Application Editor** pane.
2. Supply values in the following fields to create a presentation object for the username collection object:
    - ***Report Name***. The name of the presentation object. This name will be displayed as a column heading in the table of journal entries in the **Journal View** page. Enter "Username" in this field.
    - ***Summarization State***. Specifies whether Skylar One should display this presentation object in the **Journal View** page. Select *Enabled* in this field.
    - ***Column Order***. Specifies the order in which the columns for each presentation object will be displayed in the **Journal View** page. Columns are displayed in ascending ***Column Order***, from left to right. For this example, the Username presentation will be the first column on the left. Enter "1" in this field.
    - ***Column Width Type***. Specifies whether Skylar One should display the column with a width relative to the other columns or using a fixed width in pixels. This example uses relative column widths. Select *Relative width (x)* in this field.
    - ***Column Width***. Specifies the width of the column in the **Journal View** page, either relative to the other columns or by number of pixels. This example uses relative column widths, with the username column displayed at half the width of the other two presentation object columns. Enter "1" in this field.
    - ***Column data type***. Specifies the data type of the values displayed in this column. Usernames are text strings. Select *String template* in this field.
    - ***Indexing and sorting***. Specifies whether the table of journal entries can be sorted by the values in this column. This example allows sorting on all columns. Select *Enabled (column can be sorted according to data type)* in this field.
    - ***Display formula***. Specifies the collection objects to display in this column. Select *Username* in the select list below the ***Display formula*** field, then click the **[Add]** button.
3. Click the **[Save]** button, then click the **[Reset]** button to clear the values you entered.
4. Repeat steps two and three to create a presentation object for the login time collection object. Use the following values:

- ***Report Name***. Enter "Login Time" in this field.
- ***Active State***. Select *Enabled* in this field.
- ***Column Order***. For this example, the Login Time presentation will be the second column. Enter "2" in this field.
- ***Column Width Type***. This example uses relative column widths. Select *Relative width (x)* in this field.
- ***Column Width***. This example uses relative column widths, with the username column displayed at half the width of the other two presentation object columns. Enter "2" in this field.
- ***Column data type***. Specifies the data type of the values displayed in this column. Login time is stored as a UNIX timestamp. Select *Date and time* in this field. Skylar One will convert the collected values to human-readable timestamps based on each user's preferences.
- ***Indexing and sorting***. This example allows sorting on all columns. Select *Enabled (column can be sorted according to data type)* in this field.
- ***Display formula***. Specifies the collection objects to display in this column. Select *Login Time* in the select list below the ***Display formula*** field, then click the **[Add]** button.

5. Repeat steps two and three to create a presentation object for the logout time collection object. Use the following values:
    - ***Report Name***. Enter "Logout" in this field.
    - ***Active State***. Select *Enabled* in this field.
    - ***Column Order***. For this example, the Logout Time presentation will be the third column. Enter "3" in this field.
    - ***Column Width Type***. This example uses relative column widths. Select *Relative width (x)* in this field.
    - ***Column Width***. This example uses relative column widths, with the username column displayed at half the width of the other two presentation object columns. Enter "2" in this field.
    - ***Column data type***. Specifies the data type of the values displayed in this column. Logout time is stored as a UNIX timestamp. Select *Date and time* in this field. Skylar One will convert the collected values to human-readable timestamps based on each user's preferences.
    - ***Indexing and sorting***. This example allows sorting on all columns. Select *Enabled (column can be sorted according to data type)* in this field.
    - ***Display formula***. Specifies the collection objects to display in this column. Select *Logout Time* in the select list below the ***Display formula*** field, then click the **[Add]** button.

Creating a Credential and Using the Dynamic Application

To use the example Dynamic Application, you must first create a database credential to align with the Dynamic Application. Perform the following steps to create a database credential for a ScienceLogic Database:

1. Go to the **Credential Management** page (System > Manage > Credentials).
2. Click the **[Actions]** menu, then select *Create Database Credential*. The **Credential Editor** page appears.

3. Supply values in the following fields:
    - ***Profile Name***. Enter a name for the credential.
    - ***DB Type***. Specifies which type of database the credential can be used for. Select *MySQL* in this field.
    - ***DB Name***. Specifies the default database for the credential. The example snippet does not use this field, but a value must be specified. Enter "master" in this field.
    - ***DB User***. Specifies the username to use when connecting to the database. Enter "root" in this field.
    - ***Password***. Specifies the password to use when connecting to the database. Enter the password for the "root" user on your ScienceLogic Database in this field.
    - ***Hostname/IP***. Specifies the hostname or IP address to connect to. If you are aligning this Dynamic Application to a ScienceLogic Database modeled as a device in your system, enter "%D" in this field to use the IP address of the aligned device. If you are not aligning this Dynamic Application to a ScienceLogic Database modeled as a device in your system, enter the hostname or IP address of your ScienceLogic Database in this field.
    - ***Port***. Specifies the port to connect to. Enter "7706" in this field.
4. Click the **[Save]** button.

To align this Dynamic Application to a device, perform the following steps:

> **NOTE:** If you entered "%D" in the ***Hostname/IP*** field for your credential, you must align the Dynamic Application to a ScienceLogic Database.

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Click the wrench icon () for the device you want to align with the Dynamic Application. The **Device Properties** page appears.
3. Click the **[Collections]** tab. The **Dynamic Application Collections** page appears.
4. Click the ***Action*** menu and select *Add Dynamic Application*.
5. In the **Dynamic Application Alignment** modal page, select our example Dynamic Application, ***Snippet Journal Example***. Select the credential you created for your ScienceLogic Database.
6. Click the **[Save]** button. The page refreshes, and the *Snippet Journal Example* Dynamic Application appears in the list of Dynamic Applications.

For the Dynamic Application to collect at least one journal entry, a user must log in to the system after the Dynamic Application has been aligned to the device. Log out, then log in to the Skylar One system. To view collected data for this Dynamic Application:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Click the graph icon () for the device you aligned with the Dynamic Application. The **Device Summary** page appears.
3. Click the **[Journals]** tab. The **Journal View** page appears.

4. If multiple Journal Dynamic Applications are aligned with this device, click *Snippet Journal Example* in the left Navbar.
5. By default, only closed journal entries are displayed. Remove "closed" from the ***State*** search filter. All collected journal entries appear on the **Journal View** page.

Full Snippet Code Listing

```
import MySQLdb


COLLECTION_PROBLEM = False

create_error = False

update_error = False


try:

  meta = get_app_instance_meta_data()

except DeserializationError as e:

  INTERNAL_ALERTS.append((519, "Could not deserialize stored meta data for
  did:%s, app id:%s, error: %s" % (app_id, did, e)))


logger.debug("Credential: %s" % (cred_details,))

logger.debug("Collection objects: %s" % (collection_objects,))


if cred_details['cred_type'] != 2:

  COLLECTION_PROBLEM = True

  PROBLEM_STR = "Database credential not aligned"

else:

  try:

    conn = MySQLdb.connect(user=cred_details['cred_user'], passwd=cred_
    details['cred_pwd'], host=cred_details['cred_host'], port=cred_details
    ['cred_port'])

    cur = conn.cursor()
```

```
except MySQLdb.Error, e:

  COLLECTION_PROBLEM = True

  PROBLEM_STR = "Database connection error: %s: %s" % (e.args[0], e.args
  [1])


if COLLECTION_PROBLEM == False:

  if meta is not None and meta.has_key('time'):

    logger.debug("Have valid meta data, using last query time in WHERE
    clause")

    try:

      cur.execute("""(SELECT session_id, state, user, UNIX_TIMESTAMP
      (login_time) as login, UNIX_TIMESTAMP(logout_time) as logout_time
      FROM master_access.access_log WHERE login_time > FROM_UNIXTIME
      (%s)) UNION (SELECT UNIX_TIMESTAMP(NOW()),UNIX_TIMESTAMP(NOW
      ()),UNIX_TIMESTAMP(NOW()),UNIX_TIMESTAMP(NOW()),UNIX_TIMESTAMP(NOW
      ())) ORDER BY login DESC""", (meta['time'],))

    except MySQLdb.Error, e:

      create_error = True

      create_problem = "%s: %s" % (e.args[0], e.args[1])

  else:

    logger.debug("No valid meta data, using 60 seconds ago in WHERE
    clause")

    try:

      cur.execute("""(SELECT session_id, state, user, UNIX_TIMESTAMP
      (login_time) as login, UNIX_TIMESTAMP(logout_time) as logout_time
      FROM master_access.access_log WHERE UNIX_TIMESTAMP(login_time) >
      (UNIX_TIMESTAMP() - 60)) UNION (SELECT UNIX_TIMESTAMP(NOW()),UNIX_
      TIMESTAMP(NOW()),UNIX_TIMESTAMP(NOW()),UNIX_TIMESTAMP(NOW()),UNIX_
      TIMESTAMP(NOW())) ORDER BY login DESC""")

    except MySQLdb.Error, e:
```

```
      create_error = True

      create_problem = "%s: %s" % (e.args[0], e.args[1])

  if create_error == False:

    now = cur.fetchone()

    last_query = now[3]

    if last_query:

      meta_data = {'time':last_query}

      logger.debug("Setting Meta Data: %s" % meta_data)

      try:

        set_app_instance_meta_data(meta_data)

      except SerializationError as e:

        INTERNAL_ALERTS.append((519, "Could not serialize meta data for
        storage for did:%s, app id:%s, error: %s" % (app_id, did, e)))

        logger.debug("SerializationError when setting meta data: %s" %
        e)


    results = cur.fetchall()


    for row in results:

      if row[1] == 0:

        entry = create_entry(row[0], JOURNAL_STATE_CLOSED)

        entry_collections = {'Username':row[2], 'Login':row[3],
        'Logout':row[4]}

      elif row[1] == 3:

        entry = create_entry(row[0], JOURNAL_STATE_ABANDONED)
```

```
            entry_collections = {'Username':row[2], 'Login':row[3],
            'Logout':row[4]}
        else:
            entry = create_entry(row[0], JOURNAL_STATE_OPEN)
            entry_collections = {'Username':row[2], 'Login':row[3]}
            #logger.debug(entry_collections)
        entry.update_collected_data(entry_collections)
        EM7_RESULT.append(entry)

open_entries = bulk_get_open_entries()
for entry in open_entries:
    try:
        cur.execute("""SELECT session_id, state, logout_time FROM master_
        access.access_log WHERE session_id = %s""", (entry.entry_key,))
    except MySQLdb.Error, e:
        update_error = True
        update_problem = "%s: %s" % (e.args[0], e.args[1])

    if update_error == False:
        result=cur.fetchone()
        if result is None:
            entry.set_state(JOURNAL_STATE_ERROR)
            EM7_RESULT.append(entry)
        elif result[1] == 0:
            entry.close()
            entry_collections = {'Logout':result[2]}
```

```
            entry.update_collected_data(entry_collections)

            EM7_RESULT.append(entry)

        elif result[1] == 3:

            entry.abandon()

            entry_collections = {'Logout':result[2]}

            entry.update_collected_data(entry_collections)

            EM7_RESULT.append(entry)


if update_error == True and create_error == True:

    COLLECTION_PROBLEM = True

    PROBLEM_STR = "No database queries successful: %s: %s" % (create_
    problem, update_problem)

elif update_error == True:

    INTERNAL_ALERTS.append((519, "Query failure when updating journal
    entries, but new entry creation successful. did:%s, app id:%s,
    error: %s" % (app_id, did, update_problem)))

elif create_error == True:

    INTERNAL_ALERTS.append((519, "Query failure when creating journal
    entries, but entry update successful. did:%s, app id:%s, error: %s"
    % (app_id, did, create_problem)))
```

Caching and Cache-consuming Snippets

Configuration and performance Snippet Dynamic Applications can:

- Cache a single object that can be pickled. To cache an object, a configuration or performance Snippet Dynamic Application must have *Cache Results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor**. The cached object can be used by other configuration or performance Snippet Dynamic Applications that have *Consume Cached Results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor**.

- Use the cached results from one or more requests or snippets in Dynamic Applications that have *Cache Results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor**. To use the cached results from other Dynamic Applications, a configuration or performance Snippet Dynamic Application must have *Consume Cached Results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor**. Each snippet can consume one cached result. Unlike other Dynamic Application types that can consume cached results:
    - Snippet Dynamic Applications can consume any type of cached result.
    - The snippets defined in Snippet Dynamic Applications that consume cached results are not limited to using only the contents of the cached response. When you configure a Snippet Dynamic Application to consume cached results, your snippet code can still perform other collection and data processing in addition to using the cached response.

For more information about caching, see the ***Dynamic Application Development*** chapter.

Caching a Result in a Snippet

When a Snippet Dynamic Application is configured to *Cache Results*, you can use the following global function to cache an object:

```
em7_cache_result(object)
```

When you cache an object using the em7_cache_result() function:

- The em7_cache_result() function takes a single argument. You should pass the single object that you want to cache.
- Skylar One will pickle the passed object before the object is stored. Python must be able to pickle the object you pass to the em7_cache_result() function. Do not pickle the object in your snippet code before passing it to the em7_cache_result() function.

NOTE: Remember, the collection objects defined in a Dynamic Application that caches its results are no longer collected at a regular frequency. You must not define collection objects that are used to generate configuration tables or performance graphs in a Dynamic Application that caches results. Typically, Dynamic Applications that cache results will contain only a discovery object.

Using a Cached Result in a Snippet

When a Snippet Dynamic Application is configured to *Consume Cached Results*, the ***Cached object*** drop-down list appears in the **Dynamic Applications Snippet Editor & Registry** (**[Snippets]** tab) for that Dynamic Application.

The ***Cached object*** drop-down list will contain all requests defined for Dynamic Applications that have *Cache Results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor**. Select the cached result you want to use in your snippet code.

NOTE: Remember, the Dynamic Application that populates the cache must be aligned to the same device (or for component devices, the root device) as the Dynamic Application that consumes the cache.

When a request is selected in the ***Cached object*** drop-down list, the variable EM7_CACHED_OBJECT will be available to the snippet code. The EM7_CACHED_OBJECT variable will contain the cached result. The structure of EM7_CACHED_OBJECT is different based on the type of cached result:

- For results cached by Snippet Dynamic Applications, the contents of EM7_CACHED_OBJECT is the object that was cached using the em7_cache_result() function. Skylar One will automatically unpickle the object; you do not need to unpickle the object in your snippet code.
- For results cached by SOAP, XML, and XSLT Dynamic Applications, the contents of EM7_CACHED_OBJECT is the returned XML document. For XSLT Dynamic Applications, the XML document is the response from the device **before** the transformation is applied.
- For results cached by WMI Dynamic Applications, EM7_CACHED_OBJECT is a dictionary of values:
    - The keys in the EM7_CACHED_OBJECT dictionary are the property names (columns) from the response.
    - For convenience, each property has two keys in the EM7_CACHED_OBJECT dictionary: one is the unmodified property name, one is the property name converted to all lowercase. The values associated with the unmodified and lowercase keys will be identical for the same property name.
    - Each value in the dictionary is a list of values returned for that property. The key for each value in the list is the corresponding WMI Object Key value.

Excluded Modules and Additional Functions

This section contains a list of Python modules and other functions that are not permitted in Snippet Dynamic Applications.

Excluded Python Modules

To maintain security and maximize performance, the following Python modules are not permitted in Snippet Dynamic Applications:

- os
- os.path
- pickle
- shelve
- platform
- thread

- threading
- dummy_thread
- dummy_threading
- posix
- pwd
- spwd
- grp
- termios
- pipes
- posixfile
- resource
- commands
- subprocess
- signal
- popen2
- imp

Snippet Functions for Database, SNMP, and CURL Handles

Skylar One includes APIs that return database, SNMP, and CURL handles. You must supply a credential ID to the API, and the API returns the handle.

Database Handles

You can use the following method to obtain a database handle:

```
em7_snippets.dbc_from_cred_id(cred_ID)
```

The dbc_from_cred_id method takes the following arguments:

- *cred_ID*. Supply a numeric ID for a credential. This is the credential for the database for which you want to retrieve a handle.

The dbc_from_cred_id method can generate three types of exceptions:

- If the credential is not a database credential, the method returns the ValueError "Credential Type is not database (2)."
- If the credential does not exist, the method returns the ValueError "Credential %s does not exist."
- If the API cannot connect to the database, the method will generate a runtime error.

The method returns a cursor that adheres to the specification in the Python DB API 2.0. For information on the methods that are supported by cursors that adhere to the Python DB API 2.0, see http://www.python.org/dev/peps/pep-0249/.

NOTE: The API uses the cursor_wrapper function. The cursor_wrapper allows cursors to use auto-fetch methods.

SNMP Handles

You can use the following method to obtain an SNMP handle:

```
snmph_from_cred_id(cred_ID, IP, write=False)
```

The snmph_from_cred_id method takes the following arguments:

- *cred_ID*. Supply a numeric ID for a credential. This is the SNMP credential for which you want to retrieve a handle.
- *IP*. Supply an IP address for the device for which you want to retrieve an SNMP handle.
- *write=false*. Optional argument. Specifies whether the credential is an SNMP write credential. Default value is "false".

The snmph_from_cred_id method can generate two types of exceptions:

- If the credential is not an SNMP credential, the API returns the ValueError "Credential Type is not SNMP (1)."
- If the credential does not exist, the API returns the ValueError "Credential %s does not exist."

The returned handle supports the following methods:

- ***.get(oid)***. Takes an SNMP OID as a parameter, returns the response from an SNMP get request. On error, returns None.
- ***.walk(oid)***. Takes an SNMP OID as a parameter, returns the response from an SNMP walk request. Returns a list of tuples. Each tuple includes the SNMP index and the returned value from an OID in the response.
- ***.set(oid, datatype, value)***. Takes an SNMP OID, an SNMP datatype, and a value to set as parameters. Performs an SNMP set request on the specified OID using the specified value. Returns the response from the SNMP agent.

cURL Handles

You can use the following method to obtain a cURL handle:

```
curlh_from_cred_id(cred_ID, IP, hostname, logger=none)
```

The curlh_from_cred_id method takes the following arguments:

- *cred_ID*. Supply a numeric ID for a credential. This is the CURL credential for which you want to retrieve a handle.
- *IP*. Supply an IP address. The API will replace the %D variable in the credential URL with the IP address.

- *hostname*. Supply a hostname. The API will replace the %N variable in the credential URL with the hostname.

The curlh_from_cred_id method can generate four types of exceptions:

- If the credential is not a CURL credential, the API returns the ValueError "Credential Type is not CURL(3)."
- If the credential does not exist, the API returns the ValueError "Credential %s does not exist."
- If the CURL options are invalid, the API generates a TypeError.
- If the CURL options are invalid, the API generates a pycurl.error.

The method returns a cURL object using the pycurl library. For more information about the methods that are supported by cURL objects provided by the pycurl library, see http://pycurl.sourceforge.net/.

Snippet Functions for Polling Nagios Agents

Because command-line system calls are blocked in snippet code, Skylar One provides access to the check_nrpe and check_nt command line tools through wrapper functions. The check_nrpe and check_nt tools can be used to poll Nagios agents running on external systems.

For Linux-based external systems, you can use the following method:

```
em7_snippets.call_nrpe(host, command, arglist, use_ssl, critical_timeout,
port, timeout)
```

The call_nrpe method takes the following arguments:

- ***host***. The IP address of the external system.
- ***command***. The command option to use. The external system must have the NPRE daemon configured to associate this command option with a specific plug-in command.
- ***arglist***. An iterable list of arguments to pass to the command. If this argument is not passed, the default value is "None".
- ***use_ssl***. A boolean that specifies whether SSL should be used. If this argument is not passed, the default value is "True".
- ***critical_timeout***. A Boolean that specifies whether timeouts should return CRITICAL (critical_ timeout=True) or UNKNOWN (critical_timeout=False). If this argument is not passed, the default value is "True".
- ***port***. The port that should be used to connect to the external system. If this argument is not passed, the default value is 5666.
- ***timeout***. The number of seconds to wait for a response before a timeout occurs. If this argument is not passed, the default value is 10.

For Windows-based external systems, you can use the following method:

```
em7_snippets.call_nrpe_nt(host, variable, port, password, warning_
threshold, critical threshold, params, timeout, unknown_timeout)
```

The call_nrpe_nt method takes the following arguments:

- ***host***. The IP address of the external system.
- ***variable***. The variable to use when collecting data from the NSClient service.
- ***port***. The port that should be used to connect to the external system. If this argument is not passed, the default value is 1248.
- ***password***. The password for the external system. If this argument is not passed, the default value is "None".
- ***warning_threshold***. The threshold value to use when determining whether the response should indicate a warning state. If this argument is not passed, the default value is "None".
- ***critical_threshold***. The threshold value to use when determining whether the response should indicate a critical state. If this argument is not passed, the default value is "None".
- ***params***. Parameters to pass to the external system. If this argument is not passed, the default value is "None".
- ***timeout***. The number of seconds to wait for a response before a timeout occurs. If this argument is not passed, the default value is 10.
- ***unknown_timeout***. A boolean that specifies whether timeouts should return CRITICAL (unknown_timeout=False) or UNKNOWN (unknown_timeout=True). The default value if this argument is not passed is "False".

Both em7_snippets.call_nrpe and em7_snippets.call_nrpe_nt return the following tuple:

```
(return code, stdout, stderr)
```

Skylar One does not process the return code, output, or error. The values in the returned tuple are the values returned by the check_nrpe or check_nt command line tool.

Snippet Functions for Performing WMI and WBEM Requests

Because command-line system calls are blocked in snippet code, Skylar One provides a wrapper function that provides access to a command line tool for performing WMI and WBEM requests. This wrapper function can be used by snippet code in Performance & Configuration Dynamic Applications.

Both wrapper functions require the parameter cred_details. The cred_details parameter must be a dictionary with the same structure as self.cred_details. If the credential aligned with the Dynamic Application is a Basic/Snippet credential that can be used to perform WMI requests, you can pass self.cred_details in the cred_details parameter. If you are constructing a new dictionary to pass in the cred_details parameter, you must include the following keys:

- ***cred_host***. The hostname or IP address of the device for which the snippet is collecting.
- ***cred_port***. The TCP port that will be used to connect to the WMI or WBEM agent.
- ***cred_user***. The username to use to perform the request.
- ***cred_pwd***. The password for the user specified in ***cred_user***.
- ***cred_timeout***. The timeout to use for the request.

To perform a WMI request, use the following method:

```
em7_snippets.wmi_request(did, app_id, ip, cred_details, query, key,
delimiter, namespace)
```

The wmi_request method takes the following arguments:

- ***did***. The device ID of the device for which the snippet is collecting. Pass self.did for this parameter.
- ***app_id***. The ID for this Dynamic Application. Pass self.app_id for this parameter.
- ***ip***. The IP address of the device the method will query. To use the IP address of the device the Dynamic Application is currently collecting from, pass self.ip for this parameter.
- ***cred_details***. The credential the method will use to perform the request. You must pass a dictionary with the same structure as self.cred_details.
- ***query***. The WMI query to execute.
- ***key***. The unique key for each instance (row) returned by the request. This unique key must be a property name, and the request must include that property (column) and return values from that property name (column).
- ***delimiter***. When making a request, the command-line utility must specify a string of characters that will be used to separate the values returned in the response. Specify that string of characters in this parameter. The string of characters you pass must not appear in any value that could be included in the response. To use the default delimiter, "$$$", pass "WMI_DELIMITER" in this parameter.
- ***namespace***. Specify the namespace for the request.

To perform a WBEM request, use the following method:

```
em7_snippets.wbem_request(did, app_id, ip, cred_details, query)
```

The wbem_request method takes the following arguments:

- ***did***. The device ID of the device for which the snippet is collecting. Pass self.did for this parameter.
- ***app_id***.The ID for this Dynamic Application. Pass self.app_id for this parameter.
- ***ip***. The IP address of the device the method will query. To use the IP address of the device the Dynamic Application is currently collecting from, pass self.ip for this parameter.
- ***cred_details***. The credential the method will use to perform the request. You must pass a dictionary with the same structure as self.cred_details.
- ***query***. The WBEM query to execute.

Both the wmi_request and wbem_request methods return a dictionary:

- The keys in the dictionary are the property names (columns) from the response.
- Each value in the dictionary is a list of values returned for that property. For WMI requests, the key for each value in the list is the corresponding value returned by the property (column) you specified in the ***key*** parameter. For WBEM requests, the key for each value in the list is the corresponding value from the ***Name*** property (column).

For example, suppose you requested two properties, Name and Value, and specified Name in the key parameter. Suppose that the response includes the following three rows:

Name	Value
Bulbasaur	Grass
Charmander	Fire
Squirtle	Water

The returned dictionary has the following structure:

```
{

'Name': {'Bulbasaur':'Bulbasaur', 'Charmander':'Charmander',
'Squirtle':'Squirtle'},

'Value': {'Bulbasaur':'Grass', 'Charmander':'Fire', 'Squirtle':'Water'}

}
```

Caching Results Between Polling Periods

Skylar One includes an API that allows snippets to save values to a cache and make those values available for use in the same snippet in the same Dynamic Application.

This is most useful for storing values between polling periods. For example:

- You could store the last collection time between poll periods and use that last collection time during the following poll.
- Some APIs, such as the VMware vSphere API, allow you to perform an initial request that includes all available information and then perform subsequent requests that return only the values that have changed since the previous request. You could use the cache_result API to store the results of the initial request and update the results during each subsequent execution of the snippet.
- You could use the cache_result API to store a session handle for a SOAP web service.

Caching Values

To instantiate the cache, use the following method:

```
cache = em7_snippets.cache_api(self)
```

Each entry in the cache is associated with a key. The key is used to retrieve an entry from the cache. If you are storing only a single object in the cache, you can use the default key, which is created using the ID of the Dynamic Application and the ID of the device for which collection is being performed. You can create a custom key using the following method:

```
cache.generate_key(app_id=None, did=None, timestamp=None, use_
timestamp=True, **kwargs)
```

The key is generated using the values for Dynamic Application ID, Device ID, current timestamp, and optional custom arguments. To retrieve a value from the cache, the same key must be supplied. You can control the values that are used to generate the key by supplying values for the following arguments when calling the generate_key method:

- ***app_id***. The Dynamic Application ID to use instead of the current Dynamic Application ID.
- ***did***. The device ID to use instead of the current device ID.
- ***timestamp***. The timestamp to use instead of the current timestamp.
- ***use_timestamp***. A boolean that controls whether the timestamp will be used to generate the key.
- *****kwargs***. You can specify multiple additional parameters that will be used to generate the key.

You can then use the following method to store values in the cache:

```
cache.cache_result(result, ttl=None, commit=False, key=None)
```

The cache_result method takes the following arguments:

- ***result***. The object to be cached.
- ***ttl***. Optional argument. The default value is *None*. The number of seconds this value should live in the cache. If you supply the value "0", the object will be cached indefinitely.
- ***commit***. Optional argument. The default value is *False*. Specifies whether you want to save the object to cache immediately. Choices are:
  - *commit=True*. Save the object to cache immediately.
  - *commit=False*. Do not save the object to cache immediately.
- ***key***. Optional argument. The default value is *None*. Specifies whether to override the cache key. Choices are:
  - *key=None*. Accept the default cache key.
  - *key=keyname*. Override the cache key and use the specified cache key instead.

If you do not commit a value to the cache using the arguments for the cache_result method, you can use the following method to commit all uncommitted values to the cache:

```
cache.commit()
```

Retrieving Cached Values

To retrieve a cached value, you must have instantiated the cache:

```
cache=em7_snippets.cache_api(self)
```

You can use the following methods to retrieve a value from the cache:

```
cache.get(key)
```

```
cache.get_multi(keys)
```

The get method allows you to retrieve a single value from the cache and takes the following argument:

- ***key***. Specify the name of the key associated with the object you want to retrieve.

The get_multi method allows you to retrieve multiple values from the cache and takes the following argument:

- ***keys***. Specify a list of keys including a key for each object you want to retrieve from the cache.

Example Snippet Dynamic Applications

This section walks through three practical examples of Snippet Dynamic Applications, each demonstrating a different approach to collection.

The first example builds a simple Performance Snippet Dynamic Application that generates random numbers within a defined range, making it a useful starting point for understanding how snippet arguments, collection objects, and the `result_handler` dictionary work together without the complexity of a network connection.

The second example shows how to use snippet code to perform SNMP collection, useful for cases where Skylar One's built-in SNMP collection isn't flexible enough, such as when OIDs need to be dynamically assembled from the results of earlier polls.

The third example demonstrates how to collect performance data from a device over a Telnet connection, showing how snippets can interact with devices that only expose data through a command-line interface.

Together, these examples cover a range of snippet development patterns and can serve as starting points for your own collection workflows.

Example 1: A Random Number Generator Dynamic Application

This example describes the development of a Performance Snippet Dynamic Application that generates a random number within a specified range. The results are then presented in a graph. For simplicity, this example does not use credentials or network connections.

Creating the Dynamic Application

To create this example Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. In the **Dynamic Applications Manager** page, select the **[Actions]** button, then select *Create New Dynamic Application*. The **Create New Application** page is displayed.
3. Supply values in the following fields:
    - ***Application Name***. The name of the Dynamic Application. This example is called "Snippet Random Number Example".
    - ***Application Type***. This example is a Snippet Performance Dynamic Application. Select

Snippet Performance in this field.

- ***Poll Frequency***. To see data as quickly as possible, select *Every 1 Minute* in this field.

4. This example does not have specific requirements for the other settings defined in this page. You can leave the remaining fields set to the default values.
5. Select the **[Save]** button.

Because each collection object is assigned a specific snippet, you must create the container for the snippet code before creating the collection objects for this Dynamic Application. To create a container for the snippet code, perform the following steps:

1. Select the **[Snippets]** tab.
2. Supply values in the following fields:
   - ***Snippet Name***. The name of the snippet. The snippet in this example is called "Generate Random Number".
   - ***Active State***. To ensure that the snippet code is run by Skylar One, select *Enabled* in this field.
   - ***Snippet Code***. Leave this field blank. You will add the snippet code later in this example.
3. Select the **[Save]** button.

Creating the Collection Objects

Each collection object defines a nuermic range by specifying a high value and low value within which the random number must fall. This example will include two collection objects:

- ***Random 17-20***. The snippet will generate random numbers between 17 and 20.
- ***Random 30-50***. The snippet will generate random numbers between 30 and 50.

For each collection object, the argument will specify the range the random number must fall within for that collection object.

To create the collection objects for this Dynamic Application, perform the following steps:

1. Select the **[Collections]** tab in the **Dynamic Application Editor** pane.
2. Supply values in the following fields to create the ***Random 17-20*** object:
   - ***Object Name***. The name of the collection object. Enter "Random 17-20" in this field.
   - ***Snippet Arguments***. The arguments to pass to the snippet. Enter "17 20" in this field.
   - ***Class Type***. The collected values for this collection object will be numeric values that can go up or down. Select *4 Performance Gauge* in this field.
   - ***Snippet***. There is only one snippet for this Dynamic Application. Select *Generate Random Number* in this field.
3. This example does not have specific requirements for the other fieldss. You can accept the default values for the remaining fields.
4. Select the **[Save]** button, then select the **[Reset]** button to clear the values you entered.
5. Repeat steps two, three, and four for the ***Random 30-50*** collection object using the following values:
   - ***Object Name***. Enter "Random 30-50".
   - ***Snippet Arguments***. Enter "30 50".

- ***Class Type***. Select *4 Performance Gauge*.
- ***Snippet***. Select *Generate Random Number*.

Snippet Code

When the snippet in this Dynamic Application begins execution, the items to be collected are in the ***result_handler*** dictionary. The ***result_handler*** dictionary includes the following:

- For each collection object ***argument*** (the "oid" value), the ***result_handler*** dictionary includes a ***key***. The key has the has the same name as the argument. The argument is defined in the ***Snippet Arguments*** field for each collection object.
- For each key, there is a dictionary of collection object attributes. Each attribute is a tuple that contains two values: an *index* followed by a *result* string.
- Often the only attribute of interest are the result_handler dictionary ***keys*** themselves.

The ***result_handler*** dictionary for this Dynamic Application has the following structure, with the ***Object Name*** and ***Snippet Argument*** values highlighted in blue:

```
{

'30 50': {'prime': 0, 'error_msg': '', 'enum': '', 'name': 'Random 30-50',

'oid': '30 50', 'string_type': 0, 'class': 4, 'wm_walk_length': '', 'oid_
time': '0', 'result': '', 'oid_type': 9, 'factor': '', 'trend_col': '',
'monitor_config': 0},

'17 20': {'prime': 0, 'error_msg': '', 'enum': '', 'name': 'Random 17-20',

'oid': '17 20', 'string_type': 0, 'class': 4, 'wm_walk_length': '', 'oid_
time': '0', 'result': '', 'oid_type': 9, 'factor': '', 'trend_col': '',
'monitor_config': 0}

}
```

Before the end of a successful collection, the snippet code must assign the result of the collection to ***result_handler***. The result for each collection object must be a list of tuples. Each result tuple contains two values: an *index* followed by a *result* string. The index can be any scalar type, but no two tuples can have the same index value. There are two methods for updating the ***result*** value for a collection object using the ***result_handler*** dictionary:

- Assign a value directly to the key for that collection object. Remember that the key for a collection object is the value in the ***Snippet Arguments*** field.
- Update the result value for all the collection objects at the same time by using the **update()** function of ***result_handler***. The *results* parameter passed to the function must be a dictionary that has the same keys as ***result_handler***. Each key in the supplied dictionary must reference that value to be set as the result value for that collection object.

This example uses the first method to return results.

The following steps walk through each section of code in this example:

Import the random module, which is needed to generate random numbers:

```
import random
```

Loop through the keys in ***result_handler***. This will allow the snippet to operate on the arguments for each collection object in turn:

```
for collection in result_handler.iterkeys():
```

The low and high range values are extracted from the collection object. The snippet then validates the range values. The COLLECTION_PROBLEM and PROBLEM_STR variables are used to report invalid values:

```
range_values = collection.split()

if len(range_values) != 2 or not range_values[0].isdigit() or not
range_values[1].isdigit:

  COLLECTION_PROBLEM = True

  PROBLEM_STR = "App:%s, Snippet:%s, Invalid collection value:%s" %
  (self.app_id, req_id, collection)

  continue


low = int(range_values[0])

high = int(range_values[1])
```

If COLLECTION_PROBLEM is TRUE at the end of collection, the value of PROBLEM_STR will be used to generate an event for the associated device.

The snippet then generates the random number and assigns a list that contains a single tuple to the collection object key in result_handler. In this case there is only one value being returned for each collection object, so the index is fixed as the value 0:

```
rnum = str(random.randint(low, high))

result_handler[collection] = [(0, rnum),]
```

Using the Dynamic Application

For performance Dynamic Applications, Skylar One automatically creates presentation objects that correspond to each collection object. To add the snippet code for this example to the example Dynamic Application you created, perform the following steps:

1. Select the **[Snippets]** tab in the **Dynamic Application Editor** pane.
2. Select the wrench icon ( ) for the "Generate Random Number" snippet.
3. Insert the snippet code in the ***Snippet Code*** field. A full listing of the snippet code is included in the last section.
4. Select the **[Save]** button.

Perform the following steps to align this Dynamic Application to a device:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Select the wrench icon ( ) for the device you want to align the Dynamic Application with. The **Device Properties** page is displayed.
3. Select the **[Collections]** tab. The **Dynamic Application Collections** page is displayed.
4. Select the ***Action*** menu and choose *Add Dynamic Application*.
5. In the **Dynamic Application Alignment** modal page, select our example Dynamic Application, ***Snippet Random Number Example***. Select ***Default SNMP Credential***.
6. Select the **[Save]** button. The page refreshes, and the *Snippet Random Number Example* Dynamic Application is displayed in the list of Dynamic Applications.

The first collected value will be stored within one minute. To view collected data for this Dynamic Application:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Select the graph icon ( ) for the device you aligned the Dynamic Application with. The **Device Summary** page is displayed.
3. Select the **[Performance]** tab. The **Device Performance** page is displayed.
4. Select *Snippet Random Number Example* in the left Navbar, then select one of the two presentations for this Dynamic Application. A performance graph will be displayed.

Full Snippet Code Listing

```
import random

for collection in result_handler.iterkeys():

  range_values = collection.split()

  if len(range_values) != 2 or not range_values[0].isdigit() or not range_
values[1].isdigit:

    COLLECTION_PROBLEM = True

    PROBLEM_STR = "App:%s, Snippet:%s, Invalid collection value:%s" %
(self.app_id, req_id, collection)
```

```
        continue

    low = int(range_values[0])

    high = int(range_values[1])

    rnum = str(random.randint(low, high))

    result_handler[collection] = [(0, rnum),]
```

Example 2: Using Snippets to Collect SNMP Data

Snippets can be used to implement a variety of collection methods, including those that are built into Skylar One, such as SNMP collection. Although it is more convenient to use the built-in collection methods where possible, there are use cases when using a Snippet Dynamic Application for SNMP collection is appropriate. For example, you could use a Snippet Dynamic Application when you need to dynamically assemble OIDs using the results of several previous polls to gather the correct indexes to use. To create these types of OIDs with multiple indices, you can use a snippet that performs the polling and includes custom logic for assembly of special OIDs.

This section walks through snippet code that performs general purpose SNMP collection, and can be used as a starting point for special-purpose SNMP collection.

To use this example snippet code in a Dynamic Application, the value in the ***Snippet Argument*** field for each collection object must be an SNMP OID, the same way OIDs are specified for an SNMP Dynamic Application.

Creating the Dynamic Application

To create this example Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Select the **[Actions]** button, then *Create New Dynamic Application*. The **Create New Application** page is displayed.
3. Supply values in the following fields:
    - ***Application Name***. The name of the Dynamic Application. This example is called "Snippet SNMP Example".
    - ***Application Type***. This example is a Snippet Performance Dynamic Application. Select *Snippet Performance* in this field.
    - ***Poll Frequency***. To see data as quickly as possible, select *Every 1 Minute* in this field.
4. This example does not have specific requirements for the other settings defined in this page. You can leave the remaining fields set to the default values.
5. Select the **[Save]** button.

Creating a Container for the Snippet

Because collection objects are assigned a specific snippet, you must create the container for the snippet code before creating the collection objects for this Dynamic Application. To create a container for the snippet code, perform the following steps:

1. Select the **[Snippets]** tab.
2. Supply values in the following fields:
    - ***Snippet Name***. The name of the snippet. The snippet in this example is called "SNMP Collection".
    - ***Active State***. To ensure that the snippet code is run by Skylar One, select *Enabled* in this field.
    - ***Required***. Specifies whether this snippet is required for successful collection of all other snippet requests. Select *Required - Stop Collection*. If this snippet request fails, the platform will not attempt to execute any other snippet requests in this Dynamic Application. Dynamic Applications that consume the cache of this Dynamic Application will halt collection.
    - ***Snippet Code***. Leave this field blank. You will add the snippet code later.
3. Select the **[Save]** button.

Creating the Collection Objects

The snippet code for this example will retrieve SNMP data about the file systems on a device and write that data to the ***result_handler*** dictionary. Our collection objects will reference OID names from the HOST-RESOURCES-MIB that contain file system information.

To create the collection objects for this Dynamic Application, perform the following steps:

1. Select the **[Collections]** tab in the **Dynamic Application Editor** pane.
2. Supply values in the following fields to create the first object, which will collect the values from the ***hrStorageDescr*** OID:
    - ***Object Name***. The name of the collection object. Enter "hrStorageDescr" in this field.
    - ***Snippet Arguments***. The arguments to pass to the snippet. Enter ".1.3.6.1.2.1.25.2.3.1.3" in this field. This is the numeric OID name for the OID.
    - ***Class Type***. Select *104 Label (Always Polled)* in this field.
    - ***Snippet***. There is only one snippet for this Dynamic Application. Select *SNMP Collection* in this field.
    - ***Group Number***. Select *Group 1*.
    - ***Usage Type***. Select *Group Index*.
3. For the remaining fields, accept the default values. Select the **[Save]** button, then select the **[Reset]** button to clear the values you entered.
4. Repeat steps two and three to create the collection object that will collect the values from the ***hrStorageSize*** OID:

- ***Object Name***. Enter "hrStorageSize".
- ***Snippet Arguments***. The arguments to pass to the snippet. Enter ".1.3.6.1.2.1.25.2.3.1.5" in this field.
- ***Class Type***. Select *4 Performance Gauge* in this field.
- ***Snippet***. There is only one snippet for this Dynamic Application. Select *SNMP Collection* in this field.
- ***Group Number***. Select *Group 1*.

5. For the remaining fields, accept the default values. Select the **[Save]** button, then select the **[Reset]** button to clear the values you entered.
6. Repeat steps four and five to create the collection object that will collect the values from the ***hrStorageUsed*** OID:
   - ***Object Name***. Enter "hrStorageUsed".
   - ***Snippet Arguments***. The arguments to pass to the snippet. Enter ".1.3.6.1.2.1.25.2.3.1.6" in this field.
   - ***Class Type***. Select *4 Performance Gauge* in this field.
   - ***Snippet***. There is only one snippet for this Dynamic Application. Select *SNMP Collection* in this field.
   - ***Group Number***. Select *Group 1*.
7. For the remaining fields, accept the default values. Select the **[Save]** button.

Full Snippet Code

To enter the Snippet code:

1. Select the **[Snippets]** tab.
2. In the **Snippet Registry** pane, select the *SNMP Collection* snippet in the ***Snippet Name*** drop-down and select its wrench icon (🔧).
3. In the ***Snippet Code*** field, enter the following code:

```
try:
    snmp_h = em7_snippets.snmph_from_cred_id(self.cred_details['cred_id'],
    self.ip)
except ValueError, e:
    COLLECTION_PROBLEM = True
    PROBLEM_STR = e
else:
    for collection in result_handler.iterkeys():
```

```
if collection[-2:] == ".0":
  oid_val = snmp_h.get(collection)
  walk_tuple = ((oid, oid_val[0]),)
else:
  walk_tuple = snmp_h.walk(collection)
result_list = []
for row in walk_tuple:
  result_list.append((row[0].split('.')[-1], row[1]))
result_handler[collection] = result_list
```

Creating the Presentation Object

When you create a collection object in a Dynamic Application of type Performance, Skylar One automatically creates a presentation object that corresponds to that collection object. In this example, we will remove these presentation objects and create a new presentation object that displays file system usage in percent.

To create the presentation object:

1. Select the **[Presentations]** tab. The **Dynamic Applications Presentation Objects** page appears.
2. In the **Dynamic Applications Presentation Objects** page, the *Storage Size* and *Storage Used* collection objects have been created by default. Select each presentation object's delete icon ( ) to delete them.
3. Select the **[Reset]** button. To create a new presentation object that displays storage used, in percent, enter values in the following fields:
   - **Report Name**. Enter "Percentage Used" in this field.
   - **Summarization State**. Select *Enabled*. Skylar One will generate a report of the presentation object.
   - **Data Unit**. Enter "Percent" into this field.
   - **Abbreviation/Suffix**. Enter "%" into this field.
   - **Show as Percent**. Select *Yes*. The graph will display percent values.
   - **Formula Editor**. We want our graph to display Percentage Used, so we will use the following formula:

     *Storage Used/Storage Size * 100*

     We can enter this formula manually, or by selecting the Object IDs in the ***Formula Editor*** and selecting the **[Add]** button. Using the Object IDs provided by Skylar One, enter the following into the ***Formula Editor***:

     *(o_5511)/(o_5513)*100*

> **NOTE**: The object IDs in your Dynamic Application will be unique to your system. In the provided formula, you must replace "o_5511" and "o_5513" with the object IDs for the ***Storage Used*** and ***Storage Size*** collection objects in your system.

4. In this example, you can leave the remaining fields at their default value. Select the **[Save As]** button to save the presentation object.

Snippet Walkthrough

```
try:
    snmp_h = em7_snippets.snmph_from_cred_id(self.cred_details['cred_id'],
    self.ip)
except ValueError, e:
    COLLECTION_PROBLEM = True
    PROBLEM_STR = e
```

- The snippet uses the function snmph_from_cred_id to retrieve the SNMP handle. The function stores the SNMP handle in the variable ***snmp_h***.
- If the function ***snmph_from_cred_id*** fails, the snippet retrieves the error message from the ***ValueError*** variable and stores the error message in the variable ***e***.
- If an exception occurs, the snippet sets the value of ***COLLECTION_PROBLEM*** to *TRUE*, to indicate a collection error has occurred and the snippet sets the value of ***PROBLEM_STR*** to the error message stored in the variable ***e***.

```
else:
  for collection in result_handler.iterkeys():
    if collection[-2:] == ".0":
      oid_val = snmp_h.get(collection)
      walk_tuple = ((oid, oid_val[0]),)
    else:
      walk_tuple = snmp_h.walk(collection)
    result_list = []
    for row in walk_tuple:
      result_list.append((row[0].split('.')[-1], row[1]))
    result_handler[collection] = result_list
```

- If the **snmph_from_cred_id** function is successful, the snippet iterates over the result_handler dictionary. During each iteration "collection" is set to the key from the result_handler dictionary, which are the ***Snippet Arguments*** for each collection object. On each iteration, the **collection** variable will contain an SNMP OID to collect.
- If the last two characters in the **collection** variable are ".0", the snippet uses the **snmp_h.get** method (an SNMP get) to request the OID. The response is re-formatted as a list that contains a single tuple, which is the format for returned values.
- If the last two characters in the **collection** variable are not ".0", the snippet uses the **snmp_h.walk** method (an SNMP walk) to request the OID. The response is already formatted as a list of tuples, so no additional processing is required.

- The first element in each tuple in the walk_tuple list is the full OID for the collection. The platform expects the first element in each tuple returned for a collection object to be the index for that value. The indexes are used to associate collection objects in the same group. In this example, the name, size, and usage values for a given file system should have the same index assigned. The SNMP index is used for this purpose by iterating over the walk_tuple list to create a new list.
- The result for each collection object is returned by assigning the list of tuples to the appropriate key in the **result_handler** dictionary. The **result_handler** dictionary can be used because all the collection objects in this example are in the same group.

Using the Dynamic Application

To use the example Dynamic Application, you must first create an SNMP credential to align with the Dynamic Application. That SNMP credential must allow Skylar One to retrieve data from each subscriber device.

You can use an existing SNMP credential or create a new SNMP credential. In our example, we used the default SNMP credential EM7 Default V2.

To create a new SNMP credential, perform the following steps:

1. Go to the **Credential Management** page (System > Manage > Credentials).
2. Select the **[Actions]** menu, then select *Create SNMP Credential*. The **Credential Editor** page is displayed.
3. Supply values in the following fields:
    - ***Profile Name***. Name of the profile. Can be any combination of alphanumeric characters.
    - ***SNMP Version.*** SNMP version. Choices are SNMP V1, SNMP V2, and SNMP V3.
    - ***Port***. Port Skylar One will use to communicate with the external device or application.
    - ***Timeout (ms)***. Time, in milliseconds, after which Skylar One will stop trying to communicate with the SNMP device.
    - ***Retries***. Number of times Skylar One will try to authenticate and communicate with the external device.

    SNMP V1/V2 Settings

    These fields appear only if you selected SNMP V1 or SNMP V2 in the ***SNMP Version*** field. Otherwise, these fields are grayed out:

    - ***SNMP Community (Read Only)***. The SNMP community string (password) required for read-only access of SNMP data on the remote device or application.
    - ***SNMP Community (Read/Write)***. The SNMP community string (password) required for read and write access of SNMP data on the remote device or application.

    SNMP V3 Settings

    These fields appear only if you selected SNMP V3 in the ***SNMP Version*** field. Otherwise, these fields are grayed out:

    - ***Security Name***. Name used for SNMP authentication.
    - ***Security Pass Phrase***. Password used to authenticate the credential.

- ***Authentication Protocol***. Select an authentication algorithm for the credential. Choices are MD5 or SHA.
- ***Security Level***. Specifies the combination of security features for the credentials. Choices are:
    - *No Authentication / No Encryption*
    - *Authentication Only*
    - *Authentication and Encryption*
- ***SNMP v3 Engine ID***. The unique engine ID for the SNMP agent you want to communicate with. (SNMPv3 authentication and encryption keys are generated based on the associated passwords and the engine ID.)
- ***Context Name***. A context is a mechanism within SNMPv3 (and AgentX) that allows you to use parallel versions of the same MIB objects. For example, one version of a MIB might be associated with SNMP Version 2 and another version of the same MIB might be associated with SNMP Version 3. For SNMP Version 3, specify the context name in this field.
- ***Privacy Protocol***. The privacy service encryption and decryption algorithm. Choices are *DES* or *AES*.
- ***Privacy Protocol Pass Phrase***. Privacy password for the credential.

4. Select the **[Save]** button.

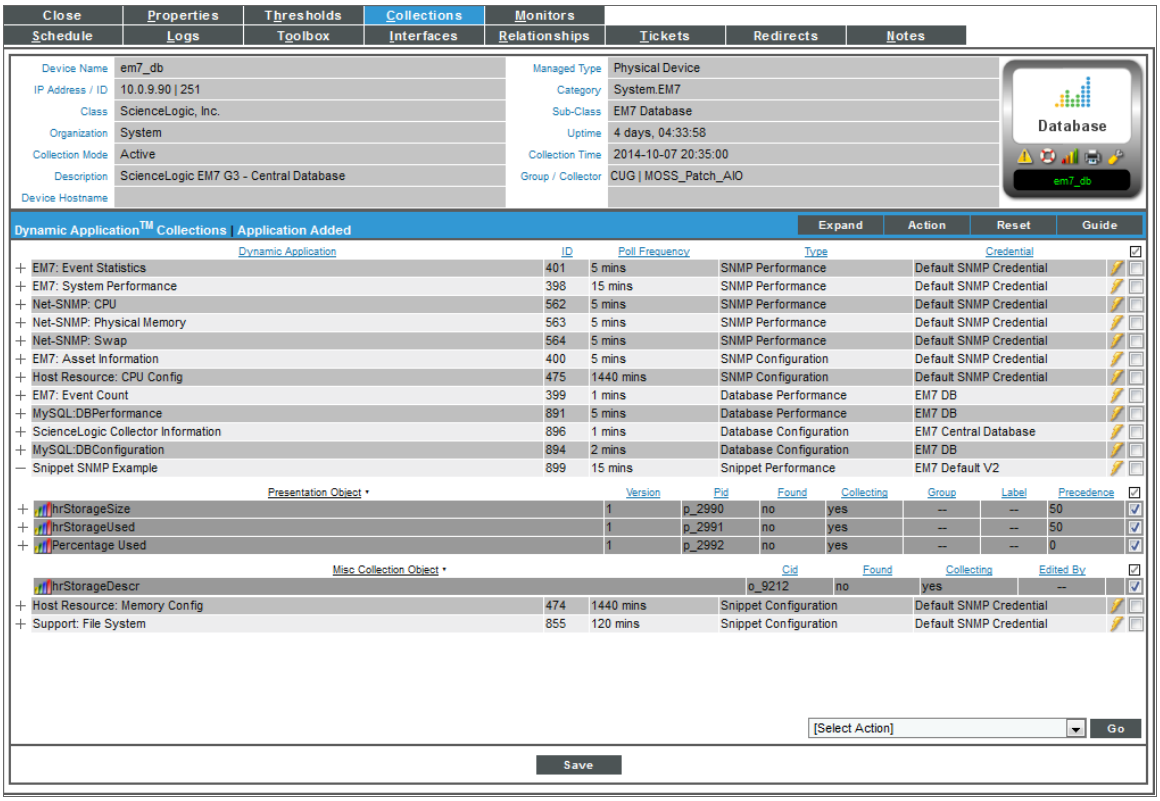
Aligning the Dynamic Application with a Device

To align this Dynamic Application to a device, perform the following steps:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Select the wrench icon () for the device you want to align the Dynamic Application with. The **Device Properties** page is displayed.

> TIP: To find a device to align with the Dynamic Application, go to the **Device Hardware** page (Devices > Hardware) and search for devices with ***Comp Type*** of *File System*.

3. Select the **[Collections]** tab. The **Dynamic Application Collections** page is displayed.

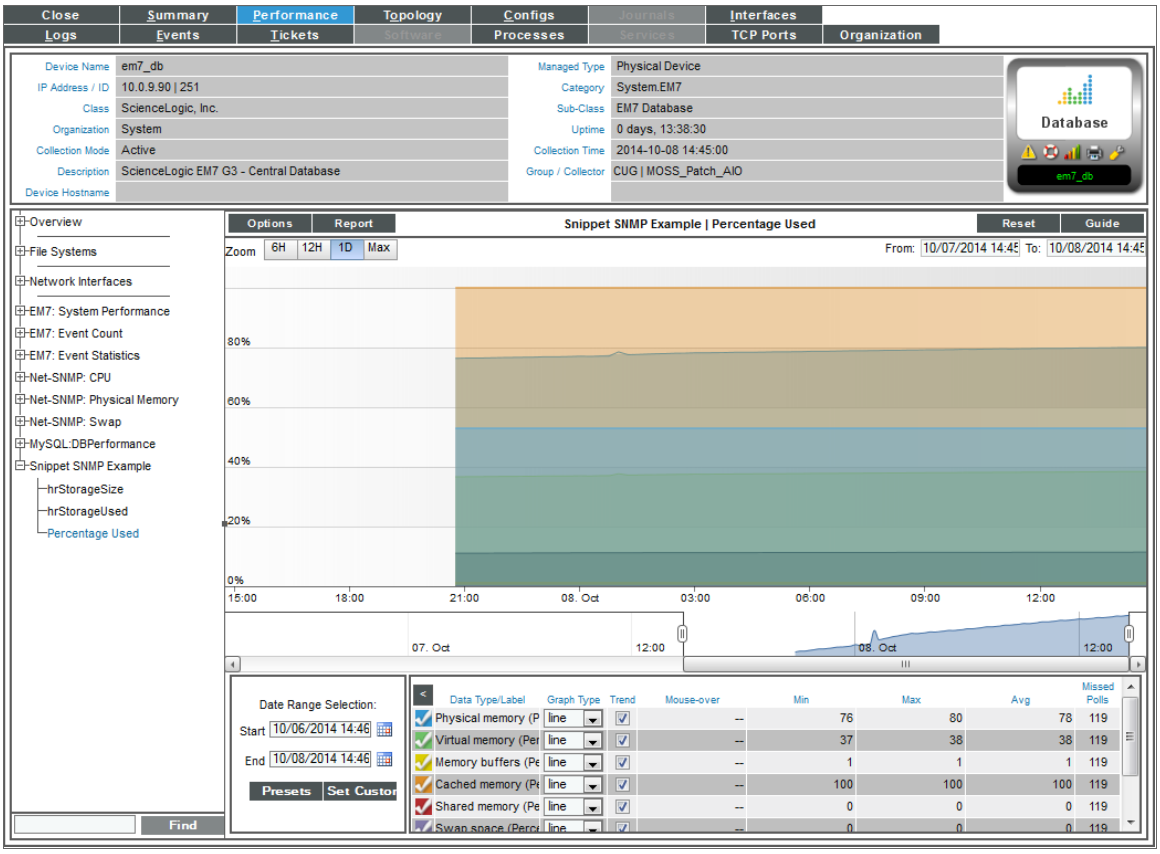


4. Select the ***Action*** menu and choose *Add Dynamic Application*.
5. In the **Dynamic Application Alignment** modal page, select our example Dynamic Application, ***Snippet SNMP Example***. Select the credential you created in the previous section.
6. Select the **[Save]** button. The page refreshes, and the *Snippet SNMP Example* Dynamic Application is displayed in the list of Dynamic Applications.

Viewing Performance Data

The first collected values will be stored within one minute. To view the performance data for this Dynamic Application:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Select the graph icon ( ) for the device you aligned the Dynamic Application with. The **Device Summary** page is displayed.
3. Select the **[Performance]** tab. The **Device Performance** page is displayed.



4. Select ***Snippet SNMP Example*** in the left Navbar, then the *Storage Percent* presentation object Dynamic Application. A performance graph will be displayed.

Example 3: Using Telnet to Collect Performance Data

Some network devices expose useful pieces of performance or configuration data via a telnet interface. Telnet may be the only method to access some information, such as configuration data.

Snippets provide a convenient way to gather data via telnet, and then present, report and alert against the results.

In this example, telnet is used to connect to a Dell Ethernet switch and collect CPU statistics. From the command line, an interaction with the switch looks like this:

```
> telnet 192.168.1.4
```

```
Trying 192.168.1.4...
```

```
Connected to 192.168.1.4.
```

```
Escape character is '^]'.
```

```
User Name:admin
```

```
Password:************

Dell-5324b# show cpu utilization

CPU utilization service is on.

CPU utilization

---------------

five seconds:8% ;one minute:8% ;five minutes:4%

Dell-5324b# exit

Connection closed by foreign host.

>
```

From the command line interface access attempt, we can see that to log in, the switch presents a "User Name:" prompt, followed by a "Password:" prompt. Once logged in, commands may be submitted at the # prompt. When CPU utilization is requested, the results come back on one line showing utilization for three different collection intervals. Finally, "exit" terminates the telnet session.

Creating the Dynamic Application

To create this example Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Select the **[Actions]** button, then *Create New Dynamic Application*. The **Create New Application** page is displayed.
3. Supply values in the following fields:
    - ***Application Name***. The name of the Dynamic Application. This example is called "Snippet Telnet Example".
    - ***Application Type***. This example is a Snippet Performance Dynamic Application. Select *Snippet Performance* in this field.
    - ***Poll Frequency***. To see data as quickly as possible, select *Every 1 Minute* in this field.
4. This example does not have specific requirements for the other settings defined in this page. You can leave the remaining fields set to the default values.
5. Select the **[Save]** button.

Because collection objects are assigned a specific snippet, you must create the container for the snippet code before creating the collection objects for this Dynamic Application. To create a container for the snippet code, perform the following steps:

1. Select the **[Snippets]** tab.
2. Supply values in the following fields:
    - ***Snippet Name***. The name of the snippet. The snippet in this example is called "CPU Collection".
    - ***Active State***. To ensure that the snippet code is run by Skylar One, select *Enabled* in this field.
    - ***Snippet Code***. Leave this field blank. You will add the snippet code later in this example.
3. Select the **[Save]** button.

Creating the Collection Objects

The snippet code for this example will use a regular expression to extract specific CPU values from the CPU response line in the switch. The collection objects will specify the command to execute (show cpu utilization) and the specific CPU value to be extracted using a regular expression for the collection object. The command to execute the specific CPU value to collect for each object will be defined in the ***Snippet Arguments*** field, separated by a pipe ('|').

To create the collection objects for this Dynamic Application, perform the following steps:

1. Select the **[Collections]** tab in the **Dynamic Application Editor** pane.
2. Supply values in the following fields to create the first object, which will collect the five second CPU data:
    - ***Object Name***. The name of the collection object. Enter "CPU Five Second" in this field.
    - ***Snippet Arguments***. The arguments to pass to the snippet. Enter "show cpu utilization|five second:" in this field.
    - ***Class Type***. The collected values for this collection object will be incrementing numeric values from which a delta must be taken. Select *1 Performance Counter* in this field.
    - ***Snippet***. There is only one snippet for this Dynamic Application. Select *CPU Collection* in this field.
3. This example does not have specific requirements for the other collection object settings. You can leave the remaining fields set to the default values.
4. Select the **[Save]** button, then select the **[Reset]** button to clear the values you entered.
5. Repeat steps two, three, and four to create a collection object for the one minute CPU collection, using the following values:
    - ***Object Name***. Enter "CPU 1 Minute".
    - ***Snippet Arguments***. The arguments to pass to the snippet. Enter "show cpu utilization|one minute:" in this field.
    - ***Class Type***. The collected values for this collection object will be incrementing numeric values from which a delta must be taken. Select *1 Performance Counter* in this field.
    - ***Snippet***. There is only one snippet for this Dynamic Application. Select *CPU Collection* in this field.
6. Repeat steps two, three, and four to create a collection object for the five minute CPU collection, using the following values:

- ***Object Name***. Enter "CPU Five Minute".
- ***Snippet Arguments***. The arguments to pass to the snippet. Enter "show cpu utilization|five minute:" in this field.
- ***Class Type***. The collected values for this collection object will be incrementing numeric values from which a delta must be taken. Select *1 Performance Counter* in this field.
- ***Snippet***. There is only one snippet for this Dynamic Application. Select *CPU Collection* in this field.

Snippet Code

This section walks through all the snippet code used in this example.

First, the telnetlib module is imported, which will be used to make the telnet connection to the switch. The regular expression module is also imported, which will be used to parse the performance data from the response:

```
import telnetlib

import re
```

The self.cred_details dictionary is used to make a connection to the switch:

```
host = self.cred_details['cred_host']

user = self.cred_details['cred_user']

password = self.cred_details['cred_pwd']

tn = telnetlib.Telnet(host)

tn.read_until("User Name:")

tn.write(user + "\n")

tn.read_until("Password:")

tn.write(password + "\n")
```

Like the snippet code in Example 1 and Example 2, this snippet iterates through the self.oids dictionary:

```
for group, oid_group in self.oids.iteritems():

  for obj_id, oid_detail in oid_group.iteritems():

    if oid_detail['oid_type'] != snippet_id:

      continue
```

The snippet expects the arguments for each collection object to be the command to run and the value to parse from the results separated by a pipe ('|'). If the collection object the snippet is currently operating on is not in this format, the snippet generates an alert, then iterates to the next collection object:

```
cmd_l = oid_detail['oid'].split('|')

if len(cmd_l) != 2:

  self.internal_alerts.append((518, "App:%s, Snippet:%s, Invalid
  collection value:%s (must be a command and a match string)" % \

  (self.app_id, req_id, oid_detail['oid'])))

  continue
```

If the collection object has valid arguments, the command is executed, then the result is parsed:

```
command = cmd_l[0]

match_val = cmd_l[1]



tn.read_until("#")

tn.write(command + "\n")

tn.read_until("-----------\r\n")

result = tn.read_until("\r\n")



match_regex = "(?<=%s).*?[0-9]{1,}" % (match_val)

match_obj = re.search(match_regex, result)
```

If a match is found, the result is stored:

```
if match_obj:

  oid_detail['result'] = [(0, match_obj.group()),]
```

Using the Dynamic Application

For performance Dynamic Applications, presentation objects that correspond to each collection object are automatically created. To add the snippet code to the example Dynamic Application you created, perform the following steps:

1. Select the **[Snippets]** tab in the **Dynamic Application Editor** pane.
2. Select the wrench icon (🔧) for the "CPU Collection" snippet.
3. Insert the snippet code in the ***Snippet Code*** field. A full listing of the snippet code is included in the last section.
4. Select the **[Save]** button.

To use the example Dynamic Application, you must first create a Basic/Snippet credential to align with the Dynamic Application. Perform the following steps to create Basic/Snippet credential for a Dell Ethernet switch:

1. Go to the **Credential Management** page (System > Manage > Credentials).
2. Select the **[Create]** button, then select *Basic/Snippet Credential*. The **Credential Editor** page is displayed.
3. Supply values in the following fields:
    - ***Profile Name***. Enter a name for the credential.
    - ***Hostname/IP***. Specifies the hostname or IP address to connect to. If you are aligning this Dynamic Application to a Dell Ethernet switch modeled as a device in your system, enter "%D" in this field to use the IP address of the aligned device. If you are not aligning this Dynamic Application to a Dell Ethernet switch modeled as a device in your system, enter the hostname or IP address of your Dell Ethernet switch in this field.
    - ***Port***. Specifies the port to connect to. Enter "23" in this field to use the default port for telnet.
    - ***Timeout***. This field is not used by the snippet code. You can leave this field blank.
    - ***Username***. Specifies the username to use when connecting to the switch. Enter the username for a user on your switch in this field.
    - ***Password***. Specifies the password to use when connecting to the switch. In this field, enter the password for the username you entered in the ***Username*** field.
4. Select the **[Save]** button.

To align this Dynamic Application to a device, perform the following steps:

> **NOTE:** If you entered "%D" in the ***Hostname/IP*** field for your credential, you must align the Dynamic Application to a Dell Ethernet switch.

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Select the wrench icon (🔧) for the device you want to align the Dynamic Application with. The **Device Properties** page is displayed.
3. Select the **[Collections]** tab. The **Dynamic Application Collections** page is displayed.
4. Select the ***Action*** menu and choose *Add Dynamic Application*.
5. In the **Dynamic Application Alignment** modal page, select our example Dynamic Application, ***Dell Switch CPU Snippet***. Select the Basic/Snippet credential you created for your Dell Ethernet switch.
6. Select the **[Save]** button. The page refreshes, and the *Snippet Telnet Example* Dynamic Application is displayed in the list of Dynamic Applications.

The first collected value will be stored within one minute. To view collected data for this Dynamic Application:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Select the graph icon ( ) for the device you aligned the Dynamic Application with. The **Device Summary** page is displayed.
3. Select the **[Performance]** tab. The **Device Performance** page is displayed.
4. Select *Snippet Telnet Example* in the left Navbar, then select one of the three presentations for this Dynamic Application. A performance graph is displayed.

Full Snippet Code Listing

```
import telnetlib

import re


host = self.cred_details['cred_host']

user = self.cred_details['cred_user']

password = self.cred_details['cred_pwd']


tn = telnetlib.Telnet(host)

tn.read_until("User Name:")

tn.write(user + "\n")

tn.read_until("Password:")

tn.write(password + "\n")


for group, oid_group in self.oids.iteritems():

  for obj_id, oid_detail in oid_group.iteritems():

    if oid_detail['oid_type'] != snippet_id:

      continue


    cmd_l = oid_detail['oid'].split('|')
```

```
if len(cmd_l) != 2:
  self.internal_alerts.append((518, "App:%s, Snippet:%s, Invalid
  collection value:%s (must be a command and a match string)" % \
  (self.app_id, req_id, oid_detail['oid'])))
  continue

command = cmd_l[0]
match_val = cmd_l[1]

tn.read_until("#")
tn.write(command + "\n")
tn.read_until("-----------\r\n")
result = tn.read_until("\r\n")

match_regex = "(?<=%s).*?[0-9]{1,}" % (match_val)
match_obj = re.search(match_regex, result)
if match_obj:
  oid_detail['result'] = [(0, match_obj.group()),]
```

Using Snippets to Collect Database Data

Snippets can be used to implement a variety of collection methods, including those that are built into Skylar One, such as Database collection. Although it is more convenient to use the built-in collection methods where possible, there are use cases when using a snippet Dynamic Application for database collection is appropriate. This section walks through snippet code that:

- Performs an SQL query to retrieve the data retention settings from a Skylar One System. This data is stored in the master.system_settings_core table. The table includes a column for each system setting and a single row that specifies the values for those system settings.
- Appends a suffix to each data retention value.

Creating the Dynamic Application

To create this example Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Select the **[Actions]** button, then *Create New Dynamic Application*. The **Create New Application** page is displayed.
3. Supply values in the following fields:
    - ***Application Name***. The name of the Dynamic Application. This example is called "Snippet Database Example".
    - ***Application Type***. This example is a Snippet Configuration Dynamic Application. Select *Snippet Configuration* in this field.
    - ***Poll Frequency***. To see data as quickly as possible, select *Every 1 Minute* in this field.
4. This example does not have specific requirements for the other settings defined in this page. You can leave the remaining fields set to the default values. Select the **[Save]** button.

Creating a Container for the Snippet

Because collection objects are assigned a specific snippet, you must create the container for the snippet code before creating the collection objects for this Dynamic Application. To create a container for the snippet code, perform the following steps:

1. Select the **[Snippets]** tab.
2. Supply values in the following fields:
    - ***Snippet Name***. The name of the snippet. The snippet in this example is called "Database Query".
    - ***Active State***. To ensure that the snippet code is run by Skylar One, select *Enabled* in this field.
    - ***Required***. Specifies whether this snippet is required for successful collection of all other snippet requests. Select *Required - Stop Collection*. If this snippet request fails, the platform will not attempt to execute any other snippet requests in this Dynamic Application. Dynamic Applications that consume the cache of this Dynamic Application will halt collection.
    - ***Snippet Code***. Leave this field blank. You will add the snippet code later.
3. Select the **[Save]** button.

Creating the Collection Objects

The snippet code for this example will retrieve data from a database and populate the ***result_handler*** dictionary with the retrieved values. The collection objects will be the names of columns in the database. The snippet code will construct the database query and process the results using the arguments provided for each collection object, i.e. new or additional fields can be added as collection objects without editing the snippet code. This example includes collection objects for the following database fields:

- ***records_perf***. The number of days for which raw performance data is retained.
- ***records_dev_logs_max***. The maximum number of device logs that should be stored.
- ***records_access_logs***. The number of months for which access log data is retained.

> **NOTE:** These three fields are included in this example were chosen to illustrate the three different ways the snippet code processes the collected values. You can add collection objects for the other data retention settings as desired. The column names for retention have the prefix "records_".

To create the collection objects for this Dynamic Application, perform the following steps:

1. Select the **[Collections]** tab in the **Dynamic Application Editor** pane.
2. Supply values in the following fields to create the first object, which will collect the values from the ***records_perf*** column of the database:
    - ***Object Name***. The name of the collection object. Enter "Raw Performance Data" in this field.
    - ***Snippet Arguments***. The arguments to pass to the snippet. The snippet code expects the arguments for each collection object to be a field name in the master.system_settings_core table. Enter "records_perf" in this field.
    - ***Snippet***. There is only one snippet for this Dynamic Application. Select *Database Query* in this field.
3. For the remaining fields, accept the default values. Select the **[Save]** button, then select the **[Reset]** button to clear the values you entered.
4. Repeat steps two and three to create the collection object that will collect the values from the ***records_dev_logs_max*** column of the database:
    - ***Object Name***. Enter "Device Logs Max".
    - ***Snippet Arguments***. The arguments to pass to the snippet. The snippet code expects the arguments for each collection object to be a field name in the master.system_settings_core table. Enter "records_dev_logs_max" in this field.
    - ***Snippet***. There is only one snippet for this Dynamic Application. Select *Database Query* in this field.
5. For the remaining fields, accept the default values. Select the **[Save]** button, then select the **[Reset]** button to clear the values you entered.
6. Repeat steps four and five to create the collection object that will collect the values from the ***records_access_logs*** column of the database:
    - ***Object Name***. Enter "Access Logs".
    - ***Snippet Arguments***. The arguments to pass to the snippet. The snippet code expects the arguments for each collection object to be a field name in the master.system_settings_core table. Enter "records_access_logs" in this field.
    - ***Snippet***. There is only one snippet for this Dynamic Application. Select *Database Query* in this field.
7. For the remaining fields, accept the default values. Select the **[Save]** button.

Snippet Code Walkthrough

The snippet code for this example starts by initializing three variables:

```
table = "master.system_settings_core"

fields = []

results = {}
```

- The snippet will be retrieving data from the table ***system_settings_core***, in the database ***master***. The table name is stored in the variable ***table***.
- The snippet will use ***fields*** to store a list of database fields that will be included in the database query.
- The snippet will use the ***results*** to store the dictionary that will be passed to the result handler.

Next, the snippet iterates through the key values from the result_handler dictionary. The keys in the result_handler dictionary are the arguments supplied for each collection object. In this example, the arguments are the field names from the master.system_settings_core table. The field names are added to the list of fields and used to initialize the ***results*** dictionary with empty lists:

```
for collection in result_handler.iterkeys():

  fields.append(collection)

  results[collection] = []
```

The list of database fields is then formatted as a comma-delimited string in the variable ***field_list***:

```
field_list = ','.join(fields)
```

Next, the snippet retrieves a database cursor using the dbc_from_cred_id method. This method is called in a try/except/else block, which will catch both types of exceptions raised by the db_from_cred_id method (ValueError and RuntimeError). If an exception is raised, the COLLECTION_PROBLEM and PROBLEM_STR variables are used to report the problem in the device log for the subscriber device:

```
try:

  dbc = em7_snippets.dbc_from_cred_id(self.cred_details['cred_id'])

except (ValueError,RuntimeError), e:

  COLLECTION_PROBLEM = True

  PROBLEM_STR = e
```

If the db_from_cred_id method does not generate an exception, the snippet executes an SQL query using the field_list and table variables to construct a SELECT statement. The query is called in a try/except block similar to the previous section of code:

```
else:
  sql = "SELECT %s FROM %s;" % (field_list,table)
  try:
    dbc.execute(sql)
    db_result = dbc.fetchall()
  except Error, e:
    COLLECTION_PROBLEM = True
    PROBLEM_STR = e
```

NOTE: Remember that the variable ***field_list*** contains a list of all key values in the ***result_handler*** dictionary. Remember that the variable ***table*** contains the value *master.system_settings_core*.

If the query executes successfully, the following block of code processes and stores the collected values:

```
else:
  i = 0
  for row in db_result:
    for index, collection in enumerate(fields):
      if collection == "records_dev_logs_max":
        result_tuple = (i, str(row[index]) + " Records")
      elif row[index] % 30 == 0 and row[index] != 30:
        result_tuple = (i, str(row[index] / 30) + " Months")
      else:
        result_tuple = (i, str(row[index]) + " Days")
      results[collection].append(result_tuple)
    i += 1
```

```
result_handler.update(results)
```

The counter ***i*** is initialized at 0 and is incremented after each row returned by the query is processed. This counter is used to assign index values to each returned value. However, the master.system_settings_core table contains only one row, so this example returns only one index (0). This counter, and the iteration over each row in the result set, is included in this example so that the snippet code can be re-used for other database tables with only minor changes.

For each row in the result set, the snippet code iterates over the ***fields*** list. For each field, ***collection*** is the field name (also the snippet argument for the corresponding collection object) and index is the list index in the ***fields*** list. Because the ***fields*** list was used to construct the database query, the list indexes for fields match the indexes in each ***row***.

For each field, the snippet determines the appropriate suffix:

- If the field is "records_dev_logs_max", the suffix is "Records". The retention setting for the maximum number of device logs is the only retention setting (fields with the prefix "records_") that is not stored as a number of days.
- All other retention settings are stored as a number of days. If the number of days is evenly divisible by 30 and greater than 30, the suffix is "Months".
- For all other values, the suffix is "Days".

The result_tuple variable is populated with the index (the value of the counter i) and a string value for the collection object. The tuple is then appended to list in the results dictionary that corresponds to the collection object.

Because ***results*** is a dictionary of lists that contains the same keys as the ***result_handler*** dictionary, we can return the collected data to Skylar One using the *update* function for the ***result_handler*** dictionary.

Adding the Snippet Code

To enter the Snippet code:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Dynamic Applications).
2. Find the example Dynamic Application, ***Example Database Collection***. Select its wrench icon ( ).
3. Select the **[Snippets]** tab.
4. In the **Snippet Registry** pane, find the ***Database Query*** snippet and select its wrench icon ( ).
5. In the ***Snippet Code*** field, enter the following code

```
table = "master.system_settings_core"

fields = []

results = {}

for collection in result_handler.iterkeys():

  fields.append(collection)

  results[collection] = []
```

```
field_list = ','.join(fields)
try:
  dbc = em7_snippets.dbc_from_cred_id(self.cred_details['cred_id'])
except (ValueError,RuntimeError), e:
  COLLECTION_PROBLEM = True
  PROBLEM_STR = e
else:
  sql = "SELECT %s FROM %s;" % (field_list,table)
  try:
    dbc.execute(sql)
    db_result = dbc.fetchall()
  except Error, e:
    COLLECTION_PROBLEM = True
    PROBLEM_STR = e
  else:
    i = 0
    for row in db_result:
      for index, collection in enumerate(fields):
        if collection == "records_dev_logs_max":
          result_tuple = (i, str(row[index]) + " Records")
        elif row[index] % 30 == 0 and row[index] != 30:
          result_tuple = (i, str(row[index] / 30) + " Months")
        else:
          result_tuple = (i, str(row[index]) + " Days")
```

```
            results[collection].append(result_tuple)

        i += 1

    result_handler.update(results)
```

Using the Dynamic Application

The Dynamic Application in this example can be aligned with any ScienceLogic Database Server or All-In-One Appliance.

Define the Credential

To use the example Dynamic Application, you must first create a Database credential to align with the Dynamic Application. Perform the following steps to create a new Database credential:

1. Go to the **Credential Management** page (System > Manage > Credentials).
2. Select the **[Actions]** menu, then select *Create Database Credential*. The **Credential Editor** page is displayed.
3. Supply values in the following fields:
    - **Profile Name**. Name of the profile. Can be any combination of alphanumeric characters.
    - ***DB Type***. Select *MySQL*.
    - ***DB Name***. Name of the database that will be accessed with the credential. Enter ***master***.
    - ***DB User***. Username associated with a valid account on the database. For Database Servers and All-In-One Appliances, this is typically the root user.
    - ***Password***. Password associated with the ***DB User*** account.
    - ***Hostname/IP***. Hostname or IP address of the Database Server or All-In-One Appliance. If you are aligning this Dynamic Application to a Database Server or All-In-One Appliance that is already discovered in your system, enter "%D" in this field to use the IP address associated with the device record.
    - ***Port***. Enter "7706".
4. Select the **[Save]** button.

Align the Dynamic Application with a Device

To align this Dynamic Application to a device, perform the following steps:

NOTE: If you entered "%D" in the ***Hostname/IP*** field for your credential, you must align the Dynamic Application to the device record for a Database Server or All-In-One Appliance.

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).

2. Select the wrench icon ( ) for the device you want to align the Dynamic Application with. The **Device Properties** page is displayed.
3. Select the **[Collections]** tab. The **Dynamic Application Collections** page is displayed.
4. Select the ***Action*** menu and choose *Add Dynamic Application*.
5. In the **Dynamic Application Alignment** modal page, select our example Dynamic Application, ***Snippet Database Example***. Select the credential you created in the previous section. Select the **[Save]** button.
6. The page refreshes, and the *Snippet Database Example* Dynamic Application is displayed in the list of Dynamic Applications.

Viewing Data

The first collected value will be stored within one minute. To view the label data for this Dynamic Application:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Select the graph icon ( ) for the device you aligned the Dynamic Application with. The **Device Summary** page is displayed.
3. Select the **[Configs]** tab. The **Configuration Report** page is displayed.
4. In the left NavBar, select ***Snippet Database Example***. The collected data is displayed.

Chapter 3

SNMP Dynamic Applications

Overview

This chapter describes Simple Network Management Protocol (SNMP), the Internet protocol for managing devices across a network. In Skylar One, an SNMP Dynamic Application is a Dynamic Application that uses SNMP to retrieve data from devices.

This chapter covers the following topics:

Concurrent SNMP Collection *248*

SNMP Collection Objects *250*

Viewing MIBs and Using the SNMP Walker Tool *257*

Creating an SNMP Performance Dynamic Application *268*

What is SNMP?

SNMP is a protocol that is supported by most enterprise-level network equipment and most server equipment as well as by many software applications.

SNMP allows Skylar One to query and collect information about a device or application. SNMP ensures that Skylar One can use the same parameters to query any type of device or application. SNMP also ensures that the data sent to Skylar One will be of the same format, regardless of the device or application that returns the data.

Many third-party hardware agents are SNMP-compliant. These agents complement standard SNMP agents by providing additional data about the device.

If SNMP is enabled on a device and a third-party agent is also running on a device, Skylar One can gather very detailed data about the device or application.

Prerequisites

This chapter does not describe how to plan, design, use, and troubleshoot Dynamic Applications for your network. This chapter assumes that you are already familiar with the common elements and concepts of Dynamic Applications. For general information on planning, designing, using, and troubleshooting Dynamic Applications, see the ***Dynamic Application Development*** chapter.

SNMP Dynamic Applications use the SNMP protocol. This manual assumes that you are familiar with the SNMP protocol.

What is an SNMP Dynamic Application?

Simple Network Management Protocol, or SNMP, is the Internet protocol for managing devices across a network. In Skylar One, an SNMP Dynamic Application is a Dynamic Application that uses SNMP to retrieve data from devices. During collection, Skylar One performs an SNMP walk command for each OID specified in the Dynamic Application. The Dynamic Application developer defines the SNMP OIDs in collection objects.

Elements of an SNMP Dynamic Application

For comprehensive information on elements that apply to SNMP Dynamic Applications, see the **Dynamic Application Development** chapter.

What is Concurrent SNMP Collection?

To increase the scale for SNMP collection, you can enable ***Concurrent SNMP Collection***. Concurrent SNMP Collection uses the standalone container called the Skylar One SNMP Collector.

The SNMP Collector is an independent service that runs as a container on a Data Collector. When you enable Concurrent SNMP Collection, each Data Collector will contain four (4) SNMP Collector containers.

NOTE: On each Data Collector, Skylar One will restart each of the SNMP Collector containers periodically to ensure that each container remains healthy. When one SNMP Collector container is restarted, the other three SNMP Collector containers continue to handle the workload.

With Concurrent SNMP Collection, SNMP collection tasks can run in parallel. A single failed task will not prevent other tasks from completing.

Concurrent SNMP Collection provides:

- Improved throughput for SNMP Dynamic Applications
- Reduced use of resources on each Data Collector
- More dependable collection from high-latency Devices

NOTE: Concurrent SNMP Collection is not available in military unique deployments (MUD). However, starting with Skylar One version 12.5.20, Concurrent SNMP Collection is available for STIG-compliant deployments.

For more information, see *Concurrent SNMP Collection* section.

Concurrent SNMP Collection

This section describes how to configure and use Concurrent SNMP Collection to run SNMP collection jobs.

Using Concurrent SNMP Collection

To increase the scale for SNMP collection, you can enable ***Concurrent SNMP Collection***. Concurrent SNMP Collection uses the standalone container called the Skylar One SNMP Collector.

The SNMP Collector is an independent service that runs as a container on a Data Collector. When you enable Concurrent SNMP Collection, each Data Collector will contain four (4) SNMP Collector containers.

NOTE: On each Data Collector, Skylar One will restart each of the SNMP Collector containers periodically to ensure that each container remains healthy. When one SNMP Collector container is restarted, the other three SNMP Collector containers continue to handle the workload.

With Concurrent SNMP Collection, SNMP collection tasks can run in parallel. A single failed task will not prevent other tasks from completing.

Concurrent SNMP Collection provides:

- Improved throughput for SNMP Dynamic Applications
- Reduced use of resources on each Data Collector
- More dependable collection from high-latency Devices

> **NOTE:** Concurrent SNMP Collection is not available in military unique deployments (MUD). However, starting with Skylar One version 12.5.20, Concurrent SNMP Collection is available for STIG-compliant deployments.

You can enable all, one, or multiple Collector Groups to use concurrent SNMP collection.

Enabling Concurrent SNMP Collection

> **NOTE:** This feature is disabled by default.

To enable Concurrent SNMP Collection in Skylar One:

1. Go to the **Behavior Settings** page (System > Settings > Behavior).
2. Check the ***Enable Concurrent SNMP Collection*** field.
3. Click **[Save]**.

> **NOTE:** If the "Data Collection: SNMP Collector" process is disabled on the **Process Manager** page (System > Settings > Admin Processes), this concurrent SNMP collection option is disabled.

> **TIP:** If you do not want all of your Skylar One Collectors to use Concurrent SNMP Collection, you can specify which Collector Units should use it in *Enabling a Collector Group to Use Concurrent SNMP Collection.*

Enabling a Collector Group to Use Concurrent SNMP Collection

Depending on the needs of your Skylar One environment, you can enable or prevent a collector group from using concurrent SNMP collection.

To enable Concurrent SNMP Collection with a Skylar One collector group:

1. Go to the **Collector Group Management** Page (System > Settings > Collector Groups).
2. Click the wrench icon () for the collector group you want to edit. The fields at the top of the page are updated with the data for that collector group.

3. Select an option in the ***Enable Concurrent SNMP Collection*** drop-down field:
    - ***Use system-wide default***. Select this option if you want this collector group to use or not use Concurrent SNMP Collection based on the ***Enable Concurrent SNMP Collection*** field on the **Behavior Settings** page. This is the default.
    - ***Yes***. Select this option to enable Concurrent SNMP Collection for this collector group, even if you did not enable it on the **Behavior Settings** page.
    - ***No***. Select this option to prevent this collector group from using Concurrent SNMP Collection, even if you did enable it on the **Behavior Settings** page.

> **NOTE:** If the "Data Collection: SNMP Collector" process is disabled on the **Process Manager** page (System > Settings > Admin Processes), this concurrent SNMP collection option is disabled.

4. Update the remaining fields as needed, and then click **[Save]**.

SNMP Collection Objects

The following sections describe how to define collection objects for SNMP Dynamic Applications.

All other elements of SNMP Dynamic Applications, such as presentation objects and alerts, behave in the same manner as other Dynamic Application types.

For details on other parts of SNMP Dynamic Applications, see the ***Dynamic ApplicationDevelopment chapter***.

Collection Objects

Like all Dynamic Application types, SNMP Dynamic Applications contain collection objects. Collection objects for SNMP Dynamic Applications share common characteristics with collection objects for other Dynamic Application types, such as naming, data typing, and grouping. For details on these fields, see the chapter ***Dynamic Application Development***.

Unlike collection objects for other Dynamic Application types, collection objects for SNMP Dynamic Applications are based on SNMP OIDs. Because of this, Collection Objects for SNMP Dynamic Applications have the following unique field:

- ***SNMP OID***. The Object Identifier associated with the object. Skylar One will use the OID to perform an SNMP walk on the device aligned with the Dynamic Application.

You can find this field and other information about Collection Objects for an SNMP Dynamic Application by clicking the wrench icon (🔧) for a Dynamic Application, and then selecting the **[Collections]** tab. A list of Collection Objects aligned with that Dynamic Application appears at the bottom of the window.

NOTE: You can also add collection objects to an SNMP Dynamic Application through the **OID Browser** page (System > Tools > OID Browser) or through the **SNMP Walker** tool (Device Administration panel > Toolbox > SNMP Walker). For details, see the section on *Viewing MIBs and Using the SNMP Walker Tool.*

TIP: ***Scalar OIDs*** represent a single object; ***tabular OIDs*** represent groups of objects in a MIB table. For improved efficiency, you can add a ".0" to the end of all scalar OIDs in an SNMP Dynamic Application. For example, for the singular object "sysDescr", you could enter ".1.3.6.1.2.1.1.1.0" in the ***SNMP OID*** field.

Extending the SNMP OID Field

Skylar One allows you to use the ***SNMP OID*** field to perform more sophisticated OID evaluations and operations when defining the value of a collection object.

TIP: You can find the ***SNMP OID*** field by clicking the wrench icon (🔧) for a Dynamic Application, and then selecting the **[Collections]** tab.

Using Multiple OIDs and/or Existing Collection Objects to Define a Single Collection Object

In the ***SNMP OID*** field, you can concatenate the values from one or more OIDs or one or more collections objects.

To concatenate the values from one or more OIDs in a single SNMP collection object, use the following syntax in the ***SNMP OID*** field:

OID number or partial OID number.collection object.integer

where:

- *OID number or partial OID number*. Enter the OID number or partial OID number. Skylar One will retrieve the value of this OID.
- *collection_object*. You can specify one or more collection object names (in Skylar One, these name usually begin with "o_"). Skylar One will retrieve the value of the collection object and include it in the collection object.
- *integer*. You can specify an integer to append to include in the collection object.

For example, you could enter the following in the ***SNMP OID*** field:

- *<partial oid>.<collection object>*

```
.1.3.6.1.2.1.1.9.1.o_1
```

- *<partial oid>.<collection object>.<integer>*

```
.1.3.6.1.2.1.1.9.1.o_1.0
```

- *<partial oid>.<collection object 1>.<collection object 2>.<integer>*

```
.1.3.6.1.2.1.1.9.o_1.o_2.0
```

NOTE: If a collection object is dependent upon the value of another collection object, during polling Skylar One polls the values in order of dependency. For example, if the definition of o_123 includes the value of o_456, Skylar One polls o_456 before polling o_123.

Using Python Operators to Define a Collection Object

The ***SNMP OID*** field can include:

Arithmetic operators (+, -, *, /), comparison operators (>, <, !=, ==), and/or boolean operators ("and", "or", "not").

NOTE: The "==" operator is used to test equality. The single "=" assignment operator is not supported, except when used inside some ScienceLogic functions.

- For example, you could enter the following in the ***SNMP OID*** field:

```
o_16752 + o_16753
```

  and the new collection object will be populated with the sum of the two collection objects.

- If you want to perform a calculation with **multiple OIDs** (*not* multiple collection objects) in the ***SNMP OID*** field, use the **em7snmp.collect** function. For example, you could enter the following in the ***SNMP OID*** field:

```
em7snmp.collect (".1.3.6.1.2.1.1.1") + em7snmp.collect
(".1.3.6.1.2.1.1.2")
```

  and the new collection object will be populated with the sum of the values from the two OIDs.

Using Variables in the SNMP Field to Specify Component Devices

For details on configuring Dynamic Applications to discover component devices, see the manual ***Dynamic Application Development***.

In the Dynamic Application that discovers component devices, Skylar One will store the value of the collection object designated as the Component Identifier into variables as follows:

Component Identifier	Variable Name
Device Name	comp_name
Distinguished Name	comp_dn
Unique Identifier	comp_unique_id
GUID	comp_guid

Suppose you have created a discovery Dynamic Application and discovered three component devices. Suppose you want to create a Dynamic Application that will collect data about these new component devices. In the ***SNMP OID*** field for each collection object, you can append ***comp_name***, ***.comp_dn*** , ***comp_unique_id***, or ***.comp_guid*** to the OID. This ensures that the collected data for each component device is easy to identify in Skylar One and can be aligned with the appropriate component device.

To specify that a value varies for each component device, use the following syntax:

OID number or partial OID number.component_identifier_variable

where:

- *OID number or partial OID number*. Enter the OID number or partial OID number. Skylar One will retrieve the value of this OID.
- *component_identifier_variable*. You can specify a variable for a component identifier. Skylar One will substitute the value for each component device into the variable.

For example:

Suppose we have will use a single SNMP Dynamic Application to collect data about two component devices:

Component Device Name	Component DN Value
Device 1	1
Device 2	2

Suppose our SNMP data looks like this:

OID	OID Value
.1.2.3.1	number1_3
.1.2.3.2	number2_3
.1.2.4.1	number1_4
.1.2.4.2	number2_4

Suppose we entered following in the ***SNMP OID*** field:

```
.1.2.3.comp_dn
```

- For device 1, Skylar One will collect the value of "1.2.3.1". That value is "number1_3".
- For device 2, Skylar One will collect the value of "1.2.3.2". That value is "number2_3".

Suppose we enter following in the ***SNMP OID*** field:

```
.1.2.4.comp_dn
```

- For device 1, Skylar One will collect the value of "1.2.4.1". That value is "number1_4".
- For device 2, Skylar One will collect the value of "1.2.4.2". That value is "number2_4".

Using Caching to Define a Collection Object

You can use SNMP objects to define a cache for use by one or more Dynamic Applications. For SNMP Dynamic Applications, caching is defined in the ***SNMP OID*** field in the **Collection Objects** page instead of for the entire Dynamic Application.

> **NOTE**: The ***Caching*** field in the **Dynamic Applications Properties Editor** page does not affect caching for SNMP Dynamic Applications.

To define a cache for an SNMP collection object, you ***define the cache in the SNMP OID field in all Dynamic Applications that will use the cache***, both those that will create the cache and those that will consume the cache. If the cache has timed out, the next request will re-populate the cache, making it available to all other Dynamic Applications. All Dynamic Applications can consume the cached value and refresh the cache.

To define a cache for an SNMP collection object, use the following syntax in the ***SNMP OID*** field:

em7snmp.collect ('*collection_object_definition*', enable_cache=*true_or_false*, time_to_expire=*time_in_minutes*, time_to_recollect=*time_in_minutes*)

where:

- *collection_object_definition*. The OID or extended OID for which you want Skylar One to cache a value or retrieve a cached value.
- enable_cache=*true_or_false*. To either create a cache or use a cached value, specify *true*.
- time_to_expire=*time_in_minutes*. Number of minutes that the collected value will remain in the cache. This value cannot be less than the value in the time_to_recollect argument..
- time_to_recollect=*time_in_minutes*. Frequency, in minutes, at which Skylar One will try to collect a value for the collection object. If collection succeeds, Skylar One will populate the collection object with the new value and store the new value in the cache. If collection fails, Skylar One will populate the collection object with the previously cached value.

For example:

```
em7snmp.collect('.1.3.6.1.2.1.25.2.3.1.3', enable_cache=True,time_to_
expire=15, time_to_recollect=15)
```

- In this example, the collection object will be populated with the value of the OID '.1.3.6.1.2.1.25.2.3.1.3'.
- The collected value will be stored in the cache.
- The value will be re-collected every 15 minutes.

- If we use this collection object in another Dynamic Application, that collection object can both populate the cache and retrieve values from the cache.

Filtering Results to Define a Collection Object

In the ***SNMP OID*** field , you can use the function **em7utils.filter** to filter the results.

The **em7utils.filter** function also supports the following operators:

- eq (equals)
- gt (greater than)
- ge (greater than or equal to)
- lt (less than)
- le (less than or equal to)

Suppose the OID .1.3.6.1.2.1.1.9.1.4 is an SNMP table, and for device 1, the data looks like this:

OID	OID Value
.1.3.6.1.2.1.1.9.1.4.1	1
.1.3.6.1.2.1.1.9.1.4.2	3
.1.3.6.1.2.1.1.9.1.4.3	9
.1.3.6.1.2.1.1.9.1.4.4	2
.1.3.6.1.2.1.1.9.1.4.5	4
.1.3.6.1.2.1.1.9.1.4.6	8
.1.3.6.1.2.1.1.9.1.4.7	6
.1.3.6.1.2.1.1.9.1.4.8	10
.1.3.6.1.2.1.1.9.1.4.9	100
.1.3.6.1.2.1.1.9.1.4.10	1000

We could enter the following in the ***SNMP OID*** field :

```
em7utils.filter(em7snmp.collect(".1.3.6.1.2.1.1.9.1.4"), gt (10))
```

For device 1, the collection object would contain the array:

.9, 100

.10, 1000

For another example, suppose you want to collect information about interfaces, but you don't want any information about loopback interfaces. Loopback interfaces use the interface type "24". You could enter the following in the ***SNMP OID*** field:

```
em7utils.filter(em7snmp.collect(".1.3.6.1.2.1.2.2.1.3", lambda x:x!="24"))
```

The OID .1.3.6.1.2.1.2.2.1.3 maps to the interfacetype object in the interface table in the IF-MIB. The "lambda x " syntax creates a sub-function called "x". The logic then says "collect values where 'x' is not

equal to '24'". Those values will be stored in the collection object. Skylar One retrieves all the interface types that do not have a value of "24".

For details on the lambda syntax, consult the Python documentation https://docs.python.org/2/tutorial/controlflow.html?highlight=lambda#lambda-expressions

Defining "Fallback" Values for a Collection Object

In the ***SNMP OID*** field, you can use syntax that tells Skylar One:

- If the first OID is available, use the first OID
- If the first OID is not available, use the second OID
- If the second OID is not available, use the third OID
- and so on

To include multiple OIDs in the ***SNMP OID*** field, use this syntax:

```
em7snmp.collect (".1.3.6.1.2.1.1.1") or em7snmp.collect
(".1.3.6.1.2.1.1.2") or em7snmp.collect (".1.3.6.1.2.1.1.3")
```

To include multiple collection objects in the ***SNMP OID*** field:

```
o_1234 or o_1234 or o_1236
```

To include a mixture of OIDs and collection objects in the ***SNMP OID*** field:

```
o_1234 or em7snmp.collect ('.1.3.6.1.2.1.1.2')
```

Using Index Values and Parsed Index Values to Define a Collection Object

If you are creating a collection object of type "Index", in the ***SMNP OID*** field, you can collect an entire index or parse an index.

- To collect an entire index, use the syntax:

  em7snmp.full_index (*object_name*)

  For example:

  ```
  em7snmp.full_index (o_1234)
  ```

This example returns the full index for the collection object o_1234 and stores the index in a new collection object.

- To parse a multi-part index, you can enter the following in the ***SNMP OID*** field:

  ```
  em7snmp.parse_index (collection_object,[index_format], index_element_
  number)
  ```

where:

- *collection_object* is the collection object for which you want to retrieve an index
- *index_format* describes the data type and order of the elements in the index. The index_format is surrounded by square brackets and each element is separated by a comma. The basic index types used in SNMP are described in detail in RFC 2578 Section 7.7, (https://tools.ietf.org/html/rfc2578#section-7). You can include the following elements:

  INDEX_TYPE_INTEGER

  INDEX_TYPE_IP

  INDEX_TYPE_STRING

  INDEX_TYPE_IMPLIEDSTRING

  INDEX_TYPE_OID

  INDEX_TYPE_IMPLIEDOID

- *index_element_number* is the position of the index_element that you want to collect. The first element in a multi-part index has the index_element_number of "0" (zero).

For example, suppose we have a collection object "o_224". Suppose the index for the OID in o_224 includes the following elements:

ipCidrRouteDest, ipCidrRouteMask, ipCidrRouteTos, ipCidrRouteNextHop

Here is the information about the example index:

OID Name	Data Type	Element Position in the Index
ipCidrRouteDest	INDEX_TYPE_IP	0
ipCidrRouteMask	INDEX_TYPE_IP	1
ipCidrRouteTos	INDEX_TYPE_INTEGER	2
ipCidrRouteNextHop	INDEX_TYPE_IP	3

To retrieve this index and assign only the value of ***ipCidrRouteNextHop*** to the new index collection object , we would enter the following in the ***SNMP OID*** field :

```
em7snmp.parse_index (o_224, [INDEX_TYPE_IP, INDEX_TYPE_IP, INDEX_TYPE_
INTEGER, INDEX_TYPE_IP], 3)
```

Viewing MIBs and Using the SNMP Walker Tool

Before you create SNMP Dynamic Applications, you need to identify the OIDs (Object Identifiers) that contain the data you want to collect. OIDs are defined in MIBs (Management Information Bases), standardized files provided by device or application manufacturers.

Skylar One includes an extensive library of MIBs by default, but you can import additional ones as needed. To explore available OIDs and see real-world examples of the data they return, use the OID Browser or the SNMP Walker Tool. The SNMP Walker Tool lets you "walk" OIDs on an actual device to preview data before you build your Dynamic Application.

This section describes a management information base (MIB) and how to view, import, and compile MIBs. This section also describes the OID Browser and the SNMP Walker Tool.

The Management Information Base

A management information base (MIB) is a collection of definitions associated with a device or application. A MIB is stored in a file and uses a standardized format defined by the SNMP protocol.

A MIB file is usually associated with a hardware manufacturer or software manufacturer. Some manufacturers use a single MIB that contains information about all their products. Other manufacturers create a separate MIB for each product. A MIB file is generic—it is not associated with a specific installed instance of a device, but rather applies to all devices or applications of that make and model.

Each aspect of a device or application that can be managed or monitored is represented within the MIB as an object ID (OID). In Skylar One, we use these OIDs to define Collection Objects. For example, suppose we want to monitor the temperatures of a Force 10 switch/router. The MIB for the Force 10 switch/router includes an OID that contains the current temperature of the entire unit. To monitor the temperature of a Force 10 switch/router, we could use this OID in a Dynamic Application.

Before creating a Dynamic Application, you must ensure that Skylar One contains the appropriate MIB file. This section will describe how to view and examine MIBs and how to import a MIB.

Viewing MIBs

By default, Skylar One includes an extensive set of MIBs. Before creating a new Dynamic Application, you must make sure that Skylar One already includes the MIB file and that the MIB file includes the data points that you want to monitor with Skylar One. If Skylar One does not include the MIB file, you must *import it*.

To view the list of MIBs included with Skylar One:

1. Go to the **MIB Compiler** page (System > Tools > MIB Compiler).
2. For each MIB file, the **MIB Compiler** page displays the following:
    - ***MIB Name***. Name of the MIB file.
    - ***Vendor***. For MIBs that use SMIv2, displays the vendor associated with the MIB.
    - ***MIB Revision Date***. Date the original author last revised and released the MIB.
    - ***Language***. Either SMIv1 or SMIv2.
    - ***Compiled***. Specifies whether or not the MIB has been compiled in Skylar One.
    - ***Type Defs***. Number of data types within the MIB.
    - ***Nodes***. Number of objects that can be monitored with this MIB
    - ***Traps***. Number of traps that can be monitored with this MIB.
    - ***Logs***. Number of error messages in the log file after compiling the MIB.

- ***Last Edit***. Date and time the MIB was created or last edited.
- ***User Edit***. User who imported or last edited the MIB.

> **TIP:** To sort the list, click on a column heading. The list will be sorted by the column value, in ascending order. To sort the list by descending order, click the column heading again. You can also filter the items on this inventory page by typing filter text or selecting filter options in one or more of the filters found above the columns on the page. For more information, see *Filtering Inventory Pages* in the ***Introduction to Skylar One*** manual.

Viewing the Contents of a MIB file

From the **MIB Compiler** page, you can view the contents of a MIB file. To do this:

1. Go to the **MIB Compiler** page (System > Tools > MIB Compiler).
2. In the **MIB Compiler** page, find the MIB you want to view. Select its information icon ( ).
3. The **MIB Viewer** modal page appears and displays the contents of the MIB file. You can view but cannot edit the MIB file from this modal page.

Importing a MIB

By default, Skylar One includes a large number of commonly-used MIB files. However, you can import additional MIB files into Skylar One. If you need to monitor a device or application for which Skylar One does not include a MIB file, you can import and compile the appropriate MIB file. You might also want to update an existing MIB file with a later version.

To import a MIB file into Skylar One:

1. Go to the **MIB Compiler** page (System > Tools > MIB Compiler).
2. Click the **[Import]** button.
3. In the **MIB Import** modal page, click **[Browse]**.
4. Navigate to the location of the MIB file on your local computer. Click the **[Import]** button.

> **NOTE:** You can select and import multiple MIB files simultaneously by holding Ctrl (Cmd on Mac) and clicking each MIB file you want to import.

5. The new MIB file appears in the list of MIB files in the **MIB Compiler** page.
6. You must compile the MIB file before Skylar One can use it. To compile a MIB, click its lightning bolt icon ( ).
7. When the MIB is done compiling, you will need to synchronize the new MIB data to all of your collectors. To sync the MIB data, go to the **OID Browser** page (System > Tools > OID Browser) and click the **[Update]** button. The status of the update displays at the top left of the page.

CAUTION: Failure to complete the update step can lead to a "no name found" error for a trap OID.

NOTE: You should also run the update step when a new collector is brought online after additional MIBs have already been added to the stack. Clicking **[Update]** on the **OID Browser** page ensures that the new collector receives the latest compiled MIBs. This update might take some time. For more information, see *Synchronizing the Latest Compiled MIBs to Collectors*.

Compiling a MIB

Before Skylar One can use a MIB file to monitor data points on a device or an application, you must ensure that the MIB file has been compiled. When you compile the MIB file, the MIB Compiler processes and validates the object definitions using the pysmi ScienceLogic Library. This significantly improves reliability of symbolic name resolution, particularly for SNMP trap-based event policies, and addresses historical issues with lost MIBs and missing or inaccessible object definitions.

To compile a MIB:

1. Go to the **MIB Compiler** page (System > Tools > MIB Compiler).
2. In the **MIB Compiler** page, find the MIB you want to compile. Select its lightning bolt icon ( ).

   NOTE: You The MIB recompile request now uses the HTTP POST method instead of GET. This prevents browser refresh actions from inadvertently triggering multiple redundant MIB recompilations. You can safely refresh your browser during or after the compilation process without triggering duplicate compilation jobs.

3. To determine if a MIB compiled successfully, you can view the ***Compile*** column for that MIB.
4. If a MIB does not compile successfully, you can view information about the compile operation. To do this, find the MIB file in the list of MIBs and select its page icon ( ).
5. If a MIB compiles successfully, but shows a value other than "0" (zero) in the ***Messages*** column, you can view information about the compile operation. To do this, find the MIB file in the list of MIBs and select its page icon ( ). In some cases, a MIB might compile successfully but might also include incorrectly defined data-points (called "objects" or "OIDs"). These data-points might not be included in the compiled MIB.

IMPORTANT: If a MIB does not compile, please view the compile-log for the MIB. MIBs that use incorrect format or syntax may fail to compile. Contact the MIB's vendor for corrections or updates. ScienceLogic is not responsible for other vendor's MIB files.

Viewing Error Logs for a MIB

After compiling a MIB file, you can view the error log for the MIB to check on the status of the compile operation. To do this:

1. Go to the **MIB Compiler** page (System > Tools > MIB Compiler).
2. In the **MIB Compiler** page, find the MIB you want to view information about. Select its page icon ( ).
3. The **MIB Compiler Log Messages** modal page displays error messages generated when a MIB is compiled and the effect of that error.
4. The format of each message is:

   MIB file name:line number:error message

   where:

   - ***MIB file name***. Is the name of the MIB file.
   - ***line number***. Is the line on which the error occurred.
   - ***error message***. Is a description of the error.
5. Some common error messages are:
   - ***access 'write-only' is no longer allowed in SMIv2***. SMIv2 changed the MAX-ACCESS write-only to read-write.
   - ***ACCESS is SMIv1 style, use MAX-ACCESS in SMIv2 MIBs instead***. A MIB that is considered SMIv2 compliant has the label MAX-ACCESS instead of ACCESS. Either remove the SMIv2 specification from the import or rename all of the ACCESS identifiers to MAX-ACCESS.
   - ***default value does not match underlying enumeration type***. The default value for an enumeration does not match the values declared in the enumeration. The values must match.
   - ***description missing in object definition***. All objects in an SMIv2 MIB have descriptions associated with them. This can be a blank string ("") or have text describing the object.
   - ***identifier 'identifier ' cannot be imported from module 'module name'***. This means either the type definition was not found in module name. The type definition probably exists in another module.
   - ***'identifier' should start with a lower case letter***. The identifier in a MIB element should start with a lower case alpha character. A type definition starts with an upper case alpha character.
   - ***index element 'index identifier' of row 'identifier' must have a size restriction***. All tables declared in a MIB have an index element. An index element should have a size restriction on the index element to make it a compliant MIB. To correct an integer index, add (mix.max) after the type definition to correct this issue.
   - ***invalid status 'mandatory' in SMIv2 MIB***. A MIB that is considered SMIv2 compliant has a STATUS of current, depreciated, or obsolete. Mandatory is an older SMIv1 standard that was changed with SMIv2.
   - ***last subidentifier assigned to 'identifier' may not be zero***. All SMIv2 MIBs must not have an identifier of zero in the MIB. The values have to be between 1 and 65536.

- ***length of hexadecimal string 'string' is not a multiple of 2***. Hexadecimal strings (also called octet strings) need to have 2 elements in the default value. For example, a declaration of '0f' (zero - f) is correct as it is 2 bytes. A declaration of 'f' is not correct as it is only 1 byte.
- ***lexically unexpected character, skipping to end of line***. There is an unknown character in the MIB file at the given line. Removing the character should allow the parser to continue parsing the file. Check for a back tick (`) and other strange characters and remove them from that line.
- ***macro 'macro type' has not been imported from module 'module name'***. A macro was called in the MIB however the macro type was not imported at the top of the MIB. To correct this, add the import statement at the top of the MIB.
- ***parse error, unexpected '{', expecting UPPERCASE_IDENTIFIER or LOWERCASE_IDENTIFIER or NUMBER***. The object did not parse correctly. There could be a spelling error for a MIB type or an extra comma or a single dash in the declaration. Check the MIBs syntax for any issues.
- ***parse error, unexpected COLON_COLON_EQUAL***. There is typically a parser error above this error. If you correct the parser error, the parser will find the ::= in the MIB and correctly build this identifier.
- ***parse error, unexpected DESCRIPTION***. There was a parser error above this line. Correct the parse error and this error will be corrected.
- ***parse error, unexpected MODULE_COMPLIANCE, expecting OBJECT***. The MODULE_COMPLIANCE macro was not imported from SNMPv2-CONF. All macros need to be imported into the MIB for the parser to correctly build the MIB tree.
- ***parse error, unexpected MODULE_IDENTITY, expecting OBJECT***. The MODULE-IDENTITY macro was not imported from SNMPv2-SMI. All macros need to be imported into the MIB for the parser to correctly build the MIB tree.
- ***parse error, unexpected VARIABLES***. There was a parser error above this line. Correct the parser error and this error will be corrected.
- ***redefinition of identifier 'identifier'***. An identifier was declared again in the MIB. Check the first declaration is correct or if the re-declaration is correct. Remove the one that is not correct.
- ***row's parent node must be a table node***. A row in a table must have a table identifier to link together. These typically show up as a parse error above this error.
- ***scalar's parent node must be simple node***. A scalar must have a container (OBJECT IDENTIFIER) as a parent node. A scalar cannot be part of a table, row, or another scalar.
- ***SMIv2 base type 'type definition' must be imported from SNMPv2-SMI***. The MIB is missing an import statement for the type definition.
- ***subtyping not allowed***. A table or scalar cannot have rows or objects typed in the scalar or table definition. They should be typed as the row declaration or as the object definition.
- ***table's SEQUENCE OF type does not match row type***. The table was either declared out of order from the row declaration or the rows have a mistyped sequence.
- ***TRAP-TYPE macro is not allowed in SMIv2***. Traps are to be declared in separate SMIv1 MIBs. You cannot add traps to an SMIv2 MIB.
- ***type of 'identifier ' in sequence and object type definition do not match***. This means the sequence of an object and the object definition are not the same type definition. This is more than likely a syntax error.

- u*nknown object identifier label 'identifier'*. The object identifier was not declared in the MIB file or it was not imported into the MIB. These typically show up as a parser error because that identifier was not parsed correctly.
- ***unknown object identifier label 'label name'***. This means an OID was cast as a new identifier and the MIB compiler was unable to find that identifier definition. These are called type definitions in the SMIv2 specification.
- ***unknown type 'type definition'***. The type definition was not declared in the MIB or it was not imported from another MIB definition. To correct this, add an import statement at the top of the MIB.

Rebuilding the SNMP Tree

If you have added new MIB files to Skylar One, you should rebuild the SNMP tree. When you rebuild, Skylar One searches for any newly added MIB files, compiles all the new MIB files, and rebuilds the SNMP tree to include the new files.

To rebuild the SNMP tree:

1. Go to the **MIB Compiler** page (System > Tools > MIB Compiler).
2. In the **MIB Compiler** page, select the **[Rebuild]** button.

Exporting a MIB

If you want to examine or edit a MIB file or if you want to install a MIB file on another computer, you can export the MIB file from Skylar One. To do this:

1. Go to the **MIB Compiler** page (System > Tools > MIB Compiler).
2. In the **MIB Compiler** page, find the MIB you want to export. Select its save icon ( ).
3. You will be prompted to save the MIB file to a location on your local computer.

Compiling Multiple MIB Files or Deleting Multiple MIB Files

The **MIB Compiler** page contains a drop-down field in the lower right called ***Select Action***. This field allows you to apply an action to multiple MIB files at once.

To apply an action to multiple MIB files:

1. In the **MIB Compiler** page, select the checkbox for each MIB file you want to apply the action to. To select all checkboxes for all MIB files, select the select the checkbox at the top of the page.
2. In the ***Select Action*** drop-down list, select one of the following actions:
   - *Delete MIB and Objects*. MIB and OIDs are removed from the ScienceLogic MIB library. Existing Dynamic Applications that use this MIB and OIDs are unaffected. However, users cannot create new Dynamic Applications based on this MIB.
   - *Compile MIB*. Compiles the selected MIBs
3. Select the **[Go]** button to apply the selected action to the selected MIB files.

The OID Browser

The **OID Browser** page (System > Tools > OID Browser) displays a list of all objects in all of the MIBs that have been compiled in Skylar One. The objects are arranged in the standard SNMP tree structure. The **OID Browser** page allows you to search a list of OIDs, "drill down" and view information under an OID, and add an OID to a Dynamic Application.

Drilling Down in OIDs

You can drill down the tree and view the information under an OID in the **OID Browser** page (System > Tools > OID Browser). To do so:

1. Go to the **OID Browser** page (System > Tools > OID Browser).
2. In the **OID Browser** page, find the OID that you want to drill down. Click on its value in the ***Object Name*** column.
3. The **OID Browser** page is refreshed and displays only the selected OID and all the OIDs in the next level of the OID.
4. You can continue clicking on values in the ***Object Name*** column to continue drilling down. When a checkbox appears for an OID, you cannot drill down any further.

> **NOTE**: You can also find the OID that you want to drill down and select its Show OID Tree icon ( ). The **OID Browser** page is refreshed and displays only the selected OID and the OIDs in the next three levels of the OID's tree.

Searching the List of OIDs

You can search the list of OIDs in all MIBs that have been compiled in Skylar One in the **OID Browser** page (System > Tools > OID Browser). To search the list of OIDs in Skylar One:

1. Go to the **OID Browser** page (System > Tools > OID Browser).
2. The search fields at the top of the **OID Browser** page allow you to search for OIDs by one of the following parameters:
    - ***Search where.*** Specifies the parameter you want to search by. You can select from the following:
        - *Where OID is like*. Searches all OID definitions for those that have the same object identifier as that entered in the regular expression field.
        - *Where Symbolic is like*. Searches all OID definitions for those that have the same symbolic name as that entered in the regular expression field.
        - *Where Name is like*. Searches all OID definitions for those that have the same object name as that entered in the regular expression field.
        - ***Where MIB Name is like***. Searches all OID definitions for those that have the same MIB name as that entered in the regular expression field.

- ***regular expression***. In this field you manually enter the text to search for. You can use the following special characters in this field:
    - * Match zero or more characters preceding the asterisk. For example:

    "dell*" would match "dell", "dell2650", "dell7250" and "dell1700N".

    "*dell*" would match "mydell", "dell", "dell2650", "dell7250" and "dell1700N".

    - % Match zero or more characters preceding the asterisk. This special character behaves in the same way as the asterisk.

3. When you select the **[Search]** button, the **OID Browser** page will be refreshed and will display only the OIDs that match the search parameters.

Adding an OID to a Dynamic Application from the OID Browser

When you add an object to a Dynamic Application, Skylar One collects values for the object from each device and application that is monitored with the Dynamic Application.

You can view OIDs in the **OID Browser** page and then add them to a Dynamic Application.

To do this:

1. Go to the**OID Browser** page (System > Tools > OID Browser).
2. In the **OID Browser** page, *drill down* to an object you want to add to a Dynamic Application.
3. Select the checkbox for the object you want to include in a Dynamic Application.
4. Repeat steps 2 and 3 for each OID you want to include in a Dynamic Application.
5. From the ***Select Action*** drop-down, choose the Dynamic Application to which you want to add the selected OIDs. You can also choose to add the selected OIDs to a new SNMP Configuration or SNMP Performance Dynamic Application.
6. Select the **[Go]** button to add the OID to the Dynamic Application.

Synchronizing the Latest Compiled MIBs to Collectors

You will need to use the **[Update]** button on the **OID Browser** page when you bring a new collector online after additional MIBs have already been added to the stack. The event engine process requires that this information is on the collectors when generating event details for SNMP traps.

To sync the latest compiled MIBs to a new collector:

1. Go to the **OID Browser** page (System > Tools > OID Browser).
2. Click the **[Update]** button. The status of the update displays at the top left of the page. This process might take some time to complete.

The SNMP Walker Tool

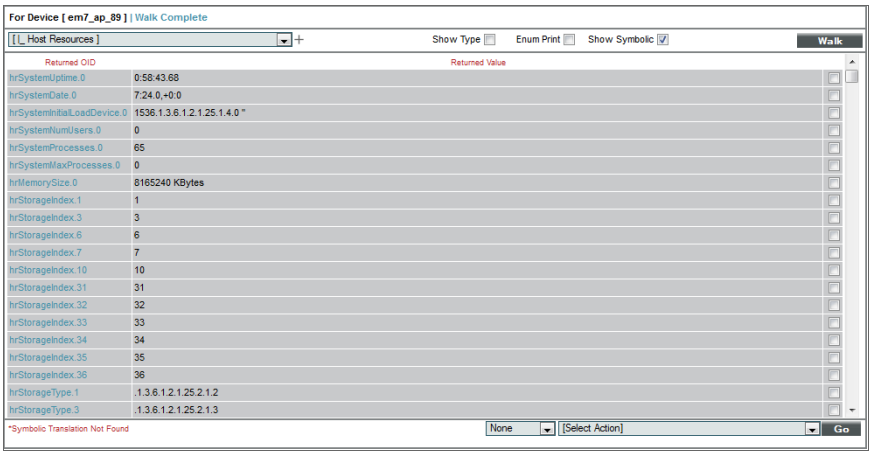
The **SNMP Walker** page allows you to "walk" one or more SNMP OIDs on a single device, so you can see a "real life" example of the type of information stored in one or more OIDs. From the **SNMP Walker** page,

Skylar One will poll the device and retrieve and display a value for each selected OID. The**SNMP Walker** page allows you to examine OIDs and also to add OIDs to new or existing Dynamic Applications.

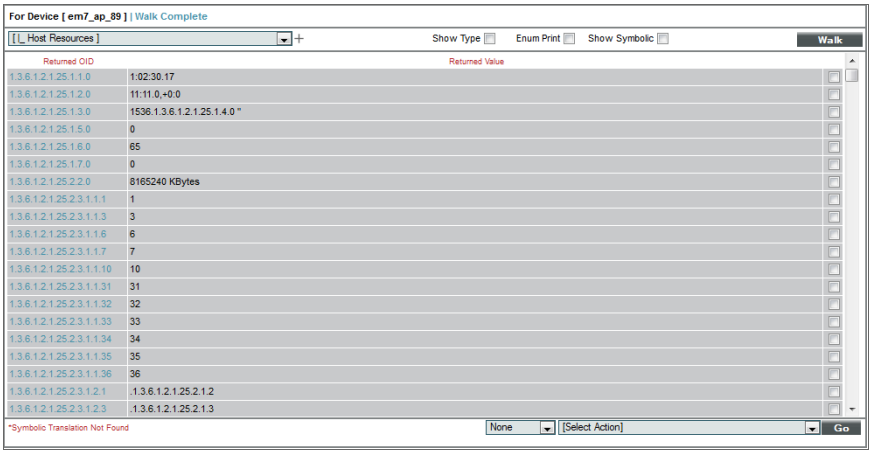
To perform an SNMP walk:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, select the wrench icon ( ) for a device on which you want to perform an SNMP walk.
3. In the **Device Properties** page, select the **[Toolbox]** tab. To perform an SNMP walk, select the SNMP Walker icon in the **Device Toolbox** pane.
4. To execute an SNMP walk provide values in the following fields:
    - ***Select OID***. Users can select from the drop-down list or manually enter an OID. The SNMP values for all sub-OIDs under the selected OID will be displayed in the results pane.
        - To enter an OID manually, select the plus sign icon. Then enter the OID in the SNMP OID field.
        - To select from the drop-down list of OIDs, highlight a MIB entry or OID entry. Choices are grouped by Common OIDs, Standard RFCs, and existing Dynamic Applications.
    - ***Show Type***. Displays the data-type (Integer, Counter, IP Address, etc.) at the start of the returned value field.
    - ***ENum Print***. Displays the list of possible values for each OID of type ENum.
    - ***Show Symbolic***. If selected, for OIDs for which Skylar One contains the MIB file, displays the text-based, descriptive name, rather than the numeric ID.
5. The results pane displays the following:
    - ***Returned OID***. Either numeric ID or symbolic (text-based) ID for each retrieved OID.
    - ***Returned Value***. Value for each OID, retrieved from the device.
    - ***Checkbox***. If the MIB is already included with Skylar One, a checkbox appears in the rows for one or more OIDs. Selecting the checkbox allows you to apply the actions from the Select Actions field to the OID

If you walked the Host Resource MIB on a device with the ***Show Symbolic*** checkbox selected, the SNMP Walker tool might return the following information:



If you walked the Host Resource MIB on a device with the ***Show Symbolic*** checkbox unselected, the SNMP Walker tool might return the following information:



Adding an OID to a Dynamic Application

From the **SNMP Walker** page, you can view data from an OID and determine if you want to add the OID to a Dynamic Application. To add an OID to a Dynamic Application (either new or existing):

1. Execute an SNMP Walk (described in the steps above).
2. In the **Results** pane, select the checkbox for each OID you want to add to a Dynamic Application.
3. In the ***Select Actions*** field, select the Dynamic Application to which you want to add the selected OID (s).
4. Select the **[Go]** button.
5. The **Collection Objects** page for the new or existing Dynamic Application appears, where you can use the OID to define a collection object, edit the parameters for the collection object, and save the collection object in the Dynamic Application.

You can also add an OID to a Dynamic Application from the **OID Browser** page. To learn more, see the section *Adding an OID to a Dynamic Application from the OID Browser*.

Creating an SNMP Performance Dynamic Application

In this section we will create an SNMP Dynamic Application. Our Dynamic Application will:

- Collect information about file systems from the Host Resources MIB.
- Include collection objects that get size and usage values.
- Include a presentation object that displays percentage used.
- Include a collection object to of type "label", to label the graph.
- Include a discovery object.

Defining the Dynamic Application Properties

To create the Dynamic Application and define the general properties for this Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the **[Actions]** button, and then select *Create New Dynamic Application*. The **Dynamic Applications Create New Application** page appears.
3. Supply values in the following fields:
    - ***Application Name***. Enter "SNMP Performance Dynamic Application" in this field.
    - ***Application Type***. Select *SNMP Performance*.
    - ***Poll Frequency***. Select *Every 1 Minute* to see data as quickly as possible.
4. For this example, you can leave the remaining fields set to their default value. Select the **[Save]** button to save the Dynamic Application.

Adding the Discovery Object

This Dynamic Application includes one discovery object. A discovery object helps Skylar One to align the Dynamic Application to devices that use the Host Resources MIB.

A discovery object is a variable that is unique to a specific hardware component or software application. Ideally, a discovery object is a variable that is always available if SNMP is running on a device, regardless of the processes running on the device or the state of the device. For example, an OID that contains a serial number of a hardware version number would make a good discovery object.

During initial discovery and nightly auto-discovery, Skylar One checks each discovered device against the list of already-defined Dynamic Applications. Skylar One searches each discovered device to find "discovery objects" and aligns devices with the appropriate Dynamic Application(s).

For example:

- Suppose your network includes servers from Dell.
- Suppose your Skylar One system includes a Dynamic Application for Dell servers.
- Suppose the Dynamic Application for Dell servers uses the SNMP variable for chassis ID to create a discovery object. This chassis-ID variable is defined by Dell and published in their MIB files. Following SNMP standards, the variable will have a unique, numeric name, used only for Dell servers.
- Because a chassis is included with each Dell server and because the chassis is not affected by software processes or the operating system, the chassis-ID variable will always be available if SNMP is running on a Dell server.
- During initial discovery and nightly auto-discovery, if Skylar One can retrieve a value for the unique chassis-ID variable, Skylar One concludes that the device is a Dell server. Skylar One will then assign the Dynamic Application for Dell servers to the device. Skylar One will later use the Dynamic Application for Dell servers to retrieve information from the device.
- During initial discovery and nightly auto-discovery, if Skylar One cannot retrieve a value for the chassis-ID variable, Skylar One concludes that the device is not a Dell server. Skylar One will not assign the Dynamic Application for Dell servers to the device.

For more details on discovery, see the manual on ***Discovery and Credentials***.

To create the discovery object to the Dynamic Application:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon () for the *SNMP Performance Dynamic Application*. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page appears.
4. Supply values in the following fields:
    - ***Object Name***. Enter "Discovery Object" in this field.
    - ***SNMP OID***. Enter ".1.3.6.1.2.1.25.2.3.1.3" in this field. This is the OID for Storage Description, as specified in the Host Resources MIB. We chose this OID to use for the discovery object because a device that uses the Host Resources MIB to report file system information will always include this OID
    - ***Class Type***. Select *100.Discovery*. Skylar One will gather values from all objects in the Dynamic Application.
5. Select the **[Save]** button, and the new object will be added and new fields will appear. Supply values in the following fields:
    - ***Alignment Condition***. Select *Align if OID is present*. Skylar One will align the Dynamic Application with a device only if the discovery object returns a value.
    - ***Validity Check***. We did not specify a value in this field.
6. Select the **[Save]** button to save the discovery object.

Adding the Collection Objects

Our example Dynamic Application contains three collection objects:

- Storage size
- Storage used
- Storage description

To create these collection objects, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon (🔧) for the *SNMP Performance Dynamic Application*. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page appears.
4. To create the collection object for ***storage size***, supply values in the following fields:
    - ***Object Name***. Enter "Storage Size" in this field.
    - ***SNMP OID***. Enter ".1.3.6.1.2.1.25.2.3.1.5". This is the OID for storage size, as specified in the Host Resources MIB.
    - ***Class Type***. Storage size is a number that can go up or down between polls. Select *4 Performance Gauge* in this field.
    - ***Group Number***. Select *Group 1*. For performance Dynamic Applications, Skylar One uses the ***Group Number*** setting to associate performance values with the appropriate labels. For the performance graph for this example to display labels correctly, all the collection objects must be in the same group.
5. For this example, you can leave the remaining fields set to their default values.
6. Select the **[Save]** button.
7. Select the **[Reset]** button to clear the form fields.
8. To create the collection object for storage used, supply values in the following fields:
    - ***Object Name***. Enter "Storage Used" in this field.
    - ***SNMP OID***. Enter ".1.3.6.1.2.1.25.2.3.1.6". This is the OID for storage description , as specified in the Host Resources MIB.
    - ***Class Type***. Storage size is a number that can go up or down between polls. Select *4 Performance Gauge* in this field.
    - ***Group Number***. Select *Group 1*. For performance Dynamic Applications, Skylar One uses the ***Group Number*** setting to associate performance values with the appropriate labels. For the performance graph for this example to display labels correctly, all the collection objects must be in the same group.
9. For this example, you can leave the remaining fields set to their default values.
10. Select the **[Save]** button.
11. Select the **[Reset]** button to clear the form fields.
12. To create the collection object for the ***storage description label***, supply values in the following fields:
    - ***Object Name***. Enter "Storage Description" in this field.
    - ***SNMP OID***. Enter ".1.3.6.1.2.1.25.2.3.1.3". This is the storage description OID specified in the Host Resources MIB.

- ***Class Type***. Select *Label (Always Polled)* in this field. In performance Dynamic Applications, collection objects that use this class type are string values that Skylar One uses to label the lines on a performance graph.
- ***Group Number***. Select *Group 1*. For performance Dynamic Applications, Skylar One uses the Group Number setting to associate performance values with the appropriate labels. For the performance graph for this example to display labels correctly, all the collection objects must be in the same group.

13. For this example, you can leave the remaining fields set to their default values.
14. Select the **[Save]** button.

Creating the Presentation Object

When you create a collection object in a Dynamic Application of type Performance, Skylar One automatically creates a presentation object that corresponds to that collection object. In this example, we will remove these presentation objects and create a new presentation object that displays file system usage in percent.

To create the presentation object:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon () for the *SNMP Performance Dynamic Application*. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[Presentations]** tab. The **Dynamic Applications Presentation Objects** page appears.
4. In the **Dynamic Applications Presentation Objects** page, the *Storage Size* and *Storage Used* collection objects have been created by default. Select each presentation object's delete icon () to delete them.
5. Select the **[Reset]** button. To create a new presentation object that displays storage used, in percent, enter values in the following fields:
    - **Report Name**. Enter "Percentage Used" in this field.
    - **Active State**. Select *Enabled*. Skylar One will generate a report of the presentation object.
    - **Data Unit**. Enter "Percent" into this field.
    - **Abbreviation/Suffix**. Enter "%" into this field.
    - **Show as Percent**. Select *Yes*. The graph will display percent values.
    - **Formula Editor**. We want our graph to display Percentage Used, so we will use the following formula:

      *Storage Used/Storage Size * 100*

      We can enter this formula manually, or by selecting the Object IDs in the ***Formula Editor*** and selecting the **[Add]** button. Using the Object IDs provided by Skylar One, enter the following into the ***Formula Editor***:

      *(o_9223)/(o_9222)*100*

> **NOTE**: The object IDs in your Dynamic Application will be unique to your system. In the provided formula, you must replace "o_5511" and "o_5513" with the object IDs for the ***Storage Used*** and ***Storage Size*** collection objects in your system.

6. In this example, you can leave the remaining fields at their default value.
7. Select the **[Save As]** button to save the presentation object.

Manually Aligning the Dynamic Application to a Device

In this example we will align the Dynamic Application to a device that supports the Host Resource MIB. By manually aligning the Dynamic Application to a device, we can immediately view the Percentage Used data in the presentation object we defined.

To manually align the Dynamic Application to a device:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the device you want to align the Dynamic Application to. In this example, we are aligning the Dynamic Application to a server. Select the device's wrench icon ( ).
3. The **Device Properties** page appears. Select the **[Collections]** tab.
4. In the **Dynamic Application Collections** page, select the **[Action]** button and select *Add Dynamic Application*.
5. The **Dynamic Application Alignment** page appears. Select *SNMP Performance Dynamic Application* to align this Dynamic Application with the server in the left pane. In the right pane, select *Default SNMP Credential*.
6. Select the **[Save]** button to add the Dynamic Application to the device.

Viewing the Report

After the Dynamic Application has collected data from the aligned server, you can view the performance report for that device. To view the performance report for a device with the *SNMP Performance Dynamic Application* aligned to it:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the device you want to align the Dynamic Application to. In this example, we are aligning the Dynamic Application to a server. Select the device's wrench icon ( ).
3. The **Device Properties** page appears. Select the **[Collections]** tab.
4. From the **Dynamic Application Collections** page, select the **[Reset]** button to update the page with the latest information.
5. Locate the *SNMP Performance Dynamic Application*. If the graph icon ( ) is colored, the performance report is available. Select the graph icon for the ***Percentage Used*** presentation object.

Or:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the device you aligned the *SNMP Performance Dynamic Application* to. Select the device's graph icon ().
3. The **Device Summary** page appears. Select the **[Performance]** tab.
4. In the left NavBar, select *SNMP Performance Dynamic Application*, then select ***Percentage Used***.
5. The SNMP Performance Dynamic Application | Percentage Used report is displayed.
    - The report returns collection objects as a percentage value. You can mouseover different data point on the report, and the report will display the percentage values for the given time
    - The percentage is displayed to the left of the report. The values for each label object are displayed in the graph key at the bottom of the report.
6. To learn more above device performance reports, see the manual ***Device Management***.

Chapter

4

Database Dynamic Applications

Overview

This chapter defines Database Dynamic Applications and explains what you can monitor with Database Dynamic Applications.

This chapter covers the following topics:

Collection Objects 277

Example of a Database Performance Dynamic Application 279

Example of a Dynamic Application of Type "Database Configuration" 294

Example of a Dynamic Application with an Identity-Based Relationship 300

Aligning the Dynamic Applications 303

Prerequisites

This chapter does not describe elements of Dynamic Application development that are common to all Dynamic Application types. Before reading this manual, you should be familiar with the common elements and concepts of Dynamic Applications. For details on the common elements of Dynamic Applications, see the manual ***Dynamic Application Development***.

You should be familiar with the query language used by your database before developing a database Dynamic Application. You must also be familiar with the database schema and the data you want to monitor before developing a database Dynamic Application. For help with these tasks, see your database administrator.

What is a Database Dynamic Application?

Dynamic Applications come in two broad categories, called ***archetypes***:

- ***Dynamic Applications of archetype Performance***. These Dynamic Applications retrieve trendable (that is, data that can be graphed) performance data from devices or applications. Only this archetype includes the **[Presentations]** tab for defining custom reports. After data has been collected, these reports can be displayed in the Device Management > Performance tab.
- ***Dynamic Applications of archetype Configuration***. These Dynamic Applications retrieve configuration data from devices or applications. Data from this archetype can be automatically linked to fields in asset records and can also be displayed in **Hardware Profile** reports and the **Software Found** page. Skylar One can automatically monitor one or more data points from this archetype for changes. If the value of the data point changes, Skylar One can automatically trigger an event.

> **NOTE**: Skylar One also includes ***Dynamic Applications of archetype Journal***. ***The Journal archetype is available only when using the Snippet protocol***. These Dynamic Applications use custom-written Python code to retrieve data from devices or applications. Skylar One will display the collected data in log format. Each log entry can contain multiple collected values and can change over time.

These archetypes contain the following types of Dynamic Applications for databases:

- ***Database Configuration***. The Dynamic Application retrieves ***configuration*** data from a database on a managed device. The Dynamic Application includes SQL queries to retrieve data. Skylar One executes these queries against a database on each subscriber device. Skylar One displays the returned data in configuration tables for each subscriber devices.
- ***Database Performance***. The Dynamic Application retrieves ***trendable performance*** data from a database on a managed device. The Dynamic Application includes SQL queries to retrieve data. Skylar One executes these queries against a database on each subscriber device. Skylar One displays the returned data in graphs in the **[Performance]** tab for each subscriber device.

Skylar One also includes Dynamic Applications for the following protocols: SNMP, SOAP, Snippet (Python), WMI, XML, and XSLT. For an overview of all types of Dynamic Applications see the ***Dynamic Application Development*** chapter. For details on each protocol, see the manual on that specific protocol (for example, for SNMP, see the ***Dynamic Application Development - SNMP*** chapter).

How Do I Allow Skylar One to Access the Database?

For Skylar One to successfully send queries to an external database:

- Skylar One must have permission to connect to the device that is hosting the database. You might have to perform some configuration tasks on the device or on your firewalls to allow Skylar One access.
- If you want Skylar One to query an external database, you must configure the appropriate security parameters on the database to give Skylar One access to the database.
- Skylar One must use a valid database username and database password to query the database. To meet this requirement, Skylar One uses ***credentials***.

Credentials are access profiles (username and password, plus additional information) for external systems. These profiles allow Skylar One to access external systems while maintaining the security of the access accounts. Users who need Skylar One to retrieve data from these external systems see only the name of the credential, not the username, password, and network information.

For more details on credentials, see the manual ***Discovery and Credentials***.

What Can I Monitor with a Database Dynamic Application?

With a Database Dynamic Application, you can monitor any value that can be retrieved with a database query.

However, many Database Dynamic Applications of type Performance query only performance data, like number of reads, number of writes, write time, read time, processor usage, memory usage, number of threads, and buffer cache hits.

Many Database Dynamic Applications of type Configuration query only configuration data, like buffer size, heap size, locks, and number of active users.

Can I Create My Own Database Dynamic Applications?

You can create your own Dynamic Applications to suit your environment and your needs. To create your own Dynamic Applications, you must:

- Determine the data you want to retrieve and monitor.
- Determine the queries you will use to retrieve that data. To write these queries, you must be familiar with the query syntax that is supported by your database, and you must be familiar with the location of data in your database.

Elements of a Database Dynamic Application

For comprehensive information on elements that apply to Database Dynamic Applications, see the Common Elements of a Dynamic Application section in the **Dynamic Application Development** chapter.

Collection Objects

A ***collection object*** is an individual data-point that will be collected by a Dynamic Application. Most Dynamic Applications collect multiple data-points. These data-points are referred to as ***objects***.

For Database Dynamic Applications, each collection object is populated with the results of a query.

For example, suppose you want to monitor the write-speed of the database every 15 minutes. You could define your Dynamic Application to execute every 15 minutes. You could define a collection object called "write_speed". You could then define a query that retrieves the value for "write speed" (usually stored in one of the administrative tables). You could align this query with the "write_speed" object and populate the object.

NOTE: This chapter describes only the fields specific to collection objects for a database Dynamic Application. All the remaining fields, for both performance and configuration archetypes, are described in detail in the ***Dynamic Application Development*** chapter.

Creating a Dynamic Application

To create a Dynamic Application, you must:

1. Define the general properties of the Dynamic Application.
2. ***Define the collection objects you want to monitor***.
3. Optionally, define thresholds for the values of collection objects.
4. Optionally, define graphs of the values of collection objects.
5. Optionally, define alerts that are triggered by the values of collection objects.

All these steps, except for defining collection objects, are the same for all types of Dynamic Applications. The steps that are the same for all types of Dynamic Applications are described in the chapter ***Dynamic Application Development***.

The step that is specific to database Dynamic Applications, defining collection objects, is described in this section.

Creating Collection Objects for a Database Dynamic Application

This section describes how to define a collection object for a Dynamic Application of type Database Performance or Database Configuration. This section describes only the fields specific to a database Dynamic Application. All the remaining fields, for both performance and configuration archetypes, are described in detail in the Collection Objects section in the ***Dynamic Application Development*** chapter.

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. ***If the Dynamic Application already exists***, in the **Dynamic Applications Manager** page, find the Dynamic Application for which you want to define a new object. Select its wrench icon ( ).
3. ***To create a new Dynamic Application***, follow the steps in the ***Dynamic Application Development*** chapter. When defining a database Dynamic Application, most of the steps are the same for all types of Dynamic Applications. Only one step, creating collection objects, is unique to database Dynamic Applications. That unique step is described in this section.
4. Select the **[Collections]** tab.
5. In the **Collection Objects** page, enter a value in each field in the top pane. This section describes only the fields specific to a database Dynamic Application. All the remaining fields are described in detail in the ***Dynamic Application Development*** chapter.
    - ***SQL Query***. Enter a valid database query in this field. The value retrieved by this query will be stored in the collection object. Be sure the query has been tested on the intended database before you include it in a Dynamic Application. Also, be sure the query can be resolved within the polling frequency of the Dynamic Application.

      If a single query will return multiple columns, and you want to view the output from multiple columns and view graphs for multiple columns, you must ***create a collection object for each returned column***. This means that for each collection object, you must enter the same query and define a different ***Object Name***, ***Class Type***, and a different ***Trended Column*** for each returned column.
    - ***Trended Column***. If a query returns multiple columns, enter the name of the column that you want to see graphed in Device Management > Performance. If a query returns multiple columns and this field is left blank for each collection object, by default only the first returned column will be graphed in the Device Management > Performance page.
6. Select the **[Save]** button to save the new collection object.
7. Repeat these steps for each collection object you want to define for the Dynamic Application.

Examples of Collection Objects

- For ***MySQL***, we could enter one of the following queries into the ***SQL Query*** field:
    - To retrieve the version number of a MySQL database:

      ```
      show global variables like 'version'
      ```

    - To retrieve the number of active events on the Skylar One system:

      ```
      SELECT count( * ) Value FROM master_events.events_active;
      ```

- For ***MS SQL***, we could enter one of the following queries into the ***SQL Query*** field:
    - To retrieve the name of the MS SQL database:

      ```
      select name from sys.databases
      ```

    - To retrieve the number of logins to the database:

```
select cntr_value from sysperfinfo where counter_name='Logins/sec'
```

- For ***Oracle***, we could enter one of the following queries into the ***SQL Query*** field:
    - To retrieve the number of times a process was delayed while waiting to access the rollback segment

      ```
      SELECT (Sum(waits) / Sum(gets)) * 100 FROM v$rollstat
      ```

    - To retrieve the hit ratio of requests to the block buffer (versus "hits" to the physical disk):

      ```
      SELECT (1 - (phys.value / (db.value + cons.value))) * 100 FROM
      v$sysstat phys,v$sysstat db,v$sysstat cons WHERE phys.name =
      'physical reads' AND db.name = 'db block gets' AND cons.name =
      'consistent gets'
      ```

Example of a Database Performance Dynamic Application

In this chapter, we will walk through a Dynamic Application that monitors the performance of a MySQL database.

The ***MySQL DB Performance*** Dynamic Application makes multiple queries to an internal table in MySQL. This internal table stores status information about the MySQL server, such as number of connections, information about index scans, information about pages in the buffer pool, reads and writes to key blocks, number of open files, information on the query cache, number of slow queries, information about table locks, and information about threads.

The ***MySQL DB Performance*** Dynamic Application includes presentation objects, threshold objects, and alert definitions that allow you to monitor the status of a MySQL server.

In this chapter, we have aligned the Dynamic Application with the Database Server. The Database Server uses a MySQL database.

NOTE: The ***MySQL DB Performance*** Dynamic Application includes multiple Collection Objects, Presentation Objects, Threshold Objects, and Alerts. This chapter will walk you though only two of each type of object.

Defining the Basic Properties for the Dynamic Application

To create the container for this Dynamic Application and define its general properties, perform the following:

> **NOTE**: For details on each field and its possible options, see the ***Dynamic Application Development*** chapter.

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Click the **[Actions]** button, and then select ***Create New Dynamic Application***. The **Dynamic Applications Create New Application** page appears.
3. Supply values in the following fields:
    - ***Application Name***. Enter *MySQL:DBPerformance*.
    - ***Application Type***. The protocol Skylar One will use and the type of data that will be collected. Select *Database Performance*.
    - ***Version Number***. Accept the default value. You can customize this value and increment it according to your change-management policies.
    - ***Operational State***. Specifies whether Skylar One will collect data from devices using this Dynamic Application. This field also specifies whether Skylar One will automatically align this Dynamic Application to devices during discovery, re-discovery, and nightly auto-discovery. Select *Enabled*.
    - ***Poll Frequency***. Frequency at which Skylar One will poll devices that use this Dynamic Application. Select *"Every 5 Minutes"*, so we can quickly view retrieved data in this example.
    - ***Abandon Collection***. Accept the default value. Specifies how many collection objects must be unavailable before the Dynamic Application should stop trying to collect data and wait until the next scheduled collection session. *Default* specifies a threshold of two collection objects.

    > **NOTE**: For all objects except those retrieved from a database, the timeout limit is specified in the credential. For database objects, the timeout limit is specified internally by Skylar One.

    - ***Context***. Leave this field blank.
4. Click the **[Save]** button to save the Dynamic Application.

Defining the Discovery Object for the Dynamic Application

A ***discovery object*** is a type of collection object. If you want Skylar One to automatically align devices with a Dynamic Application during discovery, you must include a discovery object in that Dynamic Application.

NOTE: For more details on discovery objects, see the ***Dynamic ApplicationDevelopment*** chapter.

To create a discovery object for the Dynamic Application ***MySQL:DBPerformance***, we will write a query that will return a value only if MySQL is running on a device.

To create the discovery object:

1. Go to the **Dynamic Applications Manager** page System > Manage > Applications).
2. Click the wrench icon ( ) for the Dynamic Application named ***MySQL:DBPerformance***.
3. Click the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page (System > Manage > Applications > Collections) appears.
4. Supply values in the following fields:
    - ***Object Name***. Enter *Discovery Object*.
    - ***SQL Query***. This field specifies the query that Skylar One will use to collect the discovery object. We will query a value that indicates that a MySQL database exists, regardless of the status of the data in the database. Enter the following query:

      ```
      show global status like 'Connections'
      ```

        - This query searches an internal table that stores information about all connections to the MySQL database (`show global status`).
        - The query retrieves the value of the status variable "Connections" (`like 'Connections'`).
        - The variable "Connections" contains the number of connection attempts (successful or not) to the MySQL server. The value can be zero ("0") or greater.
        - This query will return a value even if no connections have been made previously.
    - ***Class Type***. Select *[100] Discovery*.
5. Click the **[Save]** button to save the collection object. Because this collection object has been defined with a ***Class Type*** of *[100] Discovery*, the user interface displays additional fields that are specific to discovery objects.
6. Enter values in the following fields:
    - ***Alignment Condition***. Specifies how this discovery object should be evaluated. Select *Align if OID is present*. This choice tells Skylar One to automatically align the Dynamic Application with each device that returns a value for the query in the ***SQL Query*** field.
    - ***Validity Check***. Leave blank.

Defining the Collection Objects

The ***MySQL:DBPerformance*** Dynamic Application on the ScienceLogic Support Site includes 27 Collection Objects. This section will walk you through the creation of only two collection objects.

NOTE: For more details on collection objects, see the ***Dynamic Application Development*** chapter.

In this section, we will create the following collection objects:

- ***Key_reads***. This collection object monitors the MySQL status variable *key_reads*. This status variable specifies the number of times MySQL had to access the file system (instead of the key cache) to fetch database indexes. If *key_reads* is large, then the key buffer is probably too small.
- ***Key_read_requests***. This collection object monitors the MySQL status variable *key_read_requests*. This status variable specifies the total number of requests to read a key block from the cache.

If MySQL must fetch database indexes from the filesystem, queries to that database will be slower than usual. If your MySQL server must frequently fetch database indexes from the filesystem, you should increase the size of the key buffer.

To create these two collection objects, perform the following:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Click the wrench icon ( ) for the Dynamic Application named ***MySQL:DBPerformance***.
3. Click the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page (System > Manage > Applications > Collections) appears.
4. First, we will define the ***Key_reads*** collection object. This collection object monitors the MySQL status variable *key_reads*. This status variable specifies the number of times MySQL had to access the file system (instead of the key cache) to fetch database indexes.
5. Supply values in the following fields:
    - ***Object Name***. Enter *Key_reads*.
    - ***SQL Query***. Enter the following:

      ```
      show global status like 'Key_reads'
      ```

        - This query searches an internal table that stores information about the MySQL server (`show global status`).
        - The query retrieves the value of the status variable "Key reads" (`like 'Key_reads'`).
        - The variable `Key_reads` contains the number of times MySQL had to access the file system (instead of the key cache) to fetch database indexes.
    - ***Class Type***. Select *1 Performance Counter*.
    - ***Group Number***. Select *No Group*, and leave the second drop-down as *Standard*.
    - ***Trended Column***. Enter *Value*. The query returns two columns: *Variable_name*, which contains the name of the variable (Key_reads) and *Value*, which contains the value of the variable. We are interested only in the value in the *Value* column. We want Skylar One to graph the value from the *Value* column.
    - ***Enable Deviation Alerting***. Do not select these checkboxes.
    - ***Description***. Leave blank.
    - ***Formula***. Leave blank.

6. Click the **[Save]** button to save the new collection object.
7. Next, we will define the ***Key_read_requests*** collection objects. This collection object monitors the MySQL status variable *key_read_requests*. This status variable specifies the total number of requests to read a key block from the cache.
8. Supply values in the following fields:
    - ***Object Name***. Enter *Key_read_requests*.
    - ***SQL Query***. Enter the following:

      ```
      show global status like 'Key_read_requests'
      ```

        - This query searches an internal table that stores information about the MySQL server (`show global status`).
        - The query retrieves the value of the status variable "Key_read_requests" (`like 'Key_read_requests'`).
        - The variable `Key_read_requests` contains the total number of requests to read a key block from the cache.
    - ***Class Type***. Select *1 Performance Counter*.
    - ***Group Number***. Select *No Group*, and leave the second drop-down as *Standard*.
    - ***Trended Column***. Enter *Value*. The query returns two columns: *Variable_name*, which contains the name of the variable (*Key_read_requests*) and *Value*, which contains the value of the variable. We are interested only in the value in the *Value* column. We want Skylar One to graph the value from the *Value* column.
    - ***Enable Deviation Alerting***. Do not select these checkboxes.
    - ***Description***. Leave blank.
    - ***Formula***. Leave blank.
9. Click the **[Save]** button to save the new collection object.
10. In our example, you will notice that the collection objects have the following object IDs:
    - ***Key_reads*** = ***o_5500***
    - ***Key_read_requests*** = ***o_5501***

NOTE: ***On your Skylar One system, the collection objects will have different object IDs***. Whether your have imported the Dynamic Application or are creating the Dynamic Application from the steps in this chapter, the collection objects will have different object IDs than on our example Skylar One system.

Defining the Presentation Objects

Presentation objects allow you to define how Skylar One should use the values collected by the Dynamic Application to create performance graphs.

NOTE: For more details on presentation objects, see the ***Dynamic Application Development*** chapter.

The ***MySQL:DBPerformance*** Dynamic Application on the ScienceLogic Support Site includes 33 Presentation Objects. This section will walk you through the creation of only two presentation objects.

In this section, we will create the following presentation objects:

- ***Key_reads***. Displays the value of the ***Key_reads*** collection object, over time. The ***Key_reads*** collection object specifies the number of times MySQL had to access the file system (instead of the key cache) to fetch database indexes. If the value of the ***Key_reads*** collection object is large, then the key buffer is probably too small. The ***Key_reads*** presentation object will graph each collected value of the ***Key_reads*** collection object and its associated date and time.
- ***Key_read_requests***. Displays the value of the ***Key_read_requests*** collection object, over time. The ***Key_read_requests*** collection object specifies the total number of requests to read a key block from the cache. The ***Key_read_requests*** presentation object will graph each collected value of the ***Key_read_requests*** collection object and its associated date and time.

To create these two presentation objects, perform the following:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Click the wrench icon ( ) for the Dynamic Application named ***MySQL:DBPerformance***.
3. Click the **[Presentations]** tab. The **Dynamic Applications Presentation Objects** page appears.
4. First, we will define the ***Key_reads*** presentation object. The ***Key_reads*** presentation object will graph each collected value of the ***Key_reads*** collection object and its associated date and time.
5. Supply values in the following fields:
    - ***Report Name***. Enter *Key_reads*. This name will appear in the NavBar of the **Performance** page for each device that subscribes to the Dynamic Application. This name will also appear as a title for the graph.
    - ***Active State.*** Select *Enabled*. Skylar One will immediately create the graph at the next polling session.
    - ***Data Unit***. Leave blank.
    - ***Abbreviation/Suffix***. Leave blank.
    - ***Show as Percent***. Select *No*.
    - ***Vitals Link***. Select *Disabled*.
    - ***Formula Editor***. Enter the following: `(o_5500)`
        - This formula tells Skylar One to graph each value of the collection object ***o_5500***. In our example system, this is the object ID for the ***Key_reads*** collection object. Skylar One will graph each value of this collection object, along with its associated date and time.

NOTE: The object ID for the ***Key_reads*** collection object will be different on your Skylar One system. If you are creating a new Dynamic Application using the instructions in this chapter, please enter the object ID for the ***Key_reads*** collection object, as it appears on your Skylar One system.

6. Click the **[Save]** button to save the new presentation object.
7. Next, we will define the ***Key_read_requests*** presentation object. The ***Key_read_requests*** presentation object will graph each collected value of the ***Key_read_requests*** collection object and its associated date and time.
8. Supply values in the following fields:
    - ***Report Name***. Enter *Key_read_requests*. This name will appear in the NavBar of the **Performance** page for each device that subscribes to the Dynamic Application. This name will also appear as a title for the graph.
    - ***Active State.*** Select *Enabled*. Skylar One will immediately create the graph at the next polling session.
    - ***Data Unit***. Leave blank.
    - ***Abbreviation/Suffix***. Leave blank.
    - ***Show as Percent***. Select *No*.
    - ***Vitals Link***. Select *Disabled*.
    - ***Formula Editor***. Enter the following: `(o_5501)`
        - This formula tells Skylar One to graph each value of the collection object ***o_5501***. On our example Skylar One system, this is the object ID for the ***Key_read_requests*** collection object. Skylar One will graph each value of this collection object, along with its associated date and time.

NOTE: The object ID for the ***Key_read_requests*** collection object will be different on your Skylar One system. If you are creating a new Dynamic Application using the instructions in this chapter, please enter the object ID for the ***Key_read_requests*** collection object, as it appears on your Skylar One system.

9. Click the **[Save]** button to save the new presentation object.

Defining the Threshold Objects

A threshold object is an object that you can use in the formula for an alert definition or a presentation object, just as you would use a collection object.

Threshold objects can also appear as thresholds in the **Device Thresholds** page (Devices > Classic Devices > wrench icon > Thresholds, or Registry > Devices > Device Manager > wrench icon > Thresholds in the classic user interface)for each device that subscribes to the Dynamic Application.

NOTE: For more details on threshold objects, see the ***Dynamic Application Development*** chapter.

The ***MySQL:DBPerformance*** Dynamic Application on the ScienceLogic Support Site includes seven threshold objects. This section will walk you through the creation of only one threshold object.

- ***Keycache_hitrate***. We will use this threshold in the formula for two alerts. The initial alert is triggered when MySQL fetches database indexes from the filesystem instead of from the key cache. This threshold defines the percentage of fetches that can be from the filesystem instead of from the key cache before Skylar One generates an alert.

To create this threshold object, perform the following:

1. Go to the **System > Manage > Applications** page (System > Manage > Applications).
2. Click the wrench icon ( ) for the Dynamic Application named ***MySQL:DBPerformance***
3. Click the **[Thresholds]** tab. The **Dynamic Applications Threshold Objects** page appears.
4. Supply values in the following fields:
    - ***Threshold Name***. Enter *Keycache_hitrate*.
    - ***Override Threshold Value***. Select *Enabled*. This threshold will appear in the the **Device Thresholds** page (Devices > Classic Devices > wrench icon > Thresholds, or Registry > Devices > Device Manager > wrench icon > Thresholds in the classic user interfacefor devices that subscribe to the Dynamic Application.
    - ***Numeric Range: High***. Enter *100*. By default, the highest possible value for this threshold will be "100". This value will appear at the high end of the slider in the the **Device Thresholds** page (Devices > Classic Devices > wrench icon > Thresholds, or Registry > Devices > Device Manager > wrench icon > Thresholds in the classic user interface.
    - ***Numeric Range: Low***. Enter *0*. By default, the lowest possible value for this threshold will be "0". This value will appear at the low end of the slider in the the **Device Thresholds** page (Devices > Classic Devices > wrench icon > Thresholds, or Registry > Devices > Device Manager > wrench icon > Thresholds in the classic user interface.
    - ***Threshold Type***. Select *Percentage*.
    - ***Threshold Value***. Enter *99*.
5. Click the **[Save]** button to save the new threshold.
6. If you imported the Dynamic Application from the ScienceLogic Support Site, you will notice that the threshold object has the following object ID:
    - ***Keycache_hitrate*** = ***t_130***

NOTE: ***On your Skylar One system, the threshold object will have different object IDs***. Whether your have imported the Dynamic Application or are creating the Dynamic Application from the steps in this chapter, the collection objects will have different object IDs than on our example Skylar One system.

Defining the Alerts

Alerts allow you to examine and manipulate values retrieved by a Dynamic Application. An alert defines the conditions during which you would like Skylar One to insert a message in the device log. You can define events that are triggered when the alert message appears in a device log.

NOTE: For details on alerts, see the ***Dynamic Application Development*** chapter.

The ***MySQL:DBPerformance*** Dynamic Application on the ScienceLogic Support Site includes 14 alerts. This section will walk you through the creation of only two alerts.

In this section, we will create the following alerts:

- ***MySQL:Keycache_hitrate_low***. This alert compares the value of the ***Key_reads*** collection object to the value of the ***Key_read_requests*** collection object. Remember that the ***Key_reads*** collection object specifies the number of times the MySQL server has had to fetch a database index from the file system instead of from the cache. The ***Key_read_requests*** object specifies the number of times the MySQL server has fetched a database index from cache. The alert says "If the number of times MySQL has had to fetch a database index from the file system is 1% or more of the number of times MySQL has fetched a database index from cache, generate an alert of severity "minor".
- ***MySQL:Keycache_hitrate_normal***. This alert compares the value of the ***Key_reads*** collection object to the value of the ***Key_read_requests*** collection object. The alert says "If the ***MySQL:Keycache_hitrate_low*** alert is still active, and if the number of times MySQL has had to fetch a database index from the file system is less than 1% of the number of times MySQL has fetched a database index from cache, generate an alert of severity "healthy".

To create these two alerts, perform the following:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications.
2. Click the wrench icon ( ) for the Dynamic Application named ***MySQL:DBPerformance***.
3. Click the **[Alerts]** tab. The **Dynamic Applications Alert Objects** page appears.
4. First, we will define the ***MySQL:Keycache_hitrate_low*** alert.
5. Supply values in the following fields:
    - ***Policy Name***. Enter *MySQL:Keycache_hitrate_low*.
    - ***Active State***. Select *Enabled*. Skylar One will monitor this alert.
    - ***Log Message***. If this alert evaluates to TRUE, the alert will insert the following message in the device log (on the device where the condition occurred). Enter the following:

      ```
      MySQL Keycache hitrate: %V% is low. Threshold: %T.
      ```

        - The **%V** variable says "substitute the value returned by the **result** function".
        - The **%T** variable says "substitute the value returned by the **threshold** function".
    - ***Maintain State***. Select *Yes*. This alert will maintain its state until it is explicitly cleared by an event.

- ***Trigger Alert***. Select *None*. This is a deprecated field.
- ***Formula Editor***. This is where you describe the conditions under which you want Skylar One to make an entry in the device log. Enter the following:

```
result(100-((o_5500/o_5501)*100)) < threshold(t_191)
```

  - This formula says: Divide the number of ***Key_reads*** by the number of ***Key_read_requests*** and convert that value to percent. If the percentage of ***Key_Reads*** is 1% or more, the alert will evaluate to TRUE. When the alert evaluates to TRUE, it makes an entry of severity "minor" in the appropriate device log.
  - ***o_5500*** is the object ID of the ***Key_reads*** collection object.

NOTE: On your Skylar One system, the ***Key_reads*** collection object will have a different object ID. Substitute the object ID from your Skylar One system.

  - ***o_5501*** is the object ID of the ***Key_read_requests*** collection object. If you have created the ***MySQL:DBPerformance*** Dynamic Application manually (instead of importing the Dynamic Application from the ScienceLogic Support Site), the ***Key_read_requests*** collection object will have a different object ID on your Skylar One system. Substitute the object ID from your Skylar One system.

NOTE: On your Skylar One system, the ***Keycache_hitrate*** threshold will have a different object ID. Substitute the object ID from your Skylar One system.

  - ***t_191*** is the object ID of the ***Keycache_hitrate*** threshold.

NOTE: On your Skylar One system, the ***Keycache_hitrate*** threshold will have a different object ID. Substitute the object ID from your Skylar One system.

  - Remember that we set the ***Keycache_hitrate*** threshold to "99".
  - The **result** function returns the value of the formula and stores the value of the formula in the **%V** variable.
  - The **threshold** function returns the value of the threshold variable and stores the value of the threshold variable in the **%T** variable.

6. Click the **[Save]** button to save the alert.
7. In our example Skylar One system, you will notice that the alert object has the following object ID:
   - ***MySQL:Keycache_hitrate_low*** = ***a_920***

NOTE: On your Skylar One system, the alert object will have a different object ID.

8. Next we will define the ***MySQL:Keycache_hitrate_normal*** alert.
9. Supply values in the following fields:
    - ***Policy Name***. Enter *MySQL:Keycache_hitrate_normal*.
    - ***Active State***. Select *Enabled*. Skylar Onewill monitor this alert.
    - ***Log Message***. If this alert evaluates to TRUE, the alert will insert the following message in the device log (on the device where the condition occurred.)Enter the following:

      ```
      MySQL Keycache hitrate: %V% is normal.
      ```

        - The **%V** variable says "substitute the value returned by the **result** function".
    - ***Maintain State***. Select *No*. This alert will not maintain its state and does not need to be explicitly cleared by an event.
    - ***Trigger Alert***. Select *None*. This is a deprecated field.
    - ***Formula Editor***. This is where you describe the conditions under which you want Skylar One to make an entry in the device log. Enter the following:

      ```
      result(100-((o_5500/o_5501)*100)) >= threshold(t_191) and active
      (a_920)
      ```

        - This formula says: Divide the number of ***Key_reads*** by the number of ***Key_read_requests*** and convert that value to percent. If the alert ***MySQL:keycache_hitrate_low*** is still active, and if the percentage of ***Key_Reads*** is 1% or less, the alert will evaluate to TRUE. When the alert evaluates to TRUE, it makes an entry of severity "healthy" in the appropriate device log.
        - ***o_5500*** is the object ID of the ***Key_reads*** collection object.

> **NOTE**: On your Skylar One system, the ***Key_reads*** collection object will have a different object ID. Substitute the object ID from your Skylar One system.

        - ***o_5501*** is the object ID of the ***Key_read_requests*** collection object.

> **NOTE**: On your Skylar One system, the ***Key_read_requests*** collection object will have a different object ID. Substitute the object ID from your Skylar One system.

        - ***t_191*** is the object ID of the ***Keycache_hitrate*** threshold.

> **NOTE**: On your Skylar One system, the ***Keycache_hitrate*** threshold will have a different object ID . Substitute the object ID from your Skylar One system.

        - ***a_920*** is the object ID of the alert ***MySQL:keycache_hitrate_low.***

NOTE: On your Skylar One system, the ***Keycache_hitrate_low*** alert will have a different object ID. Substitute the object ID from your Skylar One system.

- Remember that we set the ***Keycache_hitrate*** threshold to "99".
- The **result** function returns the value of the formula and stores the value of the formula in the **%V** variable.
- The **threshold** function returns the value of the threshold variable and stores the value of the threshold variable in the **%T** variable.
- The **active** function checks the state of a specified alert. If the specified alert is still active, the **active** function returns the value TRUE.

10. Click the **[Save]** button to save the alert.
11. On our example Skylar One system, notice that the alert object has the following object ID:
    - ***MySQL:Keycache_hitrate_normal*** = ***a_921***

NOTE: On your Skylar One system, the alert object will have a different object ID.

Creating a Credential for the "MySQL: DBPerformance" Dynamic Application

If you want to align the ***MySQL:DBPerformance*** Dynamic Application with Skylar One's MySQL database, you must create a database credential that allows access to Skylar One's MySQL database.

Before you define the credential, you must collect the information you will need. In this case, you will need the username and password for the MySQL database. This account was defined during setup and is different than the account for logging into Skylar One.

If you can log in to the phpMyAdmin tool from the **Appliance Manager** page (System > Settings > Appliances), you can use the username and password you used to access the phpMyAdmin tool as the username and password in the credential. For details on accessing the phpMyAdmin tool, see the manual ***System Administration***. If you need help, ask your administrator.

For details on the database password, see the manual ***System Administration***.

For details on credentials, see the manual ***Discovery and Credentials***.

To create the credential for this example:

1. Go to the **Credential Management** page (System > Manage > Credentials).
2. In the the **Credential Management** page, click the **[Create]** menu. Select *Database Credential*.
3. The **Credential Editor** modal page appears. In this page, you can define the new database credential. To define the new credential, supply values in the following fields:

- ***Profile Name***. Enter *EM7 DB*.
- ***DB Type***. Select *MySQL*.
- ***DB Name***. Enter *master*.
- ***DB User***. Username associated with a valid account on the database.
- ***Password.*** Password associated with a valid account on the database.
- ***Hostname/IP.*** Hostname or IP address where the database resides. To use the localhost, in the ***Hostname/IP*** field, enter the IP address *127.0.0.1*. The credential will not work if you enter the string *localhost* in the ***Hostname/IP*** field.
- ***Port***. Enter *7706*.

4. Click the **[Save]** button to save the new database credential.

Aligning the Dynamic Application with a Device

For our example, we aligned the ***MySQL:DBPerformance*** Dynamic Application with the Database Server.

There are three ways to align the ***MySQL:DBPerformance*** Dynamic Application with a device:

- ***During initial discovery or nightly auto-discovery***. Because the ***MySQL:DBPerformance*** Dynamic Application includes a discovery object, Skylar One can automatically align this Dynamic Application with devices during discovery. For details on discovery, see the manual ***Discovery and Credentials***.
- ***Manual discovery for the Dynamic Application***. Because the ***MySQL:DBPerformance*** Dynamic Application includes a discovery object, Skylar One can automatically align this Dynamic Application with devices. From the **Dynamic Applications Manager** page, you can manually execute discovery for all devices, but only for the ***MySQL:DBPerformance*** Dynamic Application. For details on how to perform this type of discovery, see the ***Dynamic Application Development*** chapter..
- ***Manually associate the Dynamic Application with an existing device***. For details on how to perform this type of discovery, see the ***Dynamic ApplicationDevelopment*** chapter.

This section will walk you through the steps to manually align the ***MySQL:DBPerformance*** Dynamic Application with the Database Server or the All-In-One Appliance.

To manually align the Dynamic Application to the Database Server or the All-In-One Appliance:

1. Go to the **Appliance Manager** page (System > Settings > Appliances). Determine the device name of the Database Server or the All-In-One Appliance.
2. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
3. Find the device where the database resides (either the Database Server or the All-In-One Appliance). Click the device's wrench icon ( ).
4. The **Device Properties** page appears. Click the **[Collections]** tab.
5. In the **Dynamic Application Collections** page, click the **[Action]** menu and select *Add Dynamic Application*.
6. The **Dynamic Application Alignment** page appears. Select *MySQL:DBPerformance* in the **Dynamic Applications** pane, and select *EM7 DB* in the **Credentials** pane.

7. Click the **[Save]** button to add the Dynamic Application to the device.
8. At the next polling period, Skylar One should start collecting the data specified in the ***MySQL:DBPerformance*** Dynamic Application from the device where Skylar One's database resides.

Viewing Reports for the Dynamic Application

For our example, we aligned the ***MySQL:DBPerformance*** Dynamic Application with the Database Server on our device em7_ao.

To view the graphs for the presentation objects ***key_reads*** and ***key_read_requests***:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Find the device for which you want to edit thresholds. Click its bar-graph icon ().
3. Click the **[Performance]** tab. The **Device Performance** page appears.
4. In the NavBar, find the entry for the ***MySQL:DBPerformance*** Dynamic Application and expand it. Select the entry for ***Key_reads***.
5. The graph for ***Key_reads*** appears.
6. The graph displays the value of the ***Key_reads*** collection object in the y-axis and the date and time in the x-axis.
7. In the NavBar, find the entry for the ***MySQL:DBPerformance*** Dynamic Application and expand it. Select the entry for ***Key_read_requests***.
8. The graph for ***Key_read_requests*** appears.
9. The graph displays the value of the ***Key_read_requests*** collection object in the y-axis and the date and time in the x-axis.

Viewing Alerts for the Dynamic Application

To view alerts for the Dynamic Application:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Find the subscriber device for which you want to edit thresholds. Click its wrench icon ().
3. Click the **[Logs]** tab. The **Device Logs & Messages** page appears.
4. Look for alert messages from the Dynamic Application. In our example, our MySQL database did not fetch any database indexes from the file system, so only the "normal" alert appears in our device log.

Changing the Threshold for a Subscriber Device

You can change one or more threshold values for a single device. When Skylar One evaluates alerts ***for that device***, it will use the threshold values set in the **Device Threshold** page instead of the threshold value set in the **Dynamic Application Threshold Objects** page.

To edit a threshold for a single device:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Find the device for which you want to edit thresholds. Click its wrench icon (🔧).
3. Click the **[Thresholds]** tab. The **Device Thresholds** page appears.
4. In **Device Thresholds** page, move the sliders to edit one or more thresholds.
5. To save your changes, click the **[Save]** button.

Example of a Dynamic Application of Type "Database Configuration"

In this chapter, we will walk through a Dynamic Application that monitors the performance of a MySQL database.

This chapter will walk you through the Dynamic Application ***MySQL:DBConfiguration***. The ***MySQL:DBConfiguration*** Dynamic Application makes multiple queries to an internal table in MySQL. This internal table stores variables that contain configuration information for the MySQL server, such as MySQL version, the configuration of the auto-commit feature, the maximum allowed number of connections, and the sizes of various caches.

In this chapter, we have aligned the Dynamic Application with the Database Server. The Database Server uses a MySQL database.

Defining the Basic Properties for the Dynamic Application

To create the container for this Dynamic Application and define its general properties, perform the following:

> **NOTE**: For details on each field and its possible options, see the ***Dynamic Application Development*** chapter.

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the **[Actions]** button, and then select *Create New Dynamic Application*. The **Dynamic Applications Create New Application** page appears.
3. Supply values in the following fields:
   - ***Application Name***. Enter *MySQL:DBConfiguration*.
   - ***Application Type***. The protocol Skylar One will use and the type of data that will be collected. Select *Database Configuration*.
   - ***Version Number***. Accept the default value. You can customize this value and increment it according to your change-management policies.

- ***Operational State***. Specifies whether Skylar One will collect data from devices using this Dynamic Application. This field also specifies whether Skylar One will automatically align this Dynamic Application to devices during discovery, re-discovery, and nightly auto-discovery. Select *Enabled*.
- ***Poll Frequency***. Frequency at which Skylar One will poll devices that use this Dynamic Application. Select *"Every 2 Minutes"*, so we can quickly view retrieved data in this example.
- ***Abandon Collection***. Accept the default value. Specifies how many collection objects must be unavailable before the Dynamic Application should stop trying to collect data and wait until the next scheduled collection session. *Default* specifies a threshold of two collection objects.

> **NOTE**: For all objects except those retrieved from a database, the timeout limit is specified in the credential. For database objects, the timeout limit is specified internally by Skylar One.

- ***Context***. Leave this field blank.

4. Select the **[Save]** button to save the Dynamic Application.

Defining the Discovery Object for the Dynamic Application

A ***discovery object*** is a type of collection object. If you want Skylar One to automatically align devices with a Dynamic Application during discovery, you must include a discovery object in that Dynamic Application.

> **NOTE**: For more details on discovery objects, see the manual ***Dynamic Application Development***.

To create a discovery object for the Dynamic Application ***MySQL: DBConfiguration***, we will write a query that will return a value only if MySQL is running on a device.

To create the discovery object:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon ( ) for the Dynamic Application named ***MySQL:DB Configuration***.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page appears.
4. Supply values in the following fields:
   - ***Object Name***. Enter *Discovery Object*.
   - ***SQL Query***. This field specifies the query that Skylar One will use to collect the discovery object. We will query a value that indicates that a MySQL database exists, regardless of the status of the data in the database. Enter the following query:

     ```
     show global variables like 'version'
     ```

     - This query searches an internal table that stores configuration information about the MySQL database (`show global variables`).

        - The query retrieves the value of the status variable "version" (`like 'version'`).
        - The variable "version" contains the version number and name of the MySQL server.
        - This query will return a value only if MySQL is installed on a device. This query is not dependent on the data stored in the database.
    - ***Class Type***. Select *[100] Discovery*.
5. Select the **[Save]** button to save the collection object and the page will change to appear as it does below. Because this collection object has been defined with a ***Class Type*** of *[100] Discovery*, Skylar One displays additional fields that are specific to discovery objects.
6. Enter values in the following fields:
    - ***Alignment Condition***. Specifies how this discovery object should be evaluated. Select *Align if OID is present*. This choice tells Skylar One to automatically align the Dynamic Application with each device that returns a value for the query in the ***SQL Query*** field.
    - ***Validity Check***. Leave blank.

Defining the Collection Objects

This section will walk you through the creation of only six collection objects for the ***MySQL: DBConfiguration*** Dynamic Application.

> **NOTE**: For more details on collection objects, see the ***Dynamic Application Development*** chapter.

In this section, we will create the following collection objects:

- ***mysql_version***. This collection object monitors the MySQL configuration variable *version*. This configuration variable contains the version number and version name of the MySQL software.
- ***auto_commit***. This collection object monitors the MySQL configuration variable *autocommit*. This configuration variable specifies whether the autocommit mode is enabled (set to 1). If autocommit is enabled, users do not have to commit each transaction. If autocommit is not enabled, users must either commit each transaction to save it to the database.
- ***max_connections***. This collection object monitors the MySQL configuration variable *max_connections*. This configuration variable specifies the maximum allowed number of client connections to the database.
- ***key_cache_size***. This collection object monitors the MySQL configuration variable *key_buffer_size*. This configuration variable specifies the size of the cache where index information is stored.
- ***query_cache_size***. This collection object monitors the MySQL configuration variable *query_cache_size*. This configuration variable specifies the size of the cache where query results are stored.
- ***thead_cache_size***. This collection object monitors the MySQL configuration variable *thread_cache_size*. This configuration variable specifies the number of threads that should be saved in the thread cache.

To create these collection objects, perform the following:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon ( ) for the Dynamic Application named ***MySQL: DBConfiguration***.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page appears.
4. First, we will define the ***mysql_version*** collection object. This collection object monitors the MySQL configuration variable *version*. This configuration variable contains the version number and version name of the MySQL software.
5. Supply values in the following fields:
    - ***Object Name***. Enter *MySQL_version*.
    - ***SQL Query***. Enter the following:

      ```
      show global variables like 'version'
      ```

        - This query searches an internal table that stores configuration information about the MySQL database (`show global variables`).
        - The query retrieves the value of the status variable "version" (`like 'version'`).
        - The variable "version" contains the version number and version name of the MySQL server.
    - ***Class Type***. Select *10 Config Character*. The returned value contains both numbers and letters.
    - ***Group Number***. Select *Group 1* and *Standard*. In the **Configuration Report** page, this collection object will appear in table 1.
    - ***Asset/Form Link***. Leave these fields blank.
    - ***Inventory Link***. Select *Disabled*.
    - ***Change Alerting***. Select *Disabled*. Skylar One will not trigger an event if the value of this collection object changes. Notice that we will enable this option for all other collection objects in this Dynamic Application. Skylar One will then automatically generate events if the value of these collection objects change.
    - ***Table Alignment***. Select *Left*. This specifies that column-values will be left justified and will not be translated from hexadecimal unicode.
    - ***Trended Column***. Enter *Value*. The query for this collection object returns two columns: *Variable_name*, which contains the name of the variable (version) and *Value*, which contains the value of the variable. We are interested only in the value in the *Value* column and want that value to appear in the **Configuration Report** page.
    - ***Description***. Leave blank.
    - ***Formula***. Leave blank.
6. Select the **[Save]** button to save the new collection object.
7. For the remaining collection objects, enter the following and select the **[Save]** button for each collection object. If a field is not specified in this table, accept its default value.

Object Name	SQL Query	Class Type	Group Number	Change Alerting	Trended Column
auto_commit	show global variables like 'autocommit'	[15] Config Numeric	2	Enabled	Value
max_ connections	show global variables like 'max_ connections'	[15] Config Numeric	2	Enabled	Value
key_cache_ size	show global variables like 'key_buffer_ size'	[15] Config Numeric	3	Enabled	Value
thread_ cache_size	show global variables like 'thread_cache_ size'	[15] Config Numeric	3	Enabled	Value
query_ cache_size	show global variables like 'query_thread_ size'	[15] Config Numeric	3	Enabled	Value

Creating a Credential for the MySQL:DBConfiguration Dynamic Application

If you want to align the ***MySQL:DBConfiguration*** Dynamic Application with Skylar One's MySQL database, you must create a database credential that allows access to Skylar One's MySQL database.

For details on creating a credential for Skylar One's MySQL database, see the previous section.

Aligning the Dynamic Application with a Device

For our example, we aligned the ***MySQL:DBConfiguration*** Dynamic Application with the Database Server.

There are three ways to align the ***MySQL:DBConfiguration*** Dynamic Application with a Device:

- ***During initial discovery or nightly auto-discovery***. Because the ***MySQL:DBConfiguration*** Dynamic Application includes a discovery object, Skylar One can automatically align this Dynamic Application with devices during discovery. For details on discovery, see the manual ***Discovery and Credentials***.

- ***Manual discovery for the Dynamic Application***. Because the ***MySQL:DBConfiguration*** Dynamic Application includes a discovery object, Skylar One can automatically align this Dynamic Application with devices. From the **Dynamic Applications Manager** page, you can manually execute discovery for all devices, but only for the ***MySQL:DBConfiguration*** Dynamic Application. For details on how to perform this type of discovery, see the ***Dynamic Application Development*** chapter.
- ***Manually associate the Dynamic Application with an existing device***. For details on how to perform this type of discovery, see the ***Dynamic Application Development*** chapter.

This section will walk you through the steps to manually align the ***MySQL:DBConfiguration*** Dynamic Application with the Database Server or the All-In-One Appliance.

To manually align the Dynamic Application to the Database Server or the All-In-One Appliance:

1. Go the **Appliance Manager** page (System > Settings > Appliances). Determine the device name of the Database Server or the All-In-One Appliance.
2. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
3. Find the device where the database resides (either the Database Server or the All-In-One Appliance). Select the device's wrench icon ( ).
4. The **Device Properties** page appears. Select the **[Collections]** tab.
5. In the **Dynamic Application Collections** page, select the **[Action]** menu and select *Add Dynamic Application*.
6. The **Dynamic Application Alignment** page appears. Select *MySQL:DBConfiguration* in the **Dynamic Applications** pane, and select *EM7 DB* in the **Credentials** pane.
7. Select the **[Save]** button to add the Dynamic Application to the device.
8. At the next polling period, Skylar One should start collecting the data specified in the ***MySQL:DBConfiguration*** Dynamic Application from the device where Skylar One's database resides.

Viewing the Configuration Report for the Dynamic Application

For our example, we aligned the ***MySQL:DBConfiguration*** Dynamic Application with the Database Server on our device em7_ao.

Skylar One automatically creates a Configuration Report on each device that is aligned with the Dynamic Application. The Configuration Report displays the latest data collected from the device by the Dynamic Application.

> **NOTE**: For more details on the **Configuration Report** page and a description of the actions you can perform from the **Configuration Report** page, see the manual ***Monitoring Device Infrastructure Health***.

To view the Configuration Report for the Dynamic Application:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Find the device that you aligned with the Dynamic Application. (In our example, this device is ***em7_ao***). Select its bar-graph icon ( ).
3. Select the **[Configs]** tab. The **Configuration Report** page appears.
4. In the NavBar, find the entry for the ***MySQL:DBConfiguration*** Dynamic Application and expand it.
5. In the **Configuration Report** page, notice how the values are stored in tables as we specified in the ***Group*** field for each collection object. That is, mysql_version is displayed in the first table, auto_commit and max_connections are displayed in the second table, and key_cache_size, query_cache_size, and thread_cache_size are displayed in the third table.

Example of a Dynamic Application with an Identity-Based Relationship

In this example, we will describe how to build a pair of Dynamic Applications that create relationships between Sciencelogic Database Servers and the Data Collectors and Message Collectors in the same system.

Building Dynamic Applications with an Identity-Based Relationship

In this example, we will describe how to build a pair of Dynamic Applications that create relationships between Sciencelogic Database Servers and the Data Collectors and Message Collectors in the same system.

In this example, the data used to build the relationships is in the *master.system_settings_licenses* database table. This table contains information about Skylar One appliances, including the appliance type, name, IP address, etc. On a Database Server, this table contains a row for each appliance in the system. On a Data Collector or Message Collector, this table contains a row for only that Data Collector or Message Collector.

The following database fields are used in this example:

- ***id***. The automatically assigned ID number for the appliance. This will be combined with the host name of the appliance to create a unique identifier for the relationships.
- ***name***. The host name of the appliance configured in the System > Settings > Appliances page. This will be combined with the id of the appliance to create a unique identifier for the relationships.
- ***function***. The type of appliance. The appliance types are stored as integer values. This field contains 5 for Data Collectors and 6 for Message Collectors.

The following database query is used in this example:

```
SELECT CONCAT(CAST(id AS CHAR), name) as uid,

        CASE
```

```
WHEN function=5 THEN "Data Collector"

WHEN function=6 THEN "Message Collector"

END as func

FROM master.system_settings_licenses

WHERE function IN (5,6);
```

This query returns two values:

- ***uid***. A string concatenation of the id and name fields. This will be used as the unique identifier for the relationships. This example assumes that this combination of fields is unique for all appliances in all Skylar One systems.
- ***func***. A string representation of the appliance type. This will be used as the namespace for the relationships.

The query returns rows only for Data Collectors and Message Collectors.

Defining the Basic Properties for the Dynamic Application

To create the Dynamic Application that creates the identity, which will be aligned to Data Collectors and Message Collectors, perform the following steps:

> **NOTE**: For details on each field and its possible options, see the ***Dynamic Application Development*** chapter.

1. Go to the **Dynamic Applications Manager** page (System > Settings > Appliances).
2. Select the **[Actions]** button, and then select ***Create New Dynamic Application***. The **Dynamic Applications Create New Application** page appears.
3. Supply values in the following fields:
    - ***Application Name***. Enter "ScienceLogic Collector Identity"
    - ***Application Type***. Select *Database Configuration*.
    - ***Poll Frequency***. To see data as quickly as possible, select *Every 1 Minute* in this field.
4. This example does not have specific requirements for the other settings defined in this page. You can leave the remaining fields set to the default values. Select the **[Save]** button to save the Dynamic Application.

Defining the Collection Objects

To create the collection objects for the Dynamic Application that creates the identity:

NOTE: For more details on collection objects, see the ***Dynamic Application Development*** chapter.

1. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page appears.
2. Supply values in the following fields to create the collection object for the ***relationship identifier***:
    - ***Object Name***. Enter *ID*.
    - ***SQL Query***. Enter the database query for this example.
    - ***Group***. The namespace and identifier collection objects must be in the same group. Select *Group 1* in this field.
    - ***Usage Type***. This collection object will collect the uid values from the database query, which are the identifiers for the relationships. Select *Group Index* in this field.
    - ***Trended Column***. This collection object will collect the uid values from the database query. Enter "uid" in this field.
3. Select the **[Save]** button to save the new collection object.
4. Select the **[Reset]** button to clear the form fields.
5. Supply values in the following fields to create the collection object for the ***relationship namespace***:
    - ***Object Name***. Enter *Appliance Type*.
    - ***SQL Query***. Enter the database query for this example.
    - ***Group***. The namespace and identifier collection objects must be in the same group. Select *Group 1* in this field.
    - ***Usage Type***. This collection object will collect the appliance function values from the database query, which are the namespaces for the relationships. Select *Identity Namespace* in this field.
    - ***Trended Column***. This collection object will collect the appliance function values from the database query. Enter "func" in this field.
6. Select the **[Save]** button to save the new collection object.

Creating the Dynamic Application that Creates the Relationship

Both Dynamic Applications in this example collect the same data with only minor configuration differences. Therefore, to create the Dynamic Application that creates the relationship, which will be aligned to Database Servers, you can create a copy of the ScienceLogic Collector Information Dynamic Application as the starting point. To create the Dynamic Application that creates the relationship, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Settings > Appliances).
2. Select the wrench icon () for the "ScienceLogic Collector Identity" Dynamic Application.
3. Supply values in the following fields:
    - ***Application Name***. Enter "ScienceLogic Collector Information"
4. Select the **[Save As]** button to save the new Dynamic Application.

5. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page appears.
6. The difference between the collection objects in the identity-defining Dynamic Application (the previous Dynamic Application) and the relationship-defining Dynamic Application (this Dynamic Application) is the ***Usage Type*** setting for the namespace collection object. Select the wrench icon ( ) for the *Appliance Type* object.
7. In the ***Usage Type*** drop-down list, select *Relationship Namespace*.
8. Select the **[Save]** button to save the collection object.

Aligning the Dynamic Applications

To align the example Dynamic Applications:

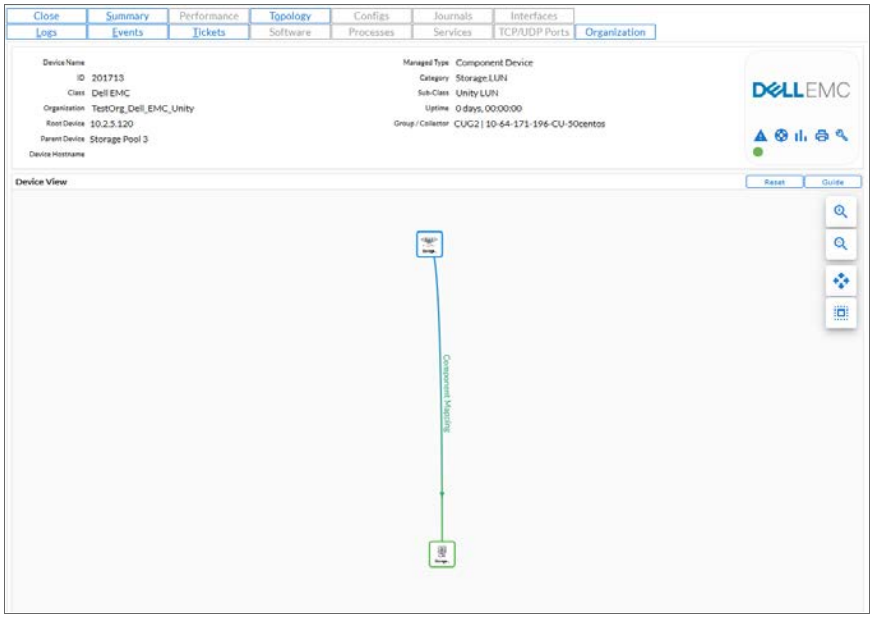
1. Discover the Database Server, Data Collectors, and Message Collectors in a Skylar One system.
2. Additionally, you must add a firewall rule to the monitored Data Collectors and Message Collectors to allow connections on port 7707 from the All-In-One Appliance or Data Collector that is performing collection.
3. Align the ***ScienceLogic Collector Information*** Dynamic Application with the Database Server and use the default Central Database credential. To do this:
4. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
5. Find the Database Server. Select its wrench icon ( ).
6. Select the **[Collections]** tab.
7. From the **[Actions]** menu, select *Add Dynamic Application*.
8. In the **Dynamic Application Alignment** modal page, select the following:
    - ***Dynamic Applications***. Select *ScienceLogic Collector Information*.
    - ***Credentials***. Select *EM7 Central Database*. If the monitored Database Server does not use the default username and password, you must edit the credential to use the new username and password.
9. Align the ***ScienceLogic Collector Identity*** Dynamic Application with the Data Collectors and Message Collectors and use the default Collector Database credential. To do this:
10. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
11. Find the Data Collector. Select its wrench icon ( ).
12. Select the **[Collections]** tab.
13. From the **[Actions]** menu, select *Add Dynamic Application*.
14. In the **Dynamic Application Alignment** modal page, select the following:
    - ***Dynamic Applications***. Select *ScienceLogic Collector Identity*.
    - ***Credentials***. Select *EM7 Central Database*. If the monitored Database Server does not use the default username and password, you must edit the credential to use the new username and password.

15. Repeat the bulleted steps for each Data Collector and Message Collector.

Viewing Dynamic Application Relationships

If the example Dynamic Applications are configured correctly and collection is successful, the relationships are displayed in the following places:

- In the **[Topology]** tab in the **Device Reports** panel for the Database Server, Data Collectors, and Message Collectors:



- In the **Device Relationships** page (Registry > Networks > Device Relationships):

Chapter

5

Internal Collection Dynamic Applications

Overview

This chapter describes how to use Internal Collection Dynamic Applications (ICDAs) to gather data from devices that do not support SNMP or to perform customized processing on collected SNMP data.

This chapter does not cover elements of Dynamic Application development that are common to all Dynamic Application types. You should be familiar with the common elements and concepts of Dynamic Applications before reading this chapter. For general information on planning, designing, using, and troubleshooting Dynamic Applications, see the chapter ***Dynamic Application Development***.

This chapter provides an overview of ICDAs. It contains the following topics:

This chapter covers the following topics:

Creating Internal Collection Dynamic Applications (ICDAs) *311*

What is an Internal Collection Dynamic Application (ICDA)?

Skylar One has traditionally used SNMP and an internal process called "internal collection" to collect the following data for each device:

- Availability and latency
- System description, system uptime, and system local
- Filesystem inventory and performance
- Interface inventory and performance

However, you might need to monitor device data that is partially available through SNMP, or not available at all through SNMP.

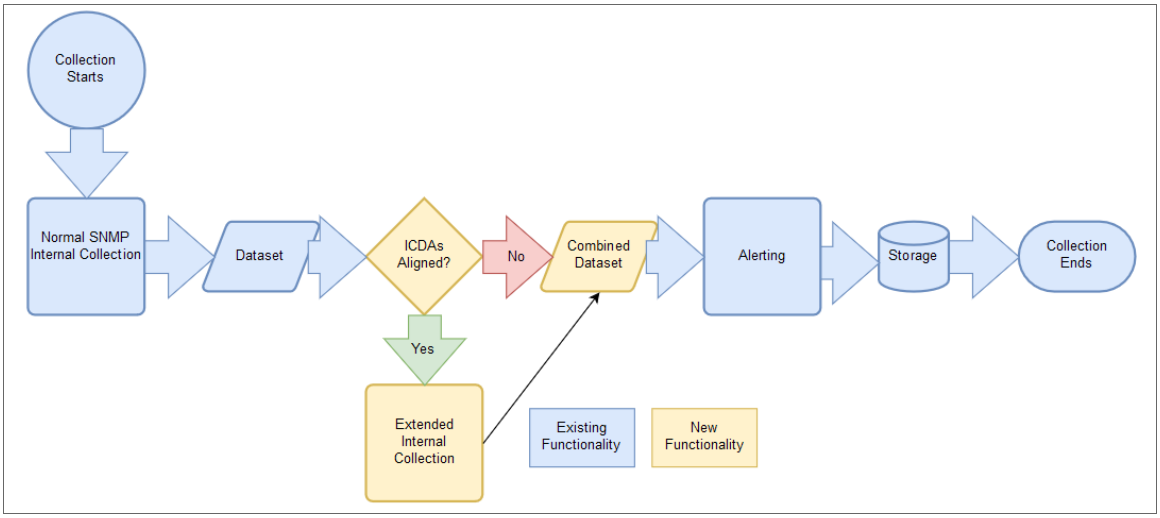
For devices that do *not* support SNMP, Skylar One provides an additional method to collect this data called ***Internal Collection Dynamic Applications (ICDAs).*** ICDAs combine the efficiency of internal collection with the flexibility of dynamic applications, ensuring uniform data across Skylar One.

How Do ICDAs Work?

Internal Collection Dynamic Applications align to devices in the same way as other Dynamic Applications. If the ICDA includes a discovery object, the ICDA automatically aligns with devices during discovery. You can also manually align ICDAs to devices and include ICDAs in device templates.

When Skylar One begins an internal collection, it checks to see if any ICDAs are aligned. If ICDAs are aligned, Skylar One performs an "Extended Internal Collection". The platform combines data it collected using ICDAs with data it collected using an internal SNMP process, and then the platform completes the collection process.

The following illustration highlights the ICDA functionality in the collection process:



NOTE: If Skylar One collects this data using the SNMP-based internal collection and also collects this data using ICDAs, the data from ICDAs takes precedence.

The ICDA data that Skylar One collects is stored in a database that is separate from the dynamic application database. ICDA results are stored where the corresponding internal collection would store its results.

ICDAs do not include presentation objects, thresholds, or alerts. Skylar One evaluates alerts and events in exactly the same way as for standard internal collection.

Types of ICDAs

There are two types of Internal Collection Dynamic Applications:

- ***Internal Collection Inventory***. These ICDAs collect configuration data about filesystems, such as storage size, filesystem type, and storage used. These ICDAs also collect configuration data about interfaces, such as physical address, operational status, and IP addresses. In Skylar One, the available Data Models for this Application Type are Filesystem Inventory and Interface Inventory. Use these ICDAs to monitor inventory processes, like **process_collect.py**, which runs every 2 hours.
- ***Internal Collection Performance***. Collects data about availability and latency, device information (system description, system uptime, system local), filesystem performance, and interface performance. In Skylar One, the available Data Models for this Application Type are Availability, Filesystem Performance, SNMP Detail (includes Uptime, Description, and Locale), and Interface Performance. Use these ICDAs to monitor performance collection processes, like **process_check.py**, which runs every 5 minutes.

NOTE: The names for the ICDA Data Models follow the existing naming conventions used by the internal collection processes.

Snippets and Execution Environments

Internal Collection Dynamic Applications collect data by executing one or more blocks of Python code, called ***snippets***. Skylar One passes credential and other configuration information to the snippet, and at the end of execution, the snippet must pass collected data back to Skylar One. The Dynamic Application developer defines snippet code that collects data and processes data.

One way in which Dynamic Applications that use snippets are unique from most other types of Dynamic Applications is that they can be aligned with a specific execution environment.

In the Skylar One user interface, an ***execution environment*** is a list of one or more ScienceLogic libraries. When a Dynamic Application snippet, Run Book Automation snippet, or credential test is executed, the execution environment defines a virtual Python environment that is deployed on-demand and includes all of the ScienceLogic libraries aligned with it for use during snippet execution.

When you create an Internal Collection Dynamic Application, you must select an execution environment to align with that Dynamic Application. All of the snippets contained in the Dynamic Application will then use that execution environment whenever the Dynamic Application runs. If you do not specify an execution environment, Skylar One will align the Dynamic Application with the default *System* environment.

You can create and manage execution environments on the **Environment Manager** page (System > Customize > ScienceLogic Libraries > Actions > Execution Environments). For more information about creating and managing execution environments, see the ***ScienceLogic Libraries and Execution Environments*** manual.

Data Collected by ICDAs

This section lists the collection objects included in each type of Data Model.

NOTE: For each Data Model, this list also describes how to index the collection objects so they can be merged with values from the internal, SNMP-based collection process. Ensure that you follow the indexing format. Improper indexing could result in duplicated entities if the two types of data cannot be merged.

The ICDA data takes precedence over any data collected during SNMP-based internal collection.

Internal Collection Inventory Application Types

Use these ICDAs to monitor inventory processes, like **process_collect.py**, which runs every 2 hours.

Filesystem Inventory

Index: ***Filesystem Name***, such as `/dev/sda1`.

Objects:

- ***Storage Size***. Measured in storage units.
- ***Filesystem Type***. Type of filesystem, such as `ext4`.
- ***Storage Used***. Measured in storage units.
- ***Storage Units***. Measured in bytes per storage units.
- ***Media Type***. Type of media, such as CD, hard drive, or flash drive.

Interface Inventory

Index: SNMP interface index or user-defined unique index if only ICDA collects the data for the interface.

Objects:

- ***Name***. String.
- ***Alias***. String.
- ***Descr***. String.
- ***Type***. Interface type as defined in IF-MIB.
- **Phys Address**. String for the MAC Address.
- ***Speed***. Measured in bps.
- ***High Speed***. Measured in Megabits per second, or Mbps.
- ***Admin Status***. Integer: up [1], down [2], testing[3].
- ***Operational Status***. Integer: up [1], down [2], testing[3], unknown [4], dormant [5], not present [6], lower layer down [7].
- ***Connector Present***. Integer.
- ***64-bit Support***. True or false.
- ***Blade***. Blade number.
- ***Port***. Port number.

- ***IP Addresses***. IP Address and Subnet, such as [(1.2.3.4, 255.255.255.0), (2.3.4.5, 255.255.255.128)].

Internal Collection Performance Application Types

Use these ICDAs to monitor performance collection processes, like **process_check.py**, which runs every five minutes.

Availability

Index: None (this impacts the entire device, so there can only be only value per index).

Objects:

- ***Availability***. True or false, depending on whether the device is available.
- ***Latency***. Measures uptime in 10 millisecond increments.

SNMP Detail

Index: None (this impacts the entire device, so there can only be only value per index).

Objects:

- ***System Description***. String.
- ***System Uptime***. Measured in 10-millisecond increments.
- ***System Locale***. String.

Filesystem Performance

Index: ***Filesystem Name***, such as `/dev/sda1`.

Objects:

- ***Storage Used***. Measured in storage units.
- ***Storage Units***. Measured in bytes per storage units.
- ***Percent Used***. Measured in percentage of storage used.

Interface Performance

Index: SNMP interface index or user-defined unique index if only ICDA collects data for the interface.

Objects:

- ***Phys Address***. String for the MAC Address.
- ***Admin Status***. Integer: up [1], down [2], testing[3].
- ***Operational Status***. Integer: up [1], down [2], testing[3], unknown [4], dormant [5], not present [6], lower layer down [7].
- ***ifInOctets***. Interface Inbound Traffic in octets.
- ***ifOutOctets***. Interface Outbound Traffic in octets.
- ***ifInDiscards***. Interface Inbound Discards.
- ***ifOutDiscards***. Interface Outbound Discards.
- ***ifOutErrors***. Interface Outbound Errors.
- ***ifInErrors***. Interface Inbound Errors.

Port Performance

Index: Data pair of port number and port protocol: TCP (0), UDP (1)

Objects:

- ***State***. Port state, up or down.

Process Inventory

> **NOTE**: ICDA process data does not merge with agent-collected data.

Index: Process Pid number

Objects:

- ***Name***. String for the Process Name.
- ***Arguments***. String for the Process Arguments.
- ***Path***. String for the Process Path.
- ***User***. String for the Process User.
- ***CPU Time***. Integer.
- ***Memory Use***. Integer.
- ***State***. Integer.

Process Performance

> **NOTE**: ICDA process data does not merge with agent-collected data.

Process Performance

Objects:

- ***Name***. String for the Process Name.
- ***Arguments***. String for the Process Arguments.
- ***Path***. String for the Process Path.
- ***User***. String for the Process User.
- ***CPU Time***. Integer.
- ***Memory Use***. Integer.
- ***State***. Integer.

Service Inventory

Index: Service Name

Objects:

- ***Start Mode***. String: Auto, Manual, Disabled, Anything Else.
- ***State***. Integer: 0 (running), 1 (not running).

Service Performance

Index: Service Name

Objects:

- ***Start Mode***. String: Auto, Manual, Disabled, Anything Else.
- ***State***. Integer: 0 (running), 1 (not running).

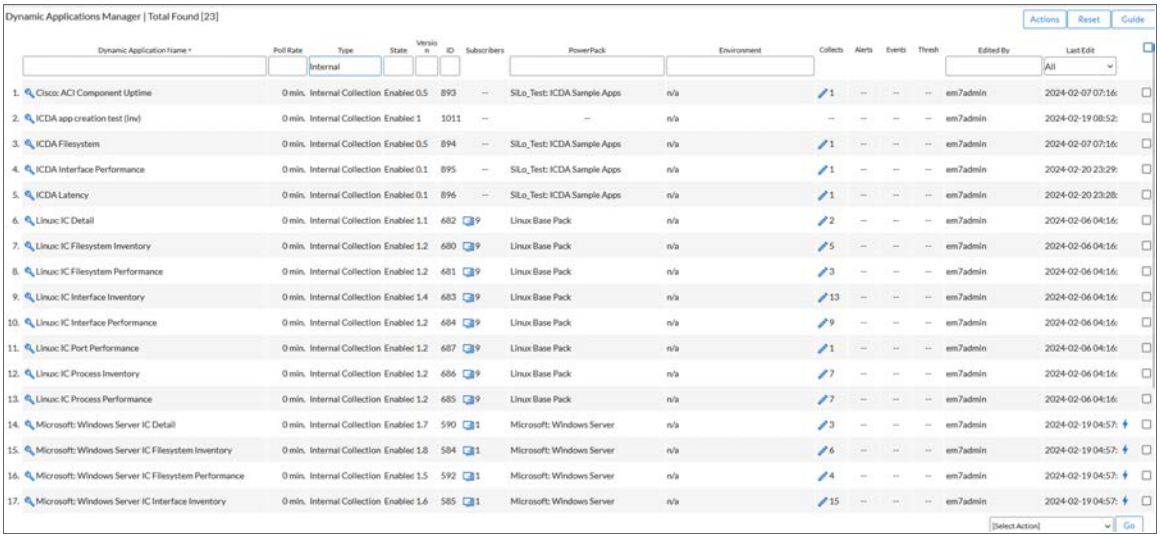
Creating Internal Collection Dynamic Applications (ICDAs)

Internal Collection Dynamic Applications (ICDAs) enable you to gather and monitor data from devices that do not support SNMP or other standard protocols. ICDAs are particularly useful for collecting both configuration data (such as filesystem and interface inventory) and performance metrics including availability, latency, device information, and interface statistics.

This section covers the elements specific to building ICDAs, the supported data types and archetype options, and the workflow for creating and managing these applications within Skylar One.

Viewing the List of ICDAs

You can view all Internal Collection Dynamic Applications on the Dynamic Applications Manager page (System > Manage > Dynamic Applications) by typing "Internal" in the ***Type*** filter-while-you-type field:



TIP: To ensure the consistency of your data, do not use an ICDA to collect data on a device that was previously monitored by another type of Dynamic Application.

Creating an ICDA

You can create custom Internal Collection Dynamic Applications (ICDAs) to gather data on devices that do not support SNMP. Currently, ICDAs support only the ***snippet*** protocol.

NOTE: If you create an ICDA with a Data Model of *Interface Inventory* and the ICDA includes a discovery Collection Object, the ICDA aligns with the device after the platform discovery runs. As a result, Skylar One does not discover interfaces until either the nightly discovery runs or the user re-discovers the device manually by running the discovery session again, creating a new discovery session, or initiating an ad hoc discovery.

To create an ICDA:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Click the **[Actions]** button and select *Create New Dynamic Application*. The **Dynamic Applications Create New Application** page appears.
3. Complete the following fields:

- ***Application Name***. Type a name for this ICDA.
- ***Application Type***. Your options for ICDAs include *Internal Collection Inventory* or *Internal Collection Performance*.

4. Click the **[Save]** button. The **[Properties]** tab updates to show the relevant fields for the ICDA.
5. In the ***Data Model*** drop-down list, select the type of data you want to collect with this ICDA. The values in the drop-down are dependent on the value in the ***Application Type*** field. The value in the ***Data Model*** field list determines the available options in the ***Object Link*** drop-down list on the **[Collections]** tab. For example, if you select *Availability* for the Data Model, then you can choose only *Availability* or *Latency*in the ***Object Link*** drop-down list.
    - If you selected *Internal Collection Inventory* as the ***Application Type***, the ***Data Model*** drop-down includes:
        - ***Filesystem Inventory***. Monitors Storage Size, Filesystem Type, Storage Used, Storage Units, or Media Type.
        - ***Interface Inventory***. Monitors Name, Alias, Description, Type, Physical Address, Speed, High Speed, Admin Status, Operational Status, Connector Present, 64-bit Support, Blade, Port, or IP Address.
    - If you selected *Internal Collection Performance* as the ***Application Type***, the ***Data Model*** drop-down includes:
        - ***Availability***. Monitors Availability or Latency.
        - ***Filesystem Performance***. Monitors Storage Used, Storage Units, or Percentage Used.
        - ***SNMP Detail***. Monitors System Description, System Uptime, or System Locale.
        - ***Interface Performance***. Monitors Octets In, Octets Out, Errors In, Errors Out, Discards In, Discards Out, Physical Address, Admin Status, or Operational Status.
6. Update the remaining fields as needed, and then click the **[Save]** button.
7. Next, create a container for your ICDA snippet code.

Creating a Container for the ICDA Snippet

In Snippet Dynamic Applications, each collection object must be associated with a snippet. Therefore, in ICDAs you must create the container for the snippet code before creating the collection objects for this ICDA.

To create a container for the snippet code:

1. Click the **[Snippets]** tab.
2. Complete the following fields:
    - ***Snippet Name***. The name of the snippet.
    - ***Active State***. Specifies whether the snippet should be executed by Skylar One when performing collection for the ICDA.
    - ***Required***. Specifies whether this snippet is required for successful collection of all other snippet requests. Choices are:

- *Required - Stop Collection*. If this snippet request fails, the platform will not attempt to execute any other snippet requests in this ICDA. Dynamic Applications that consume the cache of this ICDA will halt collection.
- *Not Required - Continue Collection*. If this snippet request fails, the platform will continue executing all remaining snippet requests in this ICDA. Dynamic Applications that consume the cache of this ICDA will continue collection.

- ***Snippet Code***. The python code that will be executed when Skylar One performs collection for this ICDA. You will add the snippet code later.

3. Click the **[Save]** button.
4. Repeat this process for each collection you want to use in conjunction with a snippet.
5. Next, create the collection objects for your ICDA.

Creating the Collection Objects

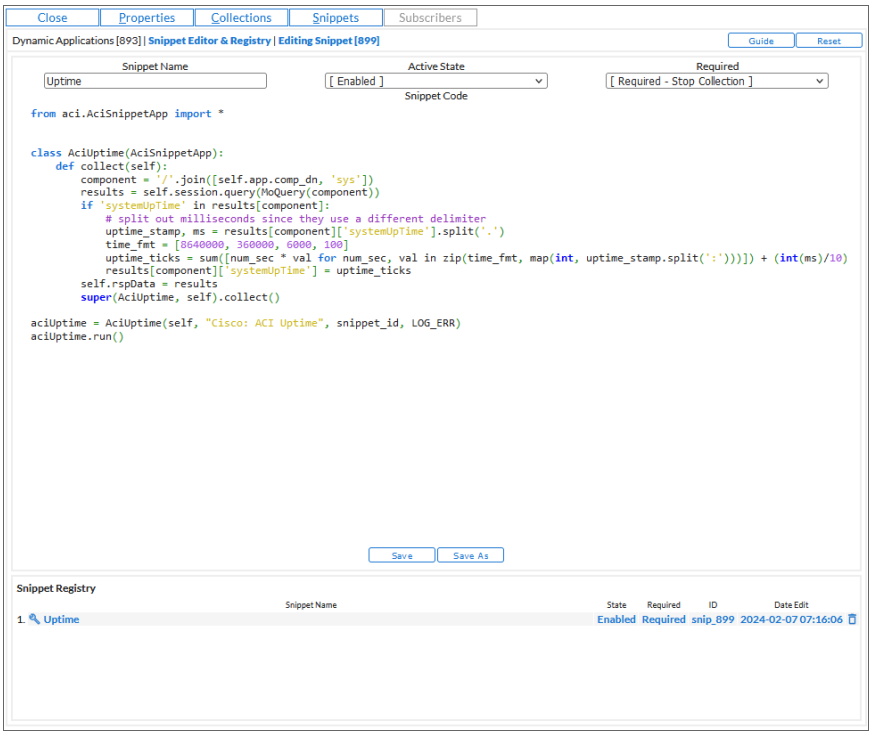
To create the collection object or objects for this ICDA:

1. Click the **[Collections]** tab.
2. Complete the following fields to create the collection object:
   - ***Object Name***. Type the name of the collection object.
   - ***Snippet Arguments***. Specify any arguments you want to pass to the snippet.
   - ***Class Type***. Depending on the type of collection object you want to create, select *[33] ICDA Object* or *[100] Discovery*.
   - ***String Type***. Define the string type for this snippet. Select *Standard* or *Hex Code*.
   - ***Object Link***. Select the specific type of data you want to collect with this ICDA. The options in this drop-down list are based on your selection for the ***Data Model*** field on the **[Properties]** tab.
   - ***Snippet***. Select the name of the snippet you created in the previous process.
3. Update the remaining fields as needed, and then click the **[Save]** button.
4. Repeat these steps to create additional collection objects you want to use in conjunction with a snippet.
5. Next, add the snippet code for your new ICDA.

Adding the Snippet Code

To add the snippet code to a new ICDA:

1. Click the **[Snippets]** tab.
2. In the **Snippet Registry** pane, select the relevant snippet and click its wrench icon (). Information about that snippet appears in the top pane:



3. In the ***Snippet Code*** field, type or paste your python code for the collection object.
4. Click the **[Save]** button and close the modal page for the ICDA.

Snippet Examples for ICDAs

The following snippets were created by ScienceLogic to use as sample snippets for ICDAs.

Availability

```
from silo_builtin_caching import get_cached_request_result

Windows_Server_Uptime_REQ_GUID = '32CA377A59190A6AB83DC5EDB05CA7E7'

result = {

    'if': get_cached_request_result(self, Windows_Server_Uptime_REQ_
    GUID)

}

print "*****************************************************"

print result
```

```
for grp_id, grp in self.oids.iteritems():
  for obj_id, obj in grp.iteritems():
    try:
      (req, oid) = obj['oid'].split('.')
      result_list = result[req].get(oid, {}).values()
    except ValueError:
      self.logger.ui_debug('Skipping OID %d' % obj_id)
      continue
    if len(result_list) >= 1 and float(result_list[0]) > 0.0:
      obj['result'] = [(0, True)]
    #obj['result'] = [(0, True if len(result_list) >= 1 and float
    (result_list[0]) > 0.0 else False)]
    print obj
```

Discovery

```
from sl_credentials.constant import CRED_SSH
import paramiko
if self.cred_details['cred_type'] == CRED_SSH:
  host = self.cred_details['cred_host']
  user = self.cred_details['cred_user']
  passwd = self.cred_details['cred_pwd']
  timeout = self.cred_details['cred_timeout'] / 1000
  port = self.cred_details['cred_port']
  ssh = paramiko.SSHClient()
  ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
```

```
    ssh.connect(host, username=user, password=passwd, port=port,
    timeout=timeout)

    stdin, stdout, stderr = ssh.exec_command("uname")

    os = stdout.readlines()

    if os[0].startswith("Linux"):

      result_handler['disc'] = True

    ssh.close()

else:

    self.logger.debug("ICDA Filesystem | DID %s: Incorrect credential type
    aligned, must be SSH" % self.did)
```

Filesystem Inventory

```
from silo_builtin_caching import get_cached_request_result

import re

#Enum to translate drive type to string

FS_ENUM_TYPES={

    0: 'Unknown',

    1: 'No Root Directory',

    2: 'Removable Disk',

    3: 'Local Disk',

    4: 'Network Drive',

    5: 'Compact Disc',

    6: 'RAM Disk'

}

result = get_cached_request_result(self, WINDOWS_SERVER_DISK_
CONFIGURATION_REQ_GUID)
```

```
indexes = set()

units_object = {}

for grp_id, grp in self.oids.iteritems():

    for obj_id, obj in grp.iteritems():

    oid = obj['oid']

    result_list = result.get(oid, {}).items()

    #Keep track of all the indexes (which are the filesystem names)

    #Rewrite the results list to clean up filesystem names

    for (i, (k, v)) in enumerate(result_list):

      #An empty string index will attempt to use the label first,

      #and fall back to " " if there is no label

      if k == '':

        k = result.get('label',{}).get('',' ')

        result_list[i] = (k, v)

      #Add a backslash to drive letters to match the output from SNMP

      elif re.match('[A-Z]:$', k):

        k += '\\'

        result_list[i] = (k, v)

      indexes.add(k)

    #If this is the 'units' object, set it aside for later processing

    if obj['oid'] == 'units':

      units_object = obj

      continue

    #Translate the drivetype enum to a string
```

```
    if obj['oid'] == 'drivetype':
      result_list = [(index, FS_ENUM_TYPES.get(int(value), value)) for
      index, value in result_list]
    obj['result'] = result_list
#Do the processing for the units object (all units are 1 - bytes)
units_object['result'] = [(i, 1) for i in indexes]
```

Filesystem Type

```
from sl_credentials.constant import CRED_SSH
import paramiko
if self.cred_details['cred_type'] == CRED_SSH:
  host = self.cred_details['cred_host']
  user = self.cred_details['cred_user']
  passwd = self.cred_details['cred_pwd']
  timeout = self.cred_details['cred_timeout'] / 1000
  port = self.cred_details['cred_port']
  ssh = paramiko.SSHClient()
  ssh.set_missing_host_key_policy(paramiko.AutoAddPolicy())
  ssh.connect(host, username=user, password=passwd, port=port,
  timeout=timeout)
  stdin, stdout, stderr = ssh.exec_command("df --output=target,fstype",
  timeout=timeout)
  results = []
  lines = stdout.readlines()[1:]
  for line in lines:
    fs_name, fs_type = map(lambda line: line.strip(), line.split(' ', 1))
```

```
        results.append((fs_name, fs_type))

    result_handler['fs_type'] = results

    ssh.close()

else:

    self.logger.debug("ICDA Filesystem | DID %s: Incorrect credential type
    aligned, must be SSH" % self.did)
```

Interface Octets

```
from sl_credentials.snmp import snmph_from_cred_array

from sl_snmp.silo_snmp import SnmpError

OID_ifInOctets = '.1.3.6.1.2.1.2.2.1.10'

results = []

try:

    snmp_h = snmph_from_cred_array(self.cred_details, self.ip)

    octets_in_walk = snmp_h.walk(OID_ifInOctets)

    for oid, value in octets_in_walk:

        if_index = oid.split('.')[-1]

        # offset to make this differentiable from the value returned by normal
        internal collection

        if_in_octets = int(value) + 100000

        results.append((if_index, if_in_octets))

except SnmpError, err:

    self.logger.ui_debug("An SNMP error occurred during interface
    performance collection")

finally:

    result_handler['ifInOctets'] = results
```

Latency

```
from silo_collect.tools import latency_ping

from silo_common.network import ip

ip_obj = ip(self.ip)

# add large offset to make ICDA data stand out

result_handler['latency'] = [('0', latency_ping(ip_obj,
logger=self.logger) + 5000)]
```

Chapter 6

XML, SOAP, and XSLT Dynamic Application

Overview

This chapter describes how to create Dynamic Applications that collect data by parsing collection objects from XML documents.

This chapter does not cover elements of Dynamic Application development that are common to all Dynamic Application types; you should be familiar with the common elements and concepts of Dynamic Applications. For details on the common elements of Dynamic Applications, see the ***Dynamic Application Development*** chapter. XML, SOAP, and XSLT Dynamic Applications use XML markup language.

You should be familiar with the XML markup language before developing XML, SOAP, or XSLT Dynamic Applications. The requests in XSLT Dynamic Applications are written using XSL transformations. You must be familiar with the structure, syntax, and elements of XSL transformations before developing XSLT Dynamic Applications.

This chapter covers the following topics:

SOAP and XSLT Requests 324

Collection Objects 331

Creating an XML Dynamic Application 335

Dynamic Component Mapping & Caching with XSLT 342

XML, SOAP, and XSLT Protocols

This section describes how to create Dynamic Applications that collect data by parsing collection objects from XML documents. There are three Dynamic Application protocols that parse collection objects from XML documents:

- ***XML***. For this Dynamic Application protocol, Skylar One performs an HTTP GET request for the XML document specified in the associated credential. The collection objects in an XML Dynamic Application specify how Skylar One should parse values from the XML document.
- ***SOAP***. For this Dynamic Application protocol, Skylar One performs one or more HTTP POST requests on the URL specified in the associated credential. The collection objects in a SOAP Dynamic Application specify which response returns values for the collection object and how Skylar One should parse values from the returned XML document. The requests in a SOAP Dynamic Application are performed in a specified order. You can substitute a collected value from a request in the POST content of a future request. The collection object that is substituted must collect a single value, not a list of values.
- ***XSLT***. For this Dynamic Application protocol, Skylar One performs one or more HTTP POST or GET requests on the URL specified in the associated credential. The collection objects in an XSLT Dynamic Application specify which response returns values for the collection object and how Skylar One should parse values from the returned XML document. The requests in a XSLT Dynamic Application are performed in a specified order. If the credential specifies a POST method, the POST content is generated by performing an XSL transformation on an XML document that contains all the values collected using the preceding requests. Therefore, unlike a SOAP Dynamic Application, you can use the values collected for a collection object that returns a list of values to constrict a request. All requests must specify an XSL transformation that is applied to the response returned for that request. The result of the XSL transformation that is performed on the response must be in a specific format from which Skylar One parses the collection objects for that request.

What is XML?

XML stands for "eXtensive Markup Language". XML was designed to transport and store data, with focus on what data is. By itself, XML does not actually do anything. XML was created to structure, store, and transport information. XML stores data in plain-text format. This allows XML data to be exchanged between disparate hardware and software.

XML data is frequently sent using HTTP requests. Users can send Skylar One data by writing client code that makes HTTP calls and sends the XML data.

When XML data is received, the receiving computer must include programs to receive, parse, and respond to XML requests. Most browsers include built-in parsers for XML.

To learn more about XML, see http://www.w3schools.com/xml/default.asp

Elements of XML, SOAP, and XSLT Dynamic Applications

For comprehensive information on elements that apply to XML, SOAP, and XSLT Dynamic Applications, see the **Dynamic Application Development** chapter.

SOAP and XSLT Requests

SOAP and XSLT Dynamic Applications must include one or more ***requests*** that define how Skylar One should request data from a device. Each request specifies an operation that will gather a response from the device.

Each collection object in a SOAP or XSLT Dynamic Application is associated with a request. The collection object specifies how to parse a data point from the response. A single request can be used to populate multiple collection objects.

For SOAP Dynamic Applications, a request specifies the POST content that Skylar One uses to perform the request. The associated collection objects are parsed from the response.

For XSLT Dynamic Applications, a request specifies:

- An XSL transformation that Skylar One applies to an XML document that contains the collected values from the previous requests. The result of this transformation is used as the POST content that Skylar One uses to perform the request.
- A second XSL transformation that Skylar One applies to the response from the device. The result of this transformation must be in a specific format from which Skylar One parses the associated collection objects.

Viewing the Requests in a Dynamic Application

To view the existing requests in a SOAP or XSLT Dynamic Application:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Find the Dynamic Application you want to view the requests for. Select its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Requests]** tab. The **Dynamic Applications Request Editor and Registry** page is displayed. The **Request Registry** pane at the bottom of the page displays the following information about each request:
    - ***Request Name***. The name of the request.
    - ***State***. The state of the request. Possible values are:
        - *Enabled*. Skylar One will perform this request during collection for this Dynamic Application.
        - *Disabled*. Skylar One will not perform this request during collection for this Dynamic Application.
    - ***Sequence***. The order in which the requests in this Dynamic Application will be performed.
    - ***ID***. The unique ID assigned to the request by Skylar One. The unique ID will always start with "req_".
    - ***Date Edit***. The last time a user edited this request.

Creating a Request

To define a request for a SOAP or XSLT Dynamic Application:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Locate the Dynamic Application for which you want to define a request. Select its wrench icon (🔧). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Requests]** tab. The **Dynamic Applications Request Editor and Registry** page is displayed.
4. Supply values in the following fields:
    - ***Request Name***. Enter a name for the request.
    - ***Execution Sequence***. Specifies the order in which Skylar One should perform the requests in this Dynamic Application Select a numeric value in this field. The request with the lowest ***Execution Sequence*** value will be performed first during collection, then the request with the next lowest ***Execution Sequence*** value, etc.
    - ***Active State***. Specifies whether Skylar One should perform this request during collection for this Dynamic Application. Choices are:
        - *Enabled*. Skylar One will perform this request during collection for this Dynamic Application.
        - *Disabled*. Skylar One will not perform this request during collection for this Dynamic Application.

> **NOTE:** If a collection object that has a ***Class Type*** of *[109] SOAP/XSLT Session ID* is associated with a request, the request is executed only under certain conditions. For more information about Session ID collection objects, see the *Collection Objects* section.

5. Depending on the type and configuration of the Dynamic Application, define the request code:
    - If you are creating a request for a SOAP Dynamic Application, supply code in the ***SOAP Request Code***. This will be the POST content that Skylar One will use for this request. You can use substitution characters in the request code.
    - If you are creating a request for an XSLT Dynamic Application that does not consume cached responses, supply code in the ***XSLT Request Code*** and ***XSLT Parser Code*** fields. For a full description of these fields, see the *Defining XSLT Request Code* and *Defining XSLT Parser Code* sections.
    - If you are creating a request for an XSLT Dynamic Application that consumes cached responses, supply a value in the ***Cached XSLT Request*** field and supply code in the ***XSLT Parser Code*** field. For a full description of these fields, see the *Defining XSLT Request Code* and *Defining XSLT Parser Code* sections.
6. Select the **[Save]** button.

Defining XSLT Request Code

If your Dynamic Application has *No Caching* or *Cache Results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor** page, you must supply an XSL transformation in the ***XSLT Request Code*** field.

If your Dynamic Application has *Consume Cached Results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor** page, you must select a cached XSLT request in the ***Cached XSLT Request*** field.

The XSLT Request Code Field

When Skylar One performs collection for an XSLT Dynamic Application, Skylar One generates the POST content for each request by applying the XSL transformation you supply in the ***XSLT Request Code*** field to an XML document. The XML document contains the values collected for collection objects that have already been collected by the Dynamic Application during this poll period.

> **NOTE:** If your Dynamic Application will be used with a credential that specifies an HTTP GET request, the result of the transformation specified in this field will not be used. However, Skylar One will still apply the supplied XSL transformation to the XML document that contains the collected values. You must always supply a valid transformation in the ***XSLT Request Code*** field.

You must supply a valid XSL transformation in the ***XSLT Request Code*** field. Skylar One will apply the specified XSL transformation to an XML document that has the following structure:

```
<objects>
  <o_XXXX>
    <i_Y></i_Y>
    <i_Z></i_Z>
    .
    .
  </o_XXXX>
  <o_AAAA>
    <i_B></i_B>
    <i_C></i_C>
```

```
        .

        .

    </o_AAAA>

</objects>
```

Where:

- o_XXXX and o_AAAA are collection object ID's
- i_Y and i_Z are indexes associated with values in the list of values collected for collection object o_XXXX.
- i_B and i_C are indexes associated with values in the list of values collected for collection object o_AAAA.

The XML file of objects contains only objects collected from other XSLT requests in the same polling period on the same device. For example, the XML file provided to the XSLT request with an ***Execution Sequence*** of "2" would contain all the objects collected in the same polling period from the XSLT request with an ***Execution Sequence*** of "0" and all the objects collected in the same polling period from the XSLT request with an ***Execution Sequence*** of "1". The XML file provided to the XSLT request with an ***Execution Sequence*** of "0" will contain no collected objects.

You can use the substitution characters described in the *Substitution Characters* section in your ***XSLT Request Code***.

The Cached XSLT Request Field

The ***Cached XSLT Request*** displays all XSLT requests defined in Dynamic Applications that have *Cache Results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor** page.

To generate the XML document that contains the collection objects associated with this XSLT request, Skylar One transforms the currently cached response for the XSLT request you select in this field using the ***XSLT Parser Code*** defined for this XSLT request.

Defining XSLT Parser Code

The ***XSLT Parser Code*** for an XSLT request is XSL transformation that will be used by Skylar One to transform the response of the SOAP request. Skylar One parses the collection objects associated with the request from the transformed response.

In the ***XSLT Parser Code*** field, you must supply an XSL transformation that transforms the response from the device in to an XML document with the following structure, where column_1 and column_2 are element names.:

```
<response>

    <row>

        <column_1></column_1>
```

```
    <column_2></column_2>
    .
    .
  </row>
  <row>
    <column_1></column_1>
    <column_2></column_2>
    .
    .
  </row>
</response>
```

The element names are used by Skylar One to parse collection object values from the XML document.

Each "<row>" block specifies an entry in the list of collection objects. For example, if the transformed response looks like this:

```
<response>
  <row>
    <label>Label 1</label>
    <value>10</label>
  </row>
  <row>
    <label>Label 2</label>
    <value>20</label>
  </row>
  <row>
    <label>Label 3</label>
```

```
        <value>30</label>
    </row>
</response>
```

A list of three values (Label 1, Label 2, and Label 3) will be collected for the collection object associated with the *label* element.

A list of three values (10, 20, and 30) will be collected for the collection object associated with the *value* element.

You can use the substitution characters described in the *Substitution Characters* section in your ***XSLT Parser Code***.

Substitution Characters

You can include substitution characters in ***SOAP Request Code***, ***XSLT Request Code***, and ***XSLT Parser Code***. Depending on the type of substitution character that you use, Skylar One will replace the substitution character with:

- The value of a collected object.
- A value specified in the associated credential.
- A property of the component device for which collection is being performed.

NOTE: Substitution characters for collected objects can be used only in ***SOAP Request Code***.

Collection Object Substitution Characters

In ***SOAP Request Code***, you can include the ID of one or more collection objects in the same Dynamic Application. All collection object IDs begin with "o_" (lowercase "oh", underscore), e.g. "o_123". To use a collection object ID in ***SOAP Request Code***, the collection object must:

- Be associated with a request that has a lower ***Execution Sequence*** setting than the request that uses the collection object in the ***SOAP Request Code***.
- Collect a single value, not a list of values.

Substitution Characters from Credentials

SOAP/XML credentials can specify up to four substitution characters that can be used in ***SOAP Request Code***, ***XSLT Request Code***, or ***XSLT Parser Code***.

When you use substitution characters from a credential in ***SOAP Request Code***, ***XSLT Request Code***, or ***XSLT Parser Code***, you must ensure that the credential(s) that are used with the Dynamic Application define values for these substitution characters:

- *%1*. Skylar One will supply the value specified in the ***Embed Value [%1]*** field for the credential associated with the Dynamic Application.
- *%2*. Skylar One will supply the value specified in the ***Embed Value [%2]*** field for the credential associated with the Dynamic Application.
- *%3*. Skylar One will supply the value specified in the ***Embed Value [%3]*** field for the credential associated with the Dynamic Application.
- *%4*. Skylar One will supply the value specified in the ***Embed Value [%4]*** field for the credential associated with the Dynamic Application.

Substitution Characters from Component Devices

In SOAP and XSLT Dynamic Applications that collect data from component devices, you can use the following variables in the ***SOAP Request Code***, ***XSLT Request Code***, or ***XSLT Parser Code***:

- *%C*. Distinguished name. Skylar One will supply the distinguished name of the component device Skylar One is currently collecting data from.
- *%N*. Device name. Skylar One will supply the device name of the component device Skylar One is currently collecting data from.
- *%U*. Unique identifier. Skylar One will supply the unique identifier of the component device Skylar One is currently collecting data from.

For more information about Dynamic Component Mapping, see the ***Dynamic Application Development*** chapter.

Editing a Request

To edit a request in a SOAP or XSLT Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Find the Dynamic Application you want to edit. Select its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Requests]** tab. The **Dynamic Applications Request Editor and Registry** page is displayed.
4. In the **Request Registry** pane at the bottom of the page, locate the request you want to edit. Select the wrench icon () for the request you want to edit.
5. The fields in the top pane will be populated with the saved values for the selected request. Edit the values in one or more fields. For a description of each field, see the *Creating a Request* section.
6. Select the **[Save]** button to save your changes to the request. Select the **[Save As]** button to save the changes as a new request.

Deleting a Request

To delete a request from a SOAP or XSLT Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Find the Dynamic Application you want to edit. Select its wrench icon (). The **Dynamic Applications Properties Editor** page is displayed.
3. Select the **[Requests]** tab. The **Dynamic Applications Request Editor and Registry** page is displayed.
4. In the **Request Registry** pane at the bottom of the page, locate the request you want to delete. Select its delete icon (). The request is deleted from Skylar One. You must edit all collection objects that were associated with the request to associate those collection objects with other requests in the Dynamic Application.

Collection Objects

This section describes how to define collection objects for XML, SOAP, and XSLT Dynamic Applications.

This section describes only the fields specific to Defining a Collection Object for a XML, SOAP, and XSLT Dynamic Application. All the remaining fields, for both performance and configuration archetypes, are described in detail in the ***Dynamic Application Development*** chapter. All other elements of XML, SOAP, and XSLT Collection Objects, such as presentation objects and alerts, behave in the same manner as other Dynamic Application types.

For details on other parts of XML, SOAP, and XSLT Dynamic Applications, see the ***Dynamic Application Development*** chapter.

Protocol-Specific Fields for Collection Objects

Similar to other Dynamic Application types, XML, SOAP, and XSLT Dynamic Applications contain collection objects. Collection objects for XML, SOAP, and XSLT Dynamic Applications have characteristics common to collection objects for other Dynamic Application types, such as naming, data typing, and grouping. This section describes the collection object fields that are specific to XML, SOAP, and XSLT Dynamic Applications.

XML Dynamic Applications

Collection objects for XML Dynamic Applications have the following unique fields:

- ***XML Tags***. Specifies how Skylar One should parse the collection object from the XML document that is requested for this Dynamic Application. For a full description of this field, see the *Specifying XML Tags and SOAP Tags* section of this chapter.
- ***XML Document***. Appears only for Dynamic Applications that have *Consume cached results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor** page. This field specifies the Dynamic Application that retrieves the XML document that this collection object will be collected from. The list of Dynamic Applications in this field is limited to XML Dynamic Applications that have *Cache results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor** page.

SOAP Dynamic Applications

Collection objects for SOAP Dynamic Applications have the following unique fields:

- ***SOAP Request***. Select the SOAP request that returns a value for this collection object.
    - For Dynamic Applications that have *No Caching* or *Cache Results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor** page, this drop-down list contains all SOAP requests defined in the same Dynamic Application.
    - For Dynamic Applications that have *Consume cached results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor**, this drop-down list contains requests defined in SOAP Dynamic Applications that have *Cache results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor** page.
- ***SOAP Tags***. Specifies how Skylar One should parse the collection object from the XML document that is collected using the request specified in the ***SOAP Request*** field. For a full description of this field, see the *Specifying XML Tags and SOAP Tags* section.

XSLT Dynamic Applications

Collection objects for XSLT Dynamic Applications have the following unique elements:

- ***XSLT Request***. Select the XSLT request that returns a value for this collection object.
    - For Dynamic Applications that have *No Caching* or *Cache Results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor** page, this drop-down list contains all XSLT requests defined in the same Dynamic Application.
    - For Dynamic Applications that have *Consume cached results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor**, this drop-down list contains requests defined in XSLT Dynamic Applications that have *Cache results* selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor** page.
- ***XSLT Tags***. Specifies how Skylar One should parse the collection object from the XML document that is returned by the request specified in the ***XSLT Request*** field. For a full description of this field, see the *Specifying XSLT Tags* section.

Specifying XML Tags and SOAP Tags

In the ***XML Tags*** field (for XML Dynamic Applications) or ***SOAP Tags*** field (for SOAP Dynamic Applications), you must specify how Skylar One should parse a value (or values) from the returned XML document.

In this field, you must specify a series of XML tags that specify an element or attribute in the XML document.

Parsing an XML Element

To specify that Skylar One should parse the value of an element (that is, a value that is inside an opening and closing XML tag), specify the series of opening XML tags that lead to that element. For example,

suppose the XML document looks like this:

```
<LoginMonitor>
  <CustomerID>
    <AuthFailures>10</AuthFailures>
  </CustomerID>
</LoginMonitor>
```

If you wanted to parse the value of the <AuthFailures> element (in this example, the value "10"), you would enter the following in the ***XML Tags*** field or ***SOAP Tags*** field:

```
<LoginMonitor><CustomerID><AuthFailures>
```

To specify that Skylar One should parse the value of an XML tag attribute, specify the series of opening XML tags that lead to the XML tag that contains the attribute. In the XML tag that contains the attribute, specify the attribute in the XML tag with the substitution character %V as the attribute value. For example, suppose the XML document looks like this:

```
<LoginMonitor>
  <CustomerID>
    <AuthFailures type="1">10</AuthFailures>
  </CustomerID>
</LoginMonitor>
```

If you wanted to parse the value of the "type" attribute from the <AuthFailures> tag (in this example, the value "1"), you would enter the following in the ***XML Tags*** field or ***SOAP Tags*** field:

```
<LoginMonitor><CustomerID><AuthFailures type="%V">
```

A collection object in an XML or SOAP Dynamic Application will return a list of values if the ***XML Tags*** or ***SOAP Tags*** specify an element or attribute that appears multiple times in the XML document. For example, if the XML document looks like this:

```
<LoginMonitor>
  <CustomerID>
    <AuthFailures type="1">10</AuthFailures>
```

```
    </CustomerID>
    <CustomerID>
      <AuthFailures type="2">20</AuthFailures>
    </CustomerID>
  </LoginMonitor>
```

The example collection objects in this section will both collect a list of two values.

Specifying XSLT Tags

Unlike XML documents in XML and SOAP Dynamic Applications, the XML document returned by an XSLT request is in a specific format. To specify how Skylar One should parse a value (or values) from the XML document returned by an XSLT request, enter the element tag name for the value you want to collect in the ***XSLT Tags*** field.

For example, suppose the result of the XSLT Parser transformation is:

```
<response>
  <row>
    <cpupercent>90</cpupercent>
  </row>
</response>
```

If you wanted to parse the value of the <cpupercent> element (in this example, the value "90"), you would enter "cpupercent" in the ***XSLT Tags*** field.

NOTE: Do not include the less-than (<) or greater-than (>) characters in the ***XSLT Tags*** field.

Session ID Objects

For SOAP and XSLT Dynamic Applications, an additional option is available in the ***Class Type*** field for collection objects: *[109] SOAP/XSLT Session ID*.

Some SOAP servers require that a management system like Skylar One use a unique, persistent session ID every time the management system authenticates. This means that when Skylar One monitors such a SOAP server, the same session ID value must be used for every poll period.

You can tell Skylar One to use the same session ID for every poll period by using the *[109] SOAP/XSLT Session ID* class type. When a collection object uses this class type:

- The request that returns this collection object will be considered a "login" request to the SOAP server. Usually, this request will be performed before all other requests in the Dynamic Application. This SOAP or XSLT request must not return collection objects that do not use the *[109] SOAP/XSLT Session ID* class type.
- At the start of each poll period, Skylar One checks to see if a value has previously been collected and stored for this collection object:
    - If a value has been previously collected and stored, Skylar One **does not perform the SOAP or XSLT request the collection object is associated with**. The previously collected value is used when this collection object is substituted in to the other requests (for SOAP Dynamic Applications) or when this collection object appears in the XML document that contains all collected values (for XSLT Dynamic Applications).
    - If a value has not been previously collected and stored, Skylar One performs the SOAP or XSLT request the collection object is associated with as normal. The collected value is stored for use in future polling periods.
- If a previously collected value is used to populate the collection object and one of the subsequent requests returns a SOAP fault, Skylar One will:
    - Perform the SOAP request that retrieves the SOAP/XSLT Session ID collection object, storing the new value for use in future polling periods.
    - Resume collection by retrying the request that returned the SOAP fault.

> **NOTE:** The request that returns a SOAP/XSLT Session ID collection object will be performed only once per poll period. After the request that returns a SOAP/XSLT Session ID collection object is performed, subsequent request that return SOAP faults will not be retried.

Creating an XML Dynamic Application

XML Dynamic Applications retrieve data by making HTTP requests to web services or APIs that return XML-formatted responses. This section walks you through the process of creating an XML Dynamic Application, including configuring the request, parsing the XML response, and extracting the data you want to monitor. This section also describes how to create and test an XML Performance Dynamic Application.

Creating the Dynamic Application

In this example, the Dynamic Application will collect data from a Dial Number Identification Service on a transaction switch. To define the properties for the Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the **[Actions]** button, then select *Create New Dynamic Application*. The **Dynamic Applications Create New Application** page appears.
3. Supply values in the following fields:

- ***Application Name***. Enter "Dial Number Identification Service" in this field.
- ***Application Type***. Select *XML Performance*.
- ***Poll Frequency***. Select *Every 1 Minute* to see data as quickly as possible.

4. For this example, you can leave the remaining fields set to their default value. Select the **[Save]** button to save the Dynamic Application.

Defining XML Data-Points to Monitor

After defining the properties for the XML Dynamic Application, we must determine what data-points are available for the Dynamic Application to collect. Each data point will be used to create a collection object.

This example Dynamic Application will collect data from an XML document that looks like this:

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<TS-IENMonitor>

  <DNIS>

    <DNIS-ID>997</DNIS-ID>

    <X25CallsSuccess>100</X25CallsSuccess>

    <X25CallsFail>10</X25CallsFail>

    <AuthsSuccess>100</AuthsSuccess>

    <AuthsFail>10</AuthsFail>

    <AuthsFailRate>10</AuthsFailRate>

    <SettlementsSuccess>100</SettlementsSuccess>

    <SettlementsFail>10</SettlementsFail>

    <SettlementsFailRate>10</SettlementsFailRate>

  </DNIS>


  <DNIS>

    <DNIS-ID>998</DNIS-ID>
```

```
      <X25CallsSuccess>100</X25CallsSuccess>
      <X25CallsFail>10</X25CallsFail>
      <AuthsSuccess>100</AuthsSuccess>
      <AuthsFail>10</AuthsFail>
      <AuthsFailRate>10</AuthsFailRate>
      <SettlementsSuccess>100</SettlementsSuccess>
      <SettlementsFail>10</SettlementsFail>
      <SettlementsFailRate>10</SettlementsFailRate>
    </DNIS>

    <DNIS-Summary>
      <X25CallsTotal>220</X25CallsTotal>
      <AuthsTotal>110</AuthsTotal>
      <SettlementsTotal>110</SettlementsTotal>
    </DNIS-Summary>
  </TS-IENMonitor>
```

We will define the following collection objects in Skylar One:

Name	XML Tag	Description
DNIS ID	<DNIS -ID>	DNIS stands for dialed number identification service. DNIS is a service sold by telecommunications companies to corporate clients that lets them determine which telephone number was dialed by a customer. For example, a company may have a different toll free number for each product line it sells. If a call center is handling calls for multiple product lines, the switch that receives the class can examine the DNIS, then play the appropriate recorded greeting. The collection object "DNIS ID" specifies the phone number called by customers
Successful X25 Calls	<X25CallsSuccess>	X.25 is a data communications protocol developed to describe how data passes into and out of public data communication networks. The protocol is used primarily by telephone companies. The collection object "Successful X25 Calls" specifies how many incoming and outgoing phone calls using X25 were successfully connected.
Failed X25 Calls	<X25CallsFail>	The collection object "Failed X25 Calls" specifies how many incoming and outgoing phone calls using X25 were not successfully connected.
Successful Authentications	<AuthsSuccess>	Authentication means the verification of the identity of a person or process placing the call. The collection object "Successful Authentication" specifies how many calls were successfully authenticated.
Failed Authentications	<AuthsFail>	The collection object "Failed Authentication" specifies how many calls were not successfully authenticated.
Authentication Failure Rate	<AuthsFailRate>	The collection object "Authentication Failure Rate" specifies the percentage of authentications that failed.

Name	XML Tag	Description
Successful Settlements	<SettlementsSuccess>	A Settlement is the process by which merchant banks and cardholder banks exchange financial data resulting from sales transactions, cash disbursements, and merchandise credits. The collection object "Successful Settlements" specifies how many settlement transactions were completed successfully.
Failed Settlements	<SettlementsFail>	The collection object "Failed Settlements" specifies how many settlement transactions were not completed successfully.
Settlement Failure Rate	<SettlementsFailRate>	The collection object "Settlement Failure Rate" specifies the percentage of settlements that failed.
Total X25 Calls	<X25CallsTotal>	The collection object "Total X25 Calls" specifies the total number of successful and failed, incoming and outgoing phone calls using X25 on all monitored DNIS numbers.
Total Authentications	<AuthsTotal>	The collection object "Total Authentications" specifies the total number of successful and failed authentications on all monitored DNIS numbers.
Total Settlements	<SettlementsTotal>	The collection object Total Settlements specifies the total number of successful and failed settlements on all monitored DNIS numbers.

Defining the Collection Objects

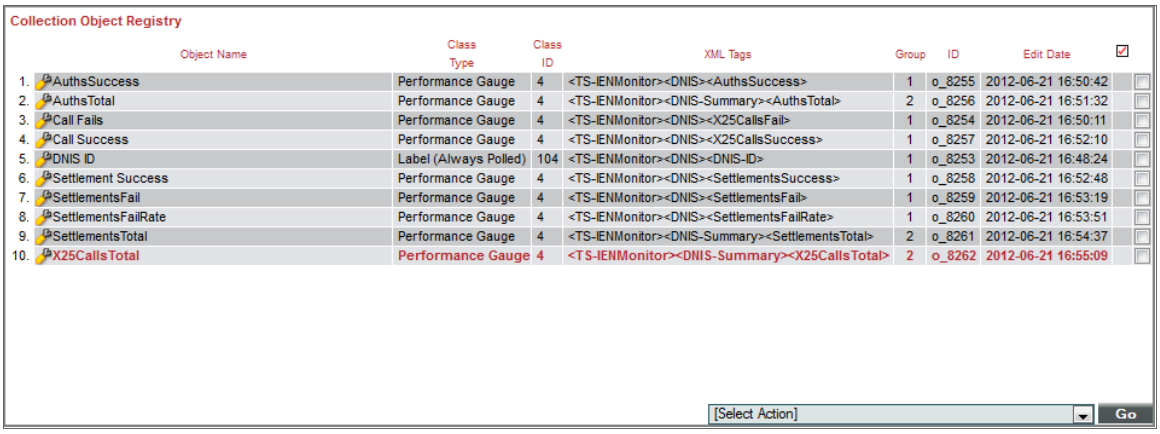
After determining what data-points are available, you can define your collection objects. To define the collection objects:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon () for the *Dial Number Identification Service* Application.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page appears.
4. To create the collection object for DNIS ID, supply values in the following fields:

- ***Object Name***. Enter "DNIS ID" in this field.
- ***XML Tags***. In this field you specify the element or attribute that should be parsed from the XML document for this collection object. Enter "<TS-IENMonitor><DNIS><DNIS-ID>" in this field
- ***Class Type***. Select *Label (Always Polled)*. This value will be used to label the graph lines on the performance graph.

- ***Group Number***. We selected *Group 1*. All collection objects that appear inside the <DNIS> section of the XML document will be in this group. Including all these collection objects in the same group means that this collection object will be used to label the graph lines on every performance graph that uses these collection objects.

5. For this example, you can leave the remaining fields set to their default values.
6. Select the **[Save]** button.
7. Select the **[Reset]** button to clear the form fields.
8. For the remaining nine collection objects, follow the example above. Once complete, the **Collection Object Registry** will look like this:



Collection Object Registry

	Object Name	Class Type	Class ID	XML Tags	Group	ID	Edit Date
1.	AuthsSuccess	Performance Gauge	4	<TS-IENMonitor><DNIS><AuthsSuccess>	1	o_8255	2012-06-21 16:50:42
2.	AuthsTotal	Performance Gauge	4	<TS-IENMonitor><DNIS-Summary><AuthsTotal>	2	o_8256	2012-06-21 16:51:32
3.	Call Fails	Performance Gauge	4	<TS-IENMonitor><DNIS><X25CallsFail>	1	o_8254	2012-06-21 16:50:11
4.	Call Success	Performance Gauge	4	<TS-IENMonitor><DNIS><X25CallsSuccess>	1	o_8257	2012-06-21 16:52:10
5.	DNIS ID	Label (Always Polled)	104	<TS-IENMonitor><DNIS><DNIS-ID>	1	o_8253	2012-06-21 16:48:24
6.	Settlement Success	Performance Gauge	4	<TS-IENMonitor><DNIS><SettlementsSuccess>	1	o_8258	2012-06-21 16:52:48
7.	SettlementsFail	Performance Gauge	4	<TS-IENMonitor><DNIS><SettlementsFail>	1	o_8259	2012-06-21 16:53:19
8.	SettlementsFailRate	Performance Gauge	4	<TS-IENMonitor><DNIS><SettlementsFailRate>	1	o_8260	2012-06-21 16:53:51
9.	SettlementsTotal	Performance Gauge	4	<TS-IENMonitor><DNIS-Summary><SettlementsTotal>	2	o_8261	2012-06-21 16:54:37
10.	X25CallsTotal	Performance Gauge	4	<TS-IENMonitor><DNIS-Summary><X25CallsTotal>	2	o_8262	2012-06-21 16:55:09

[Select Action] Go

NOTE: Select ***Performance Gauge*** as the ***Class Type*** for the remaining collection objects, because these collection objects contain performance values that can go up or down between poll periods. Include every collection object in *Group 1*, except for the **Total X25 Calls**, **Total Settlements**, and **Total Authentications**. Because these values must not be associated with the "DNS ID" labels, group these three collection objects in *Group 2*.

Creating the Presentation Object

When you create a collection object in a Performance Dynamic Application, Skylar One automatically creates a presentation object that corresponds to that collection object. You must enable these presentation objects for Skylar One to generate a graph using those presentation objects. To enable a presentation object:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon (🔧) for the *Dial Number Identification Service* Application.
3. Select the **[Presentations]** tab. The **Dynamic Applications Presentation Objects** page appears.
4. In the **Dynamic Applications Presentation Objects** page, select the wrench icon (🔧) for the presentation object you want to enable.
5. In the ***Active State*** drop-down, select *Enabled*.
6. Select the **[Save]** button.

7. Repeat the steps above for the remaining nine presentation objects.

Testing the Dynamic Application

To test the *Dial Number Identification Service* Dynamic Application in this example, you must configure a web server to host the XML document. The web server must:

- Return the XML document listed in the *Defining XML Data-Points to Monitor* section in response to a GET request.
- Be discoverable by Skylar One as an SNMP device or a Non-SNMP device.

This example uses an Administration Portal as the web server. The root directory for an Administration Portal is:

```
/usr/local/silo/gui/ap/www/
```

For this example:

- A directory called "xml" was created in the root directory of the Administration Portal.
- A file called "ienmonitor.xml", which contains the XML listed in the *Defining XML Data-Points to Monitor* section, was created in the "xml" directory.
- The permissions on the "xml" directory and the "ienmonitor.xml" file were changed to 755.
- The Administration Portal was discovered in Skylar One.

Creating a Credential

For Skylar One to collect data for the *Dial Number Identification Service* Dynamic Application from a test device, you must create a SOAP/XML credential. To create the SOAP/XML credential:

1. Go to the **Credential Management** page (System > Manage > Credentials).
2. Select the **[Create]** button, then select *SOAP/XML Host Credential* from the drop-down list. The **Create New CURL/SOAP Credential** page appears.
3. Supply values in the following fields:
    - ***Profile Name***. Enter "Dial Number XML".
    - ***Method***. Select *GET* in this field.
    - ***URL***. Enter the URL of the XML document. Use the "%D" substitution character to supply the IP address of the test device. This example uses the URL "http://%D/xml/ienmonitor.xml".
4. For this example, you can leave the reaming fields set to the default values.
5. Select the **[Save]** button.

Manually Aligning the Dynamic Application to the Test Device

After you have configured the test device, you can align the *Dial Number Identification Service* Dynamic Application to the device. After aligning the Dynamic Application to the device, Skylar One will collect data and we can view the Presentation objects we defined in the *Creating a Presentation Object* section.

To manually align the Dynamic Application to a device:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the test device you configured for this example. Select its wrench icon ( ).
3. The **Device Properties** page appears. Select the **[Collections]** tab.
4. In the **Dynamic Application Collections** page, select the **[Action]** button and select *Add Dynamic Application*. The **Dynamic Application Alignment** page appears.
5. In the Dynamic Applications pane, select the Dial Number Identification Service Dynamic Application. In the Credentials pane, select Dial Number XML.
6. Select the **[Save]** button to assign the credential.

Viewing the Reports

After the Dynamic Application has collected the data specified in the collection objects, you can view the performance report for the test device. To view the performance report for the test device with the *Dial Number Identification Service* Dynamic Application aligned to it:

1. From the **Dynamic Application Collections** page, select the **[Reset]** button to update the page with the latest information.
2. Locate the *Dial Number Identification Service*. If the graph icon ( ) is colored, the performance graph is available. Select the graph icon for the presentation object you want to view.

Or:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the test device you aligned the *Dial Number Identification Service* application to. Select the device's graph icon ( ).
3. The **Device Summary** page appears. Select the **[Performance]** tab.
4. In the left NavBar, select *Dial Number Identification Service*, then select the presentation object you want to view. For example, select ***AuthsSuccess***.
5. The AuthsSuccess report is displayed.
    - You can mouseover different data points on the report, and the report will display the collected value for the given time.
    - The values for each label object are displayed in the graph key at the bottom of the report.
6. To learn more above device performance reports, see the manual ***Monitoring Device Infrastructure Health***.

Dynamic Component Mapping & Caching with XSLT

The following sample describes the development of three XSLT Dynamic Applications. The Dynamic Applications in this example demonstrate:

- How to create Dynamic Applications to discover component devices using Dynamic Component Mapping.
- How to use caching to associate performance data with component devices.

The Dynamic Applications in this example collect component device information from a hypothetical management system. For simplicity, the management system used in this example does not require some features that would typically be used in XSLT Dynamic Applications:

- The management system reports all the necessary data in a single static XML document. This allows the Dynamic Applications to make only one request.
- The management system returns the XML document in response to a GET request. This means that the Dynamic Applications do not require XSLT Request code. This simplification also means you can try the Dynamic Applications described in this example by configuring a Web Server to act as the management system.

The XML document returned by the management system looks like this:

```
<xml>
  <component>
    <name>component-one</name>
    <id>00001</id>
    <cpu>80</cpu>
    <memory>80</memory>
  </component>
  <component>
    <name>component-two</name>
    <id>00002</id>
    <cpu>60</cpu>
    <memory>60</memory>
  </component>
  <component>
    <name>component-three</name>
    <id>00003</id>
    <cpu>70</cpu>
```

```
    <memory>70</memory>
  </component>
  <component>
    <name>component-four</name>
    <id>00004</id>
    <cpu>55</cpu>
    <memory>55</memory>
  </component>
  <component>
    <name>component-five</name>
    <id>00005</id>
    <cpu>20</cpu>
    <memory>20</memory>
  </component>
</xml>
```

Using this XML document, the Dynamic Applications in this example:

- Cache the returned XML document, so that the Dynamic Applications must make only a single request to the management system during a polling period.
- Retrieves the *name* and *id* values for each component to model each component as a separate device in Skylar One.
- Use the *cpu* and *memory* values to create performance graphs for each component device. The Dynamic Application that collects the CPU and Memory data will be aligned with each component device separately; the performance graph for each component device will show the CPU and Memory statistics for that component device only.

NOTE: Because the XML document used in this example is static, the values on the CPU and Memory graphs for each component device will be constant.

Design

This example uses three Dynamic Applications:

- ***Example Dynamic Component Mapping General***. This Dynamic Application:
    - Requests and caches the XML page from the management system.
    - Contains a discovery object so that Skylar One can automatically align the Dynamic Application with the management system during discovery.
    - Contains no other collection objects. Remember that a Dynamic Application that caches responses cannot include collection objects that need to be collected at regular intervals.
- ***Example Dynamic Component Mapping Discovery***. This Dynamic Application:
    - Contains a discovery object so that Skylar One can automatically align the Dynamic Application with the management system during discovery.
    - Uses the cached response collected by the ***Example Dynamic Component Mapping General*** Dynamic Application to collect the Name and ID values from the XML document.
    - Uses the Name and ID values to model the component devices. The Name value populates the *Device Name* component identifier, and the ID value populates the *Unique Identifier* component identifier.
    - Is associated with a Device Class. Skylar One assigns that device class to each component device that is created.
    - Automatically aligns the ***Example Component Performance*** Dynamic Application to each component device that is created.

> **NOTE:** The ***Example Dynamic Component Mapping Discovery*** meets the minimum requirements for creating component devices by including a *Device Name* component identifier, a *Unique Identifier* component identifier, and an associated device class.

- ***Example Component Performance***. This Dynamic Application:
    - Is automatically aligned to component devices by the ***Example Dynamic Component Mapping Discovery*** Dynamic Application.
    - Uses the cached response collected by the ***Example Dynamic Component Mapping General*** Dynamic Application to collect the CPU and Memory values from the XML document.
    - The XSLT Parser Code specified in this Dynamic Application uses the %U substitution (the *Unique Identifier* component identifier) to collect only the CPU and Memory values that are associated with the current component device.
    - Contains Presentation Objects to display the collected CPU and Memory values in a graph.

Creating the "Example Dynamic Component Mapping General" Dynamic Application

Defining the Dynamic Application Properties

To create the Dynamic Application and define the general properties for this Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the **[Actions]** button, then select *Create New Dynamic Application™*. The **Dynamic Applications Create New Application** page appears.
3. Supply values in the following fields:
    - ***Application Name***. Enter "Example Dynamic Component Mapping General" in this field.
    - ***Application Type***. This example uses the XSLT protocol. Dynamic Applications of type *Performance* must include a presentation object to work correctly. Because this Dynamic Application does not include collection objects that can be used in a presentation object, this Dynamic Application is of type Configuration. Select *XSLT Config [17]* in this field.
    - ***Poll Frequency***. To see data as quickly as possible, select Every 1 Minute in this field.
    - ***Caching***. This Dynamic Application must cache the response from the management system. Select *Cache Results* in this field.
    - ***Component Mapping***. This Dynamic Application does not include collection objects that are used to create component devices. Leave this checkbox unchecked.
4. For this example, you can leave the remaining fields set to the default values.
5. Select the **[Save]** button.

Adding the XSLT Request

The XSLT request in this Dynamic Application retrieves the XML document from the management system. The retrieved XML document is cached for use by the other Dynamic Applications in this example.

The XSLT request in this Dynamic Application contains XSLT Parser Code that transforms the XML document to collect the discovery object for this Dynamic Application. The XSLT Parser code applies only to this Dynamic Application. The other Dynamic Applications in this example contain different XSLT Parser code that will perform different transformations on the cached XML document.

To create the XSLT request for this Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon (🔧) for the ***Example Dynamic Component Mapping General*** Dynamic Application. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[Requests]** tab. The **Dynamic Applications Request Editor and Registry** page appears.

4. Supply values in the following fields:
    - ***Request Name***. Enter "Get Full Document" in this field.
    - ***Execution Sequence***. This Dynamic Application contains only one XSLT request. Select *0* in this field.
    - ***Active State***. This XSLT request must be executed to collect data. Select *Enabled* in this field.
    - ***XSLT Request Code***. Because Skylar One performs an HTTP GET request to retrieve the XML document from the management system, we are not required to generate an XML document to POST to the management system. However, Skylar One still performs a transformation using the ***XSLT Request Code*** before performing the request. Therefore, the XSLT Request Code used in this example performs a transformation that outputs a blank XML document. Enter the following code in this field:

```
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:output method="xml" version="1.0" encoding="iso-8859-1"
indent="yes"/>

</xsl:stylesheet>
```

    - ***XSLT Parser Code***. The discovery object in this Dynamic Application determines whether the returned XML document contains at least one "<component>" element that contains a value for the "<name>" element. Therefore, the ***XSLT Parser Code*** must transform the returned XML document into an XML document with the following structure:

```
<response>

  <row>

    <name>name value</name>

  </row>

</response>
```

      The ***XSLT Parser Code*** uses a single **xsl:value-of** element to select the first component name that appears in the returned XML document. The component name is inserted inside the required structure. Enter the following code in this field:

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:output method="xml" version="1.0" encoding="iso-8859-1"
indent="yes"/>
```

```
<xsl:template match="/">
  <response>
    <row>
      <name>
        <xsl:value-of select="xml/component/name"/>
      </name>
    </row>
  </response>
</xsl:template>
</xsl:stylesheet>
```

5. Select the **[Save]** button.

Adding the Discovery Object

This Dynamic Application includes one discovery object that tells Skylar One to automatically align the Dynamic Application to the management system. Because a Dynamic Application that caches responses cannot include collection objects that need to be collected at regular intervals, there are no other collection objects in this Dynamic Application. The discovery object in this Dynamic Application will determine whether the returned XML document contains at least one "<component>" element that contains a value for the "<name>" element.

To create the discovery object for this Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon (🔧) for the ***Example Dynamic Component Mapping Discovery*** Dynamic Application. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page appears.
4. Supply values in the following fields:
   - ***Object Name***. Enter "Discovery" in this field.
   - ***XSLT Tags***. The value you enter in this field must correspond with the name of an XML tag in the transformed response. Enter "name" in this field. The XSLT Request that you defined in the previous section returns the following transformed response:

```
<response>
  <row>
```

```
        <name></name>
    </row>
</response>
```

This discovery object looks for the presence of a component name in the transformed response.

- ***Class Type***. Select *100 Discovery* in this field. This specifies that the object is a discovery object.
- ***XSLT Request***. Select the *Get Full Document* request you created in the previous section.

5. For this example, you can leave the remaining fields set to the default values.
6. Select the **[Save]** button. After you save the discovery object, the form will change to show additional fields that apply only to discovery objects. These additional fields are not used by this example.

Creating the "Example Dynamic Component Mapping Discovery" Dynamic Application

Defining the Dynamic Application Properties

To create the Dynamic Application and define the general properties for this Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the **[Actions]** button, then select *Create New Dynamic Application™*. The **Dynamic Applications Create New Application** page appears.
3. Supply values in the following fields:
    - ***Application Name***. Enter "Example Dynamic Component Mapping Discovery" in this field.
    - ***Application Type***. This example uses the XSLT protocol. The data collected by this Dynamic Application, the component name and ID, are best displayed in tabular format, so this example uses the Configuration type. Select *XSLT Config [17]* in this field.
    - ***Poll Frequency***. To see data as quickly as possible, select Every 1 Minute in this field.
    - ***Caching***. This Dynamic Application uses the XML document collected by the ***Example Dynamic Component Mapping General*** Dynamic Application. Select *Consume Cached Results* in this field.
    - ***Component Mapping***. This Dynamic Application contains collection objects that Skylar One will use to create component devices. Check this checkbox.
4. For this example, you can leave the remaining fields set to the default values.
5. Select the **[Save]** button.

Adding the XSLT Requests

This Dynamic Application includes two XSLT requests. Each XSLT request contains XSLT Parser Code that transforms the XML document cached by the ***Example Dynamic Component Mapping General*** Dynamic Application. The transformed XML document contains:

- For the first request, the discovery object for this Dynamic Application. The discovery object in this Dynamic Application is the same as the discovery object in the ***Example Dynamic Component Mapping General*** Dynamic Application.
- For the second request, the component name and ID values that Skylar One uses to create the component devices.

To create the XSLT requests for this Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon (🔧) for the ***Example Dynamic Component Mapping Discovery*** Dynamic Application. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[Requests]** tab. The **Dynamic Applications Request Editor and Registry** page appears.
4. To create the first request, supply values in the following fields:
    - ***Request Name***. Enter "Discovery" in this field. This request returns the XML document that contains the discovery object for this Dynamic Application.
    - ***Execution Sequence***. This Dynamic Application does not require Skylar One to execute the XSLT requests in a specific order; however, the two requests must have unique values for this field. Select *0* in this field.
    - ***Active State***. This XSLT request must be executed to collect data. Select *Enabled* in this field.
    - ***Cached XSLT Request***. This field specifies the cached XML document to transform. Locate the ***Example Dynamic Component Mapping Discovery*** Dynamic Application and select *Get Full Document* from this list.
    - ***XSLT Parser Code***. This Dynamic Application uses the same discovery object as is used in the ***Example Dynamic Component Mapping General*** Dynamic Application. Therefore, this request uses the same ***XSLT Parser Code*** as the request in the ***Example Dynamic Component Mapping General*** Dynamic Application. Enter the following code in this field:

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:output method="xml" version="1.0" encoding="iso-8859-1"
indent="yes"/>

<xsl:template match="/">

  <response>

    <row>
```

```
            <name>
              <xsl:value-of select="xml/component/name"/>
            </name>
          </row>
        </response>
      </xsl:template>
      </xsl:stylesheet>
```

5. Select the **[Save]** button.
6. Select the **[Reset]** button to clear the form fields.
7. To create the second request, supply values in the following fields:
   - ***Request Name***. Enter "Get Components" in this field. This request will return the XML document that contains the name and ID values for the component devices.
   - ***Execution Sequence***. This Dynamic Application does not require Skylar One to execute the XSLT requests in a specific order; however, the two requests must have unique values for this field. Select *1* in this field.
   - ***Active State***. This XSLT request must be executed to collect data. Select *Enabled* in this field.
   - ***Cached XSLT Request***. This field specifies the cached XML document to transform. Locate the ***Example Dynamic Component Mapping Discovery*** Dynamic Application and select *Get Full Document* from this list.
   - ***XSLT Parser Code***. Two collection objects will be parsed from the output of this XSLT request: the component name and the component ID. The XML document returned by the management system might contain information about multiple component devices, so both the name and ID collection objects might be a list of values. The association of component name to component ID must be maintained for each pair of values, that is, they must reside in the same row in the transformed XML document. Therefore, the ***XSLT Parser Code*** must transform the returned XML document into an XML document with the following structure:

```
<response>
  <row>
    <name>name value for the first component device in the
    response</name>
    <id>ID value for the first component device in the
    response</id>
  </row>
```

```
    <row>
      <name>name value for the second component device in the
      response</name>
      <id>ID value for the second component device in the
      response</id>
    </row>
    .
    .
    .
    <row>
      <name>name value for the last component device in the
      response</name>
      <id>ID value for the first component device in the
      response</id>
    </row>
</response>
```

The ***XSLT Parser Code*** uses an **xsl:for-each** element to iterate through the component devices in the returned XML document, creating a <row> block for each component device. The component name and ID values are inserted inside the required structure. Enter the following code in this field:

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:output method="xml" version="1.0" encoding="iso-8859-1"
indent="yes"/>

<xsl:template match="/">
  <response>
    <xsl:for-each select="xml/component">
```

```
<row>
  <xsl:element name="name">
    <xsl:value-of select="name"/>
  </xsl:element>
  <xsl:element name="id">
    <xsl:value-of select="id"/>
  </xsl:element>
</row>
</xsl:for-each>
</response>
</xsl:template>
</xsl:stylesheet>
```

8. Select the **[Save]** button.

Adding the Collection Objects

This Dynamic Application includes a discovery object that tells Skylar One to automatically align the Dynamic Application to the management system. The discovery object in this Dynamic Application determines whether the returned XML document contains at least one "<component>" element that contains a value for the "<name>" element.

To create a component device, a Dynamic Component Mapping Dynamic Application must:

- Collect an object that maps to the *Unique Identifier* component identifier. This Dynamic Application includes a collection object that parses the component ID from the XML document. The component ID maps to the *Unique Identifier*.
- Collect an object that maps to the *Device Name* component identifier. This Dynamic Application includes a collection object that parses the component name from the XML document. The component name maps to the *Device Name*.
- Be associated with a component device class. The device class for this Dynamic Application is described in the *Creating a Device Class for the Component Devices* section.

To create the collection objects for this Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon (🔧) for the ***Example Dynamic Component Mapping Discovery*** Dynamic Application. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page appears.
4. To create the discovery object, supply values in the following fields:
    - ***Object Name***. Enter "Discovery" in this field.
    - ***XSLT Tags***. The value you enter in this field must correspond with the name of an XML tag in the transformed response. Enter "name" in this field. The "Discovery" XSLT Request you defined in the previous section returns the following transformed response:

```
<response>
  <row>
    <name></name>
  </row>
</response>
```

      This discovery object should look for the presence of a component name in the transformed response.
    - ***Class Type***. Select *100 Discovery* in this field. This specifies that the object is a discovery object.
    - ***XSLT Request***. Select the *Discovery* request you created in the previous section.
5. For this example, you can leave the remaining fields set to the default values.
6. Select the **[Reset]** button to clear the form fields.
7. To create the collection object for the component ID, supply values in the following fields:
    - ***Object Name***. Enter "Component ID" in this field.
    - ***XSLT Tags***. Enter "id" in this field. The "Get Components" XSLT Request you defined in the previous section returns the following <row> block for each component device:

```
<row>
  <name></name>
  <id></id>
</row>
```

      This collection object parses the component ID from each <row> block.
    - ***Class Type***. Select *10 Config Character* in this field. To maintain the leading zeroes in the collected component ID values, the ID is stored as a string.

- ***Group Number***. To maintain the association between the collected component ID and component name values, you must ensure that Skylar One uses the same internal indexes for each list. Skylar One maintains internal indexes for each group of collection objects. Therefore, the component ID and component name collection objects must be in the same group for Skylar One to use the same internal index for both collection objects. Select *Group 1* in this field.
- ***Index***. By default, Skylar One assigns internal indexes to the list of collected values based on the order in which they appear in the response. Suppose that "component-four" is disabled on the management system. Suppose that when "component-four" is disabled, it no longer appears in the XML response from the management system. When "component-four" appeared in the XML response, the following internal indexes would have been assigned:

  - Index 0 = component-one
  - Index 1 = component-two
  - Index 2 = component-three
  - Index 3 = component-four
  - Index 4 = component-five

  When "component-four" is disabled, it no longer appears in the XML response from the management system. During the next poll period, the index for "component-five" will change:

  - Index 0 = component-one
  - Index 1 = component-two
  - Index 2 = component-three
  - Index 3 = component-five

  In cases where the order or size of the list of values might change, you must designate a collection object as an index. This example designates the component ID collection object as the index. The collection object that is designated as an index must meet the following requirements:

  - The list of values returned by the collection object must be unique. In this example, the component ID is unique.
  - If a previously collected value in the list of values appears in a new list of collected values, that collected value is associated with the same set of values. In this example, a set of values is a component ID/component name pair. When a previously collected component ID is collected again, it is always associated with the same component name; therefore, the component ID can be used as the index.

  Select the ***Index*** checkbox.
- ***XSLT Request***. Select the *Get Components* request you created in the previous section.
- ***Component Identifiers***. Each value collected for this collection object will be used as the unique identifier for a component device. Select *Unique Identifier (%U)* in this field.

8. For this example, you can leave the remaining fields set to the default values.
9. Select the **[Save]** button.

10. Select the **[Reset]** button to clear the form fields.
11. To create the collection object for the component name, supply values in the following fields:
    - ***Object Name***. Enter "Component Name" in this field.
    - ***XSLT Tags***. Enter "name" in this field. The "Get Components" XSLT Request you defined in the previous section returns the following <row> block for each component device:

```
<row>
  <name></name>
  <id></id>
</row>
```

      This collection object parses the component name from each <row> block.
    - ***Class Type***. Select *10 Config Character* in this field. The component name is a string.
    - ***Group Number***. To maintain the association between the collected values for component ID and component name, you must ensure that Skylar One uses the same internal indexes for each list. Skylar One maintains internal indexes for each group of collection object. Therefore, the component ID and component name collection objects must be in the same group for Skylar One to use the same internal index for both collection objects. Select *Group 1* in this field.
    - ***Index***. Leave this checkbox unchecked. The collection object for the component ID has already been designated as the index for this group of collection objects.
    - ***XSLT Request***. Select the *Get Components* request you created in the previous section.
    - ***Component Identifiers***. Each value collected for this collection object will be used as the device name of a component devices. Select *Device Name (%N)* in this field.
12. For this example, you can leave the remaining fields set to the default values.
13. Select the **[Save]** button.

Creating a Device Class for the Component Devices

For Skylar One to create a component device, the Dynamic Component Mapping Dynamic Application must include collection objects that map to the *Device Name* and *Unique Identifier* component identifiers. In addition, the same Dynamic Application must be associated with a component Device Class. When you associated a component Device Class with a Dynamic Application, you are telling Skylar One to assign that device class to each component device that is created by the Dynamic Application.

To create a device class that is associated with the ***Example Dynamic Component Mapping Discovery*** Dynamic Application, perform the following steps:

1. Go to the **Device Class Editor** page (System > Customize > Device Classes).
2. In the **Device Class Editor** pane at the top of the page, supply values in the following fields:
    - ***Device Type***. This device class will be assigned to component devices. Select *Component* in this field.

- ***Root Device***. The component devices in this example will not act as root devices. Leave this checkbox unchecked.
- ***Device Class***. Enter "Example" in this field.
- ***Description***. Enter "Component Device" in this field.
- ***Dynamic App Alignment***. Select *Example Dynamic Component Mapping Discovery* in this field.

3. Select the **[Save]** button.

Creating the "Example Component Performance" Dynamic Application

Defining the Dynamic Application Properties

To create the Dynamic Application and define the general properties for this Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the **[Actions]** button, then select *Create New Dynamic Application™*. The **Dynamic Applications Create New Application** page appears.
3. Supply values in the following fields:
   - ***Application Name***. Enter "Example Component Performance" in this field.
   - ***Application Type***. This example uses the XSLT protocol. The data collected by this Dynamic Application will be displayed in a graph, so this example uses the Performance archetype. Select *XSLT Performance* in this field.
   - ***Poll Frequency***. To see data as quickly as possible, select *Every 1 Minute* in this field.
   - ***Caching***. This Dynamic Application will use the XML document collected by the ***Example Dynamic Component Mapping General*** Dynamic Application. Select *Consume Cached Results* in this field.
   - ***Component Mapping***. Although this Dynamic Application will be aligned with component devices, this Dynamic Application does not include collection objects that are used to create component devices. Leave this checkbox unchecked.
4. For this example, you can leave the remaining fields set to the default values.
5. Select the **[Save]** button.

Adding the XSLT Request

This Dynamic Application includes one XSLT request. The XSLT request will contain XSLT Parser Code that transforms the XML document cached by the ***Example Dynamic Component Mapping General*** Dynamic Application. The transformed response will return the CPU and memory utilization values for the component device.

The XSLT request uses the "%U" substitution character to filter the cached XML document. Before performing the transformation, Skylar One will replace "%U" with the unique identifier of the component

device for which data is currently being collected. By filtering the cached XML document using the unique identifier, the transformed response will contain the CPU and Memory utilization only for the component device for which data is currently being collected.

This Dynamic Application will be aligned with each of the five component devices in this example. Therefore, during each poll, the transformation will be performed five times, resulting in five different sets of CPU and memory utilization data (one set for each component device).

To create the XSLT request for this Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon (🔧) for the ***Example Component Performance*** Dynamic Application. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[Requests]** tab. The **Dynamic Applications Request Editor and Registry** page appears.
4. Supply values in the following fields:
    - ***Request Name***. Enter "Get CPU+Memory" in this field. This request will return the XML document that contains the CPU and memory utilization values for the component devices.
    - ***Execution Sequence***. This Dynamic Application will contain only one XSLT request. Select *0* in this field.
    - ***Active State***. This XSLT request must be executed to collect data. Select *Enabled* in this field.
    - ***Cached XSLT Request***. This field specifies the cached XML document to transform. Locate the ***Example Dynamic Component Mapping Discovery*** Dynamic Application and select *Get Full Document* from this list.
    - ***XSLT Parser Code***. Two collection objects will be parsed from the output from this XSLT request: the CPU utilization and the memory utilization. The transformed response must include the set of CPU and memory utilization values only for the component device for which data is currently being collected. Therefore, the ***XSLT Parser Code*** must transform the returned XML document into an XML document with the following structure:

```
<response>
  <row>
    <cpu>CPU utilization value</cpu>
    <memory>Memory utilization value</memory>
  </row>
</response>
```

      The ***XSLT Parser Code*** uses an **xsl:for-each** element to iterate through the component devices in the returned XML document. On each iteration, an **xsl:if** element is used to determine whether the ID value in the XML document matches the component ID value of the component device for which data is currently being collected (the "%U" substitution). If the ID values match, the <row> block is created using the CPU and memory values that appear in the XML document. Enter the following code in this field:

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

<xsl:output method="xml" version="1.0" encoding="iso-8859-1"
indent="yes"/>

<xsl:template match="/">

  <response>

    <xsl:for-each select="xml/component">

    <xsl:if test="id = '%U'">

    <row>

      <xsl:element name="cpu">

        <xsl:value-of select="cpu"/>

      </xsl:element>

      <xsl:element name="memory">

        <xsl:value-of select="memory"/>

      </xsl:element>

    </row>

    </xsl:if>

    </xsl:for-each>

  </response>

</xsl:template>

</xsl:stylesheet>
```

5. Select the **[Save]** button.

Adding the Collection Objects

This Dynamic Application contains one collection object for the CPU utilization value and one collection object for the memory utilization value. To create the collection objects for this Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon (🔧) for the ***Example Component Performance*** Dynamic Application. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page appears.
4. To create the collection object for the CPU utilization, supply values in the following fields:
    - ***Object Name***. Enter "CPU Usage" in this field.
    - ***XSLT Tags***. Enter "cpu" in this field. The "Get CPU+Memory" XSLT Request you defined in the previous section returns the following XML structure:

```
<row>
  <cpu></cpu>
  <memory></memory>
</row>
```

      This collection object parses the CPU utilization from the transformed response.
    - ***Class Type***. The CPU utilization is a number that can go up or down between polls. Select *4 Performance Gauge* in this field.
    - ***XSLT Request***. Select the *Get CPU+Memory* request you created in the previous section.
5. For this example, you can leave the remaining fields set to the default values.
6. Select the **[Save]** button.
7. Select the **[Reset]** button to clear the form fields.
8. To create the collection object for the memory utilization, supply values in the following fields:
    - ***Object Name***. Enter "Memory Usage" in this field.
    - ***XSLT Tags***. Enter "memory" in this field. The "Get CPU+Memory" XSLT Request you defined in the previous section returns the following XML structure:

```
<row>
  <cpu></cpu>
  <memory></memory>
</row>
```

      This collection object parses the memory utilization from the transformed response.

- ***Class Type***. The memory utilization is a number that can go up or down between polls. Select *4 Performance Gauge* in this field.
- ***XSLT Request***. Select the *Get CPU+Memory* request you created in the previous section.

9. For this example, you can leave the remaining fields set to the default values.
10. Select the **[Save]** button.

Editing the Presentation Objects

When you create a collection object in a Dynamic Application of type Performance, Skylar One automatically creates a presentation object that corresponds to that collection object. This example makes some minor changes to the presentation objects that were automatically created. To edit the presentation objects for this Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon () for the ***Example Component Performance*** Dynamic Application. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[Presentations]** tab. The **Dynamic Applications Presentation Objects** page appears.
4. Select the wrench icon for the ***CPU Usage*** presentation object. Edit the following fields:
   - ***Active State***. Select *Enabled* in this field.
   - ***Show as Percent***. The CPU utilization values are reported as percentage used. Select *Yes* in this field.
5. For this example, you can leave the remaining fields set to the default values.
6. Select the wrench icon for the ***Memory Usage*** presentation object. Edit the following fields:
   - ***Active State***. Select *Enabled* in this field.
   - ***Show as Percent***. The memory utilization values are reported as percentage used. Select *Yes* in this field.
7. For this example, you can leave the remaining fields set to the default values.
8. Select the **[Save]** button.

Automatically Aligning the Dynamic Application to Component Devices

When Skylar One creates component devices using data collected by a Dynamic Application, Skylar One can automatically align other Dynamic Applications to those component devices as they are created. In this example, the ***Example Component Performance*** Dynamic Application should be aligned to the component devices created by the ***Example Dynamic Component Mapping Discovery*** Dynamic Application.

The ***Example Component Performance*** Dynamic Application does not include a discovery object. Instead of a discovery object, we explicitly tell Skylar One to automatically align the ***Example Component Performance*** Dynamic Application with each component device that is created with the ***Example Dynamic Component Mapping Discovery*** Dynamic Application.

To do this, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon (🔧) for the ***Example Dynamic Component Mapping Discovery*** Dynamic Application. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[Component]** tab. The **Dynamic App Component Alignment** modal page appears:
4. Select the ***Example Component Performance*** Dynamic Application from the ***Unaligned Dynamic Apps*** list.
5. Select the ***>>*** button. The ***Example Component Performance*** Dynamic Application moves to the ***Aligned Dynamic Apps*** list.
6. Select the **[Save]** button.
7. Skylar One will automatically align the ***Example Component Performance*** Dynamic Application with each component device created by the ***Example Dynamic Component Mapping Discovery*** Dynamic Application.

Using the Dynamic Applications

Configuring a Test Device

To test the Dynamic Applications in this example, you must configure a web server to act as the management system. The web server must:

- Return the XML document listed in the *Overview* section in response to a GET request.
- Be discoverable by Skylar One as an SNMP device or a Non-SNMP device.

This example uses an Administration Portal as the web server. The root directory for an Administration Portal is:

```
/usr/local/silo/gui/ap/www/
```

For this example:

1. A directory called "xml" was created in the root directory of the Administration Portal.
2. A file called "components.xml", which contains the XML listed in the *Overview* section, was created in the "xml" directory.
3. The permissions on the "xml" directory and the "components.xml" file were changed to 755.

Editing the Device Class for the Test Device

You must edit the device class that Skylar One will assign to the device that hosts the web server. The device class for the test device must be enabled as a ***root device***. A ***root device*** is a device for which Skylar One can create children component devices. Perform the following steps to enable the device class as a root device:

1. Go to the **Device Class Editor** page (System > Customize > Device Classes).
2. In the **Device Class Register** at the bottom of the page, locate the device class that Skylar One will assign to the device that hosts the web server. In our example, the device that hosts the web server

will be assigned the device class with the description "EM7 G3 Admin Portal". Select the wrench icon (🔧) for the device class.

3. Check the ***Root Device*** checkbox.
4. Select the **[Save]** button.

Creating a Credential

For Skylar One to align the Dynamic Applications in this example to the device that hosts the web server, we must include a SOAP/XML credential in the discovery session. Skylar One will use the SOAP/XML credential to collect the XML document from the web server. To create the credential for this example:

1. Go to the **Credential Management** page (System > Manage > Credentials).
2. Select the **[Create]** button, and then select *SOAP/XML Host Credential* from the drop-down list. The **Create New CURL/SOAP Credential** page is displayed:
3. Supply values in the following fields:
   - ***Profile Name***. Enter "Components XML".
   - ***Method***. Select *GET* in this field.
   - ***URL***. Enter the URL of the XML document on the web server. Use the "%D" substitution character to supply the IP address of the test device. This example uses the URL "http://%D/xml/components.xml".
4. For this example, you can leave the remaining fields set to the default values.
5. Select the **[Save]** button.

Discovering the Test Device

To discover the device that hosts the web server:

1. Go to the **Discovery Control Panel** page (System > Manage > Classic Discovery or System > Manage > Discovery in the classic user interface).
2. Select the **[Create]** button. The **Discovery Session Editor** page is displayed in a new window:
3. Supply values in the following fields:
   - ***IP Address Discovery List***. Enter the IP address of your device.
   - ***SNMP Credentials***. If your device responds to SNMP, select the appropriate SNMP credential in this field.
   - ***Other Credentials***. Select the *Components XML* credential you created in the previous section.
   - ***Initial Scan Level***. Select at least *1. Initial Population of Apps* in this field.
   - ***Discover Non-SNMP***. If your device does not respond to SNMP, check this checkbox.
4. For this example, you can leave the remaining fields set to the default values.
5. Select the **[Save]** button.
6. In the **Discovery Control Panel** page, select the **[Reset]** button. The discovery session you created will appear in the list of discovery sessions.

7. Select the lightning bolt icon ( ) for the discovery session you created to run the discovery session. The Discovery Session log will appear.

Verifying the Dynamic Application Alignments

To verify that the discovery process aligned the appropriate Dynamic Applications to the test device and that Skylar One has created the component devices, perform the following steps:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Select the wrench icon ( ) for the device that hosts the web server. The **Device Properties** page is displayed in a new window.
3. Select the **[Logs]** tab. The **Device Logs & Messages** page is displayed.
4. Enter "Component" in the search bar and select the **[Search]** button. The search results should show:
    - The ***Example Dynamic Component Mapping General*** and ***Example Dynamic Component Mapping Discovery*** Dynamic Applications were aligned to the device during discovery.
    - That Skylar One created child component devices. The logs will contain one message for each component device that Skylar One created.

Viewing the Device Components Registry

The **Device Components** page (Devices > Device Components) displays a list of all root devices and component devices discovered by Skylar One. The **Device Components** page is similar to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface) page. The **Device Components** page displays all root devices and component devices in an indented view, so you can easily view the hierarchy and relationships between child devices, parent devices, and root devices. You can expand or hide the child devices for each root device and for each parent device.

To view the device that hosts the web server and its child component devices in the **Device Components** page:

1. Go to the **Device Components** page (Devices > Device Components).
2. Select the plus icon ( ) for the device that hosts the web server. The list is expanded to show the component devices.

Viewing the Component Device Map

The **Component Map** page allows you to view root devices and their children component devices in a graphical map. To view the test device and the child component devices in the **Component Map** page:

1. Go to the **Component Map** page (Classic Maps > Device Maps > Components).
2. If there are multiple root devices discovered in your Skylar One system, select the appropriate device from the drop-down list in the upper right of the page. A map of the device that hosts the web server and the child component devices is displayed.

Viewing the Performance Graphs

To view the performance graphs generated by the ***Example Component Performance*** Dynamic Application:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. Select the graph icon ( ) for one of the component devices. The **Device Summary** page is displayed in a new window.
3. Select the **[Performance]** tab. In the NavBar to the left of the page, select Example Component Performance > CPU Usage. The CPU Usage graph is displayed. Because the XSLT request in this Dynamic Application filtered the returned XML document by component ID, the graph displays only the CPU data for this component.

Expanding this Example

If you would like to create additional Dynamic Component Mapping Dynamic Applications, here are some suggestions for ways you could try modifying this example:

- Although the caching feature and the Dynamic Component Mapping feature are typically used together, you are not required to do so. Try combining the ***Example Dynamic Component Mapping General*** and ***Example Dynamic Component Mapping Discovery*** Dynamic Applications in to a single Dynamic Application that does not use the caching feature but still creates component devices. Use the ***XSLT Request Code*** from the ***Example Dynamic Component Mapping General*** Dynamic Application to create the XSLT requests for your combined Dynamic Application.
- Modify the ***Example Component Performance*** Dynamic Application so that it does not use caching. Again, use the ***XSLT Request Code*** from the ***Example Dynamic Component Mapping General*** Dynamic Application to create the XSLT requests for the new Dynamic Application. For your new Dynamic Application to collect data, you must manually align a credential that uses the "%D" substitution in the ***URL*** field. When a credential that uses "%D" in the ***URL*** field is used with a component device, Skylar One will substitute the IP address of the root device in to the URL.
- Modify the XML document on the test device to include additional layers in the component tree, i.e. include additional components that are children of the existing components. Create an additional Dynamic Application that creates component devices as children of the existing component devices. You can then add your new Dynamic Application to the list of Dynamic Applications that Skylar One should automatically align to each component device created by the ***Example Dynamic Component Mapping Discovery*** Dynamic Application. The modified XML document might look like this:

```
<xml>

  <component>

    <name>component-one</name>

    <id>00001</id>

    <cpu>80</cpu>
```

```
    <memory>80</memory>
    <sub_component>
      <name>sub-component-one</name>
      <id>10001</id>
    </sub_component>
    <sub_component>
      <name>sub-component-two</name>
      <id>10002</id>
    </sub_component>
  </component>
  <component>
    <name>component-two</name>
    <id>00002</id>
    <cpu>60</cpu>
    <memory>60</memory>
    <sub_component>
      <name>sub-component-three</name>
      <id>10003</id>
    </sub_component>
  </component>
  <component>
    <name>component-three</name>
    <id>00003</id>
    <cpu>70</cpu>
```

```
    <memory>70</memory>
    <sub_component>
      <name>sub-component-four</name>
      <id>10004</id>
    </sub_component>
    <sub_component>
      <name>sub-component-five</name>
      <id>10005</id>
    </sub_component>
    <sub_component>
      <name>sub-component-six</name>
      <id>10006</id>
    </sub_component>
  </component>
  <component>
    <name>component-four</name>
    <id>00004</id>
    <cpu>55</cpu>
    <memory>55</memory>
  </component>
  <component>
    <name>component-five</name>
    <id>00005</id>
    <cpu>20</cpu>
```

```
    <memory>20</memory>
    <sub_component>
      <name>sub-component-six</name>
      <id>10006</id>
    </sub_component>
    <sub_component>
      <name>sub-component-seven</name>
      <id>10007</id>
    </sub_component>
  </component>
</xml>
```

Chapter 7

WMI and PowerShell Dynamic Applications

Overview

This chapter describes how to use the WMI and PowerShell protocols to define collection objects and create WMI and PowerShell Dynamic Applications.

> **NOTE**: This chapter uses the WMI nomenclature with equivalent SQL nomenclature in parentheses. For example, instance (row), property (column), and class (table).

This chapter provides an overview of the WMI and PowerShell protocols and WMI and PowerShell Dynamic Applications in Skylar One. It includes the following topics:

This chapter covers the following topics:

WMI Requests 372

PowerShell Requests 376

WMI and PowerShell Collection Objects 380

Configuring Devices for Monitoring with WMI 381

Configuring Devices for Monitoring with PowerShell 413

Concurrent PowerShell Collection 455

Credentials for WMI and PowerShell Devices 465

Example: Creating a WMI Performance Dynamic Application 472

Example: Creating a PowerShell Performance Dynamic Application 479

What is WMI?

Windows Management Instrumentation, or WMI, is a Windows Service developed to access management information. WMI is a middle-layer technology that enables standardized management of Windows-based computers. It collects computer management data from a wide variety of sources and makes it accessible by using standard interfaces. WMI's specific query language is similar to SQL. For a comparison of WQL and SQL, see http://technet.microsoft.com/en-us/library/cc180454.aspx.

What is PowerShell?

Windows PowerShell is a command-line shell and scripting language for administration of Windows systems. Skylar One can execute PowerShell requests on target Windows devices via WinRM (Windows Remote Management). For an overview of Windows PowerShell, see https://learn.microsoft.com/en-us/powershell/scripting/overview?view=powershell-7.3.

Skylar One supports the following PowerShell versions for monitoring Windows devices:

- PowerShell 3.0
- PowerShell 4.0
- PowerShell 5.1

Prerequisites

This chapter does not describe how to plan, design, use, or troubleshoot Dynamic Applications for your network. This chapter assumes that you are already familiar with the common elements and concepts of Dynamic Applications. For general information on planning, designing, using, and troubleshooting Dynamic Applications, see the ***Dynamic Application Development*** chapter.

WMI Dynamic Applications use the WMI protocol. PowerShell Dynamic Applications use the PowerShell protocol. This chapter assumes that you are familiar with either the WMI or PowerShell protocols.

WMI and PowerShell Dynamic Applications

In Skylar One, a WMI Dynamic Application is a Dynamic Application that uses WMI to retrieve data from devices. WMI Dynamic Applications use WMI or WBEM requests to populate collection objects. WMI requests use WQL (WMI Query Language) to query WMI classes (tables) to retrieve data.

WBEM objects are populated with values returned by the wbemcli "get instance" command.

In Skylar One, a PowerShell Dynamic Application is a Dynamic Application that uses PowerShell to retrieve data from devices. PowerShell Dynamic Applications use PowerShell commands to populate collection objects.

Elements of WMI and PowerShell Dynamic Application

For comprehensive information on elements that apply to WMI and PowerShell Dynamic Applications, see the **Dynamic Application Development** chapter.

WMI Requests

In Skylar One, each WMI Dynamic Application must include at least one WMI or WBEM request.

Collection objects in WMI Dynamic Applications objects are populated with the values returned by a WMI request. WMI requests use WQL (WMI Query Language) to query WMI classes (tables) to retrieve data. A single WMI request can populate multiple WMI objects by querying for multiple class properties (table columns). WBEM objects are populated with values returned by the wbemcli "get instance" command.

Collection objects for both WMI and WBEM are aligned with properties (columns). The definition of each object specifies the WMI or WBEM request that will populate the collection object and the property name to align with the object. The retrieved values of the property will populate the object.

To more easily understand WMI, you can compare the terminology to standard SQL terminology:

WMI	SQL
Namespace	Database
Class	Table
Property	Column
Instance	Row

Defining a WMI Request

You can define a WMI request in the **WMI Request Editor & Registry** page. To define a WMI request:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. In the **Dynamic Applications Manager** page, find the Dynamic Application for which you want to define a WMI request. Select its wrench icon ( ).
3. Select the **[WMI Requests]** tab for the Dynamic Application.

> **NOTE:** The **[WMI Requests]** tab will only appear in Dynamic Applications of type *WMI Config* and *WMI Performance*.

4. In the **WMI Request Editor & Registry** page, enter the following:
   - ***WMI Request Name***. Name of the WMI request.
   - ***WMI Request Type***. Specifies whether to use a *WMI* request (query) or a *WBEM* request (a wbemcli "get instance" request sent over HTTP).
   - ***WMI Request Namespace***. Optional field. In this field, you can specify a WMI namespace. The WMI request will then use this namespace when requesting data. If you do not specify a value in this field, the WMI request will use the default namespace (usually *root*).
   - ***WMI Object Key***. The unique key for each instance (row) returned by the request. This unique key must be a property name, and the request must include that property (column) and return values from that property name (column). You must choose a key that remains constant over all polling periods, for example "Name" or "servicename".
   - ***Active State***. Specifies whether Skylar One should use this request when performing collection for the Dynamic Application. Choices are:
     - *Enabled*. Skylar One will use the WMI request when performing collection for the Dynamic Application.
     - *Disabled*. Skylar One will not use the WMI request when performing collection for the Dynamic Application.
   - ***WMI Request Query***. Enter the WQL query in this field. In most cases, these queries will be of the format:

     SELECT [*one or more properties (columns), separated by commas*] FROM *name of WMI class (table) where data is stored*]

     For more information on WQL, see http://msdn.microsoft.com/en-us/library/aa394606%28VS.85%29.aspx

     For more information on WMI classes, see http://msdn.microsoft.com/en-us/library/aa394554(v=VS.85).aspx

For a comparison of WQL and SQL, see http://technet.microsoft.com/en-us/library/cc180454.aspx

Example WMI Code

For our example Dynamic Application, we'll use the following WMI request:

```
SELECT TotalVisibleMemorySize,CSName,Caption,SerialNumber FROM Win32_
OperatingSystem
```

In this request, we are retrieving the property (column) values from the WMI class (table) ***Win32_OperatingSystem***. Win32_OperatingSystem is a class (table) that stores information about an instance of an operating system running on the monitored device. From this class, the WMI request retrieves the values of the following properties:

- ***TotalVisibleMemorySize***. Total amount, in kilobytes, of physical memory available to the operating system. This value does not necessarily indicate the true amount of physical memory, but only the amount that is reported to the operating system as available to it. In our example Dynamic Application, we can create an object that maps to this property. To map an object to the retrieved value from this property, we must ensure that the ***WMI Request Arguments*** field for the object (in the **Collection Objects** page) contains the value "TotalVisibleMemorySize".
- ***CSName***. Name of the computer system (device name as it appears to the operating system). In our example Dynamic Application, we can create an object that maps to this property. To map an object to the retrieved value from this property, we must ensure that the ***WMI Request Arguments*** field for the object (in the **Collection Objects** page) contains the value "CSName".
- ***Caption***. This is a short description of the operating system version. For example, "Microsoft Windows XP Professional Version = 5.1.2500". In our example Dynamic Application, we can create an object that maps to this property. To map an object to the retrieved value from this property, we must ensure that the ***WMI Request Arguments*** field (in the **Collection Objects** page) contains the value "Caption".
- ***SerialNumber***. Operating system product serial identification number. For example: "10497-OEM-0031416-71674". In our example Dynamic Application, we can create an object that maps to this property. To map an object to the retrieved value from this property, we must ensure that the ***WMI Request Arguments*** field (in the **Collection Objects** page) contains the value "SerialNumber".

Defining a WBEM Request

You can define a WBEM request in the **WMI Request Editor & Registry** page. To define a WBEM request in the **WMI Request Editor & Registry** page:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. In the **Dynamic Applications Manager** page, find the Dynamic Application for which you want to define a WBEM request. Select its wrench icon (🔧).
3. Select the **[WMI Requests]** tab for the Dynamic Application.

4. In the **WMI Request Editor & Registry** page, enter the following:
    - ***WMI Request Name***. Name of the WBEM request.
    - ***WMI Request Type***. Specifies whether to use a *WMI* request (a query) or a *WBEM* request (a wbemcli "get instance" request sent over HTTP).
    - ***WMI Request Namespace***. Optional field. In this field, you can specify a WMI namespace (database). The WBEM request will then use this namespace (databse) when requesting data. If you do not specify a value in this field, the WBEM request will use the default namespace (usually *root*).
    - ***WMI Object Key***. The unique key for each instance (row) returned by the query. This unique key must be a property name, and the query must include that property (column) and return values from that property name (column).
    - ***Active State***. Specifies whether Skylar One should use this request when performing collection for the Dynamic Application. Choices are:
        - *Enabled*. Skylar One will use the WBEM request when performing collection for the Dynamic Application.
        - *Disabled*. Skylar One will not use the WBEM request when performing collection for the Dynamic Application.
    - ***WMI Request Query***. Enter the wbemcli string in this field. In most cases, this will be of the format:

      /[name space]**:**[class name]**.**property=value (this last argument is optional)

      Usually, wbemcli requires that the request begins with the full path to the CIM object, including:

      http://username:password@hostname or IP:port,

      Skylar One uses the credentials for the Dynamic Application to automatically append this string to the front of each wbemcli request.

      For more information on wbemcli, see http://linux.die.net/man/1/wbemcli
5. Select the **[Save]** button to save the new WBEM request.

Example WBEM Request Code

For our example Dynamic Application, we'll use the following WMI request:

```
/root/cimv2:CIM_OperatingSystem
```

- In this request, we are retrieving value from the namespace (database) called ***/root/cimv2***.
- We are requesting all values from the class (table) called ***CIM_OperatingSystem***.
- This request will return all values from the class CIM_OperatingSystem. The values will be returned in the format

  property_name="value"

- Each WBEM object (defined in the **Collection Objects** page) must map to a property name returned by a WBEM request. To map each object, you must have specified a value in the ***WMI Request Arguments*** field that matches the name of a property returned by this request.

NOTE: Before defining objects for a WBEM request, you must know which property names will be returned.

Editing a WMI Request

To edit a WMI request in the **WMI Request Editor & Registry** page:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. In the **Dynamic Applications Manager** page, find the Dynamic Application for which you want to edit a WMI request. Select its wrench icon ( ).
3. Select the **[WMI Requests]** tab for the Dynamic Application.
4. In the **WMI Request Editor & Registry** page, find the WMI request in the **WMI Request Registry** pane. Select its wrench icon ( ).
5. The fields in the top pane are populated with values from the selected WMI request. You can edit the value of one or more fields. For a description of each field, see the section *Defining a WMI Request* in this manual.
6. Select the **[Save]** button to save your changes to the WMI request.

Editing a WBEM Request

You can edit a WBEM Request the same way you edit a WMI request. To view these steps, see the section *Editing a WMI Request* in this manual.

Deleting a WMI or WBEM Request

To delete a WMI or WBEM request in the **WMI Request Editor & Registry** page:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. In the **Dynamic Applications Manager** page, find the Dynamic Application for which you want to delete a WMI request or WBEM request. Select its wrench icon ( ).
3. Select the **[WMI Requests]** tab for the Dynamic Application.
4. In the **WMI Request Editor & Registry** page, find the request you want to delete in the **WMI Request Registry** pane. Select its delete icon ( ).

PowerShell Requests

PowerShell Dynamic Applications must include one or more requests that define how Skylar One should request data from a device. Each request specifies a PowerShell command that will gather a response

from the device.

Each collection object in a PowerShell Dynamic Application is associated with a request. The collection object definition specifies which property in the response should be used to populate the values for that collection object. A single request can be used to populate multiple collection objects.

Defining a PowerShell Request

To define a request for a PowerShell Dynamic Application:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Locate the Dynamic Application for which you want to define a request. Click its wrench icon (🔧). The **Dynamic Applications Properties Editor** page appears.
3. Click the **[PowerShell]** tab. The **Dynamic Applications PowerShell Command Editor & Registry** page appears.
4. Supply values in the following fields:
    - ***PowerShell Command Name***. Enter a name for the command.
    - ***Active State***. Specifies whether Skylar One should perform this request during collection for this Dynamic Application. Choices are *Enabled* or *Disabled*.
    - ***Response Object Key***. This field is not currently used.
    - ***PowerShell Implicit Remoting***. The mechanism by which a user or application accesses PowerShell cmdlets delivered via the Exchange Management Shell. This mechanism allows Dynamic Applications of type PowerShell to connect to the remote management shell for a Microsoft application and import the cmdlets published by that Microsoft application. Choices are *Enabled* or *Disabled*. If this field is set to *Enabled*, you can reference the imported cmdlets in the ***PowerShell Command Query*** field.
    - ***Configuration Name***. Grayed out unless ***PowerShell Implicit Remoting*** is set to *Enabled*. Select from a list of configuration names for PowerShell. By default, this field includes:
        - Microsoft.PowerShell
        - Microsoft.Exchange

    To enter a custom configuration name, click the plus sign icon (+) and enter the custom value.

    - ***Connection URI***. Grayed out unless ***PowerShell Implicit Remoting*** is set to *Enabled*. Specifies a Uniform Resource Identifier (URI) that defines the connection endpoint for the session. The URI must be fully qualified. By default, this field includes:
        - http://%D/PowerShell
        - http://%D:%P/WSMAN

NOTE: Skylar One will replace %D with the hostname or FQDN of the Microsoft server specified in the PowerShell credential.

To enter a custom URI, click the plus sign icon (+) and enter the custom value.

- ***Cmdlets to Import***. Grayed out unless ***PowerShell Implicit Remoting*** is set to *Enabled*. Enter a comma-separated list of one or more PowerShell cmdlet names that you want to use in this PowerShell session. If you leave this field blank, Skylar One will import all cmdlets from the remote management shell for use in this PowerShell session.
- ***PowerShell Command Query***. Enter the PowerShell command to execute. The PowerShell command must meet the following requirements:
    - The command must not end with any of the Format-* cmdlets. This includes the use of their aliases (IE. "fl" for Format-list).
    - The command must return output that can be piped to the Format-list cmdlet.
    - The command must not return whitespace over multiple lines.
    - The PowerShell cmdlets you invoke must exist on the target Windows server with the required version of PowerShell.
    - The PowerShell cmdlets you invoke must not write to the standard out pipe unless the output needs to be processed and put into a collection object.
    - The PowerShell cmdlets you invoke must be synchronous. Do not use asynchronous cmdlets, for example, Invoke-Async or Wait-Job.
    - The PowerShell cmdlets you invoke must not be an interactive cmdlet, for example, Enter-PSSession.
    - The PowerShell cmdlets you invoke must not query the same property twice.
    - If an invoked PowerShell cmdlet creates a new PSSession object, you must invoke the Remove-PSSession cmdlet before the original command ends.

  In addition to the requirements listed above, the following best practices are recommended when developing PowerShell commands:

    - Perform as much computational work in the PowerShell command as possible to reduce the workload of Skylar One.
    - Query only the pieces of information required to populate the collection objects in the Dynamic Application.
    - Per Microsoft guidelines, do not query the Win32_Product WMI class. For more information, see http://support.microsoft.com/kb/974524.

5. Click the **[Save]** button.

Editing a PowerShell Request

To edit a PowerShell request:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. In the **Dynamic Applications Manager** page, find the Dynamic Application for which you want to edit a PowerShell request. Select its wrench icon ( ).
3. Select the **[PowerShell]** tab for the Dynamic Application.

4. In the **Dynamic Applications PowerShell Command Editor & Registry** page, find the PowerShell request and select its wrench icon ( ).
5. The fields in the top pane are populated with values from the selected PowerShell request. You can edit the value of one or more fields. For a description of each field, see the section *Defining a PowerShell Request* in this manual.
6. Select the **[Save]** button to save your changes to the PowerShell request.

Deleting a PowerShell Request

To delete a PowerShell request in the **Dynamic Applications PowerShell Command Editor & Registry** page:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. In the **Dynamic Applications Manager** page, find the Dynamic Application for which you want to delete a PowerShell request. Select its wrench icon ( ).
3. Select the **[PowerShell]** tab for the Dynamic Application.
4. In the **Dynamic Applications PowerShell Command Editor & Registry** page, find the request you want to delete. Select its delete icon ( ).

Converting Legacy PowerShell Requests

The 7.3 version of Skylar One supported the use of PowerShell Dynamic Applications with an SSH-based proxy. Dynamic Applications developed for the SSH-based proxy are not compatible with the "agentless" PowerShell collection introduced in the 7.5 release.

To update a Dynamic Application developed for the SSH-based proxy for use with "agentless" PowerShell collection, edit each PowerShell request in the Dynamic Application and make the following changes to the ***PowerShell Command Query***:

- Remove "-Computer %s".
- Remove the "Format-List" cmdlet from the end of the command. If you are using a cmdlet that returns properties that you do not want to collect and you want to continue returning a sub-set of properties, use the "Select" cmdlet instead of "Format-List".

For example, suppose you need to update the following legacy PowerShell command:

```
Get-WmiObject -Computer %s Win32_DiskDrive | Format-List Partitions,
DeviceID, Model, Size, Caption
```

The new command would be:

```
Get-WmiObject Win32_DiskDrive | Select Partitions, DeviceID, Model, Size,
Caption
```

NOTE: In addition to updating the PowerShell requests in the Dynamic Application, you must also use a PowerShell credential with the Dynamic Application instead of a Basic/Snippet credential.

WMI and PowerShell Collection Objects

This section describes how to define collection objects for WMI and PowerShell Dynamic Applications.

NOTE: This section describes only the fields specific to Defining a Collection Object for a WMI or PowerShell Dynamic Application. All the remaining fields, for both performance and configuration archetypes, are described in detail in the ***Dynamic Application Development*** chapter. All other elements of WMI and PowerShell Collection Objects, such as presentation objects and alerts, behave in the same manner as other Dynamic Application types. For details on other parts of WMI or PowerShell Dynamic Applications, see the ***Dynamic Application Development*** chapter.

WMI-Specific Fields for Collection Objects

Unlike collection objects for other Dynamic Application types, collection objects for WMI Dynamic Applications are based on WMI Requests. Because of this, collection objects for WMI Dynamic Applications have the following unique fields:

- ***WMI Request Arguments***. When you request data from WMI or WBEM, you are requesting data from a class (table), which returns data in tabular form. In this field, you must specify the name of the property (table column) to associate with this object. This property name must be included in the WMI request, specified in the ***WMI Request*** field.
- ***WMI Request***. Name of the WMI request associated with this object. The WMI requests in this drop-down list are defined in one of two places:
    - If *No Caching* or *Cache Results* is selected in the ***Caching*** field in the **Dynamic Applications Properties Editor** page for this Dynamic Application, this list contains all WMI Requests defined in the **[WMI Requests]** tab for this Dynamic Application. Select from the list of WMI requests defined for this Dynamic Application.
    - If *Consume cached results* is selected in the ***Caching*** drop-down list in the **Dynamic Applications Properties Editor** page, this list contains all WMI Requests defined in WMI Dynamic Applications that have *Cache results* selected in the ***Caching*** field in the **Dynamic Applications Properties Editor** page. Select from the list of cached WMI requests.

> **NOTE**: If *Consume cached results* is selected in the ***Caching*** field in the **Dynamic Applications Properties Editor** page for a Dynamic Application, the **[WMI Requests]** tab is hidden and you cannot define WMI Requests for this Dynamic Application. You can only reference WMI Requests that reside in Dynamic Applications that cache results.

PowerShell-Specific Fields for Collection Objects

Unlike collection objects for other Dynamic Application types, collection objects for PowerShell Dynamic Applications are based on PowerShell Requests. Because of this, collection objects for PowerShell Dynamic Applications have the following unique fields:

- ***PowerShell Arguments***. Enter the property that is associated with this collection object. This property must be included in the PowerShell command you select in the ***PowerShell Request*** field.
- ***PowerShell Request***. Select the PowerShell request associated with this collection object, i.e. the PowerShell command that collects values for this collection object.

For example, suppose you have defined the following PowerShell request:

```
Get-WmiObject Win32_DiskDrive | Select Partitions, DeviceID, Model, Size,
Caption
```

This request returns the properties Partitions, DeviceID, Model, Size, and Caption. To store the values returned for each of the five properties, you would create five collection objects. For each of the five collection objects, supply the property name in the ***PowerShell Arguments*** field, e.g. "Partitions".

Configuring Devices for Monitoring with WMI

Before you can collect data from a device using WMI (Windows Management Instrumentation), you must configure the device to allow WMI queries. This section describes the configuration steps needed to enable WMI monitoring on Windows devices.

Configuring WMI on Windows 2008 Servers

Windows Management Instrumentation, or WMI, is the infrastructure that provides information about operations and management on Windows-based operating systems. WMI can be configured to respond to remote requests from Skylar One.

To configure a Windows device to respond to remote requests, you must perform the following steps:

1. *Configure Services*
2. *Configure the Windows Firewall*
3. *Configure a user account and permissions*
4. *Configuring a fixed port for WMI*

Most remote requests can be performed by a standard (non-administrator) user account that has been granted specific privileges. However, some requests can be performed only by a user with elevated permissions. For requests performed by Skylar One to a Windows server, the following users have elevated permissions:

- The default "Administrator" user account.
- A user account in the **Administrators** group on a Windows server that has User Account Control disabled.
- A user account in the **Administrators** group on a Windows server where a registry entry has been added to disable remote User Account Control filtering.

For a list of WMI classes that require elevated permissions, see http://msdn.microsoft.com/en-us/library/windows/desktop/aa826699%28v=vs.85%29.aspx

Step 1: Configuring Services

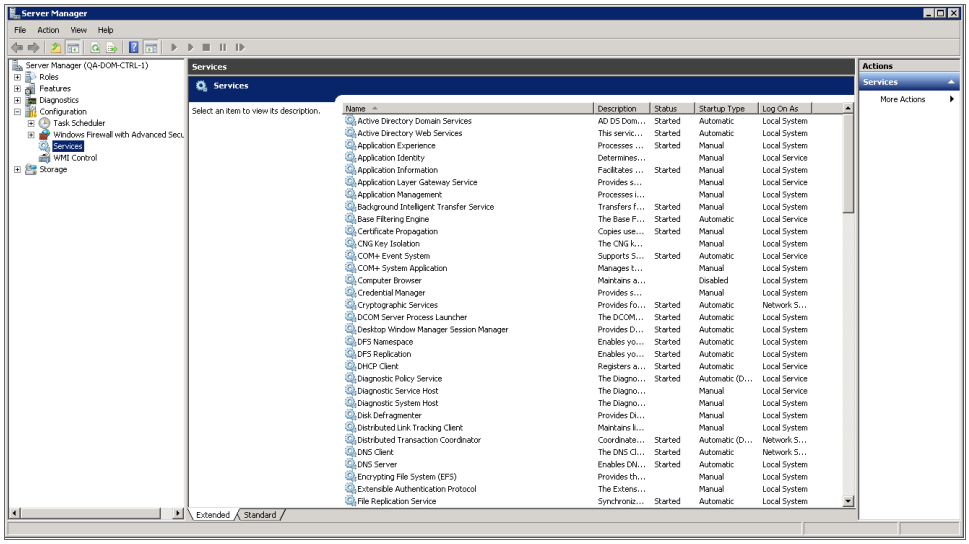
The following services must be running for a Windows device to respond to remote WMI requests:

> **NOTE:** ScienceLogic recommends you set all these services to automatically start.

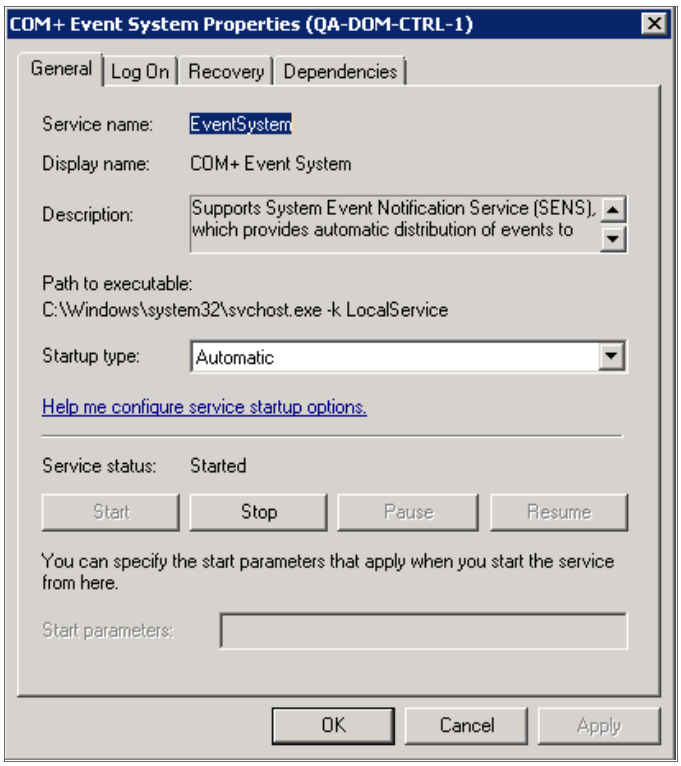
- COM+ Event System
- DCOM Server Process Launcher
- Remote Procedure Call (RPC)
- Remote Registry
- Server
- Windows Management Instrumentation

To ensure a service is running, perform the following steps:

1. In the left pane of the **Server Manager** window, expand the *Configuration* section, and then select *Services*.



2. For each required service, the ***Startup Type*** column should display *Automatic*. If a service does not have a ***Startup Type*** of *Automatic*, double-click on that service. The Properties window for that service is displayed:



3. In the ***Startup Type*** field, select *Automatic*.
4. Click the **[Apply]** button.
5. If the service has not already started, click the **[Start]** button.

Step 2: Configuring the Windows Firewall

To configure Windows Firewall to accept remote WMI requests, execute the following two commands at the console:

```
netsh advfirewall firewall set rule group="windows management
instrumentation (wmi)" new enable=yes
```

```
netsh advfirewall firewall set rule group="remote administration" new
enable=yes
```

Step 3: Configuring a User Account and Permissions

There are three ways to configure the user account that Skylar One will use to perform WMI requests:

1. To monitor the Windows server using WMI Dynamic Applications that require ***standard permissions***, you can configure a standard user account for use by Skylar One. The user account for use by Skylar One must be included in the **Distributed COM Users** and **Performance Monitor Users** groups. (For more information, consult Microsoft's documentation.)
2. To monitor the Windows server using WMI Dynamic Applications that require ***elevated permissions***, you can use the default "Administrator" user account. If you use the "Administrator" user account, you do not need to make changes to the User Account Control settings.
3. To monitor the Windows server using WMI Dynamic Applications that require ***elevated permissions***, you can also use a user account that is included in the **Administrators** group. However, you must perform **one** of the following additional steps to use this type of user account:
    - **Option 1:** Make the user a member of the **Distributed COM Users** and **Performance Monitor Users** groups, in addition to the **Administrator** group. (For more information, consult Microsoft's documentation.)
    - **Option 2:** *Configure User Access Control to allow elevated permissions*.

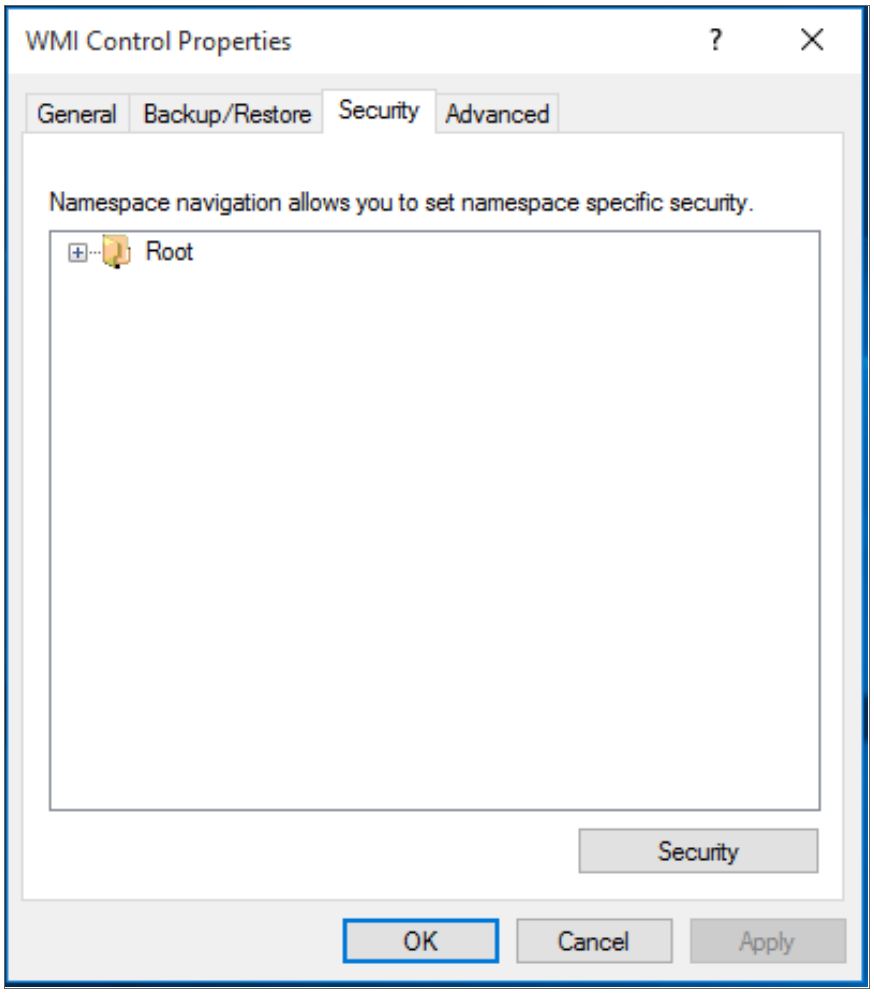
Configuring Namespace and DCOM Security Permissions

For each of these methods, you must ensure that the configured Namespace and DCOM security permissions allow that user to perform remote requests.

To configure the Namespace and DCOM security permissions:

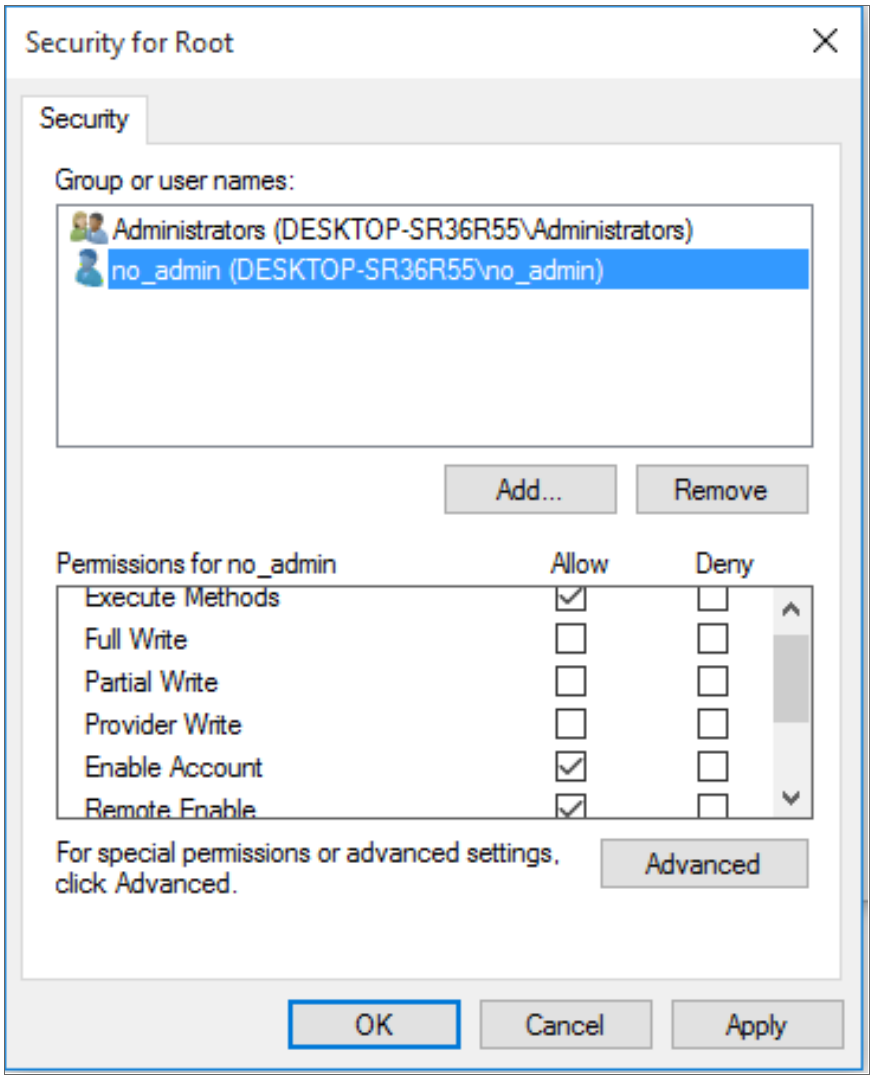
1. In the left pane of the **Server Manager** window, expand the *Configuration* section.
2. Right-click on the *WMI Control* entry and then select *Properties*.

3. In the **WMI Control Properties** window, click the **[Security]** tab:

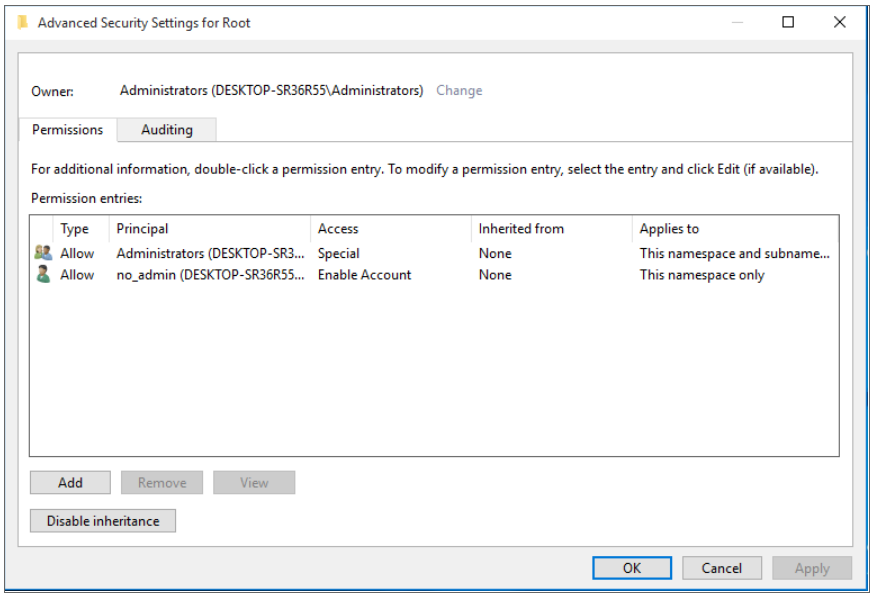


4. In the Security tab, select the *Root* entry from the navigation pane and then select the **[Security]** button. The **Security for Root** window appears.

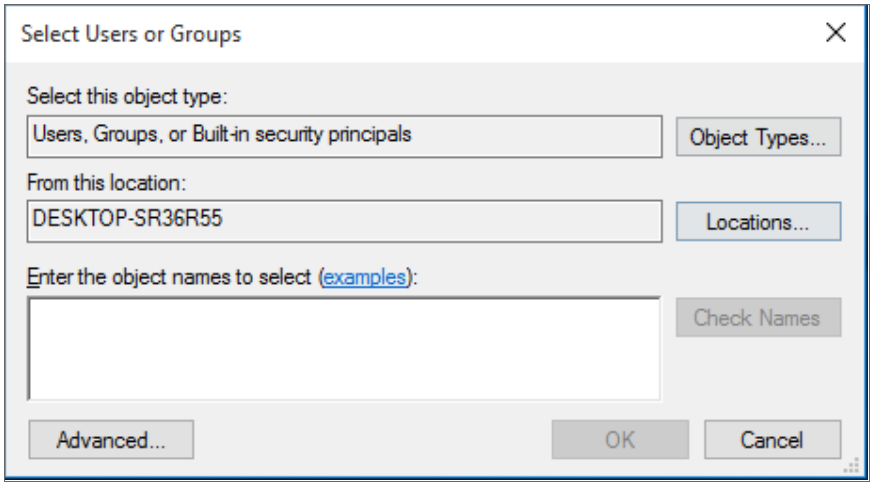
5. In the **Security for Root** window, select the **[Advanced]** button. The **Advanced Security Settings for Root** window is displayed:



6. In the **Advanced Security Settings for Root** window, click the **[Add]** button. The **Select User, Computer, Service Account, or Group** window appears.

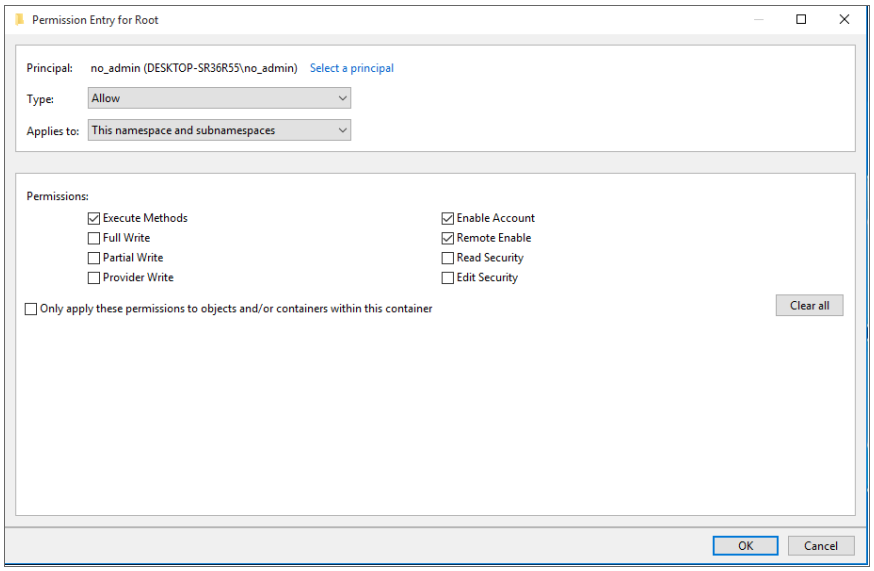


7. In the **Select User, Computer, Service Account, or Group** window :



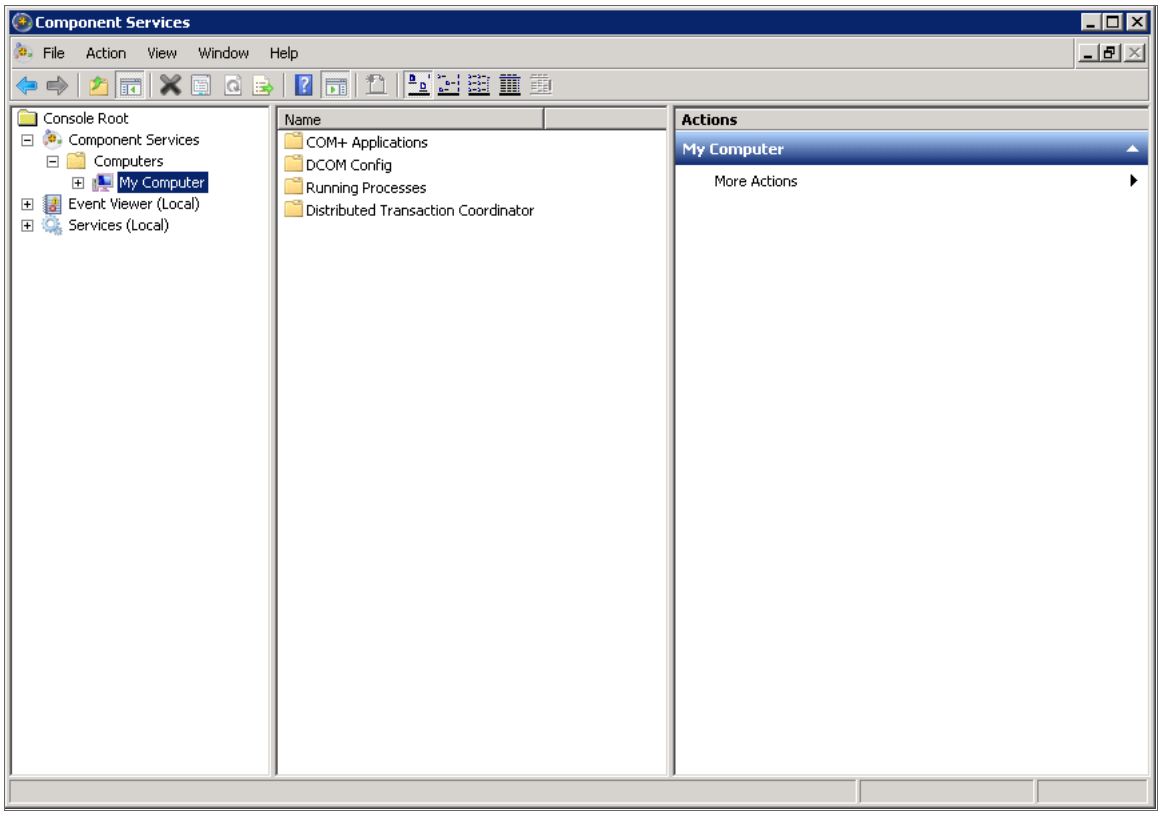
- In the ***Enter the object name to select*** field, enter the name of the user account that Skylar One will use to perform WMI requests or the name of a group that includes that user account.
- Click the **[Check Names]** button to verify the name and then click the **[OK]** button.

8. The **Permission Entry for Root** window is displayed:



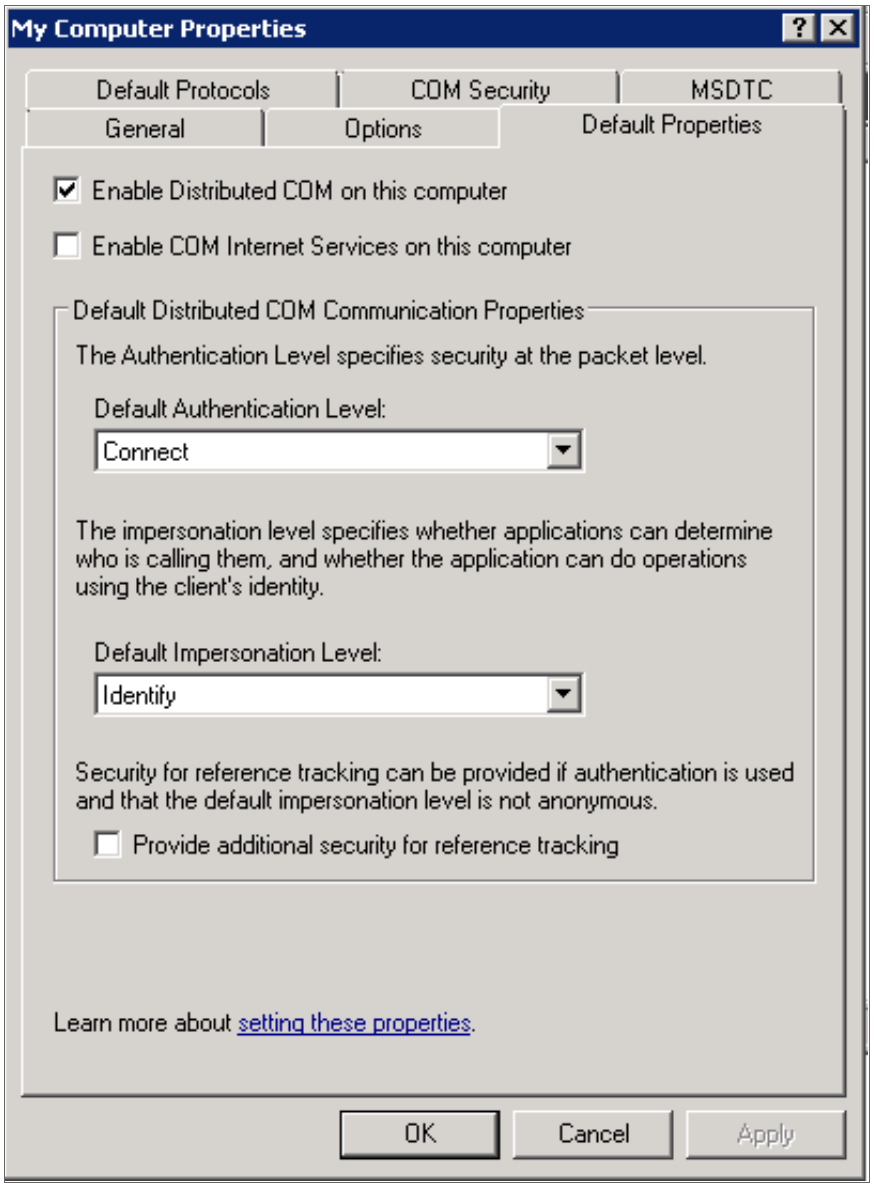
   - Select *This namespace and subnamespaces* in the ***Apply to*** field and select the ***Allow*** checkbox for all permissions.
   - Click the **[OK]** button.
9. In the **Advanced Security Settings for Root** window, click the **[Apply]** button.
10. Click the **[OK]** button in each open window to exit.
11. Go to the Start menu and select **[Run]**.

12. In the **Run** window, enter "dcomcnfg" and click **[OK]**. The **Component Services** window is displayed:



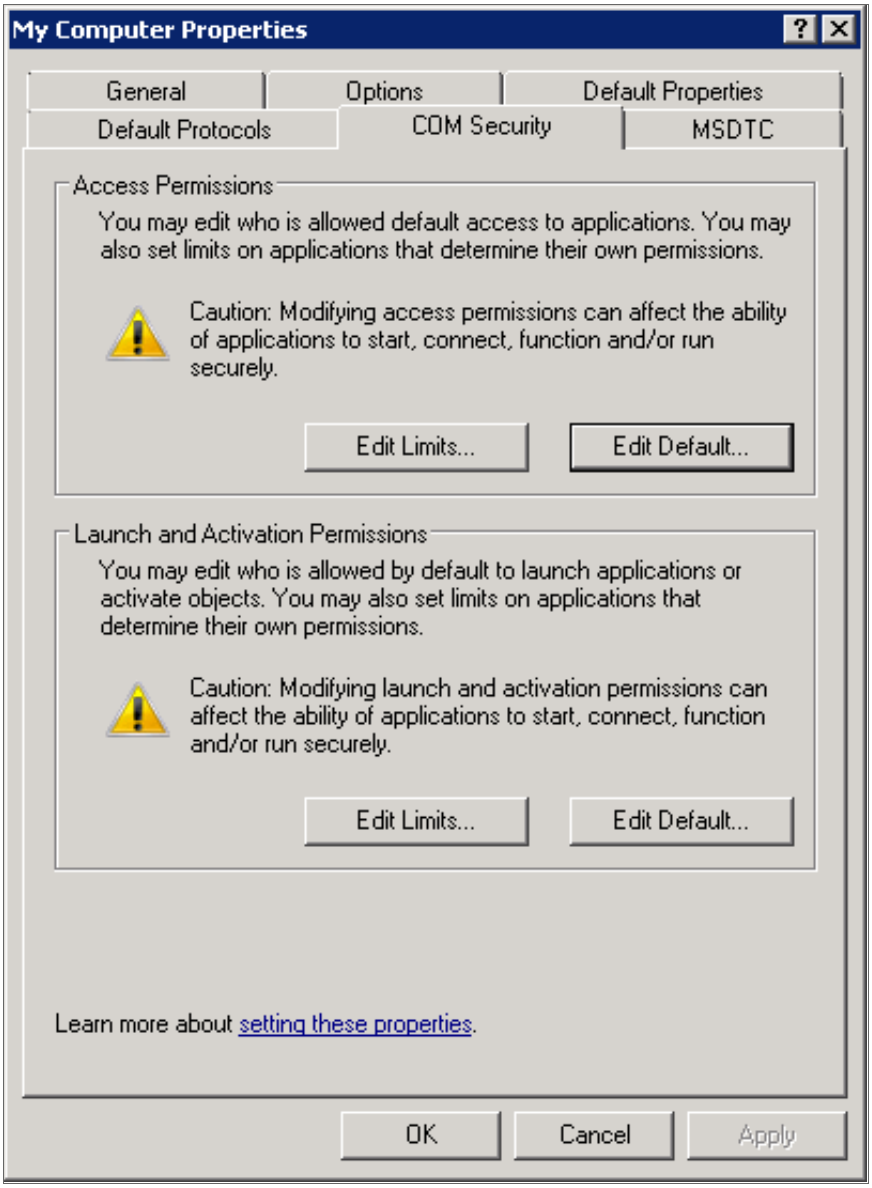
13. In the left pane, expand **Component Services > Computers**. Right-click on **My Computer** and select *Properties*. The **My Computer Properties** window is displayed.

14. In the **My Computer Properties** window, select the **[Default Properties]** tab:



- Ensure that the ***Enable Distributed COM on this computer*** checkbox is selected.
- Select *Connect* in the ***Default Authentication Level*** drop-down list.
- Select *Identify* in the ***Default Impersonation Level*** drop-down list.
- If you made changes in the **[Default Properties]** tab, click the **[Apply]** button.

15. Select the **[COM Security]** tab:



16. Select the **[Edit Limits...]** button in the **Access Permissions** pane.
17. In the window that appears, click the **[Add...]** button. The **Select Users, Computers, Service Accounts, or Groups** window is displayed.
    - Enter the name of the user account that Skylar One will use to perform WMI requests or the name of a group that includes that user account.
    - Click the **Check Names** button to verify the name and then click the **[OK]** button.
18. Select the group or user you added in the **Group or user names** pane and then select the **Allow** checkbox for all permissions.
19. Click the **[OK]** button.

20. Click the **[Edit Default...]** button in the **Access Permissions** pane, then repeat steps 16 - 19.
21. Click the **[Edit Limits...]** button in the **Launch and Activation Permissions** pane, then repeat steps 16 - 19.
22. Click the **[Edit Default...]** button in the **Launch and Activation Permissions** pane, then repeat steps 16 - 19.
23. Click the **[Apply]** button.
24. Click **[Yes]** in the confirmation window.

Configuring User Account Control to Allow Elevated Permissions

If you want to use WMI Dynamic Applications that require elevated permissions to monitor a Windows server and you are using a user account other than the default "Administrator" user account, you must perform **one** of the following two tasks:

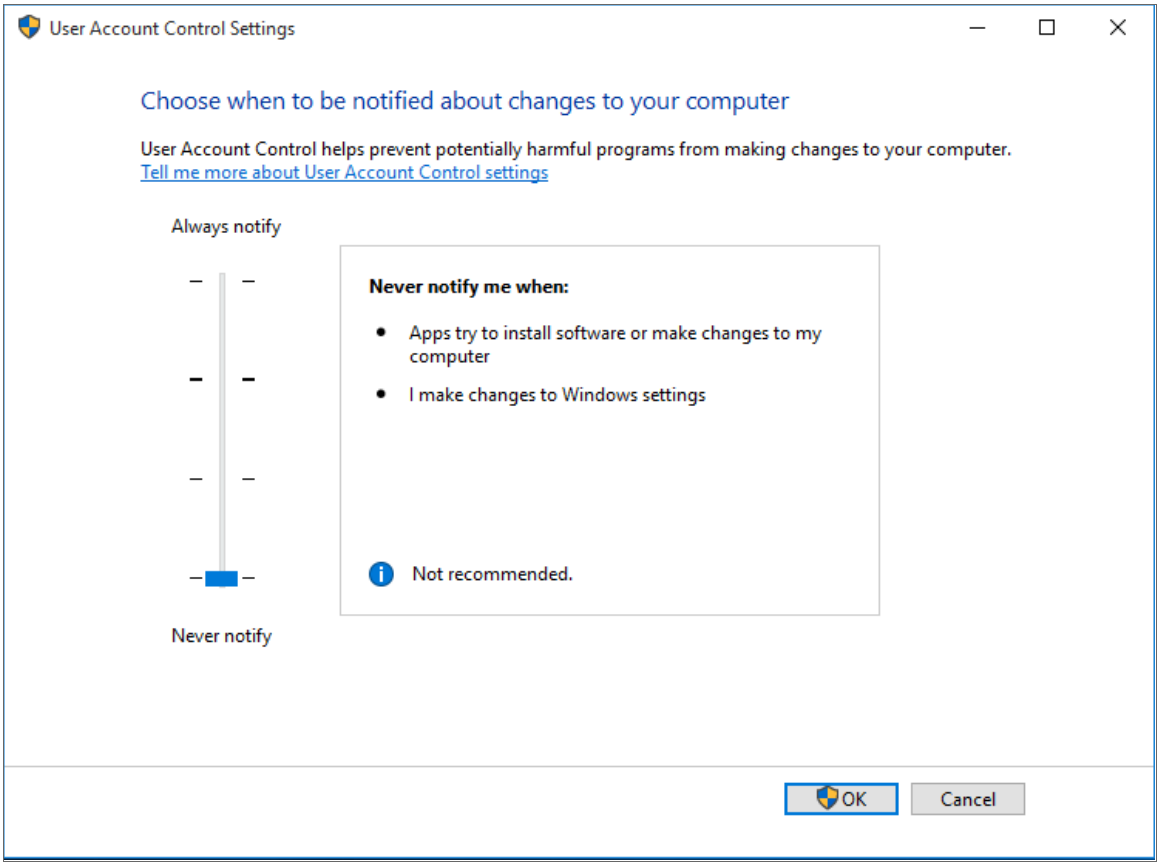
- **Option 1:** *Disable User Account Control*.
- **Option 2:** *Add a registry entry that disables remote User Account Control filtering*.

Option 1: Disabling User Account Control

To disable User Account Control:

1. Open the **Control Panel** in Large Icon or Small Icon view.
2. Select ***User Accounts***.

3. Select ***Change User Account Control Settings***. The **User Account Control Settings** window is displayed:



4. Move the slider to ***Never Notify***.
5. Click the **[OK]** button.
6. Restart the Windows server.

Option 2: Adding a Registry Entry that Disables Remote User Account Control Filtering

To add a registry entry that disables remote User Account Control filtering:

1. To disable the filter, open a text editor and add the following lines to a new file:

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_
MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System]

"LocalAccountTokenFilterPolicy"=dword:00000001
```

2. Save the file with a ".reg" extension.
3. In Windows Explorer, double click on the .reg file.

4. Select **[Yes]** in the pop-up window.

Step 4: Configuring a Fixed Port for WMI

Specific ports must be opened to allow WMI monitoring when there is a separate firewall between the Data Collector and the device. This can occur when the default configuration of the Windows Firewall blocks incoming network traffic for the Windows Management Instrumentation (WMI) connection.

For the WMI connection to succeed, the remote machine must permit incoming network traffic on TCP ports 135, 445, and additional dynamically-assigned ports, typically in the range of 1025 to 5000 and 49152 to 65535.

To set up a fixed port for WMI, see the Microsoft documentation on Setting Up a Fixed Port for WMI.

Configuring WMI for Windows Desktop Systems

This section describes how to configure devices that are running a desktop version of the Windows operating system for monitoring by Skylar One using WMI.

Before performing the tasks described in this section, you must know the IP address of each Skylar One appliance in your network. If you have not installed a Skylar One appliance, you must know the future IP address that will be used by each Skylar One appliance.

> **NOTE:** To be monitored by Skylar One, a Windows device must be running the Windows 7 operating system or later.

> **NOTE:** TCP/IP must be installed and configured before you can install SNMP on a Windows device.

Windows Management Instrumentation (WMI) is the infrastructure that provides information about operations and management on Windows-based operating systems. WMI can be configured to respond to remote requests from Skylar One. To configure a device running a desktop version of the Windows operating system to respond to remote requests, you must perform the following steps:

1. *Configure Services*
2. *Configure the Windows Firewall*
3. *Set Default Namespace Security*
4. *Set the DCOM Security Level*
5. *Disable User Account Control*
6. *Configuring a fixed port for WMI*

> **NOTE:** The following instructions describe how to configure WMI on devices running a desktop version of the Windows 10 operating system. For instructions on how to configure WMI on earlier Windows versions, consult Microsoft's documentation.

Step 1: Configuring Services

The following services must be running for a Windows device to respond to remote WMI requests:

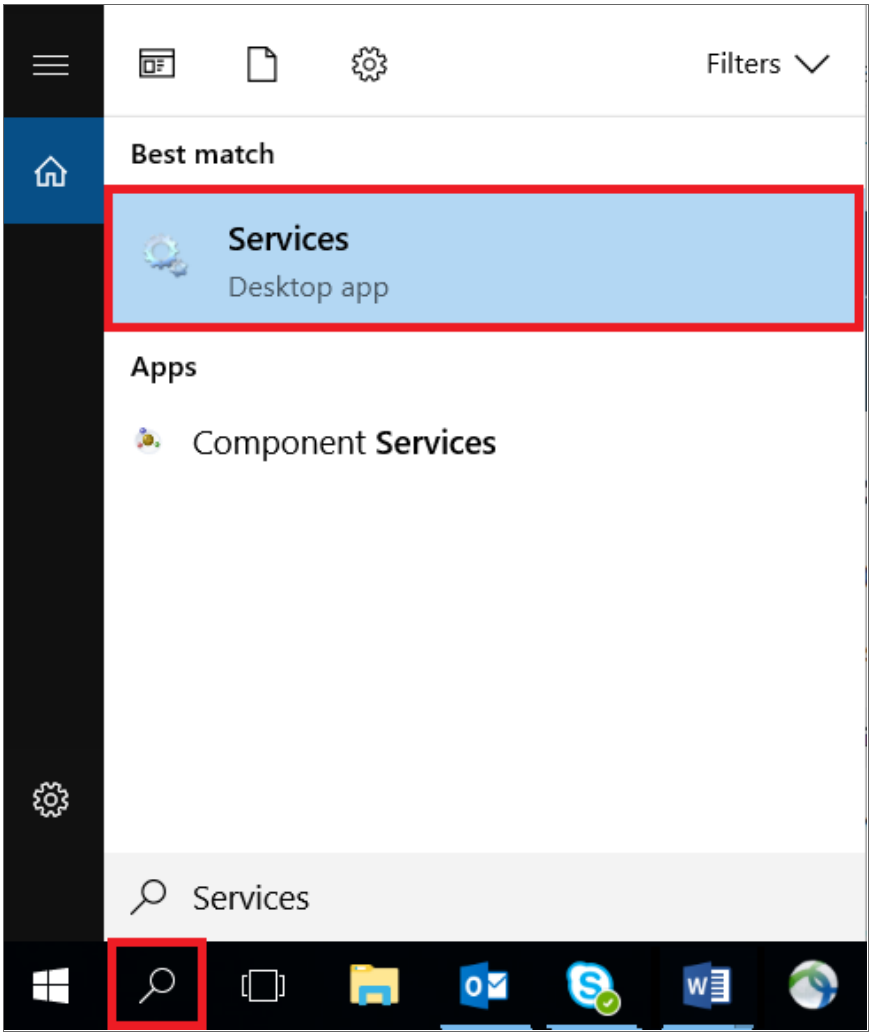
> **NOTE:** ScienceLogic recommends you set all these services to start automatically.

- COM+ Event System
- Remote Access Auto Connection Manager
- Remote Access Connection Manager
- Remote Procedure Call (RPC)
- Remote Procedure Call (RPC) Locator
- Remote Registry
- Server
- Windows Management Instrumentation
- WMI Performance Adapter
- Workstation

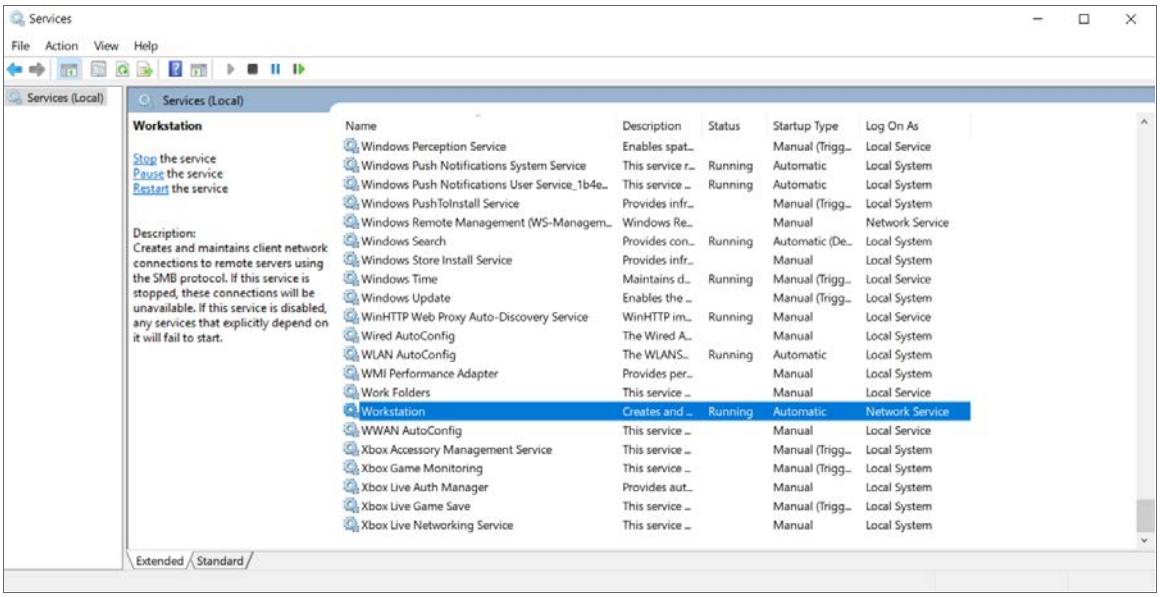
To ensure a service is running, perform the following steps:

1. Click the magnifying glass icon in the bottom-left corner and type "Services" in the ***Search Windows*** field.

2. Click the **Services** Desktop app.

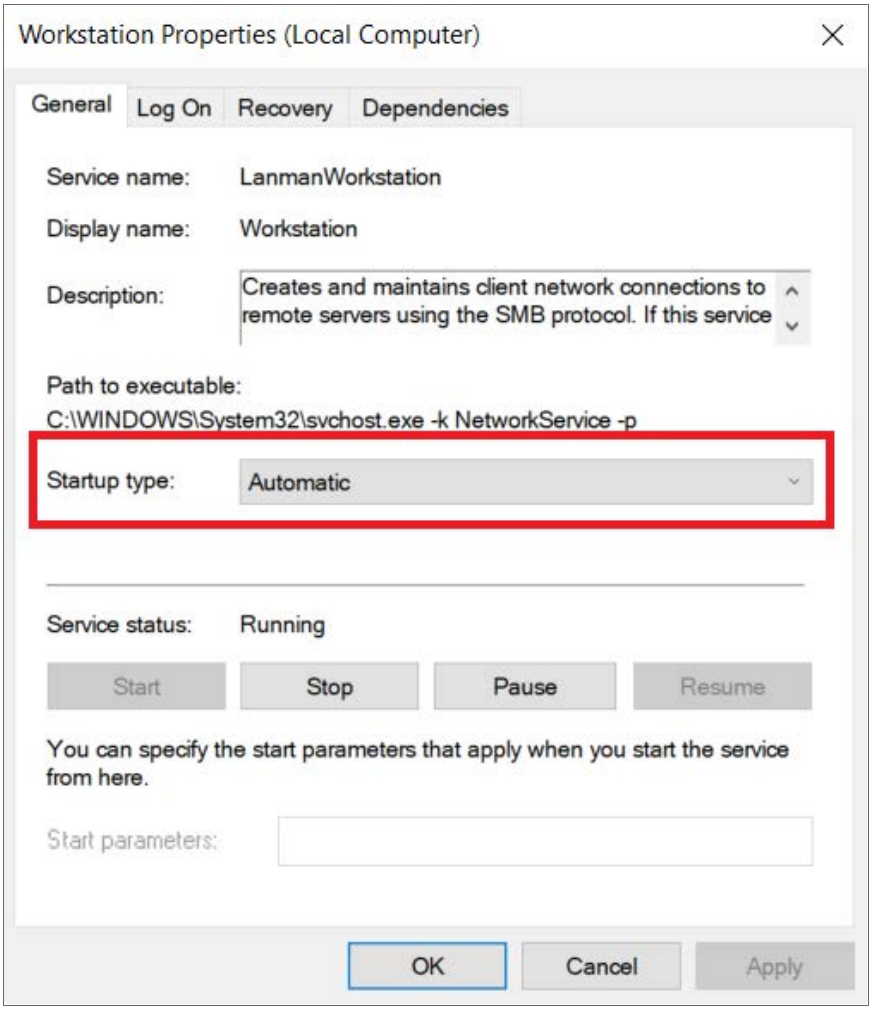


3. From the list of services in the right pane, perform the remaining steps for ***each*** of the services you want to check. This example uses **Workstation**. However, you should check each of the following services:



- COM+ Event System
- Remote Access Auto Connection Manager
- Remote Access Connection Manager
- Remote Procedure Call (RPC)
- Remote Procedure Call (RPC) Locator
- Remote Registry
- Server
- Windows Management Instrumentation
- WMI Performance Adapter
- Workstation

4. Double-click the name of the service. In this example, we double-clicked ***Workstation***.
5. In the **Workstation Properties** dialog box, click the **[General]** tab and complete the following field:



   - ***Startup Type***. Select **Automatic**.
6. Click the **[Apply]** button.
7. If the service has not already started, click the **[Start]** button.
8. Repeat steps 4-7 for each service.

Step 2: Configuring Windows Firewall

To configure Windows Firewall to accept remote WMI requests:

1. Click the magnifying glass icon in the bottom-left corner and type "Command Prompt" in the ***Search Windows*** field.
2. Execute the following two commands in the Command Prompt window:

```
netsh advfirewall firewall set rule group="windows management
instrumentation (wmi)" new enable=yes
```

```
netsh advfirewall firewall set rule group="remote administration" new
enable=yes
```

3. If the result of the second command is "No rules match the specified criteria", run the following two commands:

```
netsh firewall set service remoteadmin enable
```

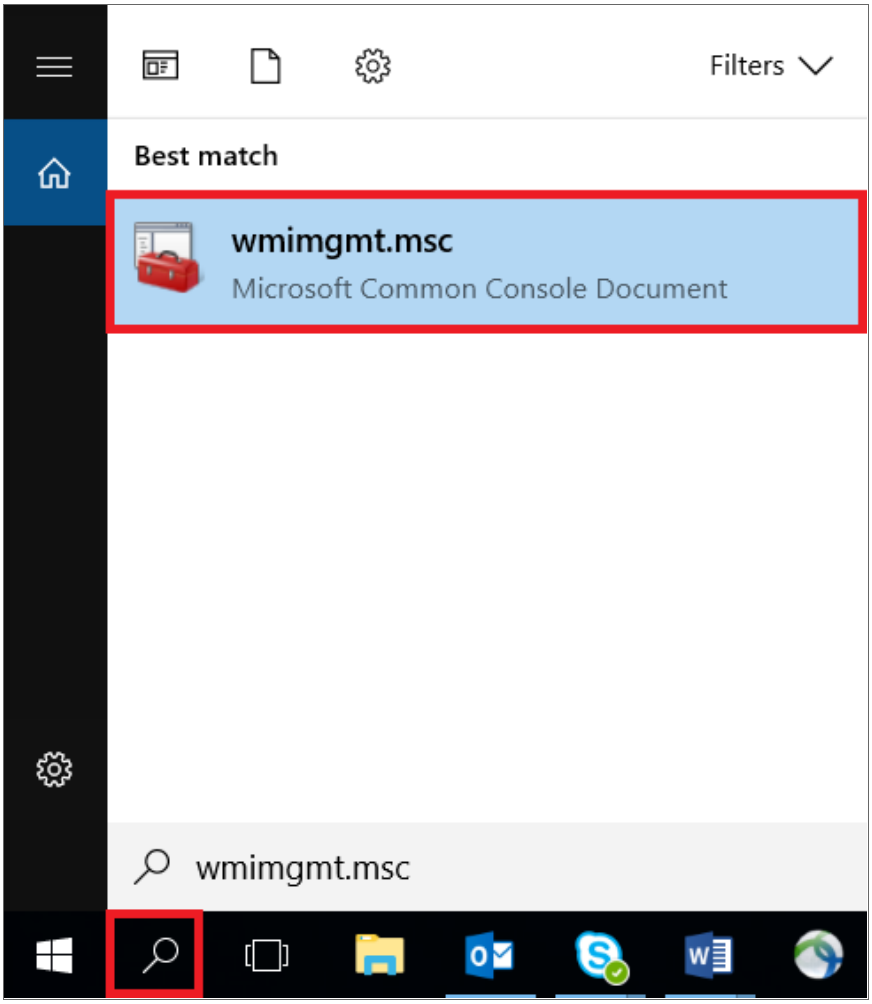
```
netsh advfirewall firewall set rule group="remote administration" new
enable=yes
```

Step 3: Setting the Default Namespace Security

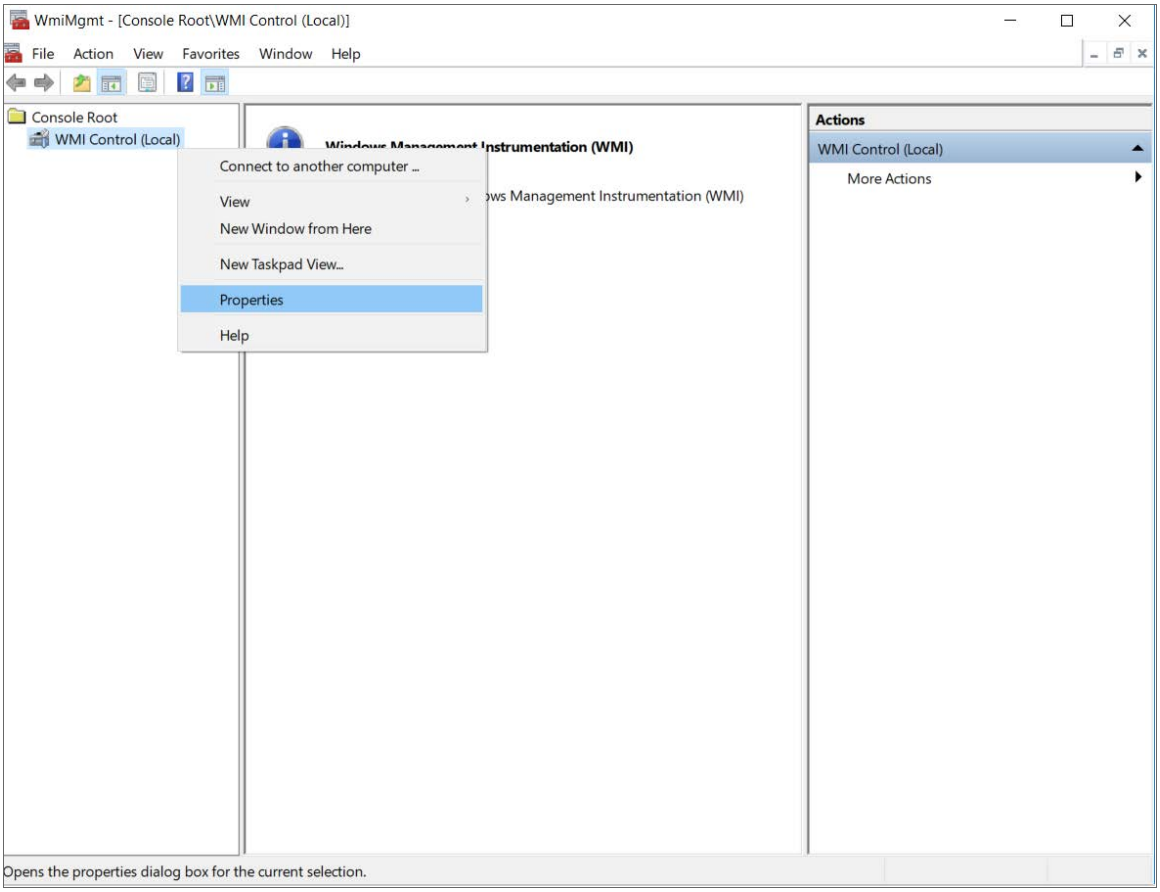
To set the default namespace security, perform the following steps:

1. Click the magnifying glass icon in the bottom-left corner and type "Services" in the ***Search Windows*** field.

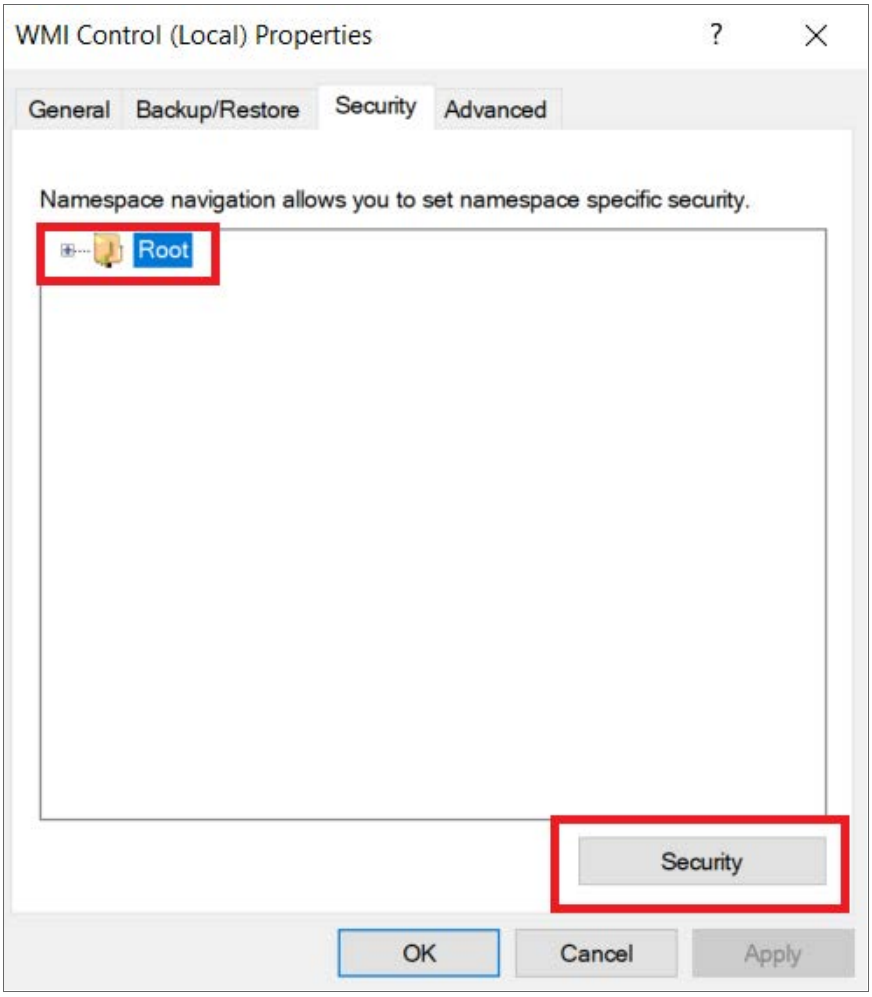
2. Click the **wmimgmt.msc** Microsoft Common Console Document.



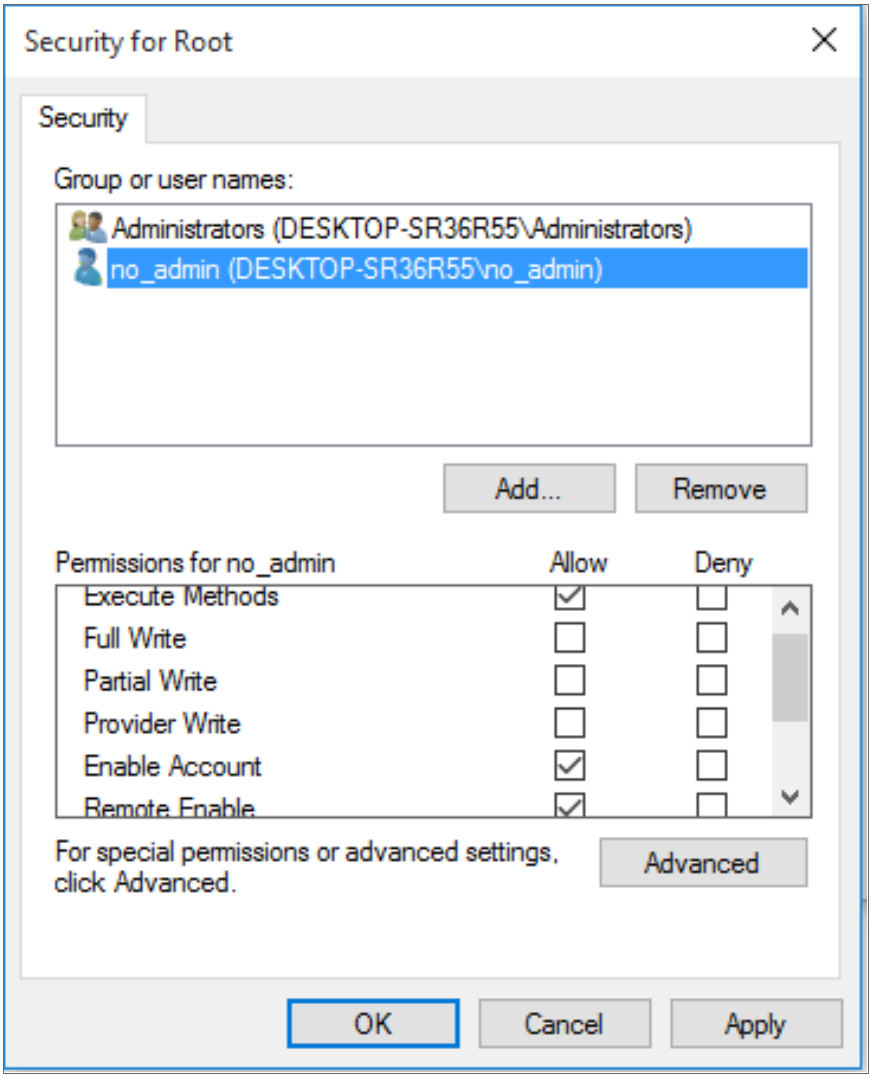
3. In the **WmiMgmt** window, right click **WMI Control (Local)** and select *Properties*.



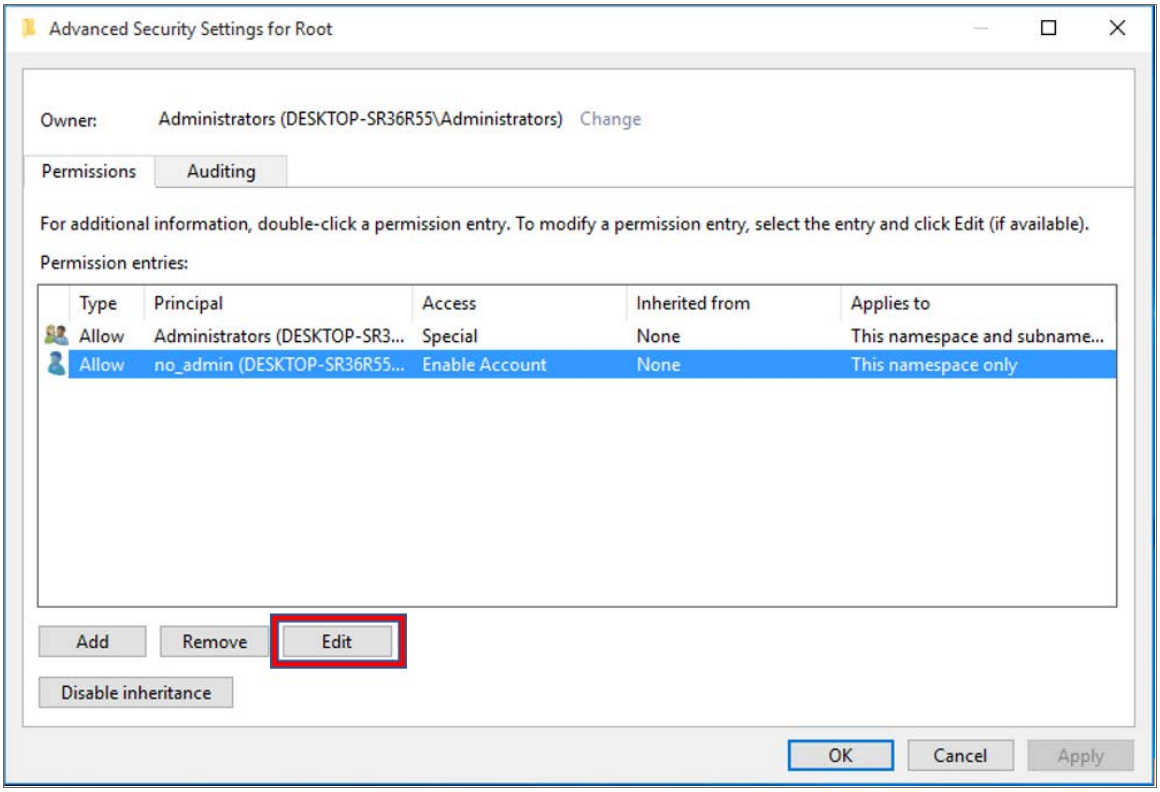
4. In the **WMI Control (Local) Properties** window, click the **[Security]** tab, click **Root**, and then click the **[Security]** button.



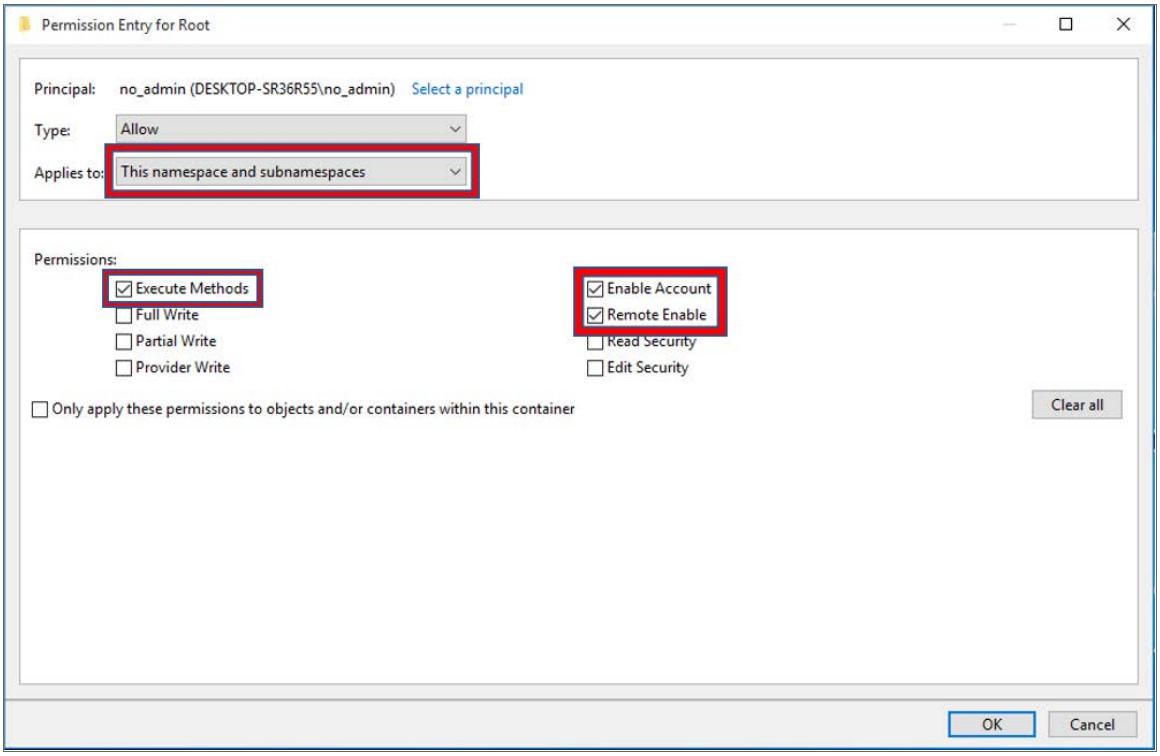
5. In the **Security for Root** window, click **Administrators**, and then click the **[Advanced]** button.



6. In the **Advanced Security Settings for Root** window, click **Administrators**, and then click the **[Edit...]**button.



7. In the **Permission Entry for Root** window, enter the following:



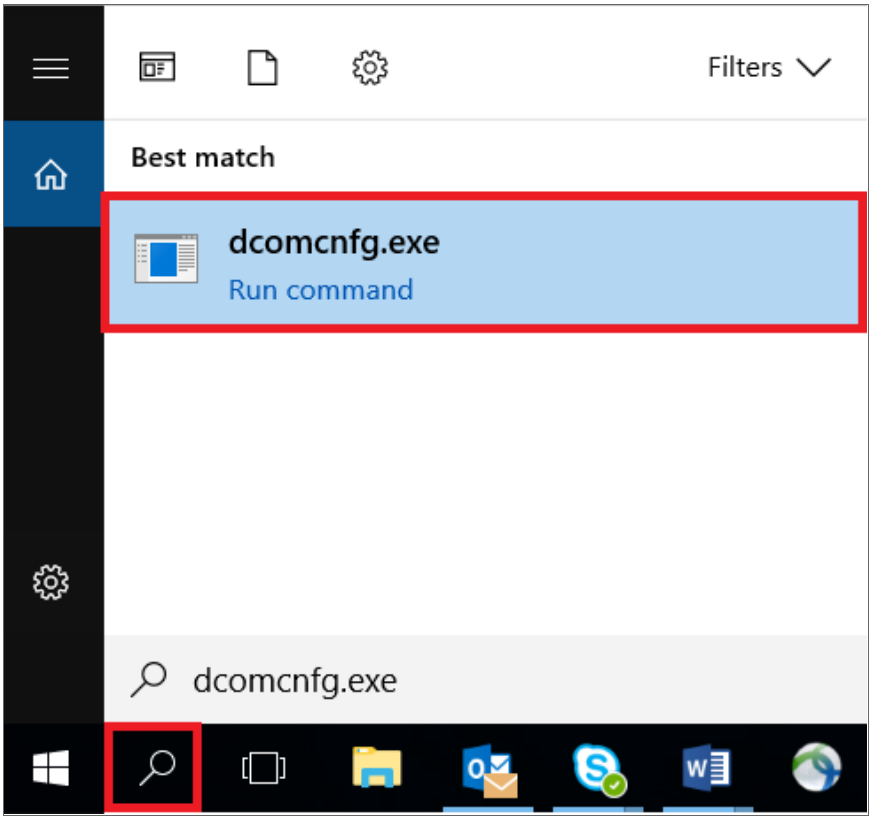
   - ***Type***. Select *Allow*.
   - ***Applies to***. Select *This namespace and subnamespaces*.
   - ***Permissions***. Select the *Execute Methods*, *Full Write*, *Partial Write*, *Provider Write*, *Enable Account*, *Remote Enable*, *Read Security*, and *Edit Security* checkboxes.
8. Click **OK** in this window and the following windows, and then close the **WmiMgmt** window.

Step 4: Setting the DCOM Security Level

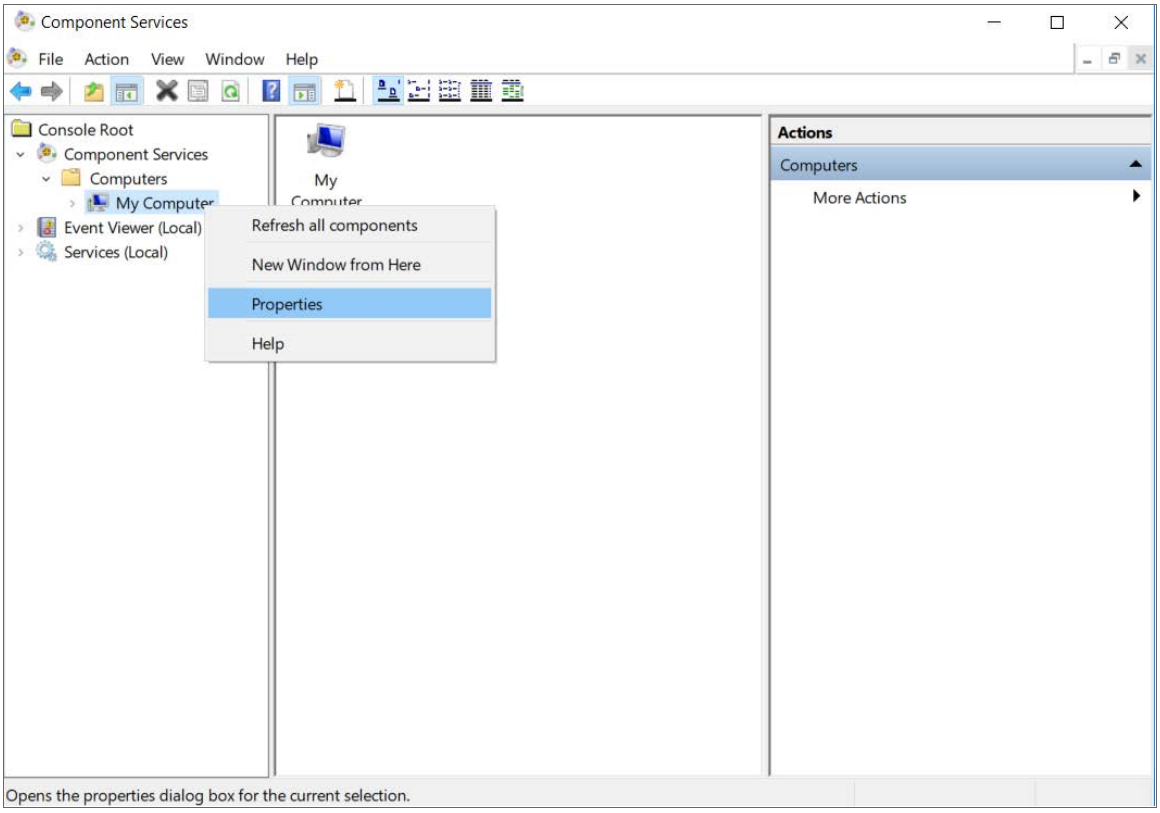
To set the DCOM Security Level, perform the following steps:

1. Click the magnifying glass icon in the bottom-left corner and type "dcomcnfg.exe" in the ***Search Windows*** field.

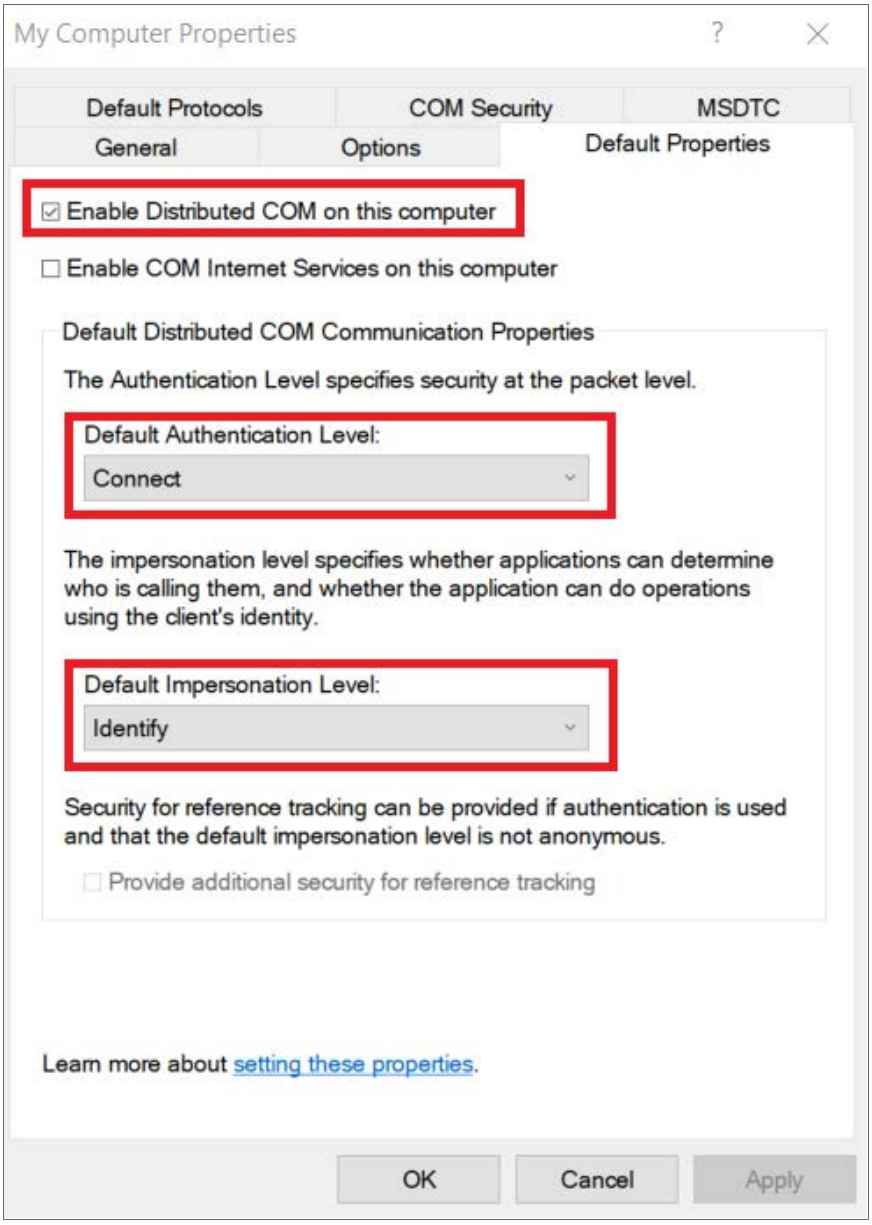
2. Click the **dcomcnfg.exe** command.



3. In the **Component Services** window, expand **Component Services > Computers**, right-click **My Computer**, and then select *Properties*.

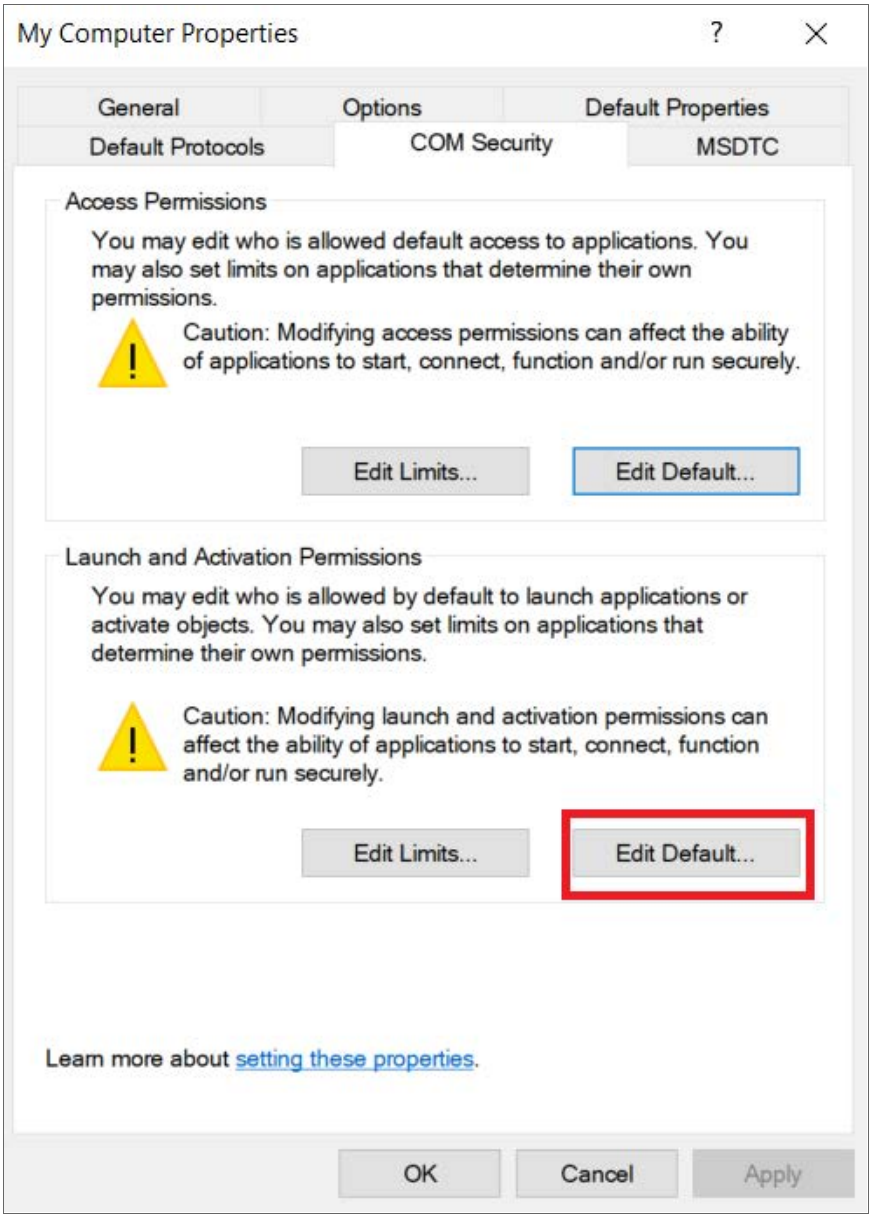


4. In the **My Computer Properties** window, click the **[Default Properties]** tab and then complete the following fields:

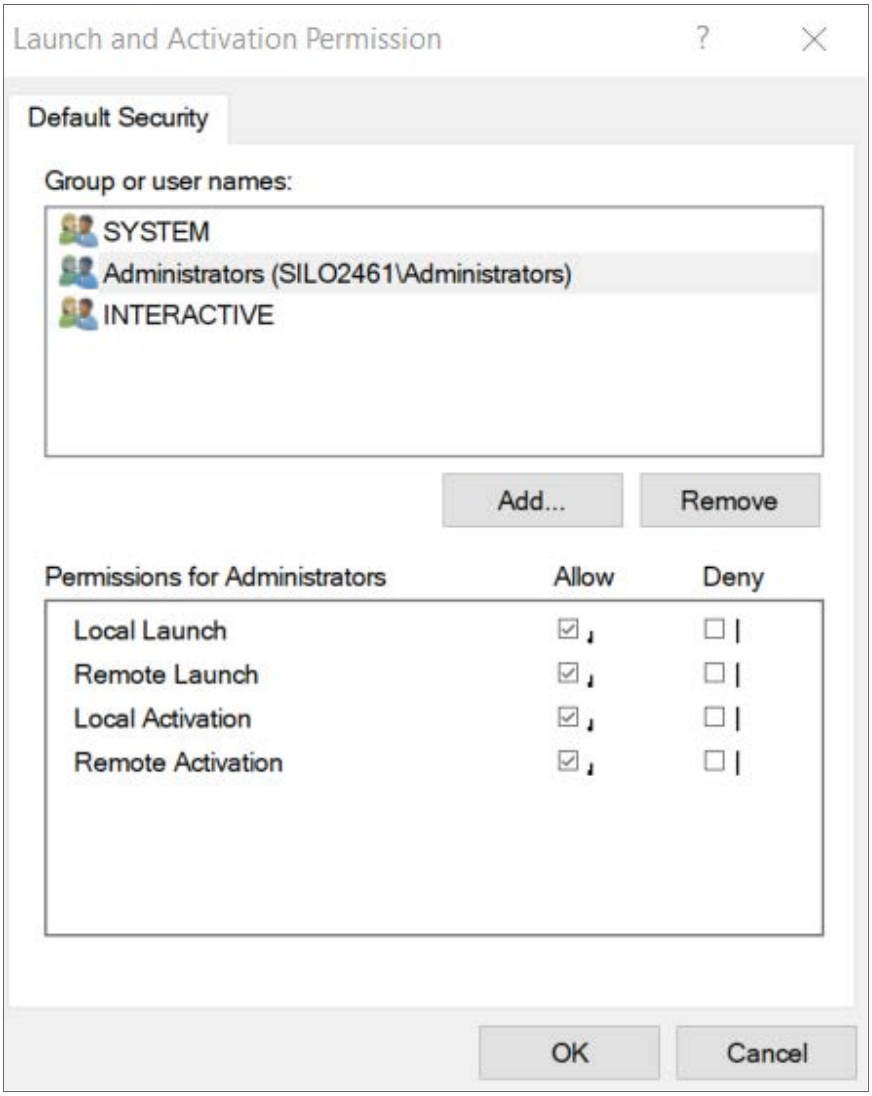


- ***Enable Distributed COM on this computer***. Select this checkbox.
- ***Default Authentication Level***. Select *Connect*.
- ***Default Impersonation Level***. Select *Identify*.

5. In the **My Computer Properties** window, click the **[COM Security]** tab. Under **Launch and Activation Permissions**, click the **[Edit: Default...]** button.



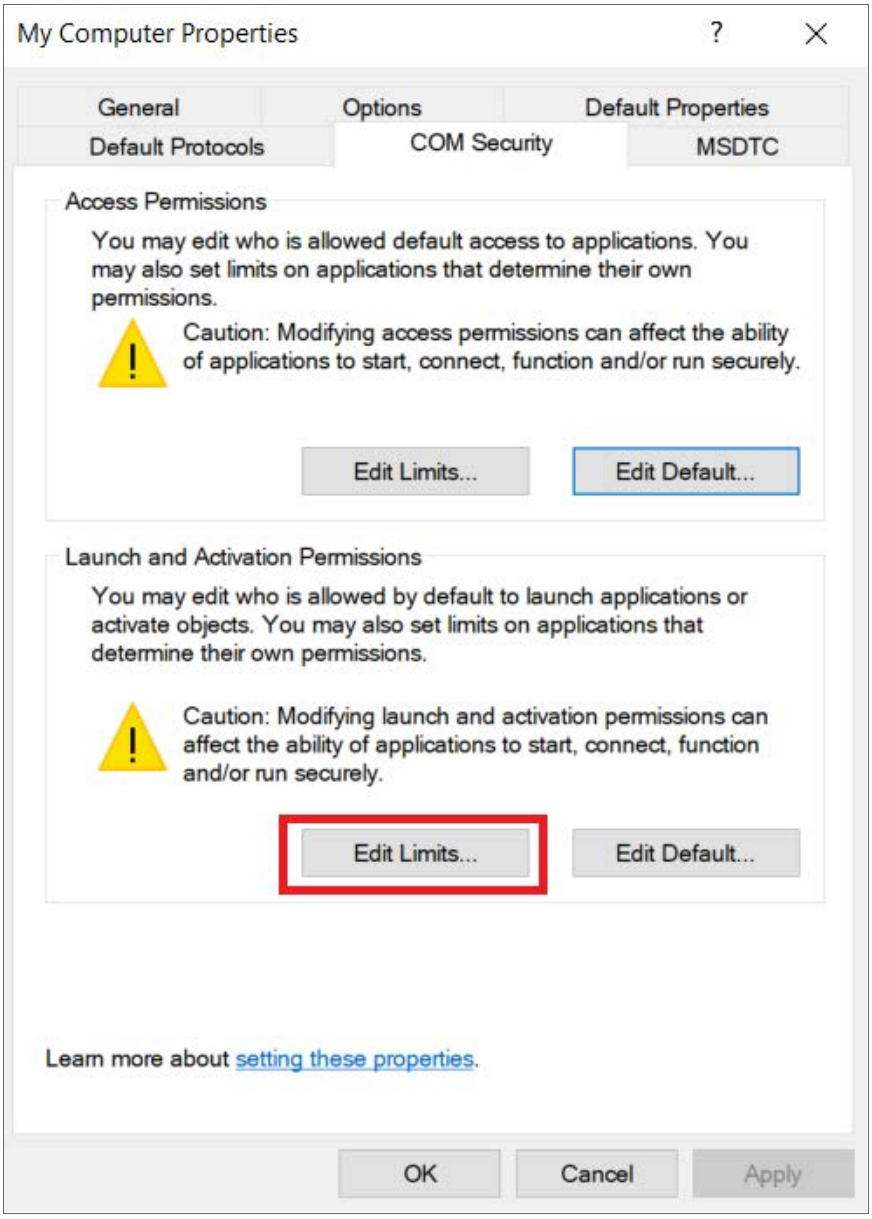
6. In the **Launch and Activation Permission** window, select the following:



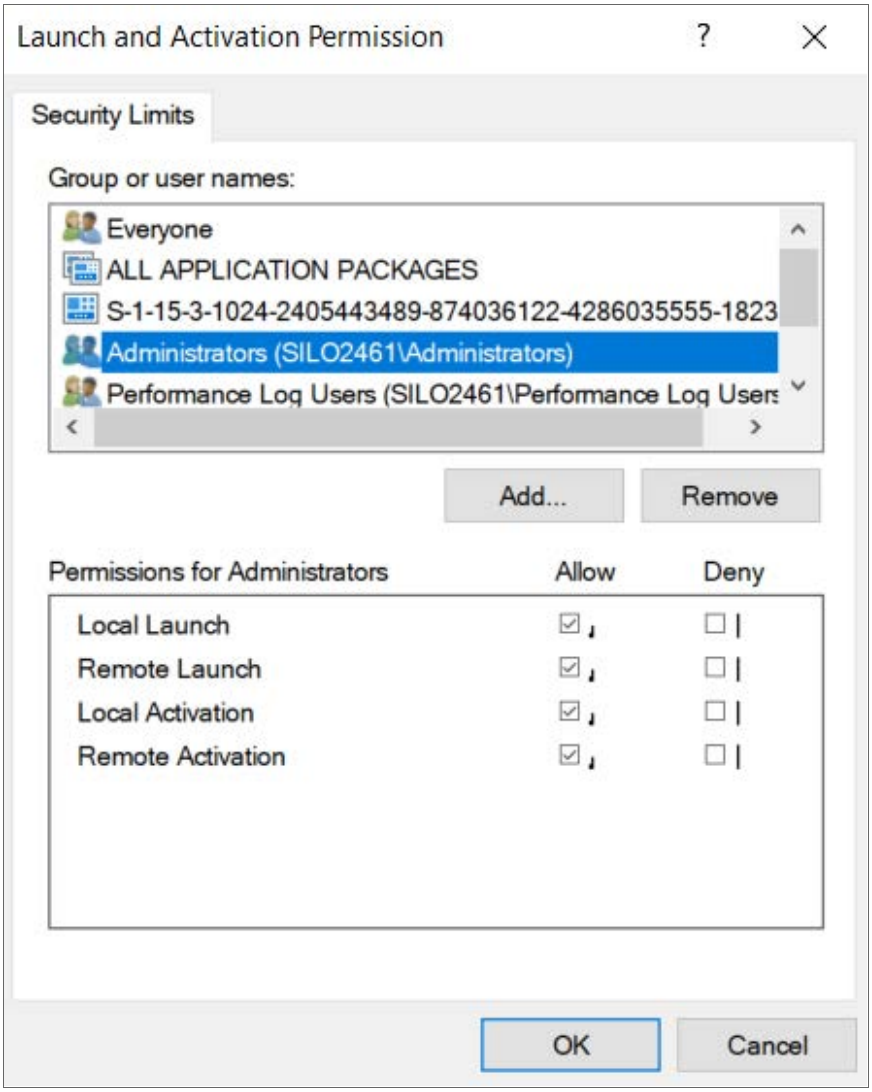
- ***Group or user names***. Select *Administrators*.
- ***Permissions for Administrators***. Set **Local Launch, Remote Launch, Local Activation**, and **Remote Activation**to ***Allow***.

7. Click **[OK]**.

8. In the **My Computer Properties** window, in the **Launch and Activation Permissions** pane, click the **[Edit Limits...]** button.



9. In the **Launch Permission** window, select the following:



   - ***Group or user names***. Select *Administrators*.
   - ***Permissions for Administrators***. Set **Local Launch, Remote Launch, Local Activation**, and **Remote Activation**to ***Allow***.
10. Click **OK** in this window and the following windows, and then close the **Component Services** window.
11. Restart the computer to save the settings.

Step 5: Disabling User Account Control

To monitor a device running Windows 7, 8, or 10, you must perform the following additional steps to disable the User Account Control (UAC) filter for remote logins:

1. Use a text editor such as Notepad to create a new file.

2. Include the following in the file.:

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_
MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System]

"LocalAccountTokenFilterPolicy"=dword:00000001
```

3. Save the file with a name of your choice, like disableUAC.reg, to the directory of your choice. Make sure to save the new file with the .reg suffix.
4. In Windows Explorer, double click on the .reg file to execute it.

Step 6: Configuring a fixed port for WMI

Specific ports must be opened to allow WMI monitoring when there is a separate firewall between the Data Collector and the device. This can occur when the default configuration of the Windows Firewall blocks incoming network traffic for the Windows Management Instrumentation (WMI) connection.

For the WMI connection to succeed, the remote machine must permit incoming network traffic on TCP ports 135, 445, and additional dynamically-assigned ports, typically in the range of 1025 to 5000 and 49152 to 65535.

To set up a fixed port for WMI, see the Microsoft documentation on Setting Up a Fixed Port for WMI.

Configuring Devices for Monitoring with PowerShell

This section describes how to configure Windows Server 2025, 2022, 2019, 2016, 2012, or 2012 R2 for monitoring by Skylar One using PowerShell.

Prerequisites

Before configuring PowerShell, ensure the following:

- Forward and Reverse DNS should be available for the target Windows server from the Skylar One Data Collector. Port 53 to the domain's DNS server should thus be available.
- When using an Active Directory user account as the Skylar One credential, port 88 on the Windows Domain Controller, for the Active Directory domain, should be open for Kerberos authentication.
- If encrypted communication between the Skylar One Data Collector and monitored Windows servers is desired, port 5986 on the Windows server should be open for HTTPS traffic. If unencrypted communications is being used, then port 5985 on the Windows server should be opened for HTTP traffic
- If multiple domains are in use, ensure that they are mapped in the [domain_realm] section of the Kerberos krb5.conf file on the Linux operating system of the Skylar One collector appliance.

Configuring PowerShell

To monitor a Windows Server using PowerShell Dynamic Applications, you must configure the Windows Server to allow remote access from Skylar One. To do so, you must perform the following general steps:

1. *Configure a user account* that Skylar One will use to connect to the Windows Server. The user account can either be a local account or an Active Directory account.

TIP: For ease of configuration, ScienceLogic recommends using an Active Directory account that is a member of the local Administrators group on the Windows Server.

2. *Configure a Server Authentication Certificate* to encrypt communication between Skylar One and the Windows Server.
3. *Configure Windows Remote Management*.
4. Optionally, *configure a Windows server as a Windows Management Proxy*.

NOTE: If you are configuring multiple Windows servers for monitoring by Skylar One, you can apply these settings using a Group Policy.

5. Optionally, you can *increase the number of PowerShell Dynamic Applications that can run simultaneously* against a single Windows server.

Step 1: Configuring the User Account for Skylar One

To enable Skylar One to monitor Windows servers, you must first configure a user account on a Windows Server that Skylar One can use to make PowerShell requests. You will include this user account information when creating the PowerShell credential that Skylar One uses to collect data from the Windows Server.

To configure the Windows Server user account that Skylar One can use to make PowerShell requests, complete one of the following options:

- **Option 1:** *Create an Active Directory Account with Administrator access*
- **Option 2:** *Create a local user account with Administrator access*
- **Option 3:** *Create a non-administrator user account*

TIP: For ease-of-configuration, ScienceLogic recommends creating an Active Directory user account.

After creating your Windows Server user account, depending on your setup and the servers you want to monitor, you might also need to configure the user account for remote PowerShell access to the following server types:

- *Microsoft Exchange Server*
- *Hyper-V Servers*

Option 1: Creating an Active Directory Account with Administrator Access

For each Windows server that you want to monitor with PowerShell or WinRM, you can create an Active Directory account that is a member of the local Administrators group on each server. For instructions, consult Microsoft's documentation. On Windows Domain Controller servers, you can use a domain account that is not in the Domain Administrators group by following the configuration instructions for *Option 3: Creating a Non-Administrator User Account*.

After creating your Active Directory account:

- If you use Skylar One to monitor Microsoft Exchange Servers, you must *configure the user account for remote PowerShell access to Microsoft Exchange Server*.
- If you use Skylar One to monitor Hyper-V Servers, you must *configure the user account for remote PowerShell access to the Hyper-V Servers*.
- Otherwise, ***you can skip the remainder of this section and proceed to Step 3***.

Option 2: Creating a Local User Account with Administrator Access

If you have local Administrator access to the servers you want to monitor and are monitoring Windows Server 2016 or Windows Server 2012, you can alternatively create a local user account with membership in the Administrators group instead of an Active Directory account. For instructions, consult Microsoft's documentation.

WARNING: This method does not work for Windows Server 2008.

After creating your local user account with Local Administrator access:

- If you use Skylar One to monitor Microsoft Exchange Servers, you must *configure the user account for remote PowerShell access to Microsoft Exchange Server*.
- If you use Skylar One to monitor Hyper-V Servers, you must *configure the user account for remote PowerShell access to the Hyper-V Servers*.
- Otherwise, ***you can skip the remainder of this section and proceed to Step 2: Configuring a Server Authentication Certificate***.

Option 3: Creating a Non-Administrator Local User Account

If you do not have Local Administrator access to the servers that you want to monitor with PowerShell or WinRM, or if the monitored Windows server is a Domain Controller that will not be in the local Administrators group, then you must first create a domain user account or create a local user account on the Windows Server. For instructions, consult Microsoft's documentation.

After creating your domain user account or local user account:

- You must configure the Windows servers to allow that non-administrator user access. To do so, ***follow the steps in this section***.
- If you use Skylar One to monitor Microsoft Exchange Servers, you must *configure the user account for remote PowerShell access to Microsoft Exchange Server*.
- If you use Skylar One to monitor Hyper-V Servers, you must *configure the user account for remote PowerShell access to the Hyper-V Servers*.

To configure Windows Servers to allow access by your non-administrator user account:

1. Start a Windows PowerShell shell with ***Run As Administrator*** and execute the following command:

   ```
   winrm configsddl default
   ```

2. On the **Permissions for Default** window, click the **[Add]** button, and then add the non-administrator user account.
3. Select the *Allow* checkbox for the ***Read (Get, Enumerate, Subscribe)*** and ***Execute (Invoke)*** permissions for the user, and then click **[OK]**.
4. Access the Management console. To do this:
   - In Windows Server 2016 and 2012, right-click the Windows icon, click **[Computer Management]**, and then expand **[Services and Applications]**.
5. Right-click on **[WMI Control]** and then select *Properties*.
6. On the **WMI Control Properties** window, click the **[Security]** tab, and then click the **[Security]** button.
7. Click the **[Add]** button, and then add the non-administrator user or group in the **Select Users, Service Accounts, or Groups** dialog, then click **[OK]**.
8. On the **Security for Root** window, select the user o group just added, then in the **Permissions** section at the bottom of the window, select the ***Allow*** checkbox for the *Execute Methods*, *Enable Account*, and *Remote Enable* permissions.
9. Under the **Permissions** section of the **Security for Root** window, click the **[Advanced]** button.
10. In the **Advanced Security Settings** window, double-click on the user account or group you are modifying.
11. On the **Permission Entry** window, in the ***Type*** field, select *Allow*.
12. In the ***Applies to*** field, select *This namespace and subnamespaces*.
13. Select the ***Execute Methods***, ***Enable Account***, and ***Remote Enable*** permission checkboxes, and then click **[OK]** several times to exit the windows opened for setting WMI permissions.
14. Restart the WMI Service from services.msc.

> **NOTE:** To open services.msc, press the Windows + R keys, type "services.msc", and then press Enter.

15. **If this is a member server**, go to the Management console, go to System Tools > Local Users and Groups > Groups. Right-click on ***Performance Monitor Users***, then select *Properties*.

16. **If this is on a domain controller**, go to the Server Manager, go to the **Tools** menu, and click **Active Directory Users and Computers**. Locate the **Builtin** folder. Inside the **Builtin** folder right-click ***Performance Monitor Users***, and then select *Properties*.
17. On the **Performance Monitor Users Properties** window, click the **[Add]** button.
18. In the ***Enter the object names to select*** field, type the non-administrator domain user or group name, and then click **[Check Names]**.
19. Select the user or group name from the list and then click **[OK]**.
20. In the **Performance Monitor Users Properties** window, click **[OK]**.
21. Perform steps 15-20 for the **Event Log Readers** user group and again for the **Distributed COM Users** user group, the R**emote Management Users** user group, and if it exists on the server, the **WinRMRemoteWMIUsers__** user group.
22. If you intend to use encrypted communications between the Skylar One collector host and your monitored Windows servers, each Windows server must have a digital certificate installed that has "Server Authentication" as an Extended Key Usage property. You can create a self-signed certificate for WinRM by executing the following command:

```
$Cert = New-SelfSignedCertificate -CertstoreLocation
Cert:\LocalMachine\My -DnsName "myHost"
```

24. Add an HTTPS listener by executing the following command:

```
New-Item -Path WSMan:\LocalHost\Listener -Transport HTTPS -Address * -
CertificateThumbPrint $Cert.Thumbprint –Force
```

NOTE: This command should be entered on a single line.

25. Ensure that your local firewall allows inbound TCP connections on port 5986 if you are going to use encrypted communications between the Skylar One collector(s) and the Windows server, or port 5985 if you will be using unencrypted communications between the two. You may have to create a new rule on Windows Firewall if one does not already exist.

Optional: Configuring the User Account for Remote PowerShell Access to Microsoft Exchange Server

If you use Skylar One to monitor Microsoft Exchange Servers:

1. Follow the steps in the section *Configuring the User Account for Skylar One*.
2. Add the new user account to the “Server Management” Exchange security group in Active Directory.
3. The user account will then be able to connect to the relevant WinRM endpoint to use cmdlets installed with the Exchange Management Shell. For example, this will give the user account access to the cmdlet “Get-ExchangeServer”.

Optional: Configuring the User Account for Remote PowerShell Access to Hyper-V Servers

To use PowerShell Dynamic Applications to monitor a Hyper-V server, you must:

- Create a user group in Active Directory
- Add the user account you will use to monitor the Hyper-V server to the group
- Set the session configuration parameters on the Hyper-V Server
- Set the group permissions on the Hyper-V Server
- Create a PowerShell credential using the new user account

Creating a User Group and Adding a User in Active Directory

To create a group in Active Directory and add a user:

1. In Active Directory, in the same DC as the Hyper-V host you want to monitor, in the OU called ***Users***, create a group. For example, we called our group ***PSSession Creators***.
2. Add a user that meets the requirements for monitoring a Windows server via PowerShell to the group. This is the user that you will specify in the PowerShell credential.

NOTE: For details on using Active Directory to perform these tasks, consult Microsoft's documentation.

Setting the Session Configuration Parameters and Group Permissions

To set the Session Configuration and the Group Permissions on the Hyper-V Server:

1. Login to the Hyper-V server.
2. Open a PowerShell session. Enter the following command:

```
Set-PSSessionConfiguration -ShowSecurityDescriptorUI -Name
Microsoft.PowerShell
```

3. When prompted, select ***A***.
4. The **Permissions** dialog appears.
5. In the **Permissions** dialog, supply values in the following fields:
   - ***Group or user names***. Select the name of the group you created in Active Directory.
   - ***Permissions for group***. For ***Full Control (All Operations)***, select the *Allow* checkbox.
6. Click the **[OK]** button.

Optional: Configuring the User Account for Access to Windows Failover Cluster

To configure Windows Servers to allow access to your Windows Failover Cluster:

1. Start a Windows PowerShell shell with ***Run As Administrator*** and execute the following command:

```
'Grant-ClusterAccess -User <domain>\<user> -ReadOnly'
```

Step 2: Configuring a Server Authentication Certificate

ScienceLogic highly recommends that you encrypt communications between Skylar One and the Windows Servers you want it to monitor.

If you have created a ***local account on the Windows Server that uses Basic Auth*** and that account will allow communication between Skylar One and the Windows server, the best practice for security is to enable HTTPS to support encrypted data transfer and authentication. To do this, you must configure WinRM to listen for HTTPS requests. This is called configuring an HTTPS listener.

NOTE: For details on configuring WinRM on your Windows servers to use HTTPS, see https://support.microsoft.com/en-us/help/2019527/how-to-configure-winrm-for-https.

The sections below describe how to configure a Server Authentication Certificate on the Windows Server. This is only one task included in configuring an HTTPS listener. However, not all users need to configure a Server Authentication Certificate. You can find out if your Windows computer has a digital certificate installed for Server Authentication by running `'Get-ChildItem -Path Cert:\LocalMachine\My -EKU "*Server Authentication*"'` from a PowerShell command shell.

To support encrypted data transfer and authentication between Skylar One and the servers, one of the following must be true:

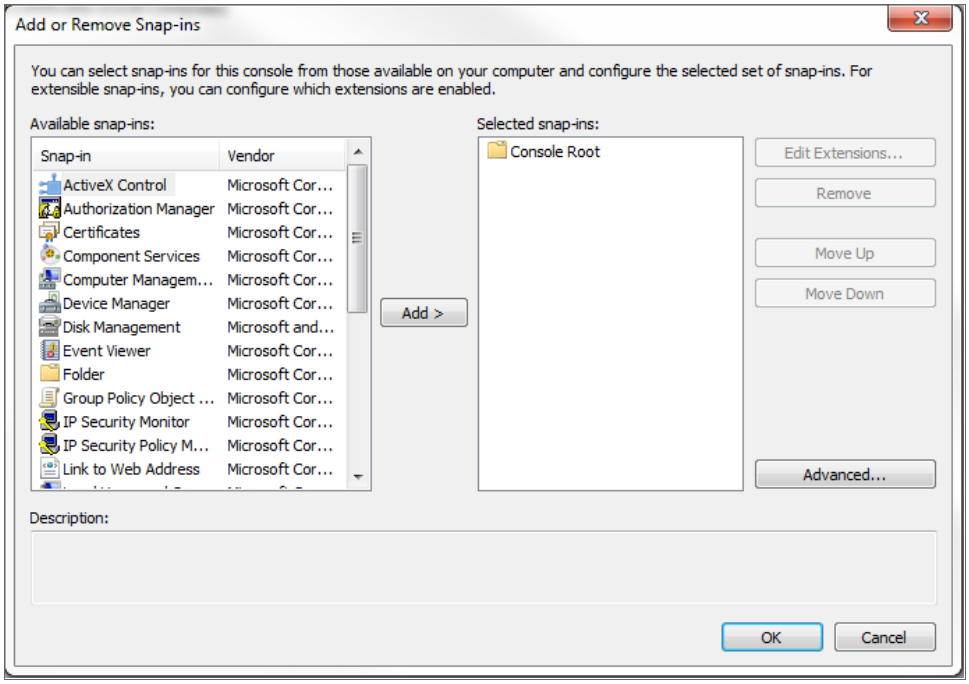
- Your network ***includes a Microsoft Certificate server***. In this scenario, you should work with your Microsoft administrator to get a certificate for your Windows Server instead of configuring a self-signed Server Authentication Certificate. ***You can skip this section and proceed to Step 3***.
- Your network ***does not include a Microsoft Certificate server***. In this scenario, you must configure a self-signed Server Authentication Certificate on the Windows Server that you want to monitor with Skylar One using one of the following methods:
    - **Option 1:** *Use the Microsoft Management Console*.
    - **Option 2:** If your Windows Server includes Windows Software Development Kit (SDK), you can *use the makecert tool*.
    - **Option 3:** If you are running PowerShell 4.0 or later, you can *use the New-SelfSignedCertificate and Export-PfxCertificate commands*.

NOTE: If you have created an Active Directory user account on the Windows Server to allow communication between Skylar One and the server, Active Directory will use Kerberos and AES-256 encryption to ensure secure authentication.

Option 1: Using the Microsoft Management Console to Create a Self-Signed Authentication Certificate

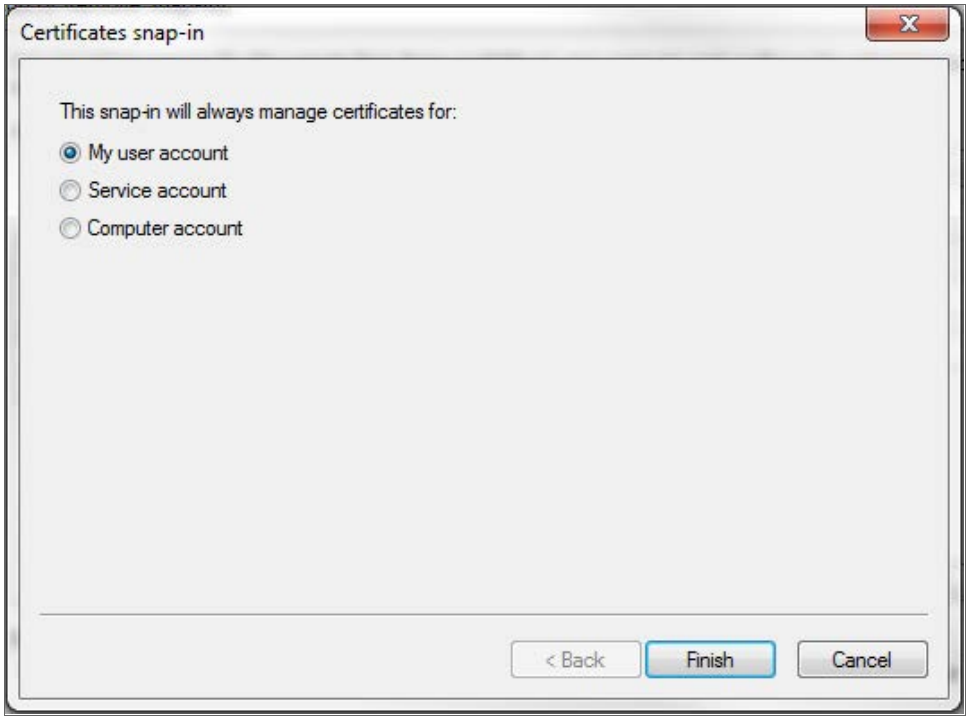
To use the **Microsoft Management Console** to create a self-signed certificate:

1. Log in to the Windows Server that you want to monitor with Skylar One.
2. In the Start menu search bar, enter "mmc" to open a **Microsoft Management Console** window.
3. Select **[File]**, then *Add/Remove Snap-Ins*. The **Add or Remove Snap-ins** window is displayed:



4. In the ***Available snap-ins*** list, select *Certificates*.

5. Click the **[Add >]** button. The **Certificates snap-in** window is displayed:



6. Select *Computer account*.
7. Click the **[Next >]** button.
8. Click the **[Finish]** button.
9. In the **Add or Remove Snap-ins** window, click the **[OK]** button.
10. In the left pane of the **Microsoft Management Console** window, navigate to Console Root > Certificates (Local Computer) > Personal.
11. Right-click in the middle pane and select *All Tasks* > *Request New Certificate...*. The **Certificate Enrollment** window is displayed.
12. Click the **[Next]** button. The **Select Certificate Enrollment Policy** page is displayed.
13. Select *Active Directory Enrollment Policy*.
14. Click the **[Next]** button. The **Request Certificates** page is displayed.
15. Select the ***Computer*** checkbox.
16. Click the **[Enroll]** button.
17. After the certificate is installed, click the **[Finish]** button.

Option 2: Using the MakeCert Tool to Create a Self-Signed Authentication Certificate

If your Windows system includes Windows Software Development Kit (SDK), you can use the MakeCert tool that is included in the kit to create a self-signed certificate. For information on the MakeCert tool, or for

details about creating a self-signed certificate with MakeCert and installing the certificate in the Trusted Root Certificate Authorities store, see the Microsoft documentation.

Option 3: Using PowerShell Commands to Create a Self-Signed Authentication Certificate

If your Windows system includes PowerShell 4.0 or later, you can use the following PowerShell commands to create a self-signed certificate:

- You can use the **New-SelfSignCertificate** command to create a self-signed certificate. For information on **New-SelfSignCertificate**, see the Microsoft documentation.
- You can use the **Export-PfxCertificate** command to export the private certificate. For information on the **Export-PfxCertificate**, see the Microsoft documentation.

Step 3: Configuring Windows Remote Management

To provide Skylar One remote access to the Windows Servers you want to monitor, you must configure Windows Remote Management.

> **NOTE:** This step is required regardless of the user account type that Skylar One will use to connect to the Windows Server.

There are three ways to configure Windows Remote Management:

- **Option 1:** *Use the script provided by ScienceLogic*.
- **Option 2:** *Manually perform the configuration*.
- **Option 3:** *Use a group policy*.

Option 1: Using a Script to Configure Windows Remote Management

ScienceLogic provides a PowerShell script in a .zip file in the PowerPack download folder that automates configuration of Windows Remote Management and permissions required for the user account that will be used in the Skylar One credential. The script configures all of the base Windows permissions required, except for opening up Windows Firewall ports for HTTP and/or HTTPS traffic. The configuration performed by the script is useful primarily for running collection with the "Microsoft: Windows Server", "Microsoft: Windows Server Event Logs", and "Microsoft: SQL Server Enhanced" PowerPacks.

> **NOTE:** The "Microsoft: SQL Server Enhanced" PowerPack requires further instance-specific permissions. See *Microsoft: SQL Server PowerPack Release Notes* for more information.

To use the PowerShell script, perform the following steps:

1. When you download the "Microsoft: Windows Server" PowerPack from the ScienceLogic Support Center, a .zip file for the ***WinRM Configuration Wizard Script (winrm_configuration_wizard_***

v3.5.ps1) will be in the folder with the PowerPack's EM7PP file.

2. Unzip the downloaded file.
3. Using the credentials for an account that is a member of the Administrator's group, log in to the Windows server you want to monitor. You can log in directly or use Remote Desktop to log in.
4. Copy the PowerShell script named ***winrm_configuration_wizard_v3.5.ps1*** to the Windows server that you want to monitor with Skylar One.
5. Right-click on the PowerShell icon and select ***Run As Administrator***.
6. At the PowerShell prompt, navigate to the directory where you copied the PowerShell script named ***winrm_configuration_wizard_v3.5.ps1***.
7. At the PowerShell prompt, enter the following to enable execution of the script:

```
Set-ExecutionPolicy -ExecutionPolicy Unrestricted -Scope Process -
Force
```

> **NOTE:** The execution policy setting persists only during the current PowerShell session.

8. After the warning text, select Y.

> **NOTE:** If your Windows configuration requires further steps to allow execution of the script, PowerShell will display prompts. Follow the prompts.

9. To run the script with interactive dialogs, enter the following at the PowerShell prompt:

```
.\winrm_configuration_wizard_v3.5.ps1 -user <domain>\<username>
```

> **NOTE:** If you have run the script previously and set HTTPS listeners, make sure you have deleted any previous HTTPS listeners with the following command: `winrm delete winrm/config/Listener?Address=*+Transport=HTTPS`

The user account you wish to use for Skylar One collection must be specified with the `-user` command-line argument regardless of other arguments used. You can obtain the full help for the PowerShell configuration script by entering the following:

```
help .\winrm_configuration_wizard_v3.5.ps1 -full
```

> **IMPORTANT:** This command works only on Windows servers, and not on Windows 11 workstations.

The most common way to run the script is silently:

```
.\winrm_configuration_wizard_v3.5.ps1 -user <domain>\<username> -
silent
```

NOTE: If you have multiple certificates installed on your server, running the script with the `-silent` flag will by default use the first certificate it encounters for your HTTP/HTTPS listeners. To set a specific certificate, run the script without the `-silent` flag and use the WinRM Installation Wizard.

You can use `"-dry_run"` command-line argument to record all PowerShell commands that would have made changes to the Windows system. These commands will be logged in the silo_winrm_ config.log file with the prefix: `"Dry Run: '<COMMAND>'"`.

```
.\winrm_configuration_wizard_v3.5.ps1 -user <domain>\<username> -
dry_run
```

You can also use `"-permissions_only"` command-line argument to run the following functions: `AddWinRMRights, AddUserToGroups, SetWMIPermissions, SetRegistryPermissions, SetSecurityRegKeyReadPermission`.

```
.\winrm_configuration_wizard_v3.5.ps1 -user <domain>\<username> -
permissions_only
```

10. If you start the script without using the `-silent` command-line argument, the **WinRM Installation Wizard** modal appears. Click **[OK]**.
11. The **Windows Account Type** modal appears. Select the appropriate choice for your environment.
12. The **Set Encryption Policy** modal appears. Select the appropriate choice for your environment.
    - ***Click YES to us only encrypted data***. Click Yes to configure an HTTPS listener for using encrypted communications between the Skylar One collectors and the Windows server. Setting up an HTTPS listener requires a digital certificate with Server Authentication EKU to be available on the server.
    - ***Click NO to allow unencrypted data***. For communication between Skylar One collectors and the Windows server, if unencrypted traffic is allowed, an HTTP listener will be configured for communication.
13. The **Change Max Requests** modal appears. Click **[Yes]**.
14. The **Change IdleTimeout** modal appears. If you would like to change the value of **IdleTimeout**, click **[Yes]**. If you click **[Yes]**, the **Set WinRM IdleTimeout** modal appears. Enter the new value in the field and click **[OK]**.
15. The **Set Ports for WinRM Traffic** modal appears, and it shows the current settings for the HTTP and HTTPS ports. If you want to make a change to these, click **[YES]**; otherwise, click **[NO]** to continue.
16. Choose which port values you would like Skylar One to use when communicating with the Windows server.
17. The **Set HTTPS Thumbprint** modal appears. Enter the information for your certificate thumbprint, which is used to create an HTTPS listener, then click **[OK]**.

> **NOTE:** If the certificate structure for your certificate thumbprint is incomplete or incorrect, an error message appears indicating that the WinRM client cannot process the request. If you think you made an error, click **[OK]** and try to correct it. Otherwise, contact a system administrator for help.

18. The **Confirm Settings** modal appears. If the settings are as you specified, click **[OK]**.
19. The **Complete** modal appears. If the settings are correct, click **[OK]**.
20. Exit the PowerShell session.

Option 2: Manually Configuring Windows Remote Management

To configure a Windows server for monitoring via PowerShell directly, perform the following steps:

1. Log in to the server with an account that is a member of the local Administrators group, or a Domain Administrator's account if on a Windows server with the Domain Controller role installed.
2. Ensure that your local firewall allows inbound TCP connections on port 5986 if you are going to use encrypted communications between the Skylar One Data Collectors and the Windows server, or port 5985 if you will be using unencrypted communications between the two. You may have to create a new rule on Windows Firewall if one does not already exist.
3. Right-click on the PowerShell icon in the taskbar or the **Start** menu, and select *Run as Administrator*.
4. Execute the following command:

```
Get-ExecutionPolicy
```

5. If the output is "Restricted", execute the following command:

```
Set-ExecutionPolicy RemoteSigned
```

6. Enter "Y" to accept.
7. Execute the following command:

```
winrm quickconfig
```

8. Enter "Y" to accept.
9. If you are configuring this Windows server for encrypted communication, execute the following command:

```
winrm quickconfig -transport:https
```

10. Enter "Y" to accept.
11. Execute the following command:

```
winrm get winrm/config
```

The output should look like this (additional lines indicated by ellipsis):

```
Config
  ...
  Client
    ...
    Auth
      Basic = true
      ...
      Kerberos = true
      ...
    ...
  Service
    ...
    AllowUnencrypted = false
    ...
    DefaultPorts
      HTTP = 5985
      HTTPS = 5986
    ...
    AllowRemoteAccess = true
  Winrs
    AllowRemoteShellAccess = true
    ...
```

12. In the Service section, if the parameter ***AllowRemoteAccess*** is set to *false*, execute the following command:

> **NOTE:** This setting does not appear for all versions of Windows. If this setting does not appear, no action is required.

```
Set-Item WSMan:\Localhost\Service\AllowRemoteAccess -value true
```

13. In the Winrs section, if the parameter ***AllowRemoteShellAccess*** is set to *false*, execute the following command:

```
Set-Item WSMan:\Localhost\Winrs\AllowRemoteShellAccess -value true
```

14. If you are configuring this Windows server for unencrypted communication and the parameter ***AllowUnencrypted*** (in the Service section) is set to *false*, execute the following command:

```
Set-Item WSMan:\Localhost\Service\AllowUnencrypted -value true
```

15. If you are configuring this Windows server for unencrypted communication, verify that "HTTP = 5985" appears in the DefaultPorts section.

> **NOTE:** ScienceLogic recommends using encrypted communication, particularly if you are also using an Active Directory account. Using an Active Directory account for encrypted authentication enables you to use Kerberos ticketing for authentication.

16. If you are configuring this Windows server for encrypted communication, verify that "HTTPS = 5986" appears in the DefaultPorts section.
17. If you are using an Active Directory account to communicate with this Windows server and in the Auth section, the parameter ***Kerberos*** is set to *false*, execute the following command:

```
Set-Item WSMan:\Localhost\Service\Auth\Kerberos -value true
```

> **NOTE:** ScienceLogic recommends using an Active Directory account.

18. If you are using a local account to communicate with this Windows server and in the Auth section, the parameter ***Basic*** is set to *false*, execute the following command:

```
Set-Item WSMan:\Localhost\Service\Auth\Basic -value true
```

19. IdleTimeout is set to 7200000 milliseconds (2 hours) by default. If an issue occurs with scheduled PowerShell monitoring and a process remains on a Windows device, it will therefore remain for up to 2 hours before being removed. To reduce the IdleTimeout and have Windows shut down idle WinRM processes after a shorter time period, execute the following command:

```
winrm s winrm/config/winrs '@{IdleTimeout="600000"}'
```

This command will change the timeout to 10 minutes (600000 ms).

> **NOTE:** When changing IdleTimeout, ensure that no other applications or utilities need a higher timeout for WinRM sessions.

Option 3: Using a Group Policy to Configure Windows Remote Management

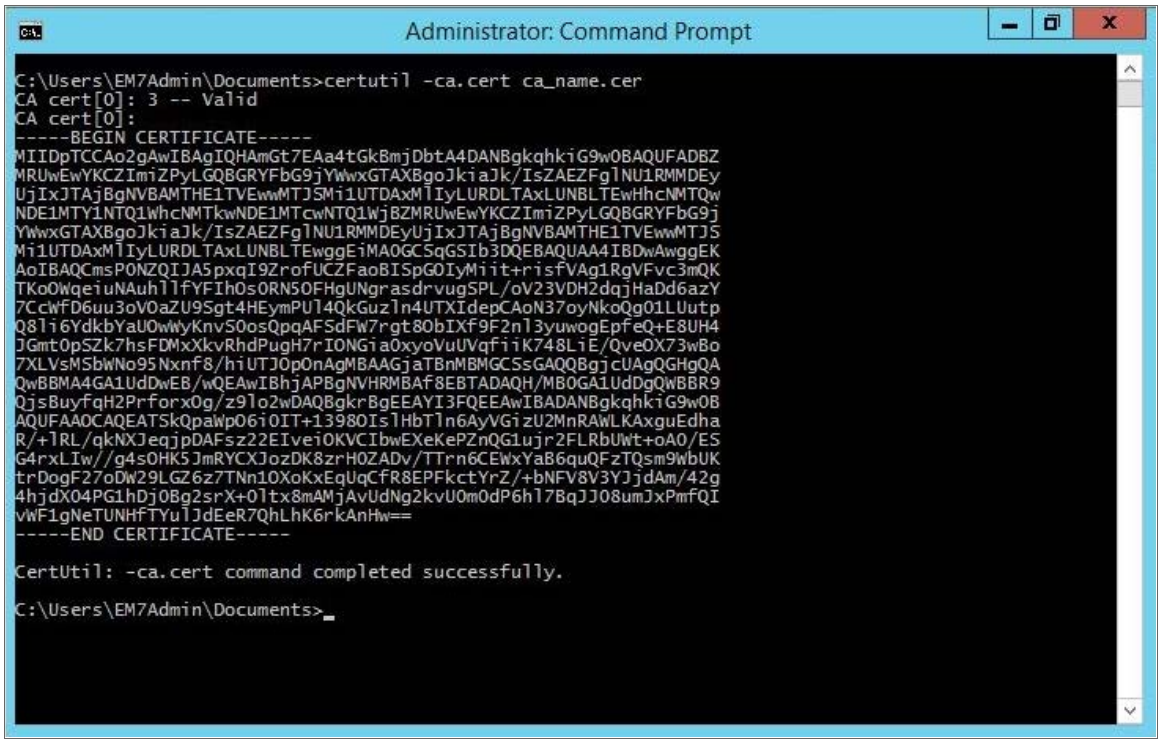
You can use a group policy object (GPO) to configure the following Windows Remote Management settings on Windows Server 2012 or Windows Server 2016:

- A registry key to enable Local Account access to Windows Remote Management
- Firewall rules
- Certificates
- HTTP and HTTPS listeners, including authentication and encryption settings
- Service start and recovery settings

To create the group policy object, perform the following steps:

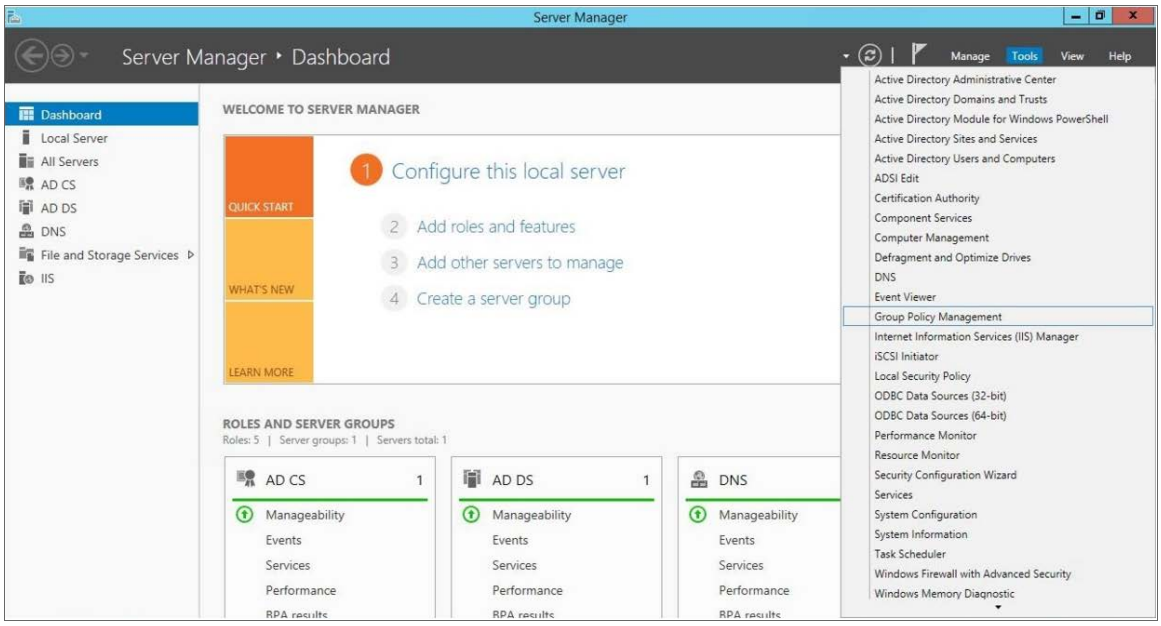
1. Log in to the CA server as an administrator.
2. Right-click on the PowerShell icon in the taskbar and select *Run as Administrator*.
3. At the PowerShell prompt, use the change directory (CD) command to navigate to a folder where you can create new files.
4. Save the root Certification Authority certificate to the local directory by executing the following command:

```
certutil.exe –ca.cert ca_name.cer
```

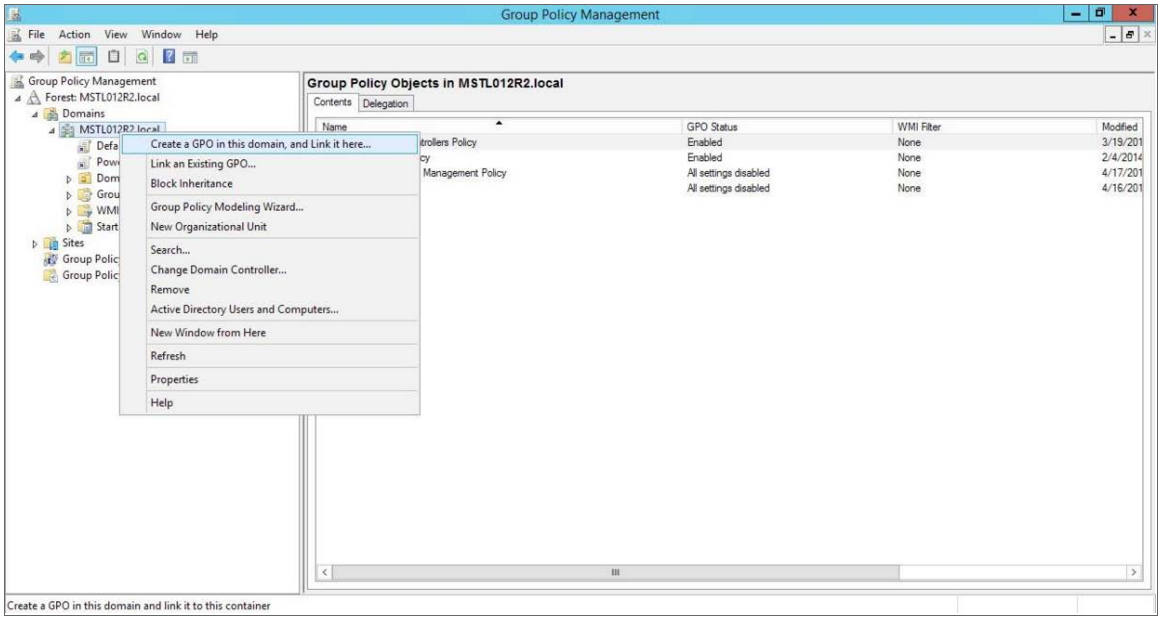


TIP: You will import this certificate into the new group policy in step 21.

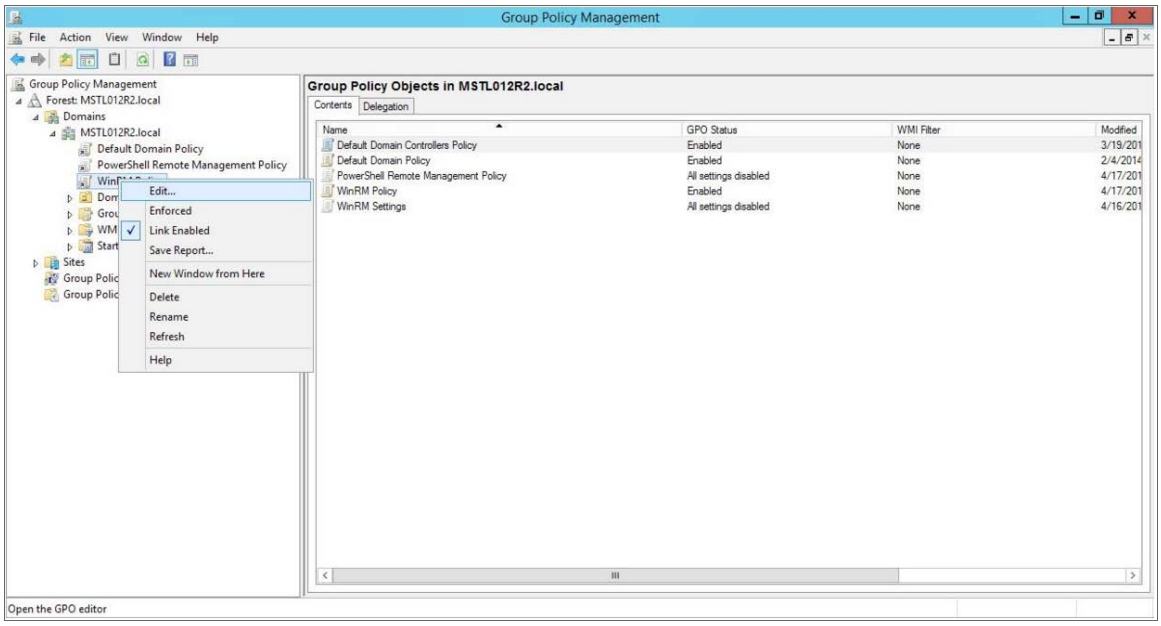
5. Exit the command prompt.
6. Log in to a domain controller in your Active Directory forest and navigate to the System Manager dashboard.
7. Click the **Tools** menu, then select *Group Policy Management*.



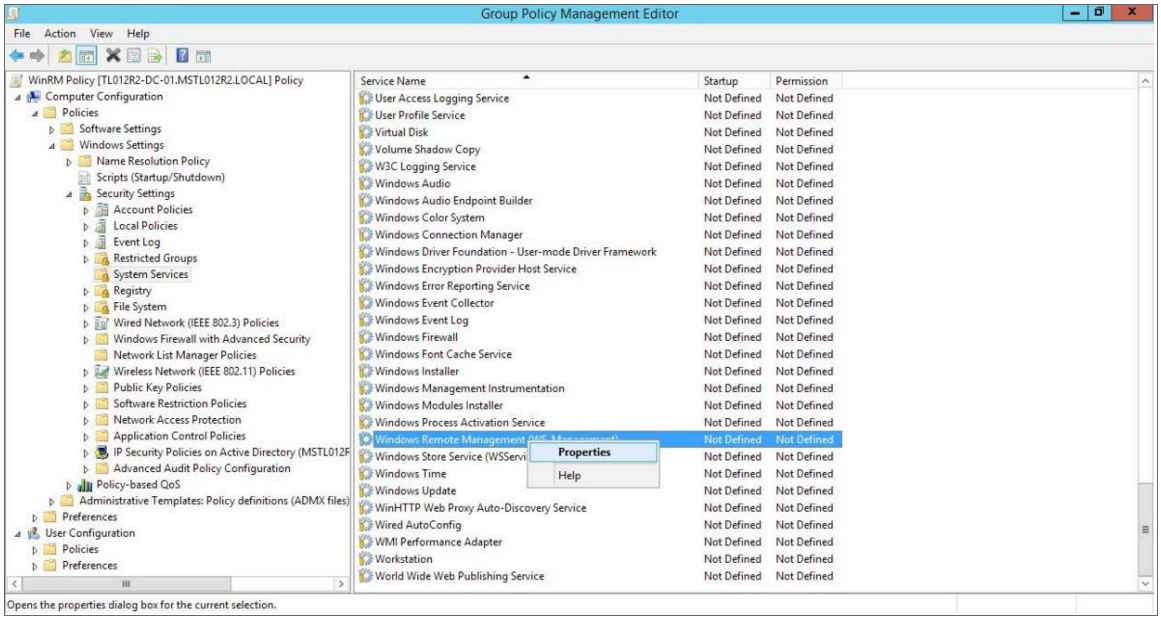
8. On the **Group Policy Management** page, in the left panel, right-click the domain name where you want the new group policy to resideand then select *Create a GPO in this domain and Link it here*.



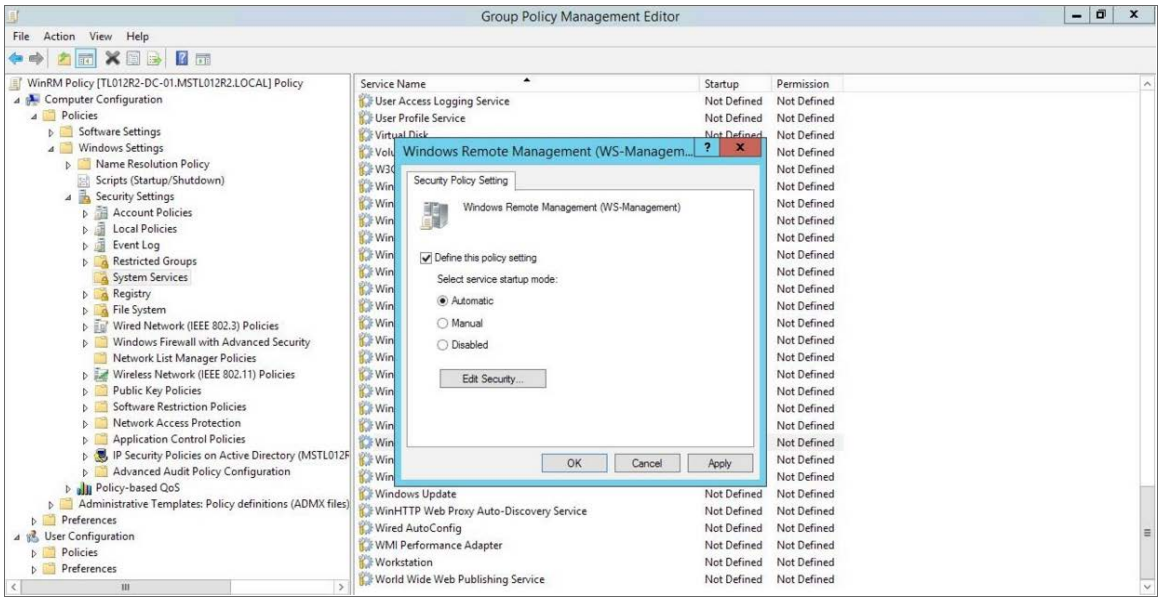
9. In the left panel, right-click the new group policy and select *Edit*. The **Group Policy Management Editor** page for the new Windows Remote Management group policy appears.



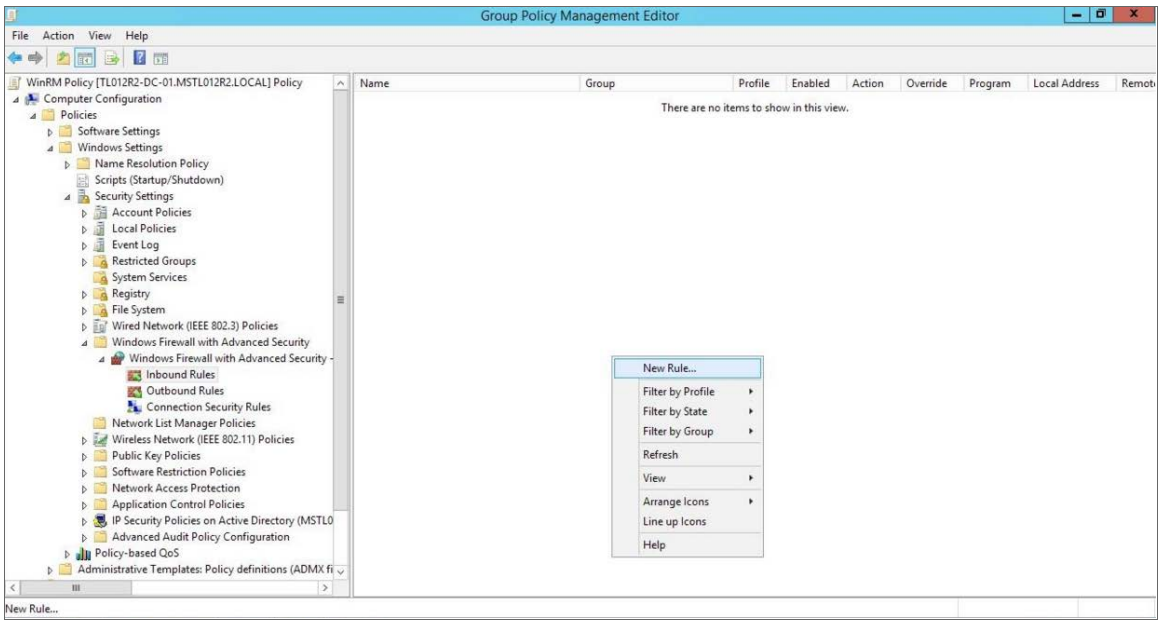
10. In the left panel, navigate to **Computer Configuration > Policies > Windows Settings > Security Settings > System Services**. In the right panel, locate the **Windows Remote Management (WS-Management)** service. Right-click the service, then select *Properties*.



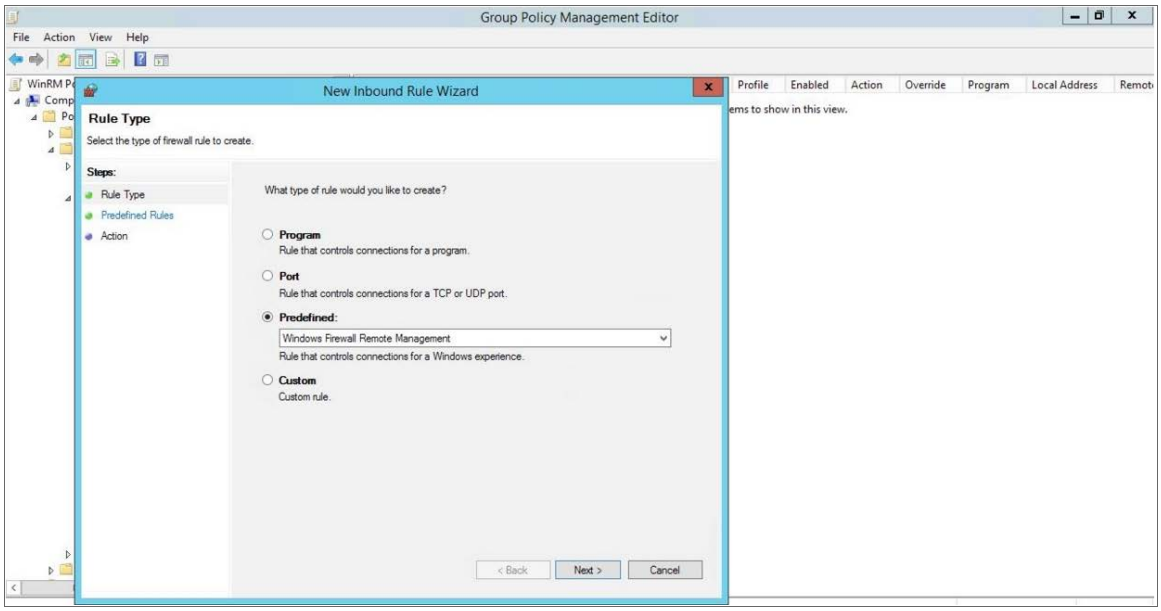
11. The **Windows Remote Management (WS-Management)** modal page appears. Select the **Define this policy setting** check box and the **Automatic** radio button, then click **[OK]**.



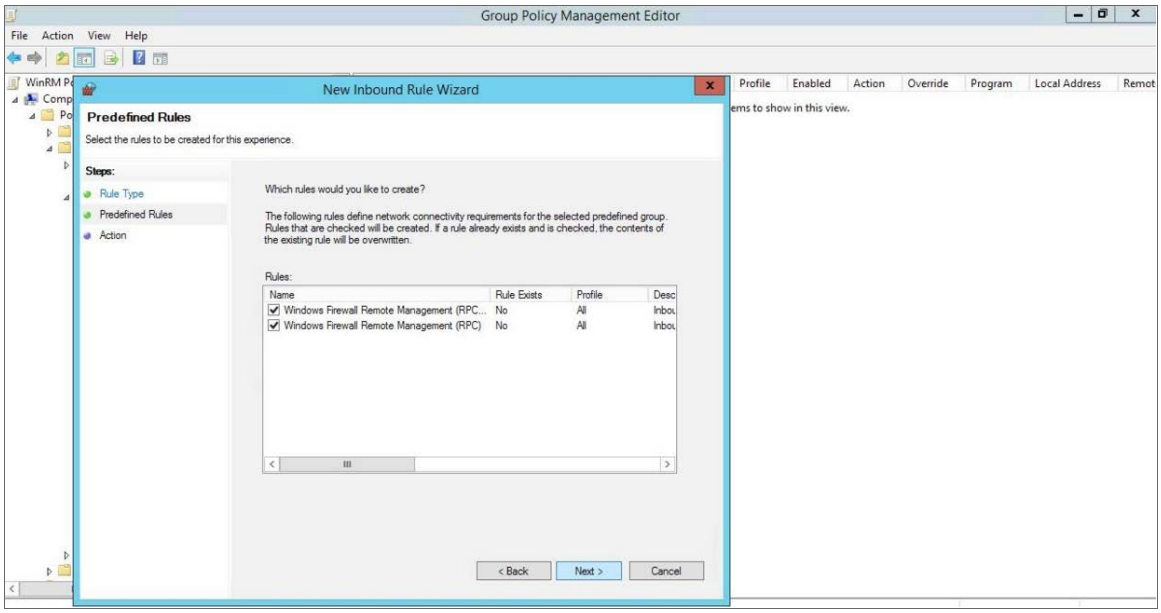
12. In the left panel of the **Group Policy Management Editor** page, navigate to **Computer Configuration > Policies > Windows Settings > Security Settings > Windows Firewall with Advanced Security > Windows Firewall with Advanced Security - LDAP > Inbound Rules**. In the right panel, right-click and select *New Rule*.



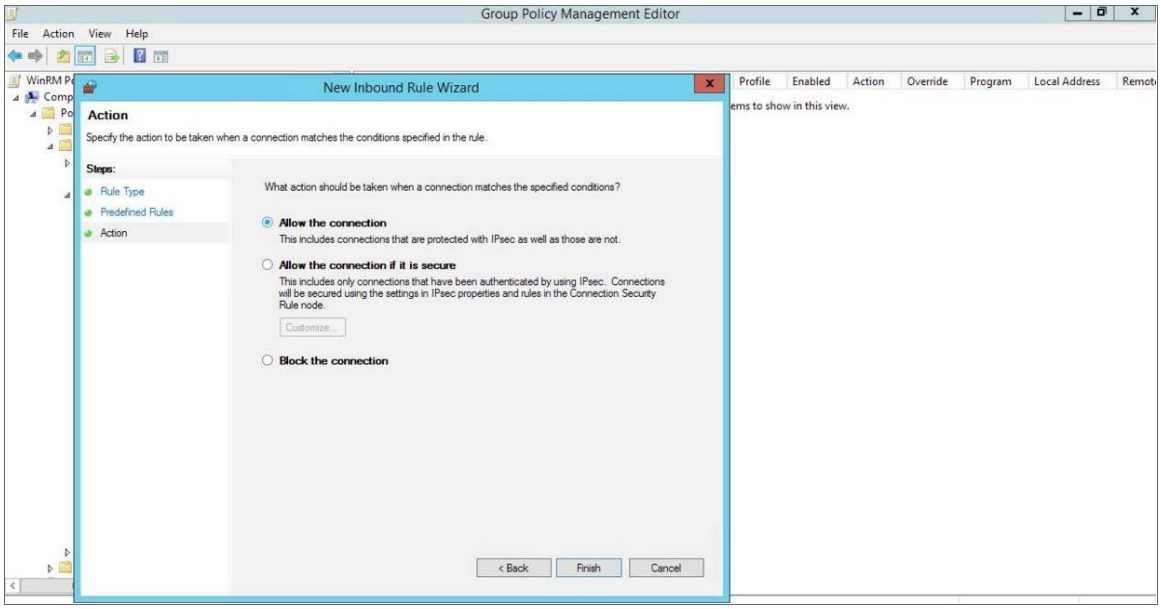
13. The **New Inbound Rule Wizard** modal page appears. Click the **Predefined** radio button, select *Windows Firewall Remote Management* from the list, and then click **[Next]**.



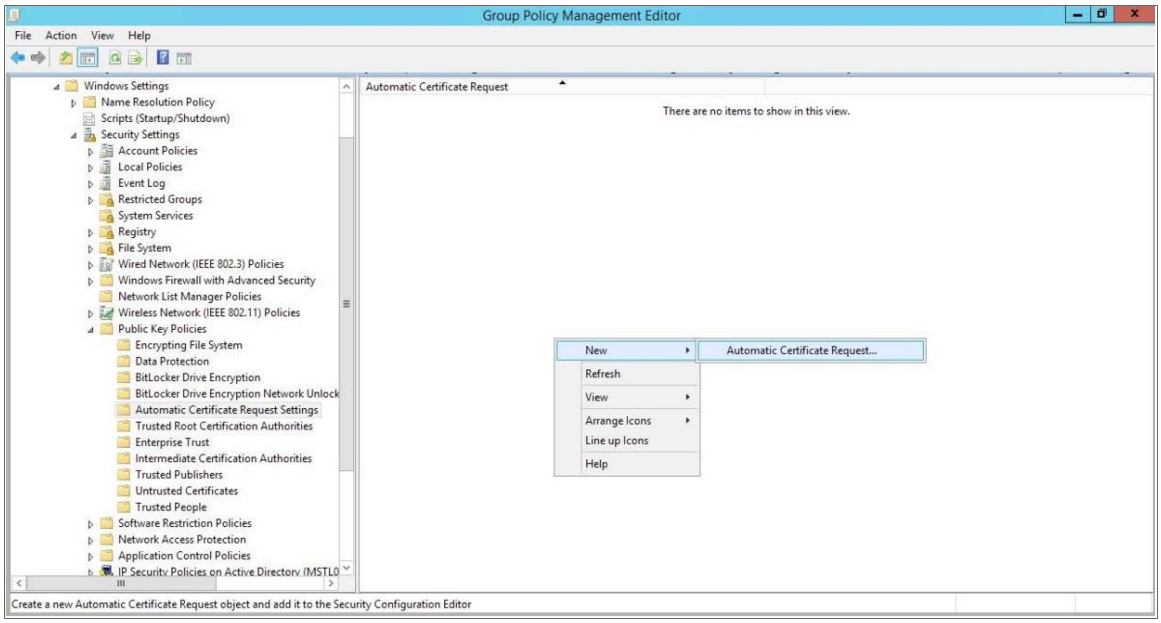
14. Select the *Windows Firewall Remote Management (RPC)* and *Windows Firewall Remote Management (RPC-EPMAP)* check boxes, then click **[Next]**.



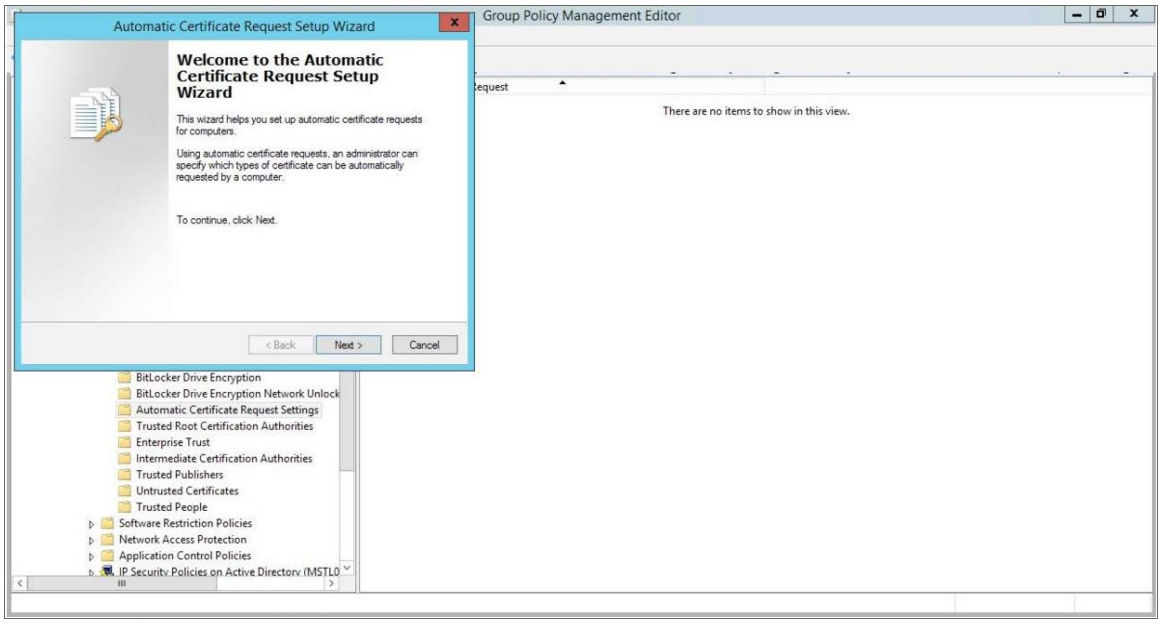
15. Select the *Allow the connection* radio button, then click **[Finish]**.



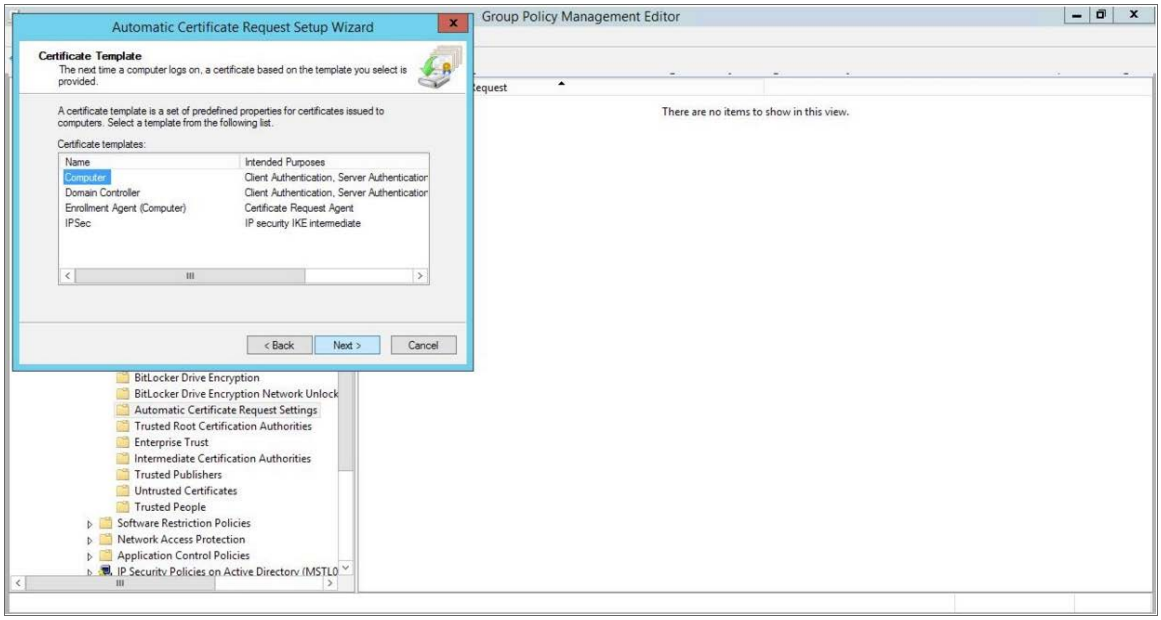
16. In the left panel of the **Group Policy Management Editor** page, navigate to **Computer Configuration > Policies > Windows Settings > Security Settings > Public Key Policies > Automatic Certificate Request Settings**. In the right panel, right-click and select *New > Automatic Certificate Request*.



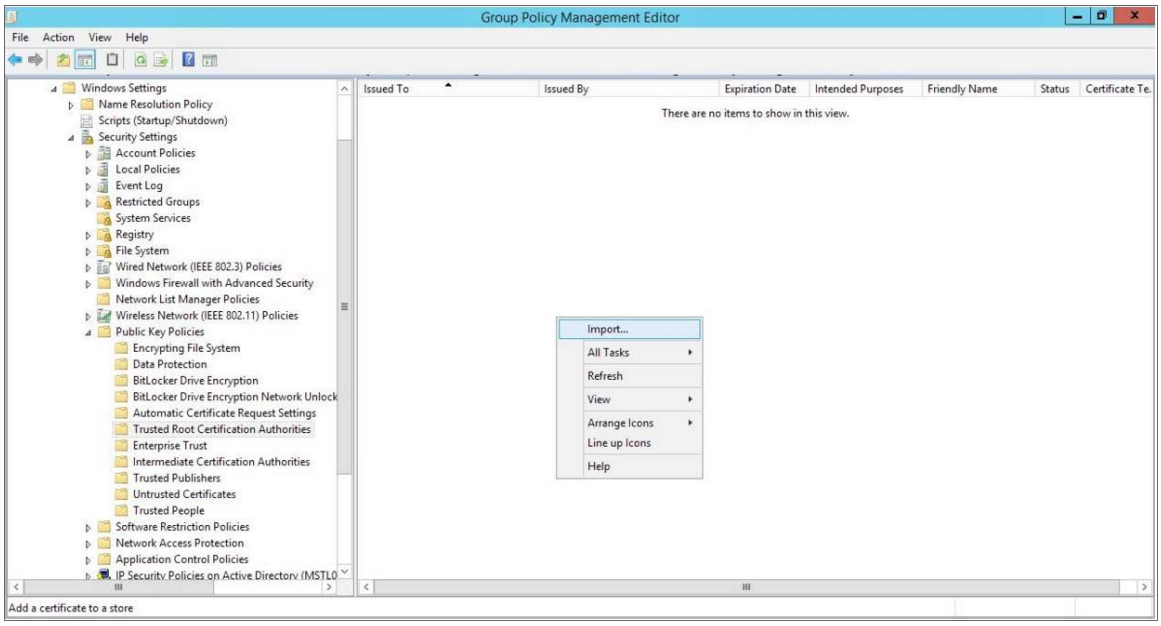
17. The **Automatic Certificate Request Setup Wizard** modal page appears. Click **[Next]**.



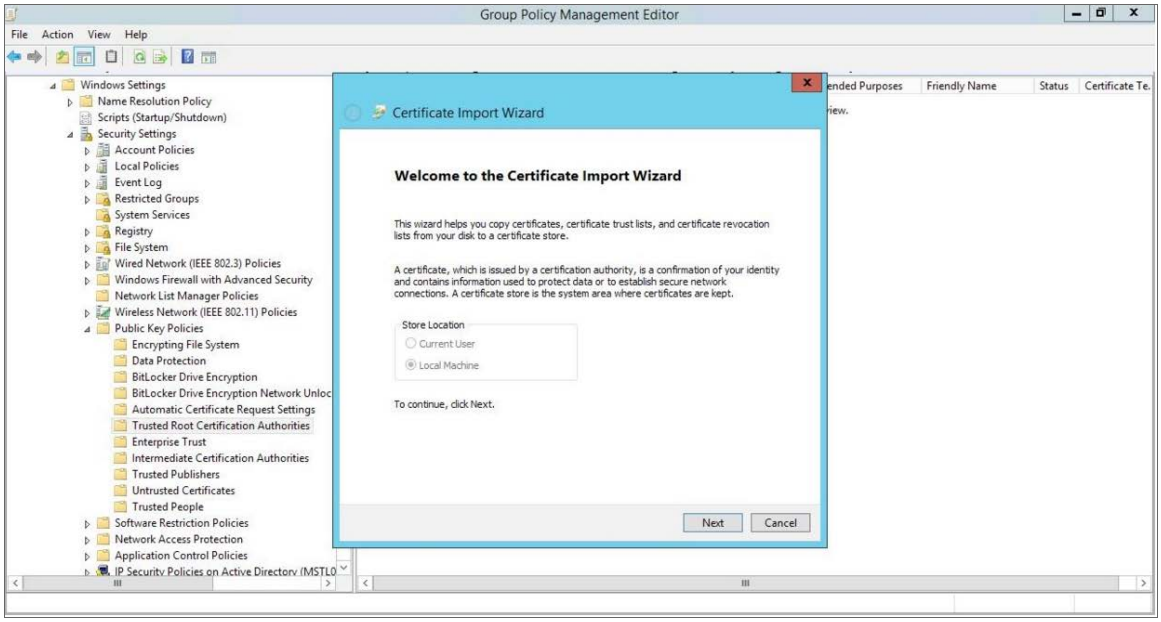
18. Select the *Computer* certificate template. Click **[Next]**, and then click **[Finish]**.



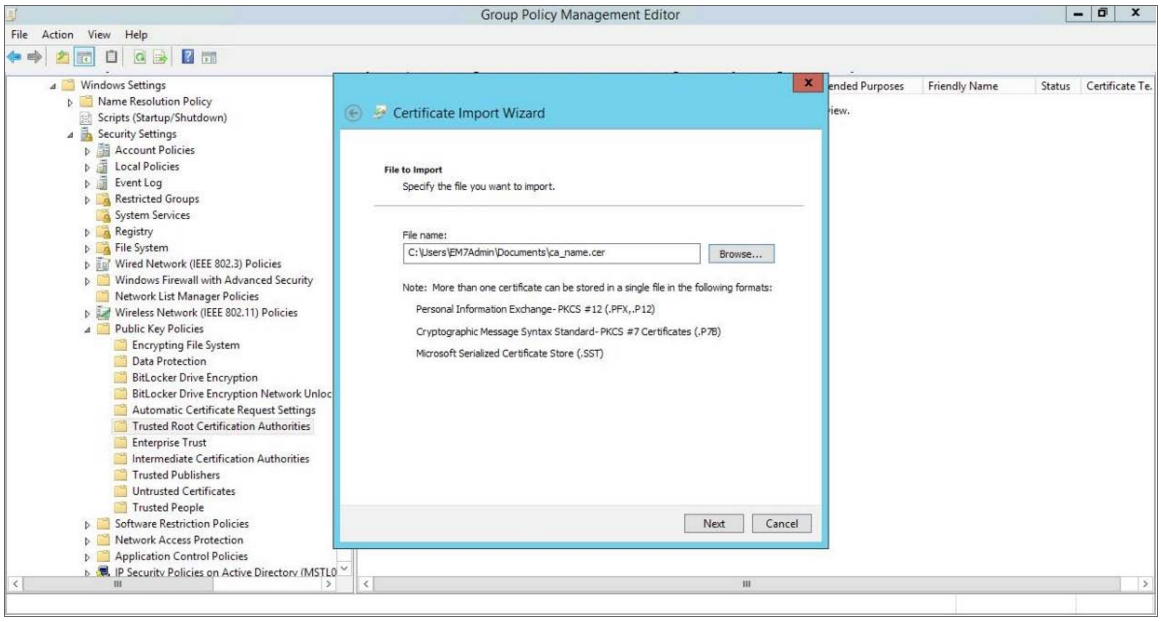
19. In the left panel of the **Group Policy Management Editor** page, navigate to **Computer Configuration > Policies > Windows Settings > Security Settings > Public Key Policies > Trusted Root Certification Authorities**. In the right panel, right-click and select *Import*.



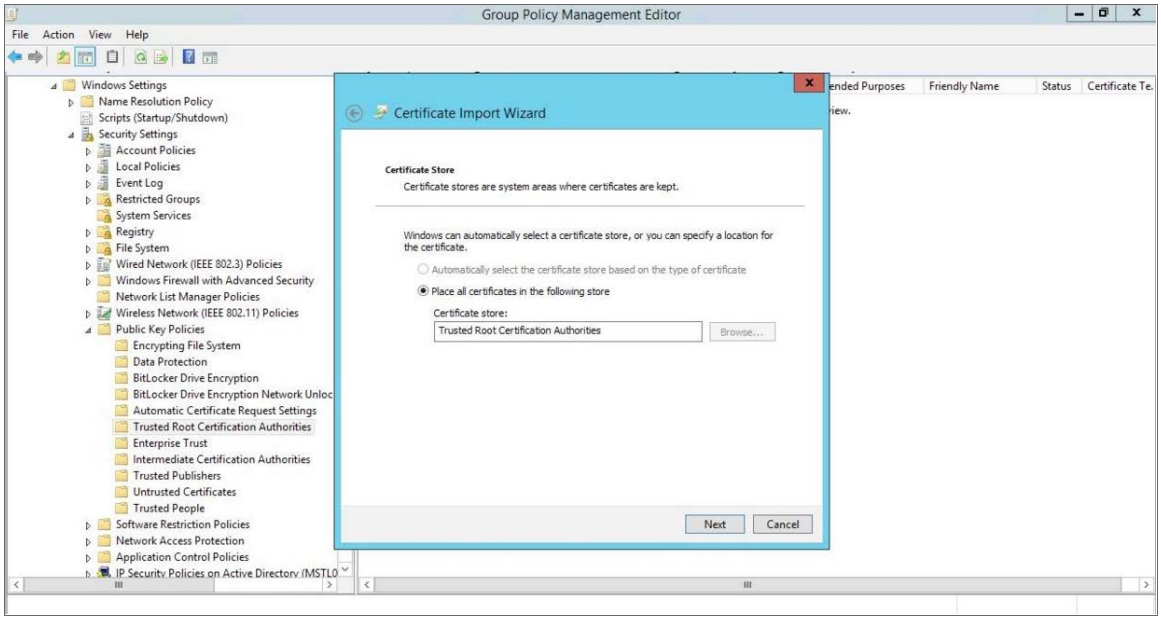
20. The **Certificate Import Wizard** modal page appears. Click **[Next]**.



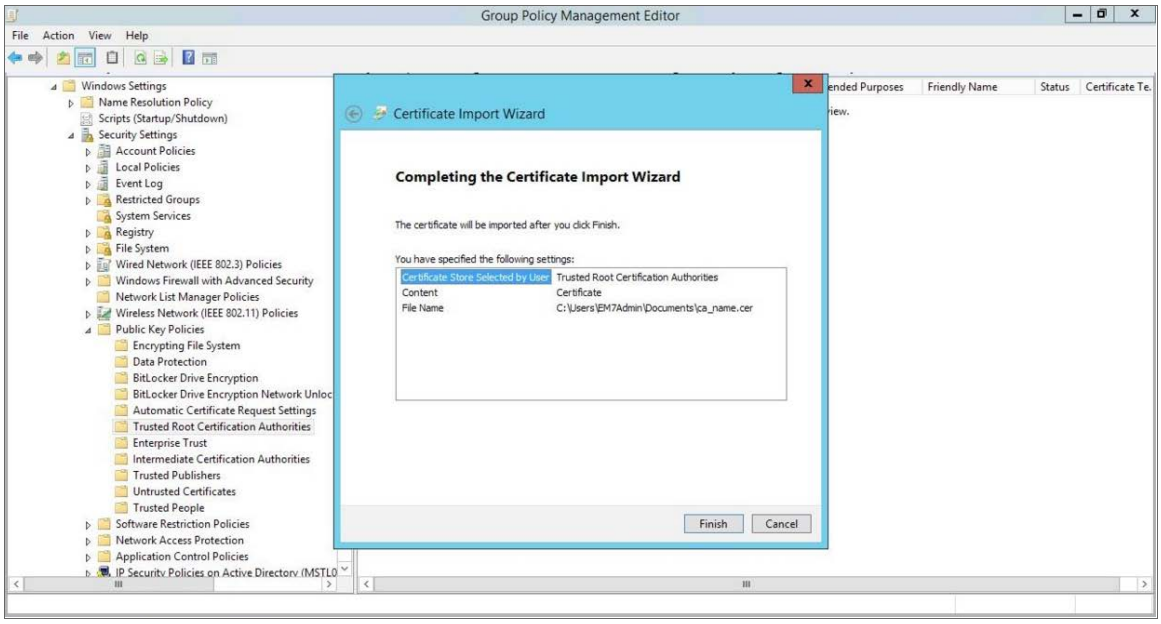
21. Browse to the Certification Authority certificate that you saved to your local directory in step 4, then click **[Next]**.



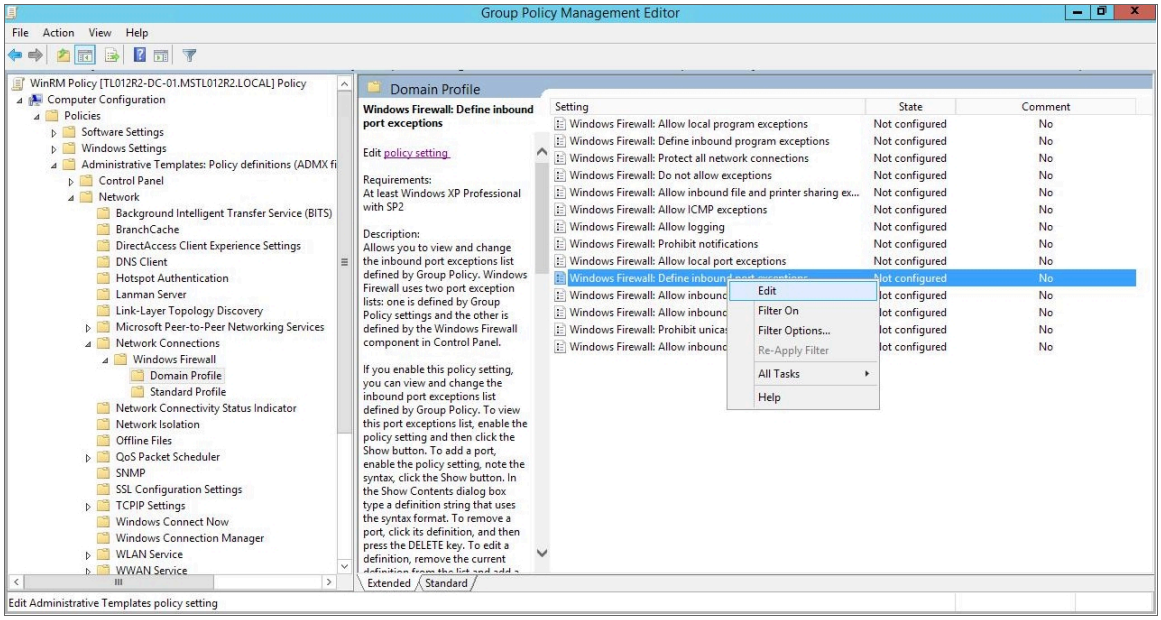
22. Select the **Place all certificates in the following store** radio button, then select the *Trusted Root Certification Authorities* certificate store and click **[Next]**.



23. Click **[OK]** to confirm that the certificate was successfully imported, and then click **[Finish]**.

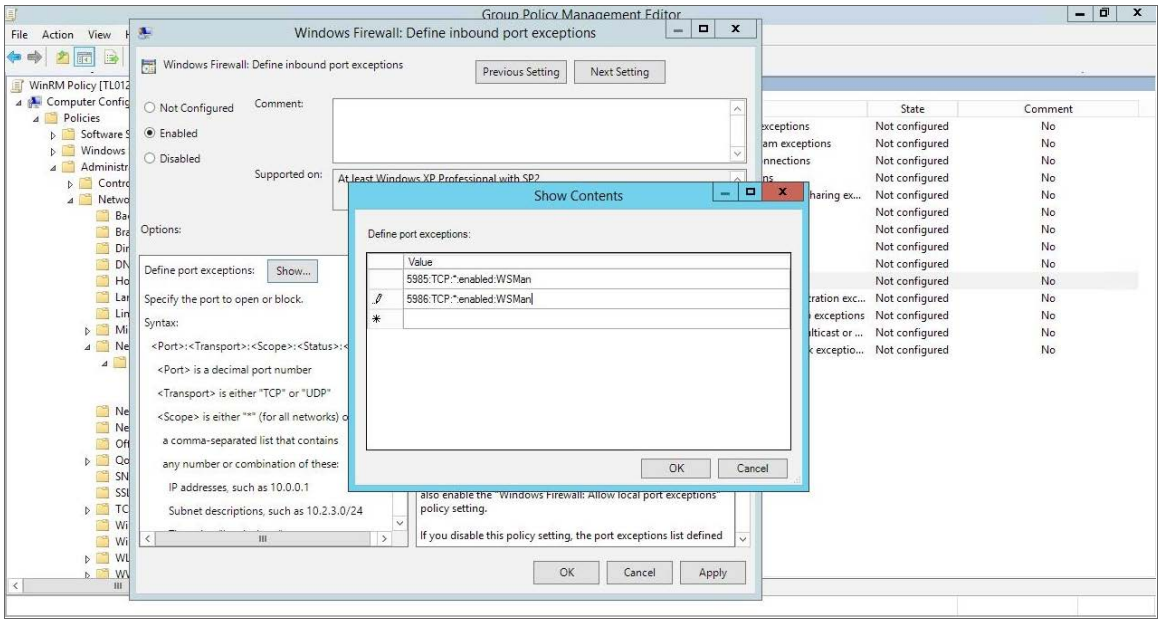


24. In the left panel of the **Group Policy Management Editor** page, navigate to **Computer Configuration > Policies > Administrative Templates > Network > Network Connections > Windows Firewall > Domain Profile**. In the right panel, right-click **Windows Firewall: Define inbound port exceptions** and select *Edit*.



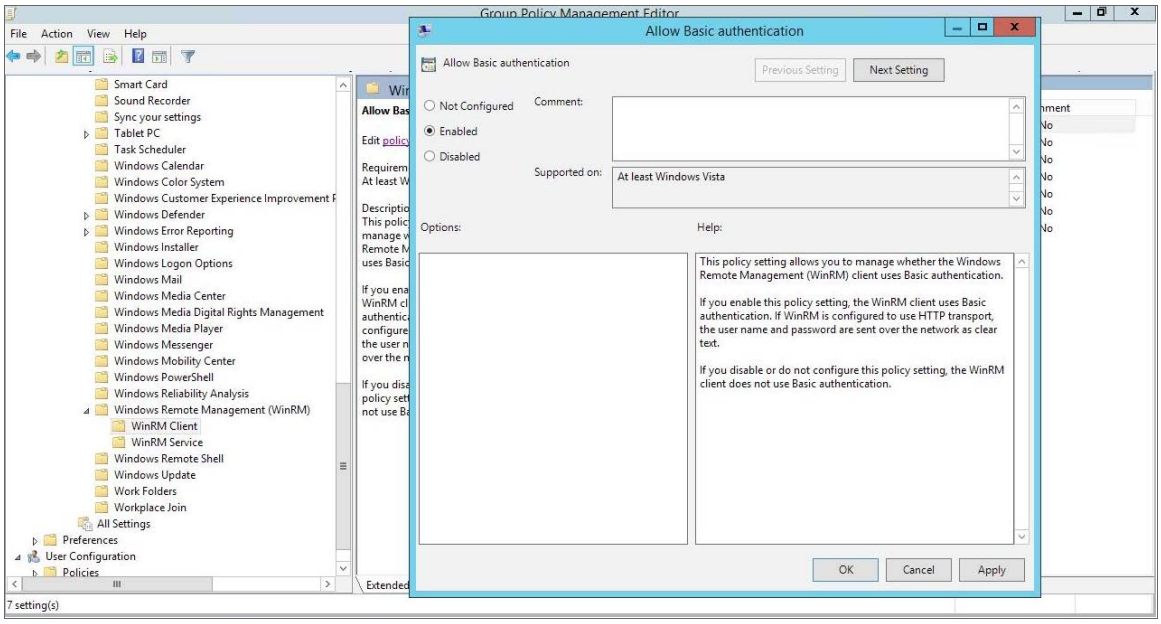
25. The **Windows Firewall: Define inbound port exceptions** modal page appears. Under **Options**, click **[Show]**.

26. The **Show Contents** modal page appears. Enter the following values:



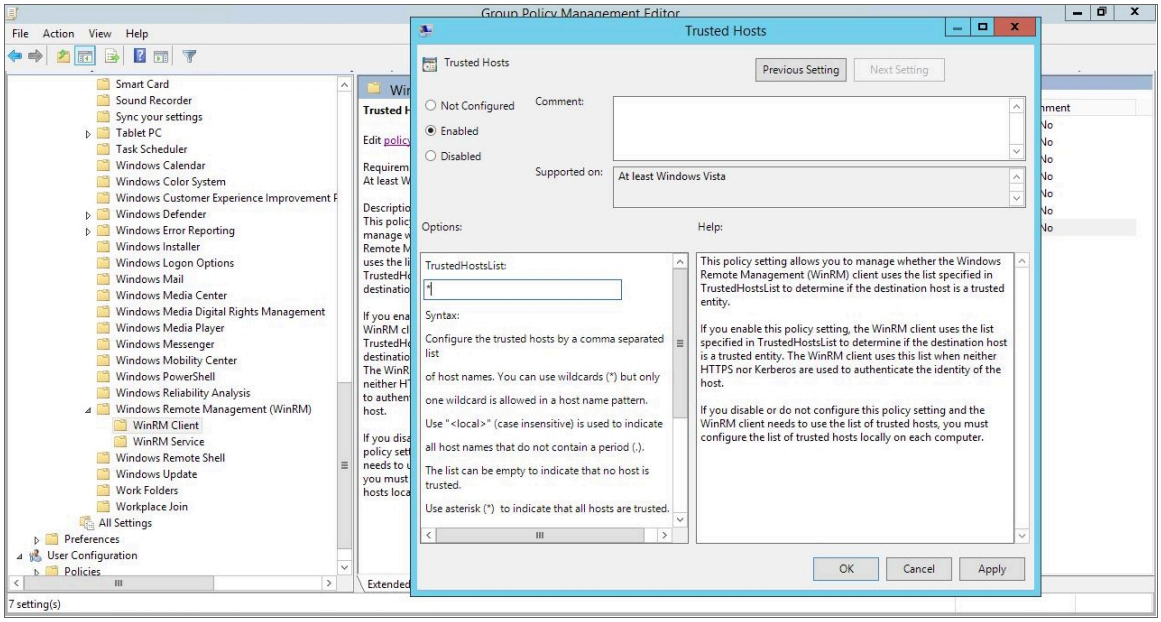
- 5985:TCP:*:enabled:WSMan
- 5986:TCP:*:enabled:WSMan

27. Click **[OK]**, then click **[OK]** again.
28. In the left panel of the **Group Policy Management Editor** page, navigate to **Computer Configuration > Policies > Administrative Templates > Windows Components > Windows Remote Management (WinRM) > WinRM Client**. In the right panel, double-click the **Allow Basic authentication** setting.

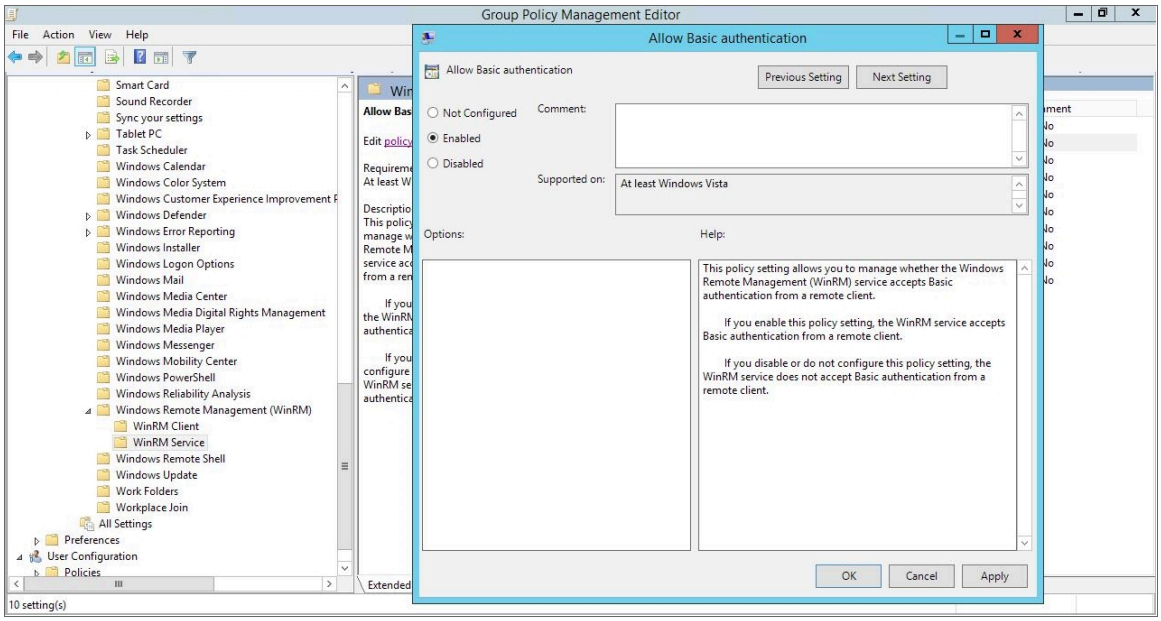


29. Select the **Enabled** radio button, then click **[OK]**.

30. Repeat steps 28 and 29 for the **Allow unencrypted traffic** setting.
31. Double-click the **Trusted Hosts** setting. Select the **Enabled** radio button, enter an asterisk (*) in the **TrustedHostsList** field (under **Options**), and then click **[OK]**.

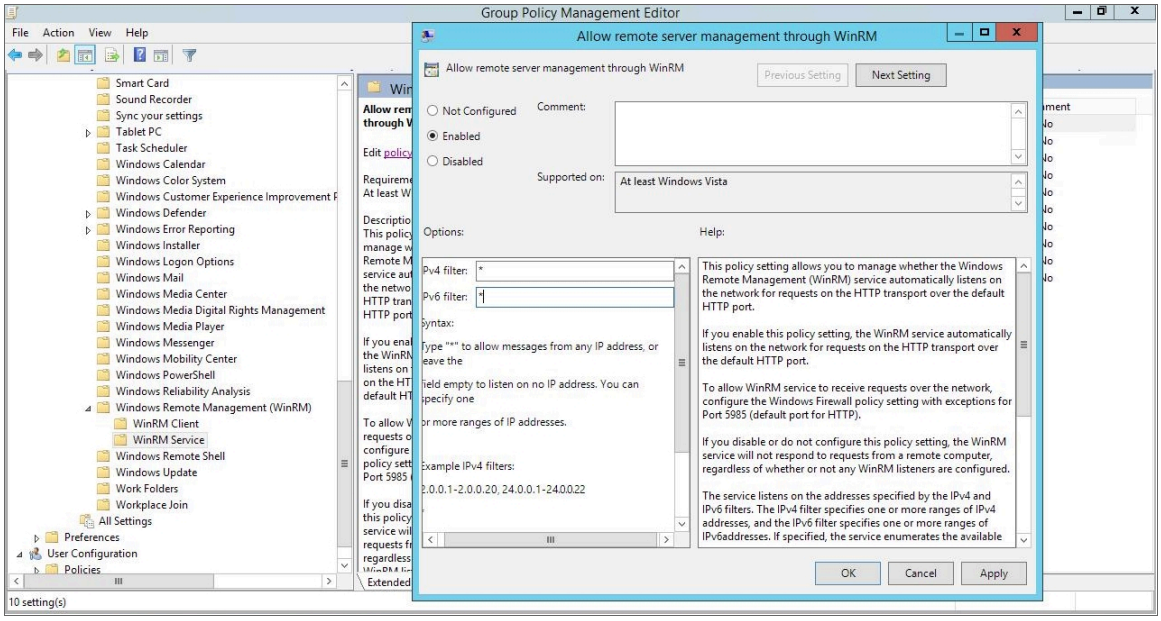


32. In the left panel of the **Group Policy Management Editor** page, navigate to **Computer Configuration > Policies > Administrative Templates > Windows Components > Windows Remote Management (WinRM) > WinRM Service**. In the right panel, double-click the **Allow Basic authentication** setting.

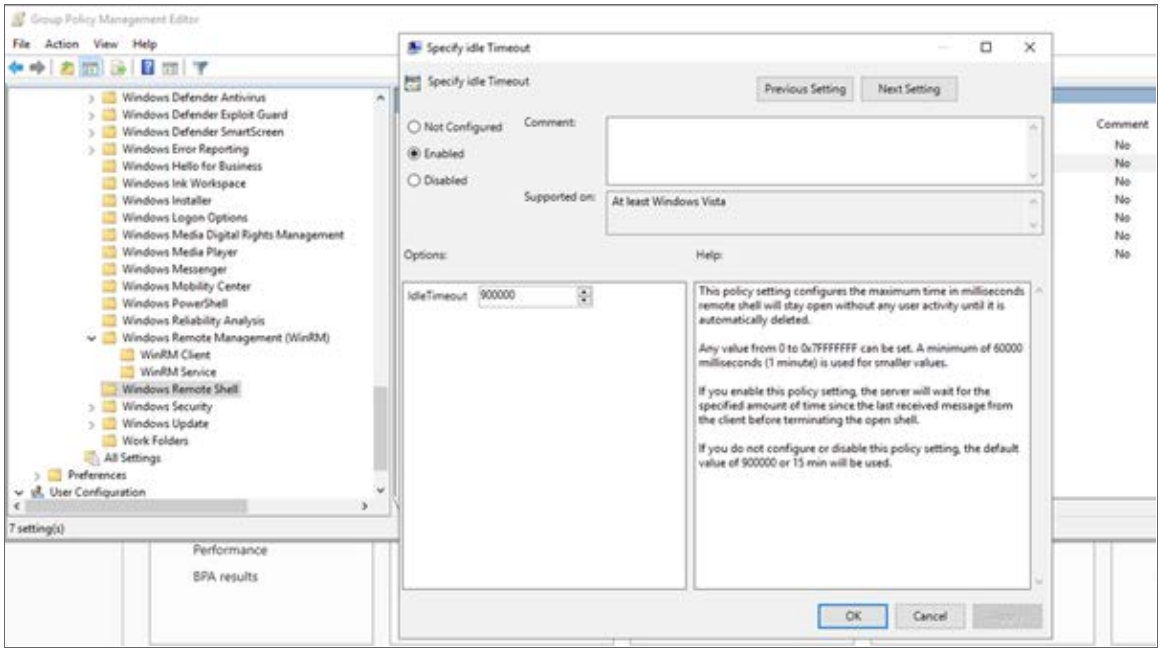


33. Select the **Enabled** radio button, then click **[OK]**.

34. Repeat steps 32 and 33 for the **Allow unencrypted traffic** setting.
35. Double-click the **Allow remote server management through WinRM** setting. Select the **Enabled** radio button, enter an asterisk (*) in the **Pv4 filter** and **Pv6 filter** fields (under **Options**), and then click **[OK]**.



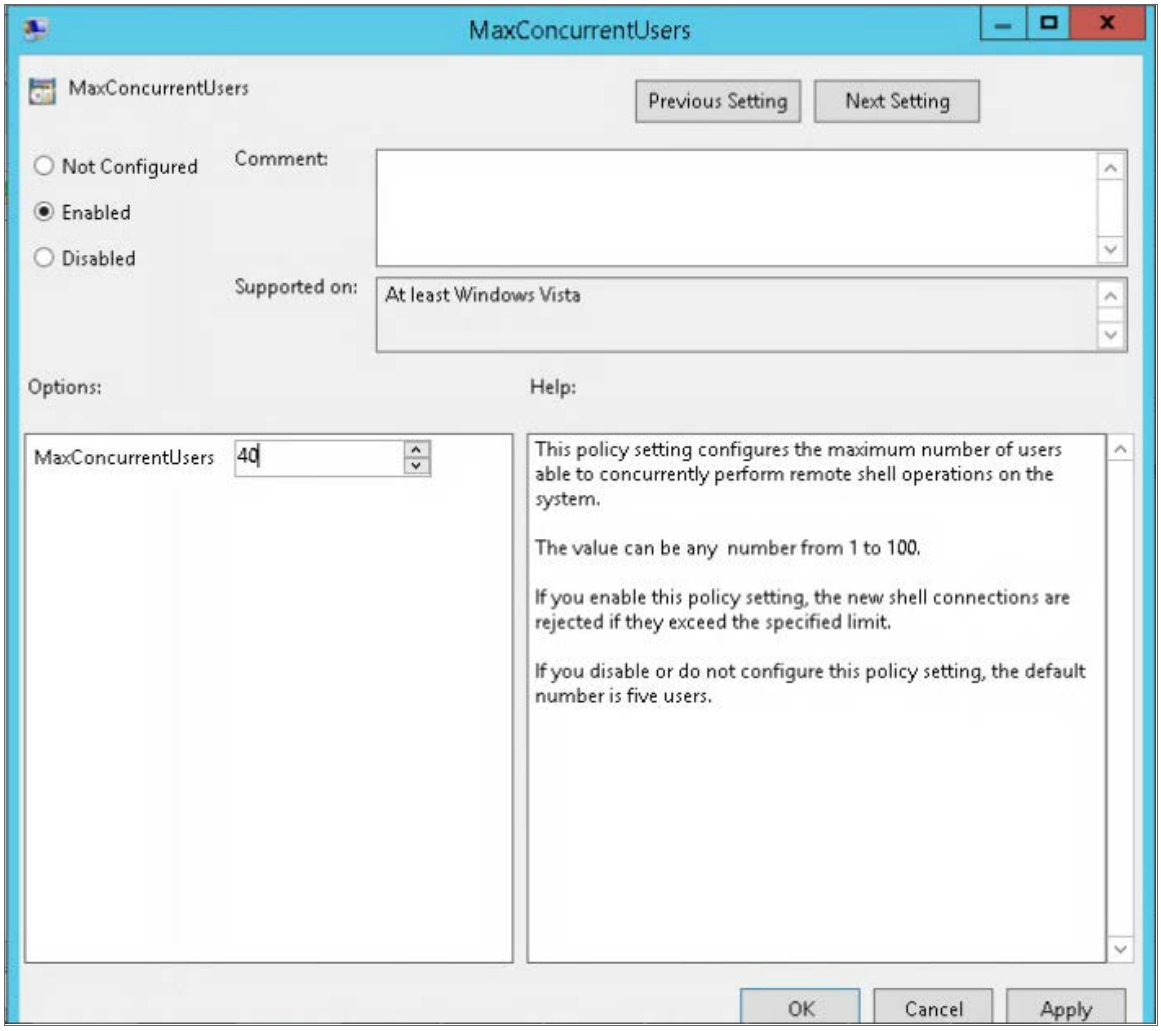
36. In the left panel of the **Group Policy Management Editor** page, navigate to **Computer Configuration > Policies > Administrative Templates: Policy Definitions > Windows Components > Windows Remote Shell**. In the right panel, double-click on **Specify Idle Timeout**:



Adjust the setting to meet your requirements. Using the value of 900000 in the image will set the timeout to 15 minutes. Once you have entered your timeout value in milliseconds, click the ***Enabled*** radio button and then click **[OK]**.

> **NOTE:** When changing IdleTimeout, ensure that no other applications or utilities need a higher timeout for WinRM sessions.

37. In the **Windows Remote Shell** folder, in the right panel, double-click on **MaxConcurrentUsers**:



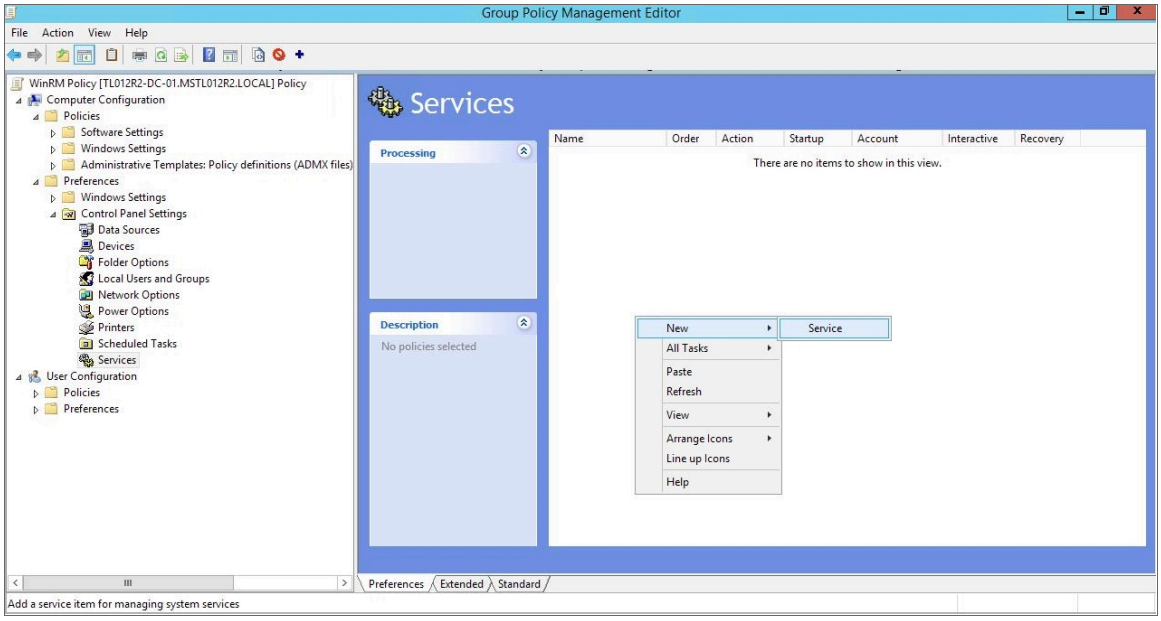
Enter "40" in the ***MaxConcurentUsers*** field. Once you have entered your value, click the ***Enabled*** radio button and then click **[OK]**.

38. **You can skip this step if you already have a group policy in place for this setting.** In the left panel of the **Group Policy Management Editor** page, navigate to **Computer Configuration > Preferences > Windows Settings > Registry**. In the right panel, right-click and select *New > Registry Item*. In the **New Registry Properties** modal page, edit the values in one or more of the following fields:

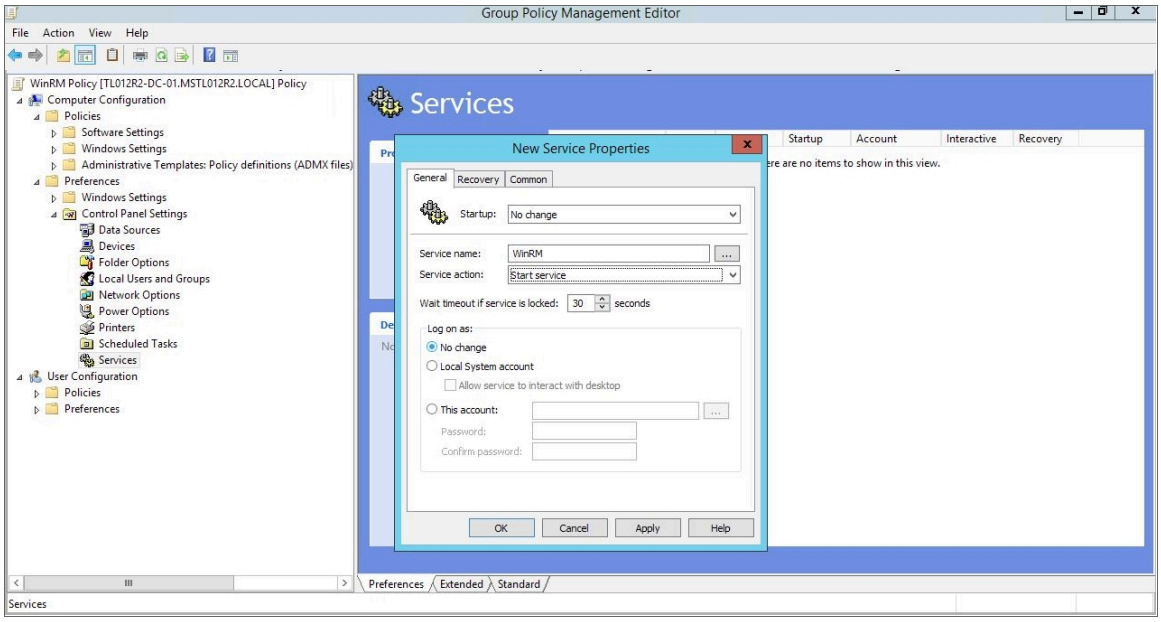
> **NOTE:** This step is required only if the user account is **not** a domain account and **not** the built-in local administrator account.

- ***Action***. Select *Create*.
- ***Hive***. Select *HKEY_LOCAL_MACHINE*.
- ***Key Path***. Enter "SOFTWARE\Microsoft\Windows\CurrentVersion\policies\system".
- ***Value name***. Enter "LocalAccountTokenFilterPolicy".
- ***Value type***. Enter "REG_DWORD".
- ***Value data***. Enter "1".
- ***Base***. Select *Decimal*.

39. In the left panel of the **Group Policy Management Editor** page, navigate to **Computer Configuration > Preferences > Control Panel Settings > Services**. In the right panel, right-click and select *New > Service*.

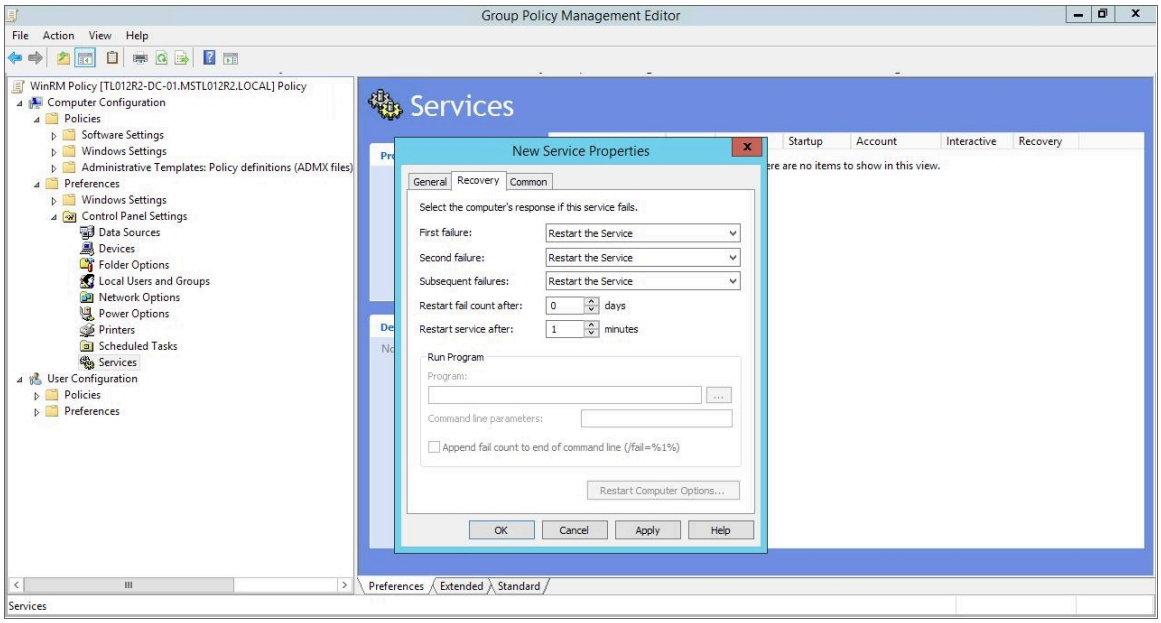


40. In the **New Service Properties** modal page, edit the values in one or more of the following fields:



- ***Startup***. Select *No change*.
- ***Service name***. Enter "WinRM".
- ***Service action***. Select *Start service*.
- ***Wait timeout if service is locked***. Select *30* seconds.
- ***Log on as***. Select *No change*.

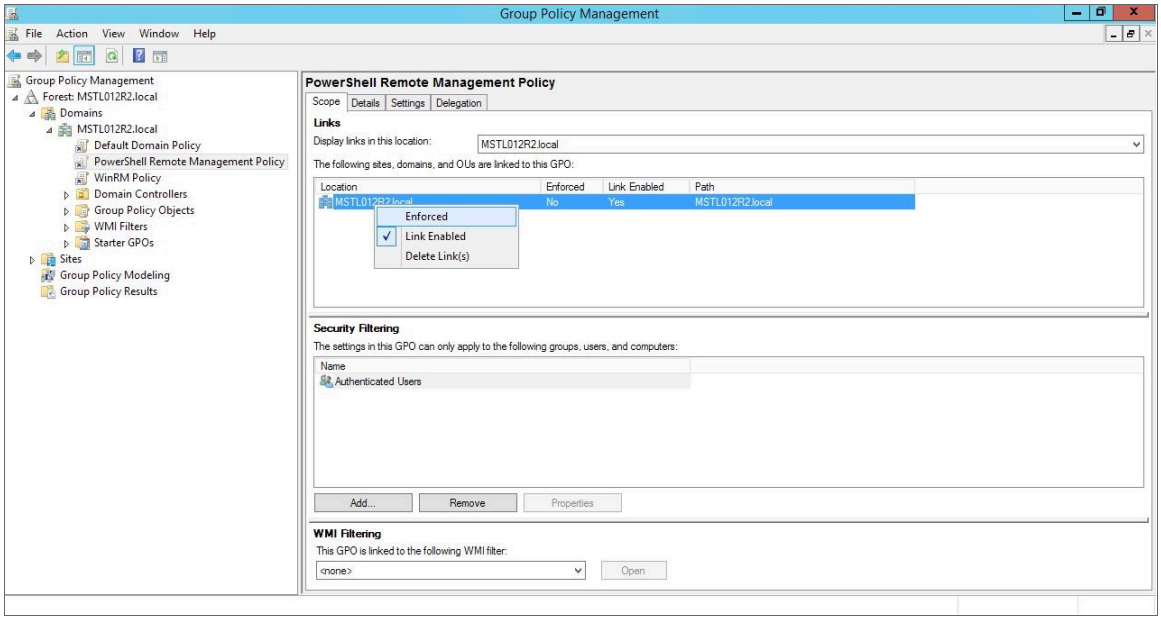
41. Click the **[Recovery]** tab, then edit the values in one or more of the following fields:



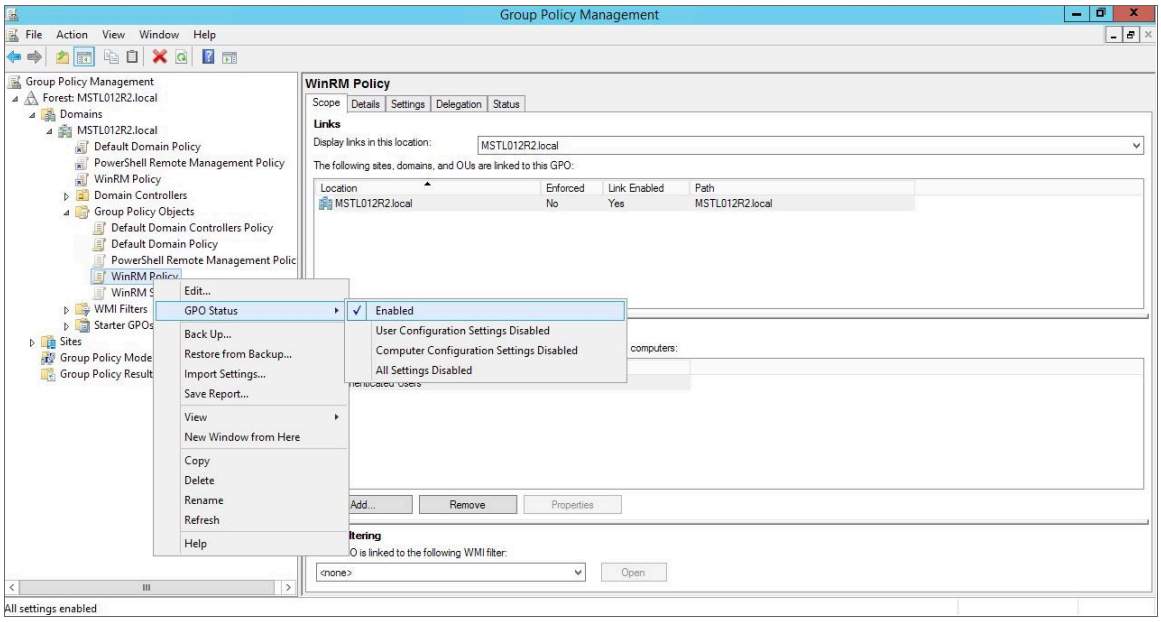
- ***First failure***. Select *Restart the Service*.

- ***Second failure***. Select *Restart the Service*.
- ***Subsequent failures***. Select *Restart the Service*.
- ***Restart fail count after***. Select *0* days.
- ***Restart service after***. Select *1* minute.

42. Click the **[OK]** button.
43. To enforce your group policy, in the left panel of the **Group Policy Management Editor** page, navigate to **Forest > Domains > [your local domain] > PowerShell Remote Management Policy**. In the **PowerShell Remote Management Policy** panel on the right, right-click the local domain name under ***The following sites, domains, and OUs are linked to this GPO*** and select *Enforced*.



44. To enable your group policy, in the left panel of the **Group Policy Management Editor** page, navigate to **Forest > Domains > [your local domain] > Group Policy Objects > WinRM Policy**. Right-click **WinRM Policy**, then select *GPO Status > Enabled*.



Configuring an HTTPS Listener with GPO Configuration

If you are using an HTTPS listener, you cannot create the listener and start it on the monitored device within group policy object (GPO) configuration. This can be done by using a startup script or an immediate task in the group policy, or by running a command manually or on the remote management tool on the device to be monitored. This command needs to be run only once as the HTTPS listener will automatically start once configured.

To perform this configuration within the group policy, perform the following steps:

1. Run the following command on the device you want to monitor:

```
winrm quickconfig -transport:https -force
```

This command will select the first available certificate enabled for server authentication. If you have multiple, valid server authentication certificates installed on your device, you will need to specify the thumbprint of the certificate and use the following command instead:

```
New-Item -Path WSMan:\LocalHost\Listener -Transport HTTPS -Address *
-CertificateThumbPrint "<CertThumbprint>" -Force
```

NOTE: The thumbprint should not contain spaces.

Using Forward and Reverse DNS for Windows Remote Management

When using Active Directory accounts for PowerShell monitoring, Kerberos and Windows Remote Management (WinRM) are used to connect to Windows devices and execute PowerShell code on those devices. Kerberos is used to request a ticket for authentication to the Windows device, and WinRM is used to execute code on the Windows device.

In a Windows Active Directory configuration, Kerberos needs to be able to communicate with the target Windows device and the Active Directory Domain Controller to verify credentials and issue a ticket for authentication. Kerberos refers to a Windows Domain as a "realm" and an Active Directory Server as a "kdc" (Key Distribution Center).

For this process, it is important that forward and reverse lookup is working for all systems involved. Forward lookup translates a host to an IP address; reverse lookup translates an IP address to a host.

This can be managed through DNS, where a forward lookup is handled through an "A" record in a forward lookup zone, and reverse lookup through a "PTR" record in a reverse lookup zone. A utility such as "nslookup" will work correctly only if the DNS record (a PTR record, in this case) is present.

Where DNS is not available or reliable, it is possible to use the hosts file (`/etc/hosts`) instead. Skylar One uses Python, which in turn can use the hosts file to provide both forward and reverse lookup. However, this approach means a higher level of server management because the hosts files on multiple Data Collector servers would need to be kept in sync. Additionally, where Concurrent PowerShell is used, the hosts files within the Docker containers would need to be updated.

Without a reliable forward and reverse lookup mechanism in place, Kerberos may not be able to validate credentials and issue a ticket for access to a Windows Device, which in turn would mean that access over WinRM to the device would be rejected.

Step 4: Configuring a Windows Management Proxy

If Skylar One cannot execute PowerShell requests directly on a Windows server, you can optionally configure an additional Windows server to act as a proxy for those PowerShell requests. To use a proxy, you must configure at least two Windows servers:

- A target server that Skylar One cannot communicate with directly.
- A proxy server that Skylar One will communicate with to execute PowerShell requests on the target server.

> **NOTE:** When monitoring a Windows device using a proxy, the account specified in the credentials is used to access both the proxy server and the target device. This account must have the correct access rights to be used on both servers. If multiple Active Directory domains are used, a trust relationship must be in place that allows the specified account access to the servers in both domains.

To configure the target and proxy servers, perform the following steps:

1. Configure a user account that Skylar One will use to connect to the proxy server and the proxy server will use to connect to the target server. The user account can either be a local account or an Active Directory account; however, the user account must have the same credentials on the target and proxy servers and be in the Local Administrator's group on both servers.
2. If you have created a local user account on the Windows Server instead of an Active Directory account, you must configure encrypted communication between Skylar One and the Windows server. To do this, you must *configure a Server Authentication certificate*.
3. *Configure Windows Remote Management* on the target server and the proxy server.
4. Log in to the proxy server as an administrator.
5. Open the PowerShell command window.
6. Right-click on the PowerShell icon in the taskbar and select *Run as Administrator*.
7. Execute one of the following commands on the proxy server to allow the proxy server to trust one or more target servers:
    - To allow the proxy server to trust all servers (not recommended), execute the following command:

      ```
      Set-Item WSMan:\Localhost\Client\TrustedHosts -value *
      ```

    - To allow the proxy server to trust only specific target servers, execute the following command, inserting a list that includes the IP address for each target server. Separate the list of IP addresses with commas.

      ```
      Set-Item WSMan:\Localhost\Client\TrustedHosts -value <comma-delimited-list-of-target-server-IPs>
      ```

NOTE: The following step is required only if the user account is **not** a domain account and **not** the built-in local administrator account.

8. Execute the following command on the proxy server to configure the LocalAccountTokenFilterPolicy:

   ```
   New-ItemProperty
   “HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System” -
   Name “LocalAccountTokenFilterPolicy” -Value 1 -PropertyType “DWORD"
   ```

NOTE: If the proxy server is in a different Windows domain (domain A) than the target servers (domain B), and the proxy server uses a user account from Active Directory, and Active Directory is in the same Windows domain as the target servers (domain B), you must perform the following to allow the proxy server to send PowerShell commands to the target servers:

- On the domain controller for each domain (domain A and domain B), create new forward-lookup zones and reverse-lookup zones that allow name resolution to work between the two domains.
- On the domain controller for each domain (domain A and domain B), create a non-transitive realm trust between the two domains.
- Login to the proxy server and add the Active Directory account (from domain A) to the Local Administrator's group for the proxy server. You should be able to select the account on the proxy server after you create the non-transitive realm trust between the two domains.

Step 5: Increasing the Number of PowerShell Dynamic Applications That Can Run Simultaneously

You can optionally execute a series of commands that will allow Skylar One to increase the default maximum number of PowerShell Dynamic Applications that can run simultaneously.

To do so:

1. Determine the number of Dynamic Applications that will be used to monitor the Windows server. Multiply this number by three.
2. Open a PowerShell command prompt. Log in as an Administrator.
3. At the prompt, execute the following commands:

```
Set-Item WSMan:\Localhost\Shell\MaxShellsPerUser –value <number you calculated in step 1>

Set-Item WSMan:\Localhost\Service\MaxConcurrentOperationsPerUser –value <number you calculated in step 1>

Restart-Service WinRM
```

4. Repeat these steps on each Windows server that will be monitored by Skylar One.

Optional PowerShell CLI Parameters

You can use the following parameters in PowerShell for the associated reasons:

- ***-NoProfile***. Does not load the PowerShell profile.
- ***-NoLogo***. Hides the copyright banner at start-up.
- ***-NonInteractive***. Does not present an interactive prompt to the user.

To enable concurrent PowerShell collection to use one of these parameters:

1. Go to the **Database Tool** page (System > Tools > DB Tool).

> **NOTE:** The **Database Tool** page is available only in versions of Skylar One prior to 12.2.1 and displays only for users that have sufficient permissions to access the page.

2. If this row does not already exist in the `master.system_custom_config` table, enter the following in the ***SQL Query*** field:

```
INSERT INTO master.system_custom_config ('field','field_value') VALUES
('powershell_prefix_setting', '<PREFIX>')
```

   where `<PREFIX>` is an integer that represents one of the prefix values described above. The integers are as follows:

   - **0**. Disabled
   - ***1***. -NoProfile
   - ***2***. -NoLogo
   - ***3***. -NoProfile and -NoLogo
   - ***4***. -NonInteractive
   - ***7***. -NoProfile, -NoLogo, and -NonInteractive

For example, if a user wanted to configure their PowerShell Data Collector to not load their PowerShell profile, they would enter the following into the ***SQL Query*** field:

```
INSERT INTO master.system_custom_config ('field','field_value') VALUES
('powershell_prefix_setting', '1')
```

3. If this row already exists in the `master.system_custom_config` table, enter the following in the ***SQL Query*** field:

```
UPDATE master.system_custom_config SET field_value = 1 WHERE field =
'powershell_prefix_setting'
```

4. After you have entered the command in the ***SQL Query*** field, click the **[Go]** button. Your changes will be picked up with the next batch of jobs that are processed.

Creating a PowerShell Credential

If you configure your Windows system to respond to PowerShell requests from Skylar One, you can use PowerShell Dynamic Applications to collect information from your Windows system.

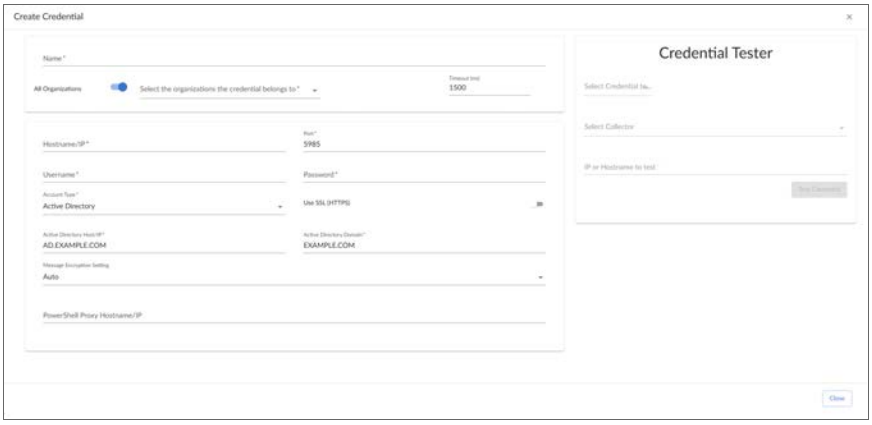
All of the PowerShell Dynamic Applications include a discovery object. If you include a credential for PowerShell Dynamic Applications in the discovery session that includes your Windows system, Skylar One will automatically align the appropriate PowerShell Dynamic Applications to the Windows system. For more information about creating a discovery session, see *Discovering Devices*.

To define a PowerShell credential in Skylar One, you will need the following information:

- The username and password for a user on the Windows device.
- If the user is an Active Directory account, the hostname or IP address of the Active Directory server and the domain.
- Determine if an encrypted connection should be used.
- If you are using a Windows Management Proxy, the hostname or IP address of the proxy server.

To create a PowerShell credential:

1. Go to the **Credentials** page (Manage > Credentials).
2. Click the **[Create New]** button and then select *Create Powershell Credential*. The **Create Credential** modal page appears:



3. Supply values in the following fields:
    - ***Name***. Name of the credential. Can be any combination of alphanumeric characters, up to 64 characters. This field is required.
    - ***All Organizations***. Toggle on (blue) to align the credential to all organizations, or toggle off (gray) and then select one or more specific organizations from the ***What organization manages this service?*** drop-down field to align the credential with those specific organizations. This field is required.
    - ***Timeout (ms)***. Time, in milliseconds, after which Skylar One will stop trying to communicate with the authenticating server. For collection to be successful, Skylar One must connect to the authenticating server, execute the PowerShell command, and receive a response within the amount of time specified in this field.
    - ***Hostname/IP***. Hostname or IP address of the device from which you want to retrieve data. This field is required.
        - You can include the variable **%D** in this field. Skylar One will replace the variable with the IP address of the device that is currently using the credential.

- You can include the variable **%N** in this field. Skylar One will replace the variable with the hostname of the device that is currently using the credential. If Skylar One cannot determine the hostname, Skylar One will replace the variable with the primary, management IP address for the current device.
- You can include the prefix **HOST** or **WSMAN** before the variable **%D** in this field if the device you want to monitor uses a service principal name (for example, "HOST://%D" or "WSMAN://%D"). Skylar One will use the WinRM service HOST or WSMan instead of HTTP and replace the variable with the IP address of the device that is currently using the credential.

- ***Port***. Type the port number used by the WinRM service on the Windows device. This field is required.
- ***Username***. Type the username for an account on the Windows device to be monitored or on the proxy server. This field is required.

> **NOTE**: The user should not include the domain name prefix in the username for Active Directory accounts. For example, use "em7admin" instead of "MSDOMAIN\em7admin".

- ***Password***. Type the password for the account on the Windows device to be monitored or on the proxy server. This field is required.
- ***Account Type***. Type of authentication for the username and password in this credential. Choices are:
  - *Active Directory*. On the Windows device, Active Directory will authenticate the username and password in this credential.
  - *Local*. Local security on the Windows device will authenticate the username and password in this credential.

- ***Use SSL (HTTPS)*** / ***Encrypted***. Select whether Skylar One will communicate with the device using an encrypted HTTP or HTTPS connection:
    - Toggle on (blue) if Skylar One will communicate with the device using an encrypted connection over HTTPS. If toggled on, when communicating with the Windows server, Skylar One will use a local user account with authentication of type "Basic Auth". You must then use HTTPS and can use a Microsoft Certificate or a self signed certificate.

      NOTE: In Skylar One versions prior to 12.3.7, this field is labeled ***Encrypted***. In versions 12.3.7 and above, it is labeled ***Use SSL (HTTPS)***.

      NOTE: In Skylar One versions 11.3.0 and later, a newer Kerberos library is used that allows for message encryption over HTTP. This feature is on by default and may eliminate the need for you to configure an HTTPS certificate depending on your security requirements.

    - Toggle off (gray) . The credential is encrypted over HTTP rather than HTTPS.
- ***Validate Certificate (when HTTPS is used)***. This field is visible when the ***Use SSL (HTTPS)*** toggle field is enabled for the connection and allows you to select whether a certificate is validated for the credential. Choices are:
    - *Ignore*. Skylar One will not validate a certificate for the credential. This is the default setting.
    - *Validate*. Skylar One will require a validated certificate for the credential. If you select *Validate*, then the target device must include a non-expired certificate issued from a certificate authority.
- ***Active Directory Host/IP***. If you selected Active Directory in the ***Account Type*** field, type the hostname or IP address of the Active Directory server that will authenticate the credential.
- ***Active Directory Domain***. If you selected Active Directory in the ***Account Type*** field, type the domain where the monitored Windows device resides.
- ***Message Encryption Setting***. If you selected Active Directory in the ***Account Type*** field, select whether Kerberos packages sent over PowerShell Remoting Protocol (PSRP) or Windows Remote Management (WinRM) are encrypted. Choices are:
    - *Auto*. Encryption is enabled if the package supports it; otherwise, encryption is disabled. This is the default setting.
    - *Never*. Messages are never encrypted. If selected, the target device must support this option.
    - *Always*. Messages are always encrypted. If selected, the target device must support this option.
- ***PowerShell Proxy Hostname/IP***. If you use a proxy server in front of the Windows devices you want to communicate with, type the fully-qualified domain name or the IP address of the proxy server in this field.

4. Click **[Save & Close]**.

> **NOTE:** If you would like to test your credential using the Credential Tester panel, click **[Save & Test]**. For detailed instructions on using the Credential Tester panel, see the *Using the Credential Tester Panel* section.

Error Messages for PowerShell Collection

The following table lists error messages that Skylar One can generate during PowerShell collection.

Error Message	Possible Issue(s)
Preauthentication failed while getting initial credentials	Incorrect Password (Active Directory Accounts only)
Client not found in Kerberos database	Username does not exist in Active Directory (Active Directory Accounts only)
KRB5 error code 68 while getting initial credentials	Incorrect domain name (Active Directory Accounts only)
Bad HTTP response returned from server. Code 401, basic auth failed	Incorrect username/password or target server does not allow user account to perform WinRM operations.
ParseError	Incorrect port specified in credential
[Errno 111] Connection refused	Mismatch between server configuration and credential, e.g. encryption option selected but not enabled on server.
Hostname cannot be canonicalized	Forward and/or reverse name resolution are not working from the Data Collector or All-In-One Appliance
Cannot resolve network address for KDC in requested realm	Forward and/or reverse name resolution are not working from the Data Collector or All-In-One Appliance
Configuration file does not specify default realm	Forward and/or reverse name resolution are not working from the Data Collector or All-In-One Appliance
No credentials cache found	Forward and/or reverse name resolution are not working from the Data Collector or All-In-One Appliance
Server not found in Kerbers database	Forward and/or reverse name resolution are not working from the Data Collector or All-In-One Appliance

Concurrent PowerShell Collection

The following sections describe how to configure and use concurrent PowerShell collection. Concurrent PowerShell collection allows multiple collection tasks to run at the same time with a reduced load on Data Collectors. Concurrent PowerShell collection also prevents missed polls and data gaps because collection will execute more quickly. As a result, Data Collectors can collect more data using fewer system resources. The PowerShell Collector is an independent service running as a container on a Data Collector.

Prerequisites

The following prerequisites are required to use concurrent PowerShell collection:

- Skylar One version 10.1.4 or greater
- "Microsoft: Windows Server" PowerPack version 110 or greater
- " Skylar One: Concurrent PowerShell Monitoring" PowerPack version 100 or greater (for Skylar One versions prior to 11.3.0).

NOTE: As of Skylar One (SL1) version 11.3.0, the "Skylar One: Concurrent PowerShell Monitoring" PowerPack is no longer a requirement for concurrent PowerShell monitoring.

Scope

When the concurrent PowerShell collection service is enabled, PowerShell Configuration and PowerShell Performance Dynamic Applications are sent to the service.

The following PowerPacks can use the concurrent PowerShell collection service:

- Microsoft: Active Directory Server
- Microsoft: DHCP Server
- Microsoft: DNS Server
- Microsoft: Exchange Server
- Microsoft: IIS Server
- Microsoft: Lync Server 2013
- Microsoft: SharePoint Server
- Microsoft: SQL Server
- Microsoft: Hyper-V Server (partially)
- Microsoft: Windows Server (partially)

Enabling and Disabling Concurrent PowerShell for Collector Groups

To improve the process of collecting data via PowerShell and to collect metrics, you can enable concurrent PowerShell collection. You can enable one or more collector groups to use concurrent PowerShell collection.

CAUTION: If you have enabled concurrent collection and you have used it to discover a very large number of devices or interfaces, disabling concurrent collection could have unintended consequences. After disabling concurrent collection, your Data Collector might become overburdened when it attempts to collect data for the same number of devices or interfaces but without the added processing capacity of concurrent collection.

CAUTION: By default, a loopback to 127.0.0.1 is configured on the collector with the line `localhost localhost.localdomain localhost4 localhost4.localdomain4` in the `/etc/hosts` file. If this line is removed, concurrent PowerShell collection will not function properly.

NOTE: Concurrent PowerShell collection is for PowerShell Performance and Performance Configuration Dynamic Application types and does not include Snippet Dynamic Applications which happen to run PowerShell commands.

Enabling and Disabling Concurrent PowerShell on All Collector Groups

To enable and disable concurrent PowerShell collection for all collector groups:

1. Go to the **Behavior Settings** page (System > Settings > Behavior).
2. Select the ***Enable Concurrent PowerShell Collection*** checkbox and click **[Save]**.

   NOTE: If the "Data Collection: PowerShell Collector" process is disabled on the **Process Manager** page (System > Settings > Admin Processes), this concurrent PowerShell collection option is disabled.

3. To disable concurrent PowerShell collection, deselect the ***Enable Concurrent PowerShell Collection*** checkbox and click **[Save]**.

Enabling and Disabling Concurrent PowerShell on a Specific Collector Group

To enable and disable concurrent PowerShell collection for a specific collector group in the classic Skylar One user interface:

1. Go to the **Collector Group Management** page (System > Settings > Collector Groups).
2. Locate the collector group for which you want to enable concurrent PowerShell, and click its wrench icon (🔧).
3. In the ***Enable Concurrent PowerShell Collection*** drop-down menu, select *Yes* and click **[Save]**.
4. To disable concurrent PowerShell collection, select *No* in the ***Enable Concurrent PowerShell Collection*** drop-down and click **[Save]**.

The Skylar One: Concurrent PowerShell Monitoring PowerPack

The "Skylar One: Concurrent PowerShell Monitoring" PowerPack includes a device template, two Dynamic Applications that use SSH to monitor collectors with concurrent PowerShell enabled, and a number of event policies.

- The "ScienceLogic: PowerShell Collector Performance" Dynamic Application is an optional Dynamic Application used for troubleshooting.
- The "ScienceLogic: PowerShell Service Log Parser" Dynamic Application parses the log file from the PowerShell servers and converts errors into events aligned to the related device.
- The "Skylar One: Concurrent PowerShell Monitoring" device template can be used to align multiple Data Collectors to the "ScienceLogic: PowerShell Service Log Parser" Dynamic Application.
- Event policies and corresponding alerts that are triggered when devices meet certain status criteria.

Aligning the "ScienceLogic: PowerShell Service Log Parser" Dynamic Application

To align the "ScienceLogic: PowerShell Service Log Parser" Dynamic Application, first you must create an SSH/Key credential:

1. Go to the **Credential Management** page (System > Manage > Credentials).
2. In the **Credential Management** page, click the **[Actions]** menu. Select ***Create SSH/Key Credential***.
3. The **Credential Editor** modal page appears. In this page, define the new SSH/Key credential using a valid username and password or SSH key for Skylar One collectors:
    - ***Credential Name***. Name of the credential. Can be any combination of alphanumeric characters.
    - ***Hostname/IP***. Hostname or IP address of the device from which you want to retrieve data.

        - You can include the variable %D in this field. Skylar One will replace the variable with the IP address of the current device (device that is currently using the credential).
        - You can include the variable %N in this field. Skylar One will replace the variable with hostname of the current device (device that is currently using the credential). If Skylar One cannot determine the hostname, Skylar One will replace the variable with the primary, management IP address for the current device.
    - ***Port***. Port number associated with the data you want to retrieve.

> **NOTE:** The default TCP port for SSH servers is 22.

    - ***Timeout (ms)***. Time, in milliseconds, after which Skylar One will stop trying to communicate with the authenticating server.
    - ***Username***. Username for the Data Collector to be monitored.
    - ***Password***. Password for the Data Collector to be monitored.
    - ***Private Key (PEM Format)***. Enter an SSH private key for the Skylar One Data Collector, in PEM format.
4. Click the **[Save]** button to save the new SSH/Key credential.

Next, you can *align the Dynamic Application manually* or *configure the device template*. Using the device template is recommended when you want to align the Dynamic Application to multiple Data Collectors.

Manually Aligning the Dynamic Application

After creating the SSH/Key credential, you will manually align the Dynamic Application.

1. Go to the **Devices** page and find the device you want to manually align the Dynamic Application to. Click on it to go to the **Device Investigator**.
2. In the **Device Investigator**, click the **[Collections]** tab. Click **[Edit]** and then click **[Align Dynamic App]**. The **Align Dynamic Application** window appears.
3. Click *Choose Dynamic Application*. The **Choose Dynamic Application** window appears.
4. Select the "ScienceLogic: PowerShell Service Log Parser" Dynamic Application and click **[Select]**. The "ScienceLogic: PowerShell Service Log Parser" Dynamic Application appears in the **Align Dynamic Application** window.
5. If a default credential is listed below the Dynamic Application and you want to use that credential, skip ahead to step 8. Otherwise, uncheck the box next to the credential name.
6. Click *Choose Credential*. The **Choose Credential** window appears.
7. Select the credential for the Dynamic Application and click the **[Select]** button. The name of the selected credential appears in the **Align Dynamic Application** window.
8. Click the **[Align Dynamic App]** button. When the Dynamic Application is successfully aligned, it is added to the **Collections** tab, and a confirmation message appears at the bottom of the tab.

To manually align the Dynamic Application using the Skylar One classic user interface:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface)
2. In the **Device Manager** page, find the device for which you want to view Dynamic Applications. Select its wrench icon (🔧)
3. In the **Device Administration** panel, select the **[Collections]** tab.
4. Click the **[Actions]** button and then select *Add Dynamic Application*. The **Dynamic Application Alignment** page appears
5. In the ***Dynamic Applications*** field, select the "ScienceLogic: PowerShell Service Log Parser" Dynamic Application.
6. In the ***Credentials*** field, select the proper credential.
7. Click the **[Save]** button.

Configuring the Device Template

After creating the SSH/Key credential, you will need to configure the device template included in the PowerPack.

> **NOTE:** If you have already manually aligned the Dynamic Application, you do not need to perform the steps in this section.

To configure the device template:

1. Go to the **Configuration Templates** page (Devices > Templates, or Registry > Devices > Templates in the classic user interface).
2. Locate the "Skylar One: Concurrent PowerShell Monitoring" sample template and click its wrench icon (🔧). The **Device Template Editor** modal page appears.
3. Type a new name for the device template in the ***Template Name*** field so the sample template is not overwritten.
4. Click the **[Dyn Apps]** tab. The **Editing Dynamic Application Subtemplates** page appears.
5. In the ***Subtemplate Selection*** pane, select the "ScienceLogic: PowerShell Service Log Parser" Dynamic Application.
6. In the ***Credentials*** drop-down list, select the SSH/Key credential that you created.
7. Click **[Save As]**.

Applying the Device Template

If your Data Collector devices already exist on your Skylar One system, perform the following steps to apply the device template:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface) and select the checkbox for each of your Data Collector devices.
2. In the ***Select Action*** menu, select *MODIFY By Template* and then click **[Go]**.

3. In the **Device Template Editor**, select the template you created in the ***Template*** field.
4. Click **[Apply]**.

If your devices have not yet been discovered, perform the following steps to discover the devices and apply the device template:

1. Go to the **Discovery Control Panel** page (System > Manage > Classic Discovery) and click **[Create]**.
2. Supply values in the following fields:
    - ***Name***. Type a name for the discovery session.
    - ***Description***. Optionally, type a description of the discovery session.
    - ***IP Address/Hostname Discovery List***. Provide a list of IP addresses for your Data Collectors.
    - ***SNMP Credentials***. Select *EM7 Default V2*.
    - ***Model Devices***. Select this checkbox.
    - ***Apply Device Template***. Select the device template that you created.
    - ***Log All***. Select this checkbox.
4. Click the **[Save]** button **to save the discovery session**. Close the **Discovery Session Editor** page.
5. In the **Discovery Control Panel** page, click the **[Reset]** button. The new discovery session will appear in the **Session Register** pane.
6. To launch the new discovery session, click its ***Queue this Session*** icon ( ).
7. If no other discovery sessions are currently running, the session will be executed immediately. If another discovery session is currently running, your discovery session will be queued for execution.

Aligning the "ScienceLogic: PowerShell Collector Performance" Dynamic Application

If you want to monitor your Data Collectors with the "ScienceLogic: PowerShell Collector Performance" Dynamic Application, you must manually align it to your Data Collectors using the SSH/Key credential. To do this:

1. Go to the **Devices** page and find the device you want to manually align the Dynamic Application to and click on it to go to the Device Investigator.
2. In the Device Investigator, click the **[Collections]** tab. Click **[Edit]** and then click **[Align Dynamic App]**. The **Align Dynamic Application** window appears.
3. Click *Choose Dynamic Application*. The **Choose Dynamic Application** window appears.
4. Select the "ScienceLogic: PowerShell Collector Performance" Dynamic Application and click **[Select]**. The "ScienceLogic: PowerShell Collector Performance" Dynamic Application appears in the **Align Dynamic Application** window.
5. If a default credential is listed below the Dynamic Application and you want to use that credential, skip ahead to step 8. Otherwise, uncheck the box next to the credential name.
6. Click *Choose Credential*. The **Choose Credential** window appears.
7. Select the credential for the Dynamic Application and click the **[Select]** button. The name of the selected credential appears in the **Align Dynamic Application** window.

8. Click the **[Align Dynamic App]** button. When the Dynamic Application is successfully aligned, it is added to the **Collections** tab, and a confirmation message appears at the bottom of the tab.

To manually align the Dynamic Application using the Skylar One classic user interface:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface)
2. In the **Device Manager** page, find the device for which you want to view Dynamic Applications. Select its wrench icon ()
3. In the **Device Administration** panel, select the **[Collections]** tab.
4. Click the **[Actions]** button and then select *Add Dynamic Application*. The **Dynamic Application Alignment** page appears
5. In the ***Dynamic Applications*** field, select the "ScienceLogic: PowerShell Collector Performance" Dynamic Application.
6. In the ***Credentials*** field, select the proper credential.
7. Click the **[Save]** button.

Enabling HTTPS Between Skylar One and the PowerShell Data Collector

You can enable or disable HTTPS as the mode of transport for communication between Skylar One and the PowerShell Data Collector. The Powershell Data Collector is a service that runs on the Collector Group. The Data Collection service on the Collector Group can optionally communicate with the PowerShell Service to queue jobs and check for results using HTTPS. You would enable HTTPS if you must meet federal requirements to encrypt all network traffic, even if it never leaves the host.

To enable or disable HTTPS as the mode of transport for communication between Skylar One and the PowerShell Data Collector, you must make some changes to the `/opt/em7/services/powershell_collector/powershell_collector.env` configuration file. This file can also be used to configure the certificates used by the container when running on HTTPS.

The keys used are:

Key	Value
USE_HTTPS	Default value is ***True***. If set to ***False***, HTTPS is disabled and the remaining SSL-related keys have no effect.
SSL_PRIVATE_KEY SSL_SERVER_CERT SSL_CA_CERT	These keys are used to specify the full path and filename of the certificate to be used by the PowerShell Data Collector when HTTPS is enabled. If these keys are not set, a self-signed certificate will be generated when the container is started. When specifying the file name and path, they must be accessible to the PowerShell Data Collector. For example, the directory `/etc/ssl/certs` is mapped to the PowerShell Data Collector, meaning any files within this directory are accessible to the PowerShell Data Collector, subject to the files' permissions. Any certificates placed in the directory on the PowerShell Data Collector must have the keys set as follows:

Key	Value
	`SSL_PRIVATE_KEY=/etc/ssl/certs/my_cert.key` Once the key is set, the Data Collector will pick up the files in the directory on startup. **NOTE: USE_HTTPS** must be set to ***True*** for these keys to work.
SSL_VERIFY	When **USE_HTTPS** is set to ***True***, this key is used by Skylar One when communicating with the PowerShell Data Collector. By default, the value is ***False***. If set to ***True***, the HTTPS connection will fail if the Data Collector is using a self-signed certificate.

Enabling and Disabling the Python PowerShell Remoting Protocol Client

If you have concurrent PowerShell enabled in Skylar One, the default Python module used for transport to monitor Windows Devices is "pyWinRm". However, the "pypsrp" Python module can provide more efficient processing of PowerShell commands, particularly when virus detection software is enabled. To use the "pypsrp" module instead, run the following SQL query on the **Database Tool** page (System > Tools > DB Tool):

1. Select your database from the Select Database list.
2. Enter the following in the SQL Query field.

```
INSERT master.system_custom_config (field, field_value) VALUES ('enable_
pypsrp_lib', 1)
```

A value of 1 will enable the "pypsrp" module. A value of 0 (or not having any setting for 'enable_pypsrp_ lib') will revert to using "pyWinRM".

3. Click **[Go]**.

To disable "pypsrp", use the following SQL query:

```
UPDATE master.system_custom_config set field_value = 0 WHERE field =
'enable_pypsrp_lib'
```

NOTE: Currently, you can only use the "pypsrp" module with concurrent PowerShell. Classic PowerShell monitoring will continue to use the "pyWinRM" module regardless of this database setting.

Optional PowerShell CLI Parameters

You can use the following parameters in PowerShell for the associated reasons:

- ***-NoProfile***. Does not load the PowerShell profile.
- ***-NoLogo***. Hides the copyright banner at start-up.
- ***-NonInteractive***. Does not present an interactive prompt to the user.

To enable concurrent PowerShell collection to use one of these parameters:

1. Go to the **Database Tool** page (System > Tools > DB Tool).

   NOTE: The **Database Tool** page is available only in versions of Skylar One prior to 12.2.1 and displays only for users that have sufficient permissions to access the page.

2. If this row does not already exist in the `master.system_custom_config` table, enter the following in the ***SQL Query*** field:

   ```
   INSERT INTO master.system_custom_config ('field','field_value') VALUES ('powershell_prefix_setting', '<PREFIX>')
   ```

   where `<PREFIX>` is an integer that represents one of the prefix values described above. The integers are as follows:

   - **0**. Disabled
   - ***1***. -NoProfile
   - ***2***. -NoLogo
   - ***3***. -NoProfile and -NoLogo
   - ***4***. -NonInteractive
   - ***7***. -NoProfile, -NoLogo, and -NonInteractive

For example, if a user wanted to configure their PowerShell Data Collector to not load their PowerShell profile, they would enter the following into the ***SQL Query*** field:

```
INSERT INTO master.system_custom_config ('field','field_value') VALUES ('powershell_prefix_setting', '1')
```

3. If this row already exists in the `master.system_custom_config` table, enter the following in the ***SQL Query*** field:

   ```
   UPDATE master.system_custom_config SET field_value = 1 WHERE field = 'powershell_prefix_setting'
   ```

4. After you have entered the command in the ***SQL Query*** field, click the **[Go]** button. Your changes will be picked up with the next batch of jobs that are processed.

Users with Windows 2008 R2 Servers

Concurrent PowerShell collection will not work for Windows 2008 R2 servers when the ***Encyrpted*** field is set to *Yes* in the PowerShell credential. Windows 2008 R2 servers are no longer covered by Microsoft's Extended Support, but if you are still using those servers you have the following options:

- Use PowerShell credentials the have ***Encryption*** set to *No*.
- Disable the Concurrent PowerShell service on the Data Collector groups that include Windows 2008 R2 servers. This will reduce the number of servers that Data Collector group can support.
- Use the *Microsoft Base Pack* (WMI-based) PowerPack for the Windows 2008 R2 servers.
- Use SNMP for the Windows 2008 R2 servers.

Scale Recommendations

The following recommendations increase the number of Windows Servers the concurrent PowerShell collector can support:

- In the **Device Properties** page for all Windows Server devices (Devices > Classic Devices > wrench icon), unselect the ***Dynamic Discovery*** checkbox. Alternatively, this can be set in bulk using a device template and device group. This prevents nightly discovery from attempting to align Dynamic Applications with a discovery object to all the devices on the collector, which does not use the concurrent PowerShell collector and will dramatically limit the number of Windows Server devices that can be monitored.
- Do not select any credentials in the discovery session used to discover new Windows Servers. Instead, use a template that includes unselecting the ***Dynamic Discovery*** checkbox and includes the desired Dynamic Applications with the appropriate credential aligned. When a credential is selected in the Discovery Session, it will attempt to align Dynamic Applications that include a discovery object, which does not use the concurrent PowerShell collector and will dramatically limit the number of Windows Server devices that can be monitored. The *Microsoft: Windows Server* PowerPack includes the "Microsoft: Windows Server Discovery Template" that you can use be used to create your template.

Additional Scale Tips

- Limit the use of the "Microsoft: Windows Server Services" PowerPack, as using this Dynamic Application can reduce the number of servers a collector can support by 40%. If you do use this PowerPack, consider slowing down the ***Poll Frequency*** of the "Microsoft: Windows Server Services" Dynamic Application.
- Limit the use of the "Microsoft: Windows Server Event Logs" PowerPack as it does not work with the concurrent PowerShell collector.
- Use the *Microsoft: SQL Server* PowerPack instead of the *Microsoft: SQL Server Enhanced* PowerPack. The *Microsoft: SQL Server Enhanced* PowerPack does not work with the concurrent PowerShell collector.

- Disable Dynamic Applications that are not providing information required to meet your Service Level Agreements. There is an enhancement in caching included with concurrent PowerShell collection that will not send a PowerShell request from a cache-producing Dynamic Application unless at least one Dynamic Application is asking for that data. Disabling a cache-consuming Dynamic Application will also disable the cache producer from collecting that data. For example, the following Dynamic Applications are now disabled by default, as they are more diagnostic in nature and may not be required for routine monitoring:
    - Microsoft: Windows Server IPStats Performance
    - Microsoft: Windows Server TCPStats Performance
    - Microsoft: Windows Server UDPStats Performance
- Slow down the ***Poll Frequency*** for Dynamic Applications that do not include events. For example, the *Microsoft: Windows Server* PowerPack's Configuration Dynamic Applications used to be set to run every two hours and are now set to run every 12 hours.

Credentials for WMI and PowerShell Devices

This section describes how to configure credentials for WMI and PowerShell Dynamic Applications. It includes the following topics:

Configuring a WMI Credential

> **NOTE:** Although Skylar One supports WMI Dynamic Applications, ScienceLogic recommends that you use PowerShell Dynamic Applications where possible. PowerShell is the preferred management platform for Microsoft products.

If you configure your Windows system to respond to WMI requests from Skylar One, you can use WMI Dynamic Applications to collect information from your Windows system.

All of the WMI Dynamic Applications include a discovery object. If you include a credential for WMI Dynamic Applications in the discovery session that includes your Windows system, Skylar One will automatically align the appropriate WMI Dynamic Applications to the Windows system. For more information about creating a discovery session, see the ***Discovery & Credentials*** manual.

To create a credential for a WMI Dynamic Application:

1. Go to the **Credentials** page (Manage > Credentials).
2. Click the **[Create New]** button and then select *Create Basic/Snippet Credential*. The **Create Credential** modal page appears.
3. Supply values in the following fields:
    - ***Name***. Name of the credential. Can be any combination of alphanumeric characters, up to 64 characters. This field is required.

- ***All Organizations***. Toggle on (blue) to align the credential to all organizations, or toggle off (gray) and then select one or more specific organizations from the ***What organization manages this service?*** drop-down field to align the credential with those specific organizations. This field is required.

> **NOTE:** To learn more about credentials and organizations, see the section on "Aligning Organizations with a Credential" in the ***Discovery & Credentials*** manual.

- ***Timeout (ms)***. Time, in milliseconds, after which the platform will stop trying to communicate with the authenticating server.
- ***Username***. Username for a user account on the device. To specify a domain user, enter the username in the format DOMAIN\username. In most cases, you should use a domain user in the credential and use the format DOMAIN\username.
- ***Password***. Password for a user account on the device.
- ***Hostname/IP***. Hostname or IP address of the device from which you want to retrieve data. To use the same WMI default credential for multiple devices, enter the variable **%D** in this field.
- ***Port***. Port number associated with the data you want to retrieve. For WMI Dynamic Applications that perform WBEM requests, supply the port used by the WBEM service on the device. For WMI Dynamic Applications that perform WMI requests, which includes all default WMI Dynamic Applications in Skylar One, enter any valid port number in this field; the platform does not specify a port number when performing WMI requests.

4. Click **[Save & Close]**.

Configuring a WMI Credential in the Classic Skylar One User Interface

> **NOTE:** Although Skylar One supports WMI Dynamic Applications, ScienceLogic recommends that you use PowerShell Dynamic Applications where possible. PowerShell is the preferred management platform for Microsoft products.

If you configure your Windows system to respond to WMI requests from Skylar One, you can use WMI Dynamic Applications to collect information from your Windows system.

All of the WMI Dynamic Applications include a discovery object. If you include a credential for WMI Dynamic Applications in the discovery session that includes your Windows system, Skylar One will automatically align the appropriate WMI Dynamic Applications to the Windows system. For more information about creating a discovery session, see the ***Discovery & Credentials*** manual.

You can create a credential for WMI Dynamic Applications from the **Credential Management** page.

To create a credential for a WMI Dynamic Application in the classic Skylar One user interface:

1. Go to the **Credential Management** page (System > Manage > Credentials).
2. Click the **[Create]** button, then select *Basic/Snippet Credential*.
3. The **Credential Editor** page appears, where you can define the following fields:

- ***Credential Name***. Name of the credential. Can be any combination of alphanumeric characters.
- ***Hostname/IP***. Hostname or IP address of the device from which you want to retrieve data. To use the same WMI default credential for multiple devices, enter *%D* in this field.
- ***Port***. Port number associated with the data you want to retrieve. For WMI Dynamic Applications that perform WBEM requests, supply the port used by the WBEM service on the device. For WMI Dynamic Applications that perform WMI requests, which includes all default WMI Dynamic Applications in Skylar One, enter any valid port number in this field; the platform does not specify a port number when performing WMI requests.
- ***Timeout (ms)***. Time, in milliseconds, after which the platform will stop trying to communicate with the authenticating server.
- ***Username***. Username for a user account on the device. To specify a domain user, enter the username in the format DOMAIN\username. In most cases, you should use a domain user in the credential and use the format DOMAIN\username.
- ***Password***. Password for a user account on the device.

4. To save the credential, select the **[Save]** button. To clear the values you set, select the **[Reset]** button.

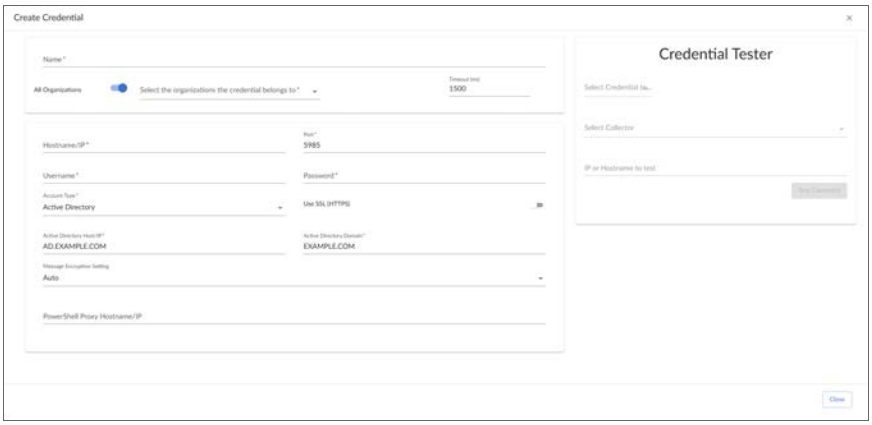
Configuring a PowerShell Credential

To define a PowerShell credential in Skylar One, you will need the following information:

- The username and password for a user on the Windows device.
- If the user is an Active Directory account, the hostname or IP address of the Active Directory server and the domain.
- Determine if an encrypted connection should be used.
- If you are using a Windows Management Proxy, the hostname or IP address of the proxy server.

To create a PowerShell credential:

1. Go to the **Credentials** page (Manage > Credentials).
2. Click the **[Create New]** button and then select *Create Powershell Credential*. The **Create Credential** modal page appears:



3. Supply values in the following fields:
    - ***Name***. Name of the credential. Can be any combination of alphanumeric characters, up to 64 characters. This field is required.
    - ***All Organizations***. Toggle on (blue) to align the credential to all organizations, or toggle off (gray) and then select one or more specific organizations from the ***What organization manages this service?*** drop-down field to align the credential with those specific organizations. This field is required.
    - ***Timeout (ms)***. Time, in milliseconds, after which Skylar One will stop trying to communicate with the authenticating server. For collection to be successful, Skylar One must connect to the authenticating server, execute the PowerShell command, and receive a response within the amount of time specified in this field.
    - ***Hostname/IP***. Hostname or IP address of the device from which you want to retrieve data. This field is required.
        - You can include the variable **%D** in this field. Skylar One will replace the variable with the IP address of the device that is currently using the credential.
        - You can include the variable **%N** in this field. Skylar One will replace the variable with the hostname of the device that is currently using the credential. If Skylar One cannot determine the hostname, Skylar One will replace the variable with the primary, management IP address for the current device.
        - You can include the prefix **HOST** or **WSMAN** before the variable **%D** in this field if the device you want to monitor uses a service principal name (for example, "HOST://%D" or "WSMAN://%D"). Skylar One will use the WinRM service HOST or WSMan instead of HTTP and replace the variable with the IP address of the device that is currently using the credential.
    - ***Port***. Type the port number used by the WinRM service on the Windows device. This field is required.
    - ***Username***. Type the username for an account on the Windows device to be monitored or on the proxy server. This field is required.

      NOTE: The user should not include the domain name prefix in the username for Active Directory accounts. For example, use "em7admin" instead of "MSDOMAIN\em7admin".

    - ***Password***. Type the password for the account on the Windows device to be monitored or on the proxy server. This field is required.
    - ***Account Type***. Type of authentication for the username and password in this credential. Choices are:
        - *Active Directory*. On the Windows device, Active Directory will authenticate the username and password in this credential.
        - *Local*. Local security on the Windows device will authenticate the username and password in this credential.

- ***Use SSL (HTTPS)*** / ***Encrypted***. Select whether Skylar One will communicate with the device using an encrypted HTTP or HTTPS connection:
    - Toggle on (blue) if Skylar One will communicate with the device using an encrypted connection over HTTPS. If toggled on, when communicating with the Windows server, Skylar One will use a local user account with authentication of type "Basic Auth". You must then use HTTPS and can use a Microsoft Certificate or a self signed certificate.

      NOTE: In Skylar One versions prior to 12.3.7, this field is labeled ***Encrypted***. In versions 12.3.7 and above, it is labeled ***Use SSL (HTTPS)***.

      NOTE: In Skylar One versions 11.3.0 and later, a newer Kerberos library is used that allows for message encryption over HTTP. This feature is on by default and may eliminate the need for you to configure an HTTPS certificate depending on your security requirements.

    - Toggle off (gray) . The credential is encrypted over HTTP rather than HTTPS.
- ***Validate Certificate (when HTTPS is used)***. This field is visible when the ***Use SSL (HTTPS)*** toggle field is enabled for the connection and allows you to select whether a certificate is validated for the credential. Choices are:
    - *Ignore*. Skylar One will not validate a certificate for the credential. This is the default setting.
    - *Validate*. Skylar One will require a validated certificate for the credential. If you select *Validate*, then the target device must include a non-expired certificate issued from a certificate authority.
- ***Active Directory Host/IP***. If you selected Active Directory in the ***Account Type*** field, type the hostname or IP address of the Active Directory server that will authenticate the credential.
- ***Active Directory Domain***. If you selected Active Directory in the ***Account Type*** field, type the domain where the monitored Windows device resides.
- ***Message Encryption Setting***. If you selected Active Directory in the ***Account Type*** field, select whether Kerberos packages sent over PowerShell Remoting Protocol (PSRP) or Windows Remote Management (WinRM) are encrypted. Choices are:
    - *Auto*. Encryption is enabled if the package supports it; otherwise, encryption is disabled. This is the default setting.
    - *Never*. Messages are never encrypted. If selected, the target device must support this option.
    - *Always*. Messages are always encrypted. If selected, the target device must support this option.
- ***PowerShell Proxy Hostname/IP***. If you use a proxy server in front of the Windows devices you want to communicate with, type the fully-qualified domain name or the IP address of the proxy server in this field.

4. Click **[Save & Close]**.

> **NOTE:** If you would like to test your credential using the Credential Tester panel, click **[Save & Test]**. For detailed instructions on using the Credential Tester panel, see the *Using the Credential Tester Panel* section.

Configuring a PowerShell Credential in the Classic Skylar One User Interface

To define a PowerShell credential in the classic Skylar One user interface:

1. Collect the information you need to create the credential:
    - The username and password for a user on the Windows device.
    - If the user is an Active Directory account, the hostname or IP address of the Active Directory server and the domain.
    - Determine if an encrypted connection should be used.
    - If you are using a Windows Management Proxy, the hostname or IP address of the proxy server.
2. Go to the **Credential Management** page (System > Manage > Credentials).
3. In the **Credential Management** page, click the **[Actions]** menu. Select ***Create PowerShell Credential***.
4. The **Credential Editor** page appears, where you can define the following fields:
    - ***Profile Name***. Name of the credential. Can be any combination of alphanumeric characters. This field is required.
    - ***Hostname/IP***. Hostname or IP address of the device from which you want to retrieve data. This field is required.
        - You can include the variable **%D** in this field. Skylar One will replace the variable with the IP address of the device that is currently using the credential.
        - You can include the variable **%N** in this field. Skylar One will replace the variable with the hostname of the device that is currently using the credential. If Skylar One cannot determine the hostname, Skylar One will replace the variable with the primary, management IP address for the current device.
        - You can include the prefix **HOST** or **WSMAN** before the variable **%D** in this field if the device you want to monitor uses a service principal name (for example, "HOST://%D" or "WSMAN://%D"). Skylar One will use the WinRM service HOST or WSMan instead of HTTP and replace the variable with the IP address of the device that is currently using the credential.
    - ***Username***. Type the username for an account on the Windows device to be monitored or on the proxy server. This field is required.

NOTE: The user should not include the domain name prefix in the username for Active Directory accounts. For example, use "em7admin" instead of "MSDOMAIN\em7admin".

- ***Encrypted***. Select whether Skylar One will communicate with the device using an encrypted connection. Choices are:
    - *yes*. When communicating with the Windows server, Skylar One will use a local user account with authentication of type "Basic Auth". You must then use HTTPS and can use a Microsoft Certificate or a self-signed certificate.
    - *no*. When communicating with the Windows server, Skylar One will not encrypt the connection.
- ***Port***. Type the port number used by the WinRM service on the Windows device. This field is automatically populated with the default port based on the value you selected in the ***Encrypted*** field. This field is required.
- ***Account Type***. Type of authentication for the username and password in this credential. Choices are:
    - *Active Directory*. On the Windows device, Active Directory will authenticate the username and password in this credential.
    - *Local*. Local security on the Windows device will authenticate the username and password in this credential.
- ***Timeout (ms)***. Type the time, in milliseconds, after which Skylar One will stop trying to collect data from the authenticating server. For collection to be successful, Skylar One must connect to the authenticating server, execute the PowerShell command, and receive a response within the amount of time specified in this field.
- ***Password***. Type the password for the account on the Windows device to be monitored or on the proxy server. This field is required.
- ***PowerShell Proxy Hostname/IP***. If you use a proxy server in front of the Windows devices you want to communicate with, type the fully-qualified domain name or the IP address of the proxy server in this field.
- ***Active Directory Hostname/IP***. If you selected Active Directory in the ***Account Type*** field, type the hostname or IP address of the Active Directory server that will authenticate the credential.
- ***Domain***. If you selected Active Directory in the ***Account Type*** field, type the domain where the monitored Windows device resides.

5. To save the credential, click the **[Save]** button. To clear the values you set, click the **[Reset]** button.

Example: Creating a WMI Performance Dynamic Application

In this example, we will create a WMI Dynamic Application. Our Dynamic Application will collect the following information from a network interface running on a Windows computer: Total bytes, current bandwidth, name, packets per second, outbound errors, and received errors.

> **NOTE**: This example Dynamic Application is included in the "Microsoft Base Pack" PowerPack, version 1.5 and later.

Defining the WMI Request

To create the Dynamic Application and define the general properties for this Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the **[Actions]** button, and then select *Create New Dynamic Application*. The **Dynamic Applications Create New Application** page appears.
3. Supply values in the following fields:
   - ***Application Name***. Enter *Windows Interface* in this field.
   - ***Application Type***. Select *WMI Performance*.
   - ***Poll Frequency***. Select *Every 5 Minutes*.
4. For this example, you can leave the remaining fields at their default value. Select the **[Save]** button to save the Dynamic Application.

Adding the WMI Request

In Skylar One, each WMI Dynamic Application must include at least one WMI or WBEM request.

WMI objects are populated when the Dynamic Application executes a WMI request. WMI requests use WQL (WMI Query Language) to query WMI classes (tables) to retrieve data. A single WMI request can populate multiple WMI objects by querying for multiple class properties (table columns).

WMI objects are aligned with properties (column). The definition of each object specifies the WMI request that will populate the object and the property name to align with the object. The retrieved values of the property will populate the object.

For more details on WMI requests, see the *WMI Requests* section.

To create the WMI request for this Dynamic Application:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon (🔧) for the *Windows Interface* Dynamic Application. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[WMI Requests]** tab. The **WMI Request Editor & Registry** page appears.
4. Supply values in the following fields:
    - ***WMI Request Name***. We named our WMI Request "Win32_PerfFormattedData_Tcpip_NetworkInterface"
    - ***WMI Request Type***. Select *WMI*.
    - ***WMI Object Key***. The unique key for each instance (row) returned by the request. This unique key must be a property (column) name, and the request must include that property (column) and return values from that property name (column).The selected property (column) must return the same values over all polling periods. The "Name" property (column) meets these criteria. Enter "Name" in this field.
    - ***Active State***. Select *Enabled*.
    - ***WMI Request Query***. This Dynamic Application is getting values from the Win32_PerfFormattedData_Tcpip_NetworkInterface class (table), and will collect the following values: Interface name, total bytes, current bandwidth, name, outbound errors, and received errors. We entered the following in the WMI Request Query:

      Select Name,BytesTotalPerSec,PacketsPerSec,CurrentBandwidth,PacketsOutboundErrors,PacketsReceivedErrors From Win32_PerfFormattedData_Tcpip_NetworkInterface
5. Select the **[Save As]** button to save the WMI Request.

Adding the Collection Objects

Our example Dynamic Application has six collection objects:

- Total Bytes per second
- Current Bandwidth
- Interface Name
- Packets per second
- Outbound errors
- Received errors

To create these collection objects, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon (🔧) for the *Windows Interface* Dynamic Application. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page appears.
4. To create the collection object for *Total Bytes*, supply values in the following fields:

- ***Object Name***. We named our collection object "Network Interface\Bytes Total/sec"
- ***WMI Request Argument***. In this field, you must specify the name of the property (table column) to associate with this object. Enter "BytesTotalPersec" in this field.
- ***Class Type***. Total bytes per second is a number that can go up or down between polls. Select *4 Performance Gauge* in this field.
- ***WMI Request***. Name of the WMI request associated with this object. Select *Win32_PerfFormattedData_Tcpip_NetworkInterface*.
- ***Group Number***. Select *Group 1*. For performance Dynamic Applications, Skylar One uses the ***Group Number*** setting to associate performance values with the appropriate labels. For the performance graph for this example to display labels correctly, all the collection objects must be in the same group.
- ***Description***. A description of the object. This is an optional field. We provided a summary of the object in this field.

5. For this example, you can leave the remaining fields set to their default values.
6. Select the **[Save]** button.
7. Select the **[Reset]** button to clear the form fields.
8. To create the following Collection Objects for Current Bandwidth, Packets Per Second, Outbound Errors, and Received Errors collection objects, repeat step 4, using the following values in the ***WMI Request Argument*** field. These values match the properties defined in the WMI Request for this Dynamic Application.

Collection Object	WMI Request Argument
Network Interface/Current Bandwidth	Current Bandwidth
Network Interface/Packets/sec	PacketsPerSec
Network Interface/Outbound Errors	PacketsOutbandErrors
Network Interface/Received Errors	PacketsReceivedErrors

9. To create the Interface Name collection object, which will be the label for the performance report, supply the following values in the **Dynamic Applications | Collections Objects** page:
    - ***Object Name***. We named this collection object "Network Interface\Name".
    - ***WMI Request Argument***. In this field, you must specify the name of the property (table column) to associate with this object. Enter "Name" in this field.
    - ***Class Type***. Select *Label (Always Polled)* in this field. In performance Dynamic Applications, collection objects that use this class type are string values that Skylar One uses to label the lines on a performance graph.
    - ***WMI Request***. Name of the WMI request associated with this object. Select *Win32_PerfFormattedData_Tcpip_NetworkInterface*.

- ***Group Number***. Select *Group 1*. For performance Dynamic Applications, Skylar One uses the ***Group Number*** setting to associate performance values with the appropriate labels. For the performance graph for this example to display labels correctly, all the collection objects must be in the same group.
- ***Description***. A description of the object. This is an optional field. We provided a summary of the object in this field.

10. For this example, you can leave the remaining fields set to their default values.
11. Select the **[Save]** button.

Creating the Presentation Objects

When you create a collection object in a Dynamic Application of type Performance, Skylar One automatically creates a presentation object that corresponds to that collection object. In this example, we will remove these presentation objects and create new presentation objects for each collection object defined above.

To create the presentation objects:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon () for the *Windows Interface* Dynamic Application.
3. Select the **[Presentations]** tab. The **Dynamic Applications Presentation Objects** page appears.
4. In the **Dynamic Applications Presentation Objects** page, each collection object created in the *Adding the Collection Objects* section has been created by default. Select each presentation object's delete icon () to delete them.
5. Select the **[Reset]** button. To create the presentation object that displays Total Bytes Per Second, enter values in the following fields:
   - ***Report Name***. Enter "Bytes Total per Second" in this field.
   - ***Active State***. Select *Enabled*. Skylar One will generate a report of the presentation object.
   - ***Data Unit***. Enter "Bytes/Second" into this field.
   - ***Abbreviation / Suffix***. Enter "Bps" into this field.
   - ***Show as Percent***. Select *No*. The graph will not display percent values.
   - '***Formula Editor***. In this field, enter the object ID for the Total Bytes Per Second collection object, surrounded by parentheses.
6. For this example, you can leave the remaining fields set to their default values.
7. Select the **[Save]** button to save the presentation object.
8. Select the **[Reset]** button. For the Current Bandwidth presentation object, enter the following values in the **Dynamic Applications Presentation Objects** page:
   - ***Report Name***. Enter "CurrentBandwidth" in this field.
   - ***Active State***. Select *Enabled*. Skylar One will generate a report of the presentation object.
   - ***Data Unit***. Enter "Bytes per second" into this field.
   - ***Abbreviation / Suffix***. Enter "Bps" into this field.
   - ***Show as Percent***. Select *No*. The graph will not display percent values.

- ***Vitals Link***. Select *Disabled*.
- ***Formula Editor***. In this field, enter the object ID for the Current Bandwidth collection object, surrounded by parentheses.

9. For this example, you can leave the remaining fields set to their default values. Select the **[Save]** button to save the presentation object.
10. Select the **[Reset]** button. For the Interface Utilization presentation object, enter the following values in the **Dynamic Applications Presentation Objects** page:
    - ***Report Name***. Enter "Interface Utilization" in this field.
    - ***Active State***. Select *Enabled*. Skylar One will generate a report of the presentation object.
    - ***Data Unit***. Enter "Percent" into this field.
    - ***Abbreviation / Suffix***. Enter "%" into this field.
    - ***Show as Percent***. Select *Yes*. The graph will display percent values.
    - ***Formula Editor***. In this field, enter the following formula:

      *((<object ID for Current Bandwidth> > 0) ? ((8 * <object ID for Bytes Total/sec>) / <object ID for Current Bandwidth>) : 0)*

      For example, if the object ID for Current Bandwidth is o_7034 and the object ID for Bytes Total/sec is o_7031, enter:

      *((o_7034 > 0) ? ((8 * o_7031) / o_7034) : 0)*

      This formula includes a collection object as a divisor. To prevent an error from occurring when the divisor returns zero, the formula includes a ternary operator that tests to see if the divisor is zero. If the divisor is zero, the formula returns zero. If the divisor is greater than zero, the formula converts the "Bytes Total/sec collection object in to Bits Total/sec, then divides the total bits/second by the speed of the interface.

11. For this example, you can leave the remaining fields set to their default values. Select the **[Save]** button to save the presentation object.
12. Select the **[Reset]** button. For the Packets Per Second presentation object, enter the following values in the **Dynamic Applications Manager** page:
    - ***Report Name***. Enter "Packets per Second" in this field.
    - ***Active State***. Select *Enabled*. Skylar One will generate a report of the presentation object.
    - ***Data Unit***. Enter "Packets/Second" into this field.
    - ***Abbreviation / Suffix***. Enter "P/s" into this field.
    - ***Show as Percent***. Select *No*. The graph will not display percent values.
    - ***Formula Editor***. In this field, enter the object ID for the Packets Per Second collection object, surrounded by parentheses.
13. For this example, you can leave the remaining fields set to their default values. Select the **[Save]** button to save the presentation object.
14. Select the **[Reset]** button. For the Outbound Errors presentation object, enter the following values in the **Dynamic Applications Manager** page:

- ***Report Name***. Enter "PacketsOutboundErrors" in this field.
- ***Active State***. Select *Enabled*. Skylar One will generate a report of the presentation object.
- ***Data Unit***. Enter "Errors" into this field.
- ***Abbreviation / Suffix***. Enter "Errors" into this field.
- ***Show as Percent***. Select *No*. The graph will not display percent values.
- ***Formula Editor***. In this field, enter the object ID for the Outbound Errors collection object, surrounded by parentheses.

15. For this example, you can leave the remaining fields set to their default values. Select the **[Save]** button to save the presentation object.
16. Select the **[Reset]** button. For the Inbound Errors presentation object, enter the following values in the **Dynamic Applications Manager** page:
    - ***Report Name***. Enter "PacketsInboundErrors" in this field.
    - ***Active State***. Select *Enabled*. Skylar One will generate a report of the presentation object.
    - ***Data Unit***. Enter "Errors" into this field.
    - ***Abbreviation / Suffix***. Enter "Errors" into this field.
    - ***Show as Percent***. Select *No*. The graph will not display percent values.
    - ***Formula Editor***. In this field, enter the object ID for the Inbound Errors collection object, surrounded by parentheses.
17. For this example, you can leave the remaining fields set to their default values. Select the **[Save]** button to save the presentation object.

Creating a Credential

To use the Windows Interface Dynamic Application, we must include a Basic/Snippet credential. To create the Basic/Snippet credential:

1. Go to the **Credential Management** page (System > Manage > Credentials).
2. Select the **[Create]** button, and then select *Basic/Snippet Credential*. The **Create New Basic/Snippet Credential** page appears.
3. Define values in the following fields:
    - ***Credential Name***. Enter "Windows Interface WMI" in this field.
    - **[Hostname/IP]**. Enter "%D" in this field. Skylar One will replace the variable with the IP address of the device that is currently using the credential.
    - ***Port***. Enter "1521" in this field. This is the default port for WMI.
    - ***Timeout***. Enter "5000" in this field. Skylar One will stop trying to communicate with the authenticating server after 5000 seconds.
    - ***Username***. Enter the username for a user account in this field that will provide access to the monitored Windows device.
    - ***Password***. Enter a password for a user account that will provide access to the monitored Windows device

4. Select the **[Save]** button to save the credential.

Manually Aligning the Dynamic Application to a Device

In this example we will align the Dynamic Application to a Windows device running WMI. By manually aligning the Dynamic Application to a device, we can immediately view the interface data in the presentation objects we defined.

To manually align the Dynamic Application to a device:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the device you want to align the Dynamic Application to. In this example, we are aligning the Dynamic Application to a Windows device running WMI. Select the device's wrench icon ( ).
3. The **Device Properties** page appears. Select the **[Collections]** tab.
4. In the **Dynamic Application Collections** page, select the **[Action]** button and select *Add Dynamic Application*. The **Dynamic Application Alignment** page appears.
5. Select the ***Windows Interface*** Dynamic Application in the **Dynamic Applications** pane, and select ***Windows Interface WMI*** in the **Credentials** pane.
6. Select the **[Save]** button to add the Dynamic Application.

Viewing the Performance Reports

After the Dynamic Application has collected the data specified in the collection objects, you can view the performance report for the device. To view the performance report for the device with the *Windows Interface* Dynamic Application aligned to it:

1. From the **Dynamic Application Collections** page, select the **[Reset]** button to update the page with the latest information.
2. Locate the ***Windows Interface*** Dynamic Application. If the graph icon ( ) is colored, the performance report is available. Select the graph icon for the presentation object you want to view.

Or:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the test device you aligned the *Windows Interface* Dynamic Application to. Select the device's graph icon ( ).
3. The **Device Summary** page appears. Select the **[Performance]** tab.
4. In the left NavBar, select any of the presentation objects you created in the section *Creating the Presentation Objects* in this chapter. In this example, we selected *Bytes Total per Second*.
5. The Windows Interface | Bytes Total per Second report is displayed.
    - The report displays the collected values from the collection object Network Interface\Bytes Total/sec.

- You can mouse over different data points on the report, and the report will display the total bytes moving through the interface at the time selected on the graph.
- The amount of bytes is shown to the left of the report.
- The values for each label object are displayed in the graph key at the bottom of the page.

6. To learn more about performance reports, see the manual ***Monitoring Device Infrastructure Health***.

Example: Creating a PowerShell Performance Dynamic Application

In this example, we will create a PowerShell Dynamic Application. Our Dynamic Application will collect the Processor Queue Length from a Windows computer.

> **NOTE**: This example Dynamic Application is included in the "Microsoft: Windows Server" PowerPack, version 1.0 and later. This example describes how to create only one of the requests, collection objects, and presentation objects that are included in the "Microsoft: Windows Server" PowerPack version of this Dynamic Application.

Creating the Dynamic Application

To create the Dynamic Application and define the general properties for this Dynamic Application, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the **[Actions]** button, and then select *Create New Dynamic Application*. The **Dynamic Applications Create New Application** page appears.
3. Supply values in the following fields:
   - ***Application Name***. Enter "Microsoft: Windows Server CPU Performance" in this field.
   - ***Application Type***. Select *PowerShell Performance*.
   - ***Polling Frequency***. Select *Every 5 Minutes*.
4. For this example, you can leave the remaining fields at their default value. Select the **[Save]** button to save the Dynamic Application.

Adding the PowerShell Command

In Skylar One, each PowerShell Dynamic Application must include at least one PowerShell Command.

The collection objects in a PowerShell Dynamic Application are populated when Skylar One executes a PowerShell Command.

Collection objects in PowerShell Dynamic Applications are aligned with properties (columns). The definition of each object specifies the PowerShell command that will populate the object and the property name to align with the object. The retrieved values of the property will populate the object.

To create the PowerShell command for this Dynamic Application:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon for the "Microsoft: Windows Server CPU Performance" Dynamic Application. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[PowerShell]** tab. The **PowerShell Command Editor and Registry** page appears.
4. Supply values in the following fields:
    - ***PowerShell Command Name***. We named our PowerShell Command "Server CPU Processor Queue-Length".
    - ***Active State***. Select *Enabled*.
    - ***PowerShell Command Query***. This Dynamic Application collects the CookedValue property from \System\Processor Queue Length: . We entered the following in the ***PowerShell Command Query*** field:

```
(Get-Counter "\System\Processor Queue Length").CounterSamples |
Select-Object CookedValue
```

5. Select the **[Save As]** button to save the PowerShell command.

Adding the Collection Object

Our example Dynamic Application has one collection object: Processor Queue Length.

To create the collection object, perform the following steps:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon () for the "Microsoft: Windows Server CPU Performance" Dynamic Application. The **Dynamic Applications Properties Editor** page appears.
3. Select the **[Collections]** tab. The **Dynamic Applications | Collections Objects** page appears.
4. To create the collection object for Processor Queue Length, supply values in the following fields:
    - ***Object Name***. We named our collection object "Processor Queue Length".
    - ***PowerShell Argument***. In this field, you must specify the name of the property to associate with this object. Enter "CookedValue" in this field.
    - ***Class Type***. Processor Queue Length is a number that can go up or down between polls. Select *4 Performance Gauge* in this field.
    - ***PowerShell Request***. Name of the PowerShell request associated with this object. Select *Server CPU Processor Queue Length*.
    - ***Group Number***. Select *No Group*. For performance Dynamic Applications, Skylar One uses the Group Number setting to associate performance values with the appropriate labels.
    - ***Description***. A description of the object. This is an optional field. We provided a summary of the object in this field.
5. For this example, you can leave the remaining fields set to their default values.
6. Select the **[Save]** button.

Creating the Presentation Object

When you create a collection object in a Dynamic Application of type Performance, Skylar One automatically creates a presentation object that corresponds to that collection object. In this example, we will edit the presentation object for the Processor Queue Length to create a new presentation object for the Processor Queue Length.

To create the Processor Queue Length presentation object:

1. Go to the **Dynamic Applications Manager** page (System > Manage > Applications).
2. Select the wrench icon () for the "Microsoft: Windows Server CPU Performance".
3. Select the **[Presentation]** tab. The **Dynamic Applications Presentation Objects** page appears.
4. In the **Dynamic Applications Presentation Objects** page, the Processor Queue Length collection object created in the *Adding the Collection Objects* section has been created by default. Select the Processor Queue Length presentation object's wrench icon () to edit it.
5. To create the presentation object that displays the Processor Queue Length, supply values in the following fields:
    - ***Report Name***. Enter "Processor Queue Length".
    - ***Active State***. Select *Enabled*. Skylar One will generate a report of the presentation object.
    - ***Data Unit***. Enter "Threads" into this field.
    - ***Abbreviation / Suffix***. Enter "Threads" into this field.
    - ***Show as Percent***. Select *No*. The graph will not display percent values.
6. For this example, you can leave the remaining fields set to their default values. Select the **[Save]** button to save the presentation object.

Creating a Credential

To use the "Microsoft: Windows Server CPU Performance" Dynamic Application, we must create a PowerShell credential. To create the PowerShell credential:

1. Go to the **Credentials** page (Manage > Credentials).
2. Click the **[Create New]** button and then select *Create Powershell Credential*. The **Create Credential** modal page appears.
3. Supply values in the following fields:
    - ***Name***. Enter a credential name in this field. We entered "PowerShell 2k12R2 [AD]" in this example.

- ***All Organizations***. Toggle on (blue) to align the credential to all organizations, or toggle off (gray) and then select one or more specific organizations from the ***What organization manages this service?*** drop-down field to align the credential with those specific organizations. This field is required.

> **NOTE:** To learn more about credentials and organizations, see see the section on "Aligning Organizations with a Credential" in the ***Discovery & Credentials*** manual.

- ***Timeout (ms)***. Enter "3000" in this field. Skylar One will stop trying to communicate with the authenticated server after 3000 ms.
- ***Hostname/IP***. Enter "%D" in this field. Skylar One will replace the variable with the IP address of the device that is currently using the credential.
- ***Port***. The port should be automatically selected after selecting a value in the ***Encrypted*** field.
- ***Username***. Enter the username for a user that will provide access to the monitored Windows device.
- ***Password***. Enter the password for the user account you entered in the ***Username*** field.
- ***Account Type***. Select the account type of the user that will provide access to the monitored Windows device.
- ***Use SSL (HTTPS)*** / ***Encrypted***. Select whether Skylar One will communicate with the device using an encrypted HTTP or HTTPS connection.

> **NOTE:** In Skylar One versions prior to 12.3.7, this field is labeled ***Encrypted***. In versions 12.3.7 and above, it is labeled ***Use SSL (HTTPS)***.

> **NOTE:** In Skylar One versions 11.3.0 and later, a newer Kerberos library is used that allows for message encryption over HTTP. This feature is on by default and may eliminate the need for you to configure an HTTPS certificate depending on your security requirements.

- ***Validate Certificate (when HTTPS is used)***. If you enabled the ***Use SSL (HTTPS)*** toggle field, select whether a certificate is validated for the credential. If you select *Validate*, then the target device must include a non-expired certificate issued from a certificate authority.
- ***Active Directory Host/IP***. If you are using an Active Directory user account, enter the hostname or IP address of the managed device's corresponding domain controller from the active directory forest.
- ***Active Directory Domain***. If you are using an Active Directory user account, enter your Active Directory domain.

- ***Message Encryption Setting***. If you are using an Active Directory user account, select whether Kerberos packages sent over PowerShell Remoting Protocol (PSRP) or Windows Remote Management (WinRM) are encrypted. Choices are *Auto*, *Never*, or *Always*.
- ***PowerShell Proxy Hostname/IP***. Do not enter a value this field unless you have configured a Windows device to serve as an intermediary proxy to retrieve PowerShell data from the target Windows device.

4. Click **[Save & Close]**.

Manually Aligning the Dynamic Application to a Device

In this example we will align the Dynamic Application to a Windows device that is configured for monitoring via PowerShell. By manually aligning the Dynamic Application to a device, we can immediately view the performance data in the presentation object we defined.

To manually align the Dynamic Application to a device:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the device you want to align with the Dynamic Application. In this example, we are aligning the Dynamic Application to a Windows device that is configured for monitoring via PowerShell. Select the wrench icon for the device ( ).
3. The **Device Properties** page appears. Select the **[Collections]** tab.
4. In the **Dynamic Application Collections** page, select the **[Action]** button and select *Add Dynamic Application*. The **Dynamic Application Alignment** page appears:
5. Select the "Microsoft: Windows Server CPU Performance" Dynamic Application in the **Dynamic Applications** field, and select the appropriate PowerShell credential in the **Credentials** field.
6. Select the **[Save]** button to add the Dynamic Application.

Viewing the Performance Report

After the Dynamic Application has collected the data specified in the collection objects, you can view the performance report for the device. To view the performance report for the device with the "Microsoft: Windows Server CPU Performance" Dynamic Application aligned to it:

1. From the **Dynamic Application Collections** page, select the **[Reset]** button to update the page with the latest information.
2. Locate the "Microsoft: Windows Server CPU Performance" Dynamic Application. If the graph icon is colored, the performance report is available. Select the graph icon ( ) for the presentation object you want to view.

Or:

1. Go to the **Device Manager** page (Devices > Classic Devices, or Registry > Devices > Device Manager in the classic user interface).
2. In the **Device Manager** page, find the test device you aligned the "Microsoft: Windows Server CPU Performance" Dynamic Application. Select the device's graph icon.
3. The **Device Summary** page appears. Select the **[Performance]** tab.

4. In the left NavBar, select the presentation object you created in the section *Creating the Presentation Objects*:
5. The Microsoft: Windows Server CPU Performance | Processor Queue Length report is displayed.
    - The report displays the collected values from the collection object Processor Queue Length.
    - You can mouse over different data points on the report, and the report will display the queue length value at the time selected on the graph.
    - The values for the Processor Queue Length label object are displayed in the graph key at the bottom of the page.

To learn more about performance reports, see the section on *Device Management*.

800-SCI-LOGIC (1-800-724-5644)

International: +1-703-354-1010