ScienceLogic

ServiceNow CMDB SyncPack

Version 3.6.3

Table of Contents

Chapter

1

Introduction to the ServiceNow CMDB SyncPack

Overview

This chapter gives an overview of the "ServiceNow Configuration Management Database (CMDB)" SyncPack, which lets you integrate SL1 with the ServiceNow Configuration Management Database (CMDB). This includes syncing SL1 organizations with ServiceNow companies, SL1 devices with ServiceNow CIs, and SL1 device details with ServiceNow CIs.

For more information about how SL1 integrates with ServiceNow, including uses cases, certifications, and customer feedback, see ServiceNow Integration at the ScienceLogic website.

This chapter covers the following topics:

What Can I Do with this SyncPack?

The "ServiceNow Configuration Management Database (CMDB)" SyncPack is the ScienceLogic integration with the ServiceNow Configuration Management Database (CMDB) Module. This SyncPack maintains and enhances the ServiceNow CMDB by discovering device information bi-directionally between SL1 and ServiceNow and by automatically maintaining ServiceNow Configuration Item (CI) relationships.

This SyncPack includes the following applications:

- **Organization Sync**. The "Sync Organizations from SL1 to ServiceNow" application syncs organizations from SL1 with ServiceNow companies. In this context, *sync* means that if you update a company in ServiceNow, the Organization Sync process will update the SL1 organization with that information, and vice versa. For more information, see *Syncing Organizations*.

- **Device Sync**. The "Sync Devices from SL1 to ServiceNow" application syncs devices and virtual device relationships from SL1 to ServiceNow. You can sync devices based on organization and collector group, and you can uses "mappings" to connect an SL1 device class to a ServiceNow CI class. You can also collect manufacturer/model attributes from asset records aligned with devices in SL1 and sync that information with ServiceNow. For more information, see *Syncing Devices from SL1 to ServiceNow*.

- **Business Service Sync**. The "Sync Business Services from ServiceNow to SL1" application syncs services that were defined in ServiceNow with business services in SL1, recreating the service structure in SL1, where you can see the relationships between all the components. For more information, see *Syncing Business Services*.

- **CI Attribute Sync**. The "Sync CI Attributes from ServiceNow to SL1" application imports CI attributes from ServiceNow to the relevant asset and attribute fields in SL1. The CI Attribute Sync supports assets, asset configuration, asset maintenance, location, production statuses, and custom attributes. For more information, see *Syncing CI Attributes from ServiceNow to SL1*.

- **Discovery Sync**. The Discovery Sync integration lets you use SL1 for discovering and syncing ServiceNow devices. With Discovery Sync, you start an SL1 discovery session from ServiceNow and then sync the newly discovered SL1 devices or virtual devices and their data with ServiceNow. For more information, see *Discovery Sync*.

- **File System Sync**. The "Sync File Systems from SL1 to ServiceNow" application reads file systems discovered in SL1 and then maps them to a parent CI record in ServiceNow. For more information, see *Syncing File Systems from SL1 to ServiceNow*.

- **Installed Software Sync**. The "Sync Software Packages from SL1 to ServiceNow" reads all software packages from SL1 and creates new CIs in ServiceNow. The "Sync Installed Software from SL1 to ServiceNow" then reads all available software packages from SL1 and the devices aligned to that software by region and syncs them with ServiceNow. For more information, see *Syncing Installed Software between SL1 and ServiceNow*.

- **Interface Sync**. The "Sync Interfaces from SL1 to ServiceNow" application collects interface data from ServiceNow and SL1 and runs multiple CI syncs for each interface to be synced. For more information, see *Syncing Interfaces from SL1 to ServiceNow*.

Contents of the ServiceNow CMDB SyncPack

This section lists the PowerFlow applications that are in the "ServiceNow CMDB" SyncPack.

> **NOTE:** Starting with version 3.5.0 of this SyncPack, the "Sync Advanced Topology from SL1 to ServiceNow" application was removed. The functionality in that application was added to the "Sync Devices from SL1 to ServiceNow" and the "Sync Interfaces from SL1 to ServiceNow" applications.

PowerFlow Applications

The following applications are included with the "ServiceNow CMDB" SyncPack:

- *Cache ServiceNow Companies, CIs and SL1 Orgs, Device Classes*. Reads all existing SL1 Organizations, SL1 Device Classes, ServiceNow Companies, and ServiceNow CIs and writes them to a cache. To perform a Device Sync, you must run this application before you run the "Sync Devices from SL1 to ServiceNow" application. Before version 3.5.0 of this SyncPack, this application was named "Cache ServiceNow CIs and SL1 Device Classes".

- *Create Custom Attributes and ServiceNow Custom Link in SL1*. Creates custom device attributes in SL1, which SL1 uses to create a custom link in SL1 that redirects to the ServiceNow CI. In SL1 the link appears as a *ServiceNow* link in the **Tools** menu on the **Device Investigator** window for the corresponding device.

- *Delete Devices from SL1*. Lets you delete devices in a specific SL1 Virtual Collector Group if those devices have not been modified in SL1 for a specified amount of time that is set in the application.

- *Generate Required CI Relations for ServiceNow*. Pulls device class mappings from the "Sync Devices from SL1 to ServiceNow" and the "Sync CI Attributes from ServiceNow to SL1" applications to prevent you from having to add a separate set of class mappings. The application also lists any missing relationships in the Step Log in the PowerFlow user interface. Please note that this application is a report used by PowerFlow, and it does not send any data to ServiceNow.

- *Report: Identify Unmapped Devices Classes*. Pulls the class mappings from Device Sync and Attribute Sync and compares the mappings with the full list of device classes of discovered devices in SL1. The application generates a report on the **Reports** page that lists missing mappings, and if any device classes are unmapped, the application generates an event in the target SL1 system.

- *Sync Business Services from SL1 to ServiceNow*. Reads Business Services, IT Services, and Device Services in SL1 and syncs them with business services in ServiceNow. This application creates and updates services, but it does not delete services.

- *Sync Business Services from ServiceNow to SL1*. Syncs services that were defined in ServiceNow with Business Services in SL1. By syncing services from Service Now to SL1, you can see the relationships between the service components, the application components, and the infrastructure components in SL1.

- *Sync CI Attributes from ServiceNow to SL1*. Reads CI attributes from ServiceNow and maps those attributes to asset and attribute fields in SL1. This application uses the mappings and additional attributes options from the "Sync Devices from SL1 to ServiceNow" application. This application can also sync the location and production state attributes from ServiceNow to SL1.

- *Sync Device Groups from SL1 to ServiceNow*. Collects all device groups and group IDs from SL1 and posts device group data to ServiceNow. To prevent errors when running this application or a device sync, make sure that the device group names are not already being used by existing groups in ServiceNow.

- *Sync Devices from SL1 to ServiceNow*. Syncs devices and their properties and relationships from SL1 to ServiceNow.

- *Sync Discovery Requirements*. Processes credentials from SL1, processes collector groups, device templates, virtual device classes, and collectors, and then syncs organizations and device groups.

- *Sync Discovery Session Status from SL1 to ServiceNow*. Collects and processes Discovery sessions from SL1, and collects Discovery session logs.

- *Sync Discovery Templates from SL1 to ServiceNow*. Syncs SL1 discovery sessions that contain a configured string to ServiceNow and creates Service Catalog templates in ServiceNow. You can use those templates for discovering or monitoring CIs.

- *Sync File Systems from SL1 to ServiceNow*. Reads file systems discovered in SL1 and then maps them to a parent CI record in ServiceNow.

- *Sync Installed Software from SL1 to ServiceNow*. Reads all available software packages from ServiceNow and the devices aligned to that software by region and syncs them with SL1. Run this application after running the "Sync Software Packages from SL1 to ServiceNow" application.

- *Sync Interfaces from SL1 to ServiceNow*. Collects network interface data from ServiceNow and SL1, and then runs multiple CI syncs for each interface to be synced.

- *Sync Organizations from SL1 to ServiceNow*. Pulls organizations from SL1 and syncs to ServiceNow.

- *Sync Service Requests from ServiceNow to SL1*. Processes Discovery sessions and posts Discovery sessions and new virtual devices to SL1. Also enables device decommissioning for devices you no longer want to monitor. This application was formerly named "Sync Discovery Session Requests from ServiceNow to SL1".

- *Sync Software Packages from SL1 to ServiceNow*. Reads all software packages from and creates new CIs in ServiceNow. Run this application before running the "Sync Installed Software" application.

PowerFlow Applications (Internal)

To view the internal PowerFlow applications, click the Filter icon ( ) on the **Applications** page and select *Show Hidden Applications*. Internal applications are hidden by default. The following applications are "internal" applications that should not be run directly, but are automatically run by applications from the previous list:

- *Bulk Delete Devices*. Deletes devices from SL1.

- *Create Discovery Session in SL1*. Creates and starts a Discovery session in SL1 and updates the ServiceNow service request.

- *Create ServiceNow CI*. Creates a new ServiceNow CI with a mappings dictionary, but does not attempt to look up new CIs.

- *Create Virtual Device in SL1*. Creates a virtual device in SL1 and updates the Requested Item (RITM) value.

- *Post Attribute DB Calls to SL1*. Posts attribute database calls to SL1.

- *Post Attribute Rest Calls to SL1*. Posts attribute REST calls to SL1.

- *Post Company and Organization Updates*. Posts company and organization updates to ServiceNow or SL1.

- *Post Discovery-dependent Data to ServiceNow*. Posts data used by a Discovery session to ServiceNow.

- *Post Installed Software to ServiceNow*. Posts installed software data to ServiceNow.

- *Post New Companies to ServiceNow*. Posts new companies to ServiceNow.

- *Post New Organization to SL1*. Posts a new organization to SL1.

- *Process Remove Device Requests from ServiceNow to SL1*. Pulls requested device information form SL1 and validates the requests to remove a device from monitoring. Removed devices are placed in an SL1 Virtual Collector Group.

- *Pull and Post Discovery Logs*. Pulls Discovery session logs from SL1 and posts updates to ServiceNow.

Chapter

2

Installing ServiceNow CMDB SyncPack

Overview

This chapter explains how to install and configure the various applications used by the "ServiceNow CMDB" SyncPack.

The following workflow covers how to install and configure this SyncPack:

1. *Review the prerequisites and background information for this SyncPack*.

2. *In PowerFlow, download, import, and install the "ServiceNow CMDB" SyncPack*.

3. *In ServiceNow, enable cross-scoped access* (if needed).

4. *In ServiceNow, install the "ScienceLogic SL1: CMDB & Incident Automation" application* (also called the "Scoped Application").

5. *In ServiceNow, create a ServiceNow Group* and then *create a ServiceNow User* .

6. *In ServiceNow, install and activate the relevant plugins*.

7. *In ServiceNow, enable the ServiceNow Identification and Reconciliation Module*.

8. *In ServiceNow, install the "ScienceLogic Domain Separation (Global)" update set* (for domain-separated ServiceNow instances only).

Prerequisites for the ServiceNow CMDB SyncPack

This section describes the prerequisites for the ServiceNow SyncPacks. For more information about the specific software versions required by a ServiceNow SyncPack, see the *release notes* for that SyncPack.

Administrator Access

To install this SyncPack, you must have administrator access to both SL1 and ServiceNow. Specifically, you will need:

- ScienceLogic root SSH access
- ScienceLogic administrator access to the Administration Portal
- ServiceNow administrator access

> **NOTE:** ScienceLogic highly recommends that you disable all firewall session-limiting policies. Firewalls will drop HTTPS requests, which results in data loss.

Port Access

The following table lists the port access required by PowerFlow and this SyncPack:

Source	Port	Requirement
SL1 API	443	SL1 API Access
ServiceNow API	443	ServiceNow API Access
SL1 Database	7706	SL1 Database Access

Architecture Overview for ServiceNow SyncPacks SL1

The following diagram details the various elements that are contained in SL1 and the PowerFlow system, and how PowerFlow sits between the core SL1 platform and an external data platform:
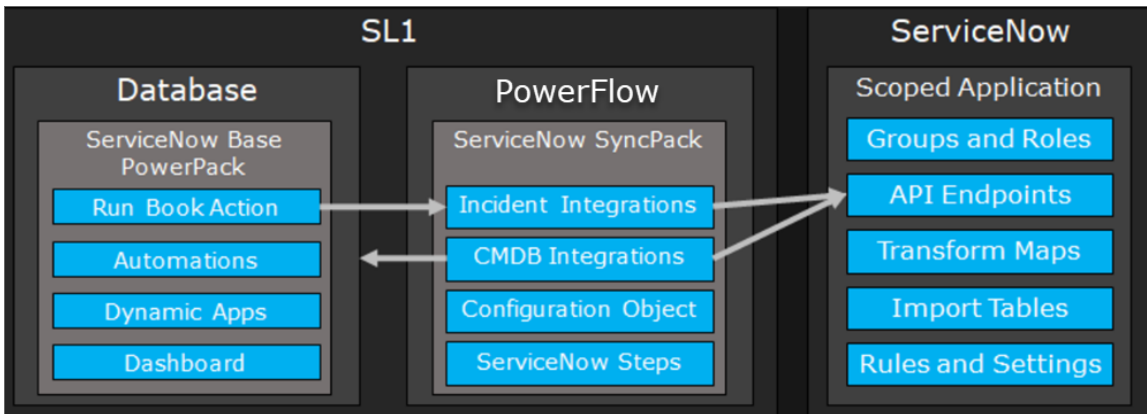
The following diagram provides an example of the high-level architecture of a PowerFlow system with High Availability, Disaster Recovery, and a proxy configured:
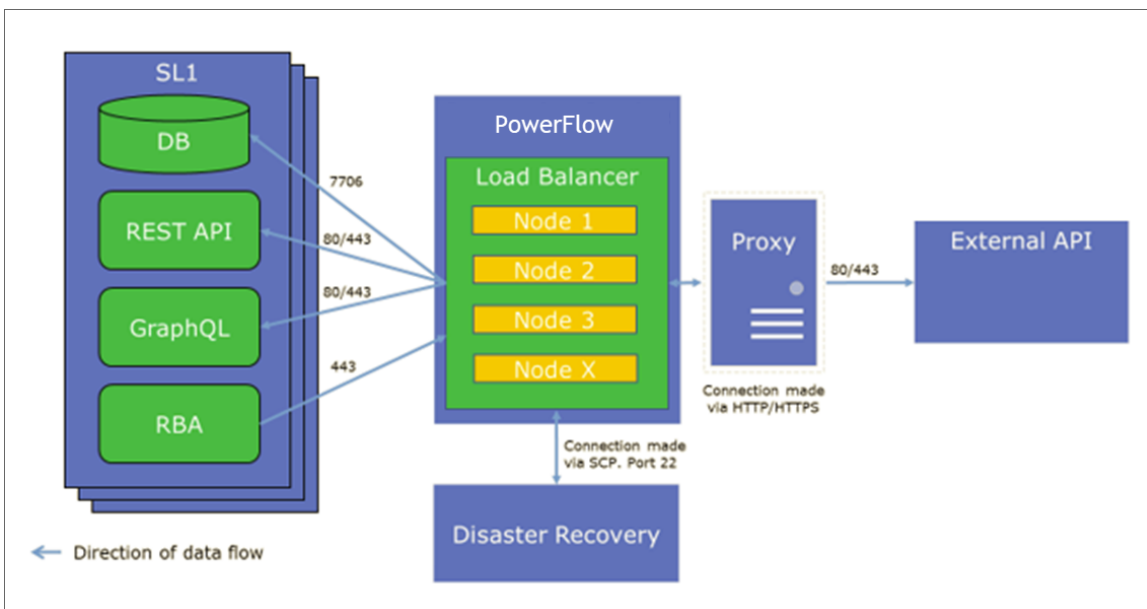
SL1 and ServiceNow Terminology

The following table lists the different names for the shared elements in SL1 and ServiceNow:

SL1	ServiceNow
Asset, Custom Attribute	Asset (ITAM)
Device	CI (Configuration Item)
Discovery Session	Service Request, Catalog Request
Event	Incident, Event, or Case (depending on the SyncPack you are using)
Alert	Event
Organization	Company, Domain
Schedule, Maintenance Schedule	Change Request, Change Schedule
Topology, Relationships, Dynamic Component Mapping and Relationships (DCM+R)	Dependency View, Affected CIs

Dependency Map for ServiceNow SyncPacks

The following graphic describes which SyncPacks depend on other SyncPacks:
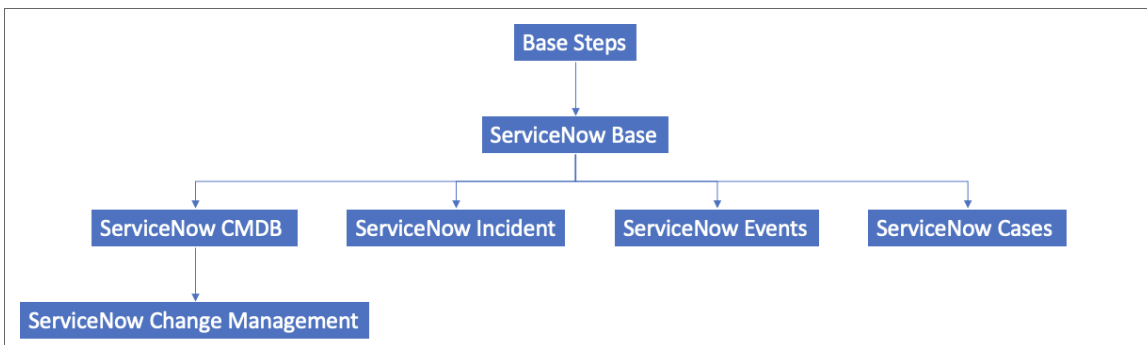
> **TIP:** For more information about the "Base Steps" SyncPack, see the **SL1 PowerFlow Platform** manual.

Downloading, Importing, and Installing the SyncPack

A SyncPack file has the **.whl** file extension type. You can download the SyncPack file from the ScienceLogic Support site.

> IMPORTANT: If you are planning to *upgrade* from an older version of this SyncPack to version 3.5.x, see ***Upgrading to Version 3.5.x***.

Downloading the SyncPack

> NOTE: If you are installing or upgrading to the latest version of this SyncPack in an offline deployment, see ***Installing or Upgrading in an Offline Environment*** to ensure you install any external dependencies.

To locate and download the SyncPack:

1. Go to the ScienceLogic Support Site at https://support.sciencelogic.com/s/.

2. Click the **[Product Downloads]** tab and select *PowerPacks & SyncPacks*.

3. In the *Search* field, search for the SyncPack and select it from the search results. The **Release Version** page appears.

4. On the **[Files]** tab, click the down arrow next to the SyncPack version that you want to install, and select *Show File Details*. The **Release File Details** page appears.

5. Click the **[Download File]** button to download the SyncPack.

After you download the SyncPack, you can import it to your PowerFlow system using the PowerFlow user interface.

Importing the SyncPack

You must import and install the "ServiceNow Base" SyncPack before uploading and installing any of the other ServiceNow SyncPacks.

If you are upgrading to version 3.5.0 or later of this SyncPack, be sure to install and activate version 1.5.0 or later of the "Base Steps" SyncPack. Older versions of the "Base Steps" SyncPack might cause installation errors for this SyncPack.

To import a SyncPack in the PowerFlow user interface:

1. On the **SyncPacks** page (☺) of the PowerFlow user interface, click **[Import SyncPack]**. The **Import SyncPack** page appears.

2. Click **[Browse]** and select the **.whl** file for the SyncPack you want to install. You can also drag and drop a **.whl** file to the **Import SyncPack** page.

3. Click **[Import]**. PowerFlow registers and uploads the SyncPack. The SyncPack is added to the **SyncPacks** page.

4. You will need to activate and install the SyncPack in PowerFlow. For more information, see the following topic.

> **NOTE**: You cannot edit the content package in a SyncPack published by ScienceLogic. You must make a copy of a ScienceLogic SyncPack and save your changes to the new SyncPack to prevent overwriting any information in the original SyncPack when upgrading.

Installing the SyncPack

> **WARNING**: If you are *upgrading* to this version of the SyncPack from a previous version, make a note of any settings you made on the **Configuration** pane of the various PowerFlow applications in this SyncPack, as these settings are *not* retained when you upgrade. However, any mappings you added to the **attribute_mappings** section for the "Sync Devices from SL1 to ServiceNow" application *are* retained when you upgrade.

To activate and install a SyncPack in the PowerFlow user interface:

1. On the **SyncPacks** page of the PowerFlow user interface, click the **[Actions]** button (⋮) for the SyncPack you want to install and select *Activate & Install*. The **Activate & Install SyncPack** modal appears.

   > **NOTE**: If you try to activate and install a SyncPack that is already activated and installed, you can choose to "force" installation across all the nodes in the PowerFlow system.

   > **TIP**: If you do not see the SyncPack that you want to install, click the Filter icon (☰) on the **SyncPacks** page and select *Toggle Inactive SyncPacks* to see a list of the imported PowerPacks.

2. Click **[Yes]** to confirm the activation and installation. When the SyncPack is activated, the **SyncPacks** page displays a green check mark icon (✅) for that SyncPack. If the activation or installation failed, then a red exclamation mark icon (❗) appears.

3. For more information about the activation and installation process, click the check mark icon (✅) or the exclamation mark icon (❗) in the **Activated** column for that SyncPack. For a successful installation, the "Activate & Install SyncPack" application appears, and you can view the Step Log for the steps. For a failed installation, the **Error Logs** window appears.

4. If you have other versions of the same SyncPack on your PowerFlow system, you can click the **[Actions]** button (⋮) for that SyncPack and select *Change active version* to activate a different version other than the version that is currently running.

Installing or Upgrading in an Offline Deployment

Use the following procedure if you are installing or upgrading this SyncPack in an offline deployment.

Downloading, Importing, and Installing the SyncPack

This release of the SyncPack requires the following external files for an offline deployment:

- **Jinja2-2.11.1-py2.py3-none-any.whl**, available at this location at pypi.org.
- **MarkupSafe-1.1.1-cp37-cp37m-manylinux1_x86_64.whl**, available at this location at pypi.org.

To upload the required external files:

1. After downloading the two required external files, SCP the files to the master node.

2. Run the following commands on the host:

```
devpi use 'https://<is_username>:<is_password>@<is_hostip>:3141/isadmin/dependencies'

devpi login <is_username> --password=<is_password>

cd /tmp/

devpi upload Jinja2-2.11.1-py2.py3-none-any.whl

devpi upload MarkupSafe-1.1.1-cp37-cp37m-manylinux1_x86_64.whl –force
```

> **NOTE**: If you cannot run these commands on the host, you can instead run them on the pypiserver container.

3. Perform a docker copy to both files (if you are running commands in the pypiserver container):

```
docker cp <file1-location> $(docker ps -q -f name=iservices_pypiserver):/tmp

docker cp <file2-location> $(docker ps -q -f name=iservices_pypiserver):/tmp
```

4. Follow the steps in *Installing the Synchronization PowerPack* to install and activate this SyncPack.

Allowing Cross-Scoped Access in ServiceNow

When using custom tables, you might need to configure cross-scope access for the ScienceLogic plugin. The following examples contain errors that might occur when cross-scope access is required.

Example of an API response:

```
{"results":[{"error":
{"message":"com.glide.script.fencing.access.ScopeAccessNotGrantedExceptio
n: read access to ui_test_hardware not
granted","detail":""},"status":"failure"}
```

Example of navigating to a URL directly from a web browser when cross-scope access is required:

This page contains the following errors:

error on line 1 at column 1: Document is empty

Below is a rendering of the page up to the first error.

In this example, the table requires that you grant access to the ScienceLogic Scope to allow the API call to run correctly. In the above example, the target table is **u_test_hardware**.

> **NOTE:** A ServiceNow account with System Administrator is required.

To grant access to the ScienceLogic Scope in ServiceNow:

1. Log in to your ServiceNow instance.

2. For newer releases of ServiceNow, click the globe icon (🌐) to change the scope. On older releases, click the **Settings** icon (⚙️) and select the **Developer** tab. The **Developer System Settings** window appears.

3. From the **Application** drop-down list, select *ScienceLogic ServiceNow Integration*.

4. Close the **Developer System Settings** window and navigate to the **Cross scope privileges** page (System Applications > Application Cross-Scope Access). Make sure you are in the "ScienceLogic ServiceNow Application" scope and track these updates in an update set.

5. Click the **[New]** button to create a new record on the **Cross scope privileges** page:

6. Verify that the *Source Scope* and *Application* fields are set to *ScienceLogic ServiceNow Integration*. If they are not, repeats steps 2-3.

7. Complete the following fields:

    - *Target Scope*. Specify the scope of the target table, such as *Global*. Be sure to verify the application to which the table belongs, and use that value as the target scope in this field.

    - *Operation*. Select *Read*.

    - *Target Name*. Specify the name of the target table.

    - *Status*. Select *Allowed*.

    - *Target Type*. Select *Table*.

8. Click the **[Submit]** button.

For more information, see the [Cross-scope privilege record](#) topic in the ServiceNow documentation.

Installing the "ScienceLogic SL1: CMDB & Incident Automation" Application in ServiceNow

You must install the "ScienceLogic SL1: CMDB & Incident Automation" application on the ServiceNow instance to enable this SyncPack. The "ScienceLogic SL1: CMDB & Incident Automation" application is also known as the "Certified" or "Scoped" application.

Version 1.0.76 and later of the "ScienceLogic SL1: CMDB & Incident Automation" certified application includes updates to the ServiceNow Identification & Reconciliation Engine (IRE). As a result, the latest "ScienceLogic Domain Separation (Global)" update set is required if your ServiceNow instance is domain-separated.

> **NOTE:** Ask your ScienceLogic contact for access to this update set.

In version 1.0.76 and later of the "ScienceLogic SL1: CMDB & Incident Automation" application, the Incident portion of the application was turned off by default as part of an ongoing effort to make each individual application easier to support. Please note that the none of the Incident content was removed, but it is no longer enabled by default.

- For users of the "ServiceNow CMDB" SyncPack, there will be no impact, as the Incident portion is turned off by default in version 1.0.76 of the "ScienceLogic SL1: CMDB & Incident Automation" application. For more information about turning Incident features on and off, see below.

- ScienceLogic strongly recommends that users of the "ServiceNow Incident" SyncPack migrate to the "ScienceLogic SL1: Incident Automation" application. You can download this application from the ServiceNow Store at https://store.servicenow.com.

- In future releases of the "ServiceNow Incident" SyncPack, you will need to download and install the "ScienceLogic SL1: Incident Automation" application to get any updates to the Incident module. The two SyncPacks and their corresponding applications will be completely separate going forward.

Requesting and Installing the Certified Application

You must first request the "ScienceLogic SL1: CMDB & Incident Automation" application from the ServiceNow Store, and then you can install it. You must have a ServiceNow HI Service Account to request this application and download it onto your ServiceNow instance.

To request and install the certified application:

1. Go to the ServiceNow Store at https://store.servicenow.com and search for "ScienceLogic SL1".

2. Select the "ScienceLogic SL1: CMDB & Incident Automation" application. The detail page for the application appears.

3. Click the **[Get]** button and log in with your HI credentials.

4. After the request is approved, log in to ServiceNow as an administrator and navigate to **Application Manager** (System Applications > Applications or My Company Applications).

5. Click **[Downloads]** in the menu header or search for "ScienceLogic".

6. Click the version drop-down for the "ScienceLogic SL1: CMDB & Incident Automation" application listing to make sure you are using the correct version of the application that is compatible with your version of this SyncPack.

7. Click the **[Install]** button for the application. The installation is complete when the button changes to **[Installed]**.

8. In the filter navigator, search for "ScienceLogic" and locate the application in the left-hand navigation menu to verify that the application was installed. You might need to log out of ServiceNow and log in again to see the updated left-hand navigation menu.

Incident Features Turned Off in Version 1.0.76 of the Certified Application

The following elements in version 1.0.76 of the "ScienceLogic SL1: CMDB & Incident Automation" application were turned off by default.

Table Transform Maps

To reactivate the following transforms:

1. In ServiceNow, go to **Systems Import Sets > Administration > Transform Maps**.

2. Search for *ScienceLogic Incident* or *ScienceLogic Event*.

3. For ScienceLogic Incident and ScienceLogic Event, the **Active** field was unchecked. To continue to use one or both of these transforms, check the **Active** box.

Application Modules

To reactivate the following application modules so they display again in the Application Menu:

1. In ServiceNow, go to **System Definition > Application Menus**.

2. Search for *ScienceLogic* with filter *Application = ScienceLogic ServiceNow Integration*.

3. You can reactivate the following application modules in the module Related Lists section:

   - *Event*: To continue to use the Module, check the **Active** box.

   - *Event Properties*: To continue to use the Module, check the **Active** box.

   - *Events*: To continue to use the Module, check the **Active** box.

   - *Severity Lookup Rules*: To continue to use the Module, check the **Active** box.

Creating a ServiceNow Group

For best practice and security, create a dedicated ServiceNow account that has restricted access to only the groups, access control lists (ACLs), and roles needed for ScienceLogic integration.

To create a ServiceNow Account for ScienceLogic Incident management:

1. In ServiceNow, go to the **Groups** page (System Security > Users and Groups > Groups) and click **[New]**. A **New record** page appears.

2. In the **New record** page, type the group name and any additional information. *Name* is the only required field.

3. Click **[Submit]**.

4. Select the new group from the **Groups** page, and at the bottom of the Group record, select the **[Roles]** tab and click **[Edit]**.

5. Search for *x_sclo_scilogic.Admin* and move it to the **Roles List** column using the arrow buttons.

6. Click **[Save]**. Your ServiceNow Group now has an assigned Role.

7. Next, create a ServiceNow user to use with this Group. See the following procedure for the details.

Creating a ServiceNow User

NOTE: The ServiceNow user you create in this procedure will *not* be able to log into the ServiceNow user interface with the username and password you give this user. However, you will use the username and password in the relevant configuration objects in the PowerFlow user interface to run applications. For more information about configuration objects, see *Creating and Aligning a Configuration Object*.

To create a ServiceNow Account:

1. In ServiceNow, go to the **Users** page (System Security > Users and Groups > Users) and click **[New]**. A **New record** page appears.

2. Complete the following fields:

    - *User ID*. Type a user ID. Required.

    - *First Name*. Type the user's first name.

    - *Last Name*. Type the user's last name.

    - *Password*. Type a password. Required.

    - *Active*. Select this checkbox. Required.

    - *Web Service Access Only*. Select this checkbox. Required.

    - *Time Zone*. Select *GMT*. Required.

    - *Date Format*. Select *System (yyyy-MM-dd)*.

3. Right-click the gray header and click *Save* to save the user.

4. Select the **[Groups]** tab at the bottom of the record and click the **[Edit]** button.

5. Find the group you created previously and move the group to the right-hand column using the arrow buttons.

6. Click **[Save]**. After the user has been added to the group, you can see their Roles and Groups at bottom of the record.

> **NOTE**: As a best practice, you should use a non-administrator ServiceNow user for the PowerFlow
> configuration object.

Installing and Activating ServiceNow Plugins

This SyncPack requires the "ServiceNow Configuration Management for Scoped Apps (CMDB)" plugin (com.snc.cmdb.scoped), which gives the ScienceLogic scoped application access to the Identification Engine APIs. You will need to install this plugin on your ServiceNow instance.

In addition, ScienceLogic strongly recommends that you install the "CMDB CI Class Models " plugin (sn_cmdb_ci_class). This plugin contains all new class models provided by ServiceNow.

To install and activate plugins in ServiceNow, you must have ServiceNow Instance administrator rights.

To install one or both of the plugins in ServiceNow:

1. In ServiceNow, log in and navigate to **Plugins** (System Definition > Plugins). This page is only available with administrative rights.

2. Search for a plugin by its name or its ID (such as sn_cmdb_ci_class) and select the plugin. The options to **Install** or **Update** appear, depending on the status of the plugin:

   - **Install**. Installs the plugin in ServiceNow if it has not been installed before. After you install the plugin, the button is grayed out and the text changed to **Installed**.

   - **Update**. Shows that newer versions of the plugin are available.

3. Click **Install** or **Update** to complete the installation.

Enabling the ServiceNow Identification and Reconciliation Module

This SyncPack uses the "ServiceNow Identification and Reconciliation" module to create and de-duplicate CI records. PowerFlow builds a JSON-formatted string that is sent to the "ServiceNow Identification and Reconciliation" module. The following link provides additional detail about the formatting of the JSON-formatted string: IdentificationEngineScriptableApi.

> **IMPORTANT:** Version 3.5.0 and later of this SyncPack uses enhanced versions of the IdentificationEngine API: createOrUpdateCIEnhanced and identifyCIEnhanced.

The JSON-formatted string is sent directly to a custom-scripted API endpoint and run through the IdentificationEngineScriptable API. Identification (Insert or Update) of Configuration Items (CIs) is handled by the ServiceNow Identification and Reconciliation module.

For more information about how SL1 and ServiceNow work with the ServiceNow Identification and Reconciliation module to discover and module other applications, such as VMware, see *Mappings between SL1, ServiceNow, and Other Applications*.

For more information about the "ServiceNow Identification and Reconciliation" module, see [CMDB Identify and Reconcile](). See also [Reconciliation Rules](), [CMDB Identification Rules](), and [Identification engine error messages]().

> **NOTE**: As a best practice, do not use information for identification that is specific to only ScienceLogic. An example is the device ID (DID) | SL1 ID, which is a number that SL1 automatically assigns and is only unique to that specific SL1 system.

Configuring Service Rules for Device Sync

> **NOTE**: The default device relationships are based on ServiceNow defaults. Hardware defaults to SNMP OID Classifications (**discovery_snmp_oid**). vCenter, File systems and network adapters are all matched against ServiceNow Discovery.

The *ServiceNow CMDB* SyncPack utilizes class hierarchies to build relationships in ServiceNow. This requires building **service rules** (containment rules and hosting rules) in ServiceNow to correctly identify dependent CIs during the business discovery process and service mapping. **Containment rules** describe which CIs are contained by a given CI. **Hosting rules** describe the environment on which a CI runs.

ScienceLogic recommends packaging all of the service rules into a ServiceNow update set so that you can be easily package and deploy these changes across environments. For more information, see *Creating a ServiceNow Update Set*.

These rules or "mappings" are defined in the "Sync Devices from SL1 to ServiceNow" application in the PowerFlow user interface. These mappings connect an **SL1 device class** to a **ServiceNow CI class**, which determines the CI class that ServiceNow uses when creating the CI in ServiceNow.

For more information about the "ServiceNow Identification and Reconciliation" module, see [CMDB dependent relationship rules]() and [CMDB Identification Rules]() at the ServiceNow website.

For example, if you experience error messages about missing relationships in ServiceNow when you run the "Sync Devices from SL1 to ServiceNow" application in the PowerFlow user interface, you might be missing certain containment rules or mappings that are needed to complete the export process:

Containment Rules

Containment rules are chained to each other in a group, with a CI type that is the top-level (root) parent of the group.

To create containment rules:

1. In ServiceNow, type "cmdb_metadata_containment.list" in the filter navigator to access the **CMDB Metadata Containment Rules** page.

2. Click **[New]**. A new **CMDB Metadata Containment Rules** record appears.

3. Complete the following fields:

   - *Configuration item class*. Specify the child CI class.

   - *Relation type*. Specify the relationship type. The common relationship types used by the ServiceNow integration are "contained" or "contained by", depending on your CMDB. Click the magnifying glass icon to select the correct value.

   - *Parent*. Specify the parent CI class.

4. Click **[Submit]**.

5. In the PowerFlow user interface, go to the **Applications** page and manually run the "Cache ServiceNow CIs and SL1 Device Classes" application.

6. Run the "Sync Devices from SL1 to ServiceNow" application and make sure that no errors exist due to missing CI relationships.

Hosting Rules

Hosting rules can only be one level, and they always involve resources such as physical or virtual hardware.

1. In the ServiceNow filter navigator, type "cmdb_metadata_hosting.list" to view the **CMDB Metadata Hosting Rules** page.

2. Click **[New]**. A new **CMDB Metadata Hosting Rules** record appears.

3. In the **New Metadata Hosting Rules** record, complete the following fields:

   - *Parent type*. Specify the parent CI class.

   - *Child type*. Specify the child CI class.

   - *Relation type*. Specify the relationship type. The common relationship types used by the ServiceNow integration are "Hosts" or "Hosted on", depending on your CMDB. Click the magnifying glass icon to select the correct value.

4. Click **[Submit]**.

5. Add any additional containment and hosting rules that are needed to build the CI relationships in ServiceNow.

6. In the PowerFlow user interface, go to the **Applications** page and manually run the "Cache ServiceNow CIs and SL1 Device Classes" application.

7. Run the "Sync Devices from SL1 to ServiceNow" application and make sure that no errors exist due to missing CI relationships.

Creating a ServiceNow Update Set

ScienceLogic recommends packaging the service rules into a standalone ServiceNow update set that you can export if needed. An *update set* is an XML file containing a group of customizations that can be moved from one

ServiceNow instance to another, similar to the way you can use a PowerPack in SL1 to move content from one SL1 system to another.

This update set should include any changes or configurations to the service rules for the ServiceNow Identification and Reconciliation Module.

To create a standalone update set in ServiceNow:

1. In ServiceNow, go to the **Update Sets** page (System Update Sets > Local Update Sets) and click **[New]**. A new **Update Set** record appears.

2. Complete the following fields:

   - *Name*. Specify a name that describes the rules of this update set.

   - *State*. Set to *In Progress*.

   - *Application*. Set the application scope to *Global*.

   - Complete the remaining fields as needed.

3. Click **[Submit]** or **[Submit and Make Current]**.

4. If you selected **[Submit and Make Current]**, go to step 6.

5. If you clicked **[Submit]**, you can select the update set in the picker in the header or navigate to the update set and select *Make This My Current Set* in the **Related links** section. You are now ready to make changes in your ServiceNow Instances.

6. When you are done with all updates in the update set, change the update set *State* field to *Complete*.

Adding Service Rules to an Update Set

If you submitted your new update set and made it "Current" in *Creating a ServiceNow Update Set*, skip this step and go to *Exporting an Update Set*.

If you did not make your update set current, you will need to identify your current update set and move all of the service rules you need into your update set. You can find this information by clicking the globe icon (⊚) on the ServiceNow navigation bar and reviewing the contents of the drop-down.

All of the service rules that you defined are tracked in the update set record under the **[Customer Updates]** tab.

To add all created service rules to your update set:

1. In ServiceNow, go to the **Update Sets** page (System Update Sets > Local Update Sets) to view a list of update sets on the ServiceNow instance.

2. Identify your current update set, which should have all of the created service rules tracked.

3. Identify the self-created update set that you want to contain all the service rules. This is the update set that you want to export.

4. Select the current update set that has all of the already-created service rules.

5. On the **[Customer Updates]** tab, identify all of the records with a *Type* of either *CMDB Metadata Containment Rules* or *CMDB Metadata Hosting Rules*.

6. Select each of the relevant service rule records and set the *Update set* field to match the update set you want to export. Click the magnifying glass icon to select the correct value.

7. Click **[Update]**.

8. Repeat steps 6-7 until all relevant containment and hosting rules are in the new update set, and then go to *Exporting an Update Set*.

Exporting an Update Set

After you have created your update set and defined the service rules, mark your update set as *Complete* and export it to an XML file.

To export an update set:

1. In ServiceNow, go to the **Update Sets** page (System Update Sets > Local Update Sets) to view a list of update sets on the ServiceNow instance.

2. Select your update set from the list.

3. Set the *State* to *Complete* and click **[Update]**.

4. From the **Update Sets** page, select your completed update set from the list.

5. Under the *Related Links* section, click **Export to XML**.

6. Save the downloaded XML file.

Installing the ScienceLogic Domain Separation (Global) Update Set in ServiceNow

If your ServiceNow environment is ***domain-separated***, where the data, processes, and administrative tasks have been organized into logical groupings called domains, you will need to install the latest version of the "ScienceLogic Domain Separation (Global)" update set in ServiceNow. This update set is *not* included in the "ScienceLogic SL1: CMDB & Incident Automation" application (also called the Certified application).

NOTE: Ask your ScienceLogic contact for access to this update set.

For more information about ServiceNow domain separation, see *Using ServiceNow Domain Separation with PowerFlow.*

> IMPORTANT: If your ServiceNow environment does *not* use domain separation, you can skip this topic.

Overview of the Update Set

The "ScienceLogic Domain Separation (Global)" update set contains the following items:

- Scripted REST API
- Scripted REST Resource
- Scripted REST Query Parameter
- Scripted REST Query Parameter Association
- Script Include

This update set completely separates the ServiceNow Identification Engine REST resource that is used in the "ScienceLogic ServiceNow Integration" application and all of the required resources and duplicates it in the Global scope.

A Scripted REST Service in the Global application is a direct copy of the application endpoint with a new name: `api/10693/sciencelogic_domain_separation`. This REST Service includes only one Resource: `Device IdentificationEngine POST`. This resource works exactly like the application version, but it points to the new Script Include "SciLoDomainSepUtil". This version of the REST resource takes the same formatted JSON as the Certified application.

The Script Include "SciLoDomainSepUtil" includes all of the functionality needed to run the ServiceNow Identification Engine API.

Additional resources for the ServiceNow API:

- [CMDB Identification and Reconciliation](#)
- [identifyCI(String jsonString)](#)
- [createOrUpdateCI(String source, String input)](#)
- [Identification engine error messages](#)

> NOTE: The only resource shared with this update set and the Certified application is the Device Properties page. These properties are located in the Certified application at ScienceLogic > Device > Device Properties.

Limitations of the Identification Engine

For more information about how the Identification Engine handles incoming payloads in domain-separated systems, see the following ServiceNow Knowledge Base article: [KB0695949](#).

The payload and the user domain must match, or the ServiceNow Identification Engine (IDE) will by default insert the CMDB record. Safeguards within the PowerFlow Device Sync application were put in place for payloads that

have relationships. The application will drop the payload if all Configuration Items do not share the same domain.

Installing the Update Set

To install the "ScienceLogic Domain Separation (Global)" update set:

1. Ask your ScienceLogic contact for access to this update set.

2. In ServiceNow, navigate to the **Retrieved Update Sets** page (System Update Sets > Retrieved Update Sets).

3. Click the **Import Update Set from XML** link under **Related Links**.

4. Click **[Browse]** and navigate to the update set XML file you downloaded. Select the XML file and click **[Upload]**.

5. After the file is uploaded, the **Retrieved Update Sets** page appears. Click the link for the "ScienceLogic Domain Separation (Global)" update set. The **Retrieved Update Set** page appears.

6. Click **[Preview Update Set]** . After the preview set runs, a status page appears.

7. Ensure that "Success" appears in the **Completion code** field. If "Success" does not appear in the **Completion code** field, contact ScienceLogic Support to assist with reviewing any conflicts that might exist. Do not proceed until those conflicts are resolved and "Success" appears in the **Completion code** field.

8. Click **[Commit]** to commit the fix script after running the preview set.

9. Before you start to sync devices, you must select the **Domain Separation** option on the **Configuration** pane in the "Sync Devices from SL1 to ServiceNow" application. This option ensures that PowerFlow gets re-pointed to the API endpoint after you install the "ScienceLogic Domain Separation (Global)" update set. For more information, see *Running a Device Sync*.

Configuring Domain Separation without Using the Update Set

You can sync to a domain-separated ServiceNow CMDB without installing "ScienceLogic Domain Separation (Global)" update set, but you will need to manage multiple configuration objects and schedules in PowerFlow. You will need to create multiple schedules, and each schedule will reference a unique configuration object that is specific to a specific domain in ServiceNow.

Creating the Configuration Objects for the ServiceNow Domains

To create a configuration object for each ServiceNow domain:

1. In the PowerFlow user interface, go to the **Configurations** page and click **[Create Configuration]**:

2. Click **[Toggle JSON Editor]** to open the JSON viewer.

3. In the **Configuration Data** section, make sure that this object is configured with a ServiceNow domain-specific user. For example:

```
{
    "encrypted": false,
    "name": "snow_user",
    "value": "domainA_user"
},
{
    "encrypted": false,
    "name": "snow_password",
    "value": "domainA_password"
},
```

4. Update the **include_orgs** value with a list of organizations that map to the domain to which you are syncing.

```
{
  "encrypted": false,
  "name": "include_orgs",
  "value": [1,2,3]
},
```

5. Define the class mapping for the configuration object. For example:

```
{
  "encrypted": false,
  "name": "mapping",
  "value": {
            "cmdb_ci_computer": [
              "IBM | IBM OS/400 V5R1M0",
              "IBM | Main Frame",
              "IBM | AIX RS/6000"
            ],
            "cmdb_ci_esx_resource_pool": [
              "VMware | Resource Pool"
            ]
          }
},
```

6. The **region** value should be unique to SL1 stack that is being synced. For example:

```
{
  "encrypted": false,
  "name": "region",
  "value": "StackA"
}
```

7. Repeat steps 1-6 for each ServiceNow domain you want to use.

Example JSON Code for a Configuration Object

The following JSON code is for an example configuration object:

```
[
  {
    "encrypted": false,
    "name": "sl1_host",
    "value": "SL1_StackA"
```

```
  },
  {
    "encrypted": false,
    "name": "sl1_db_host",
    "value": "${config.sl1_host}"
  },
  {
    "encrypted": false,
    "name": "sl1_password",
    "value": "password"
  },
  {
    "encrypted": false,
    "name": "sl1_user",
    "value": "StackA_user"
  },
  {
    "encrypted": false,
    "name": "sl1_db_user",
    "value": "root"
  },
  {
    "encrypted": false,
    "name": "sl1_db_password",
    "value": "StackA_password"
  },
  {
    "encrypted": false,
    "name": "snow_host",
    "value": "example.service-now.com"
  },
  {
    "encrypted": false,
    "name": "snow_user",
    "value": "domainA_user"
  },
  {
    "encrypted": false,
    "name": "snow_password",
    "value": "domainA_password"
```

```
  },
  {
    "encrypted": false,
    "name": "include_orgs",
    "value": [1,2,3]
  },
  {
    "encrypted": false,
    "name": "mapping",
    "value": {
            "cmdb_ci_computer": [
              "IBM | IBM OS/400 V5R1M0",
              "IBM | Main Frame",
              "IBM | AIX RS/6000"
            ],
            "cmdb_ci_esx_resource_pool": [
              "VMware | Resource Pool"
            ]
          }
  },
  {
    "encrypted": false,
    "name": "region",
    "value": "StackA"
  }

]
```

Aligning a Schedule with a ServiceNow Domain

After you have created the configuration objects, you can define multiple schedules, one schedule for each domain. When scheduling the application, you will need to create multiple schedules, where each schedule uses a domain-specific configuration object.

NOTE: When specifying a configuration object to use in the custom parameters, use the ID of the configuration object.

The following image shows how you can create a schedule that uses a specific configuration object using the *Custom Parameters* field in the PowerFlow Scheduler:

The following image shows how you could schedule Device Syncs for multiple ServiceNow domains:

Using ServiceNow Domain Separation with PowerFlow

The following topics provide more information about ServiceNow domain separation and how it relates to PowerFlow. For more information, see [Domain separation](#) in the ServiceNow Documentation.

> **NOTE:** When either multiple SL1 stacks or multiple ServiceNow systems are involved with PowerFlow, you should create an individual configuration object for each SL1 stack or ServiceNow system. Next, create an individual schedule for each configuration object. Each schedule should use a configuration object that is specific to that single SL1 stack or ServiceNow system. Creating copies of a PowerFlow application from a SyncPack for the purpose of distinguishing between domains is not supported, and will result in issues on upgrades.

> **NOTE:** If you get an error message stating that a device or CI has no *domain_sys_id* value, that means that the company for that CI has no domain.

User Setup

Company and domain setup is critical for the domain separation integration to work using the Identification Engine provided by ServiceNow. This solution requires only one user and will require proper setup depending on where the user is located within the domain tree.

Example 1

In the following example, **ScienceLogic (1)** is both the domain and the company. The ScienceLogic user service account is associated with **ScienceLogic (2)** company, and it will have access to all child domains. You do not need to set visibility to any domain. This is the best way to set up this user, because placing it in the top domain ensures that it always has access to all children:
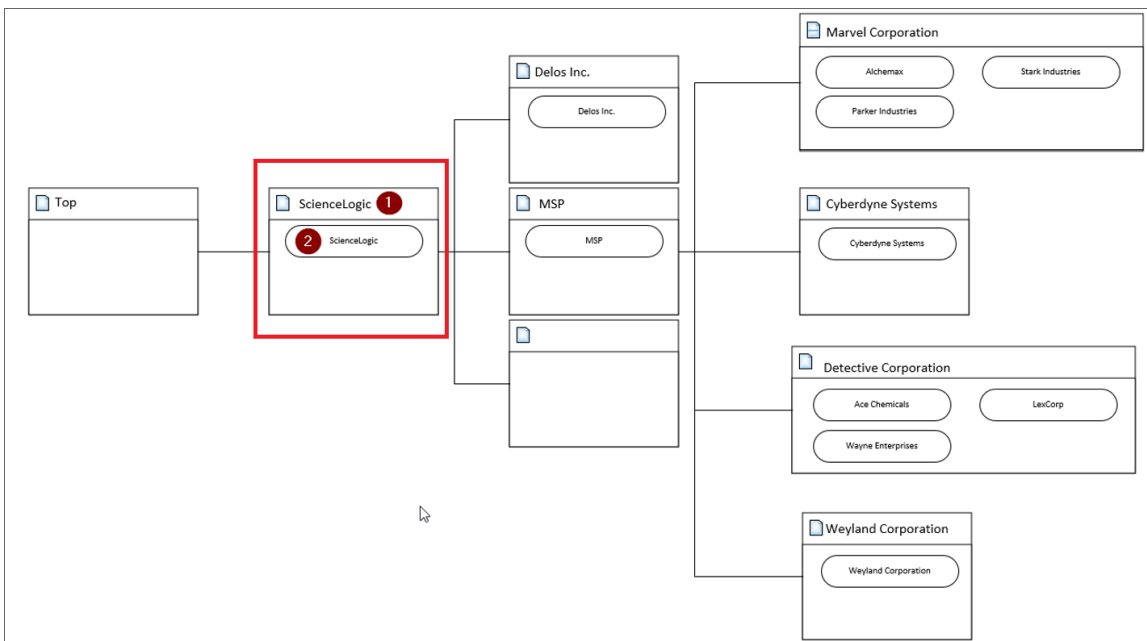
Example 2

In the following example, **Delos Inc. (1)** is the company within the **Delos Inc.** domain. The PowerFlow service account is associated with the **Delos Inc. (1)** company. The **Delos Inc.** domain has no children domains, and if domain visibility is not assigned, PowerFlow will not properly update the CMDB. This setup works, but it requires that proper domain visibility is set up for the service account to work correctly.
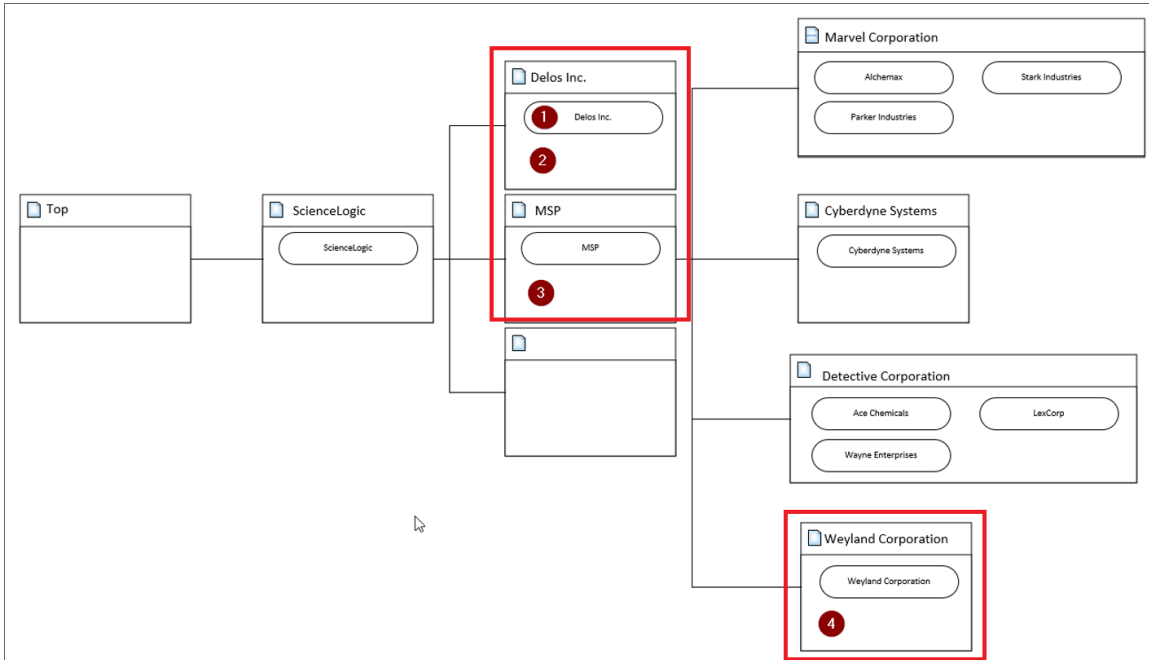
> **NOTE:** Assigning visibility to **MSP (3)** will grant the service account access to all child domains. Assigning visibility to **Weyland Corporation (4)** will only allow access to the **Delos Inc.** domain and the **Weyland** domain; all other domains will not work.

Workflow

The API endpoint is based on the API query parameter "test" being true or not, which determines which Identification Engine API resource should be used. There are two primary avenues supplied with this REST resource: "createOrUpdateCI" or "identifyCI", and the only difference is that "identifyCI" does not commit the results:
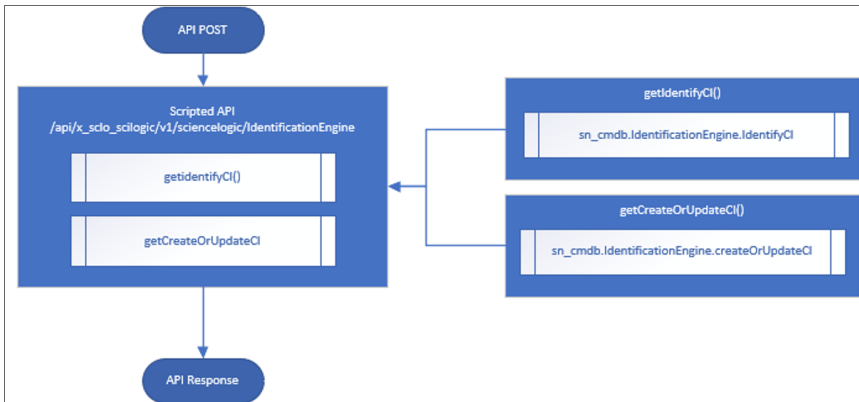
The "getCreateOrUpdateCI" function uses the following workflow:

1. Retrieves the current session Domain ID (`sys_id`).

2. Sets the user Domain ID by creating an array of domain `sys_id` values and returning only the unique domains, or setting the domain if the array has only one unique domain.

3. Submits the JSON formatted string to "createOrUpdateCI()" or "identifyCI()" API.

4. Sets the user's Domain ID back to the original session ID.

The following image shows this workflow:

Installing the ScienceLogic Domain Separation (Global) Update Set in ServiceNow
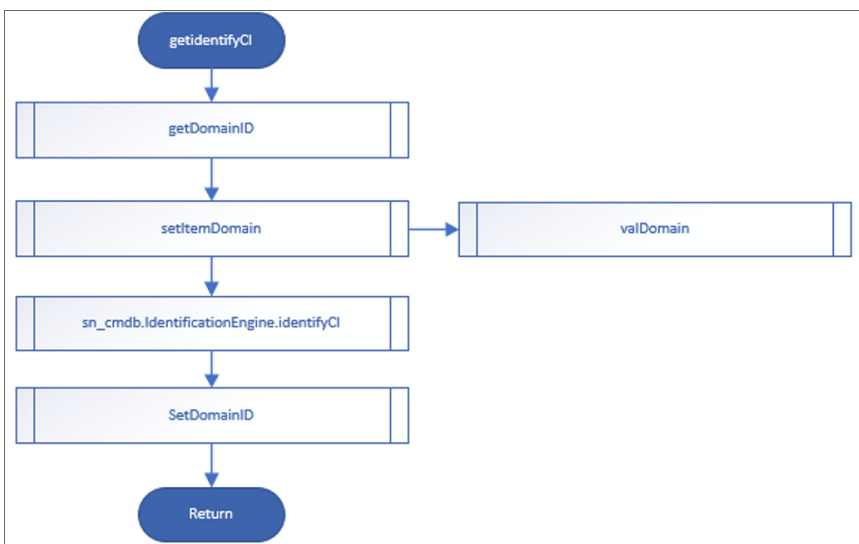
Chapter

3

Configuring Organization Sync and Device Sync for the CMDB SyncPack

Overview

This chapter describes the how to configure and run Organization Sync and Device Sync in the "ServiceNow CMDB" SyncPack.

For more information about the other syncs in this SyncPack, see *Configuring Additional Syncs for the CMDB SyncPack*.

This chapter covers the following topics:

Creating and Aligning a Configuration Object

A *configuration object* supplies the login credentials and other required information needed to execute the steps for an application in PowerFlow. The **Configurations** page (⚙) of the PowerFlow user interface lists all available configuration objects for that system.

You can create as many configuration objects as you need. A PowerFlow application can only use one configuration object at a time, but you can use (or "align") the same configuration object with multiple applications.

To use this SyncPack, you will need to create one or more configuration objects in the PowerFlow user interface and align that configuration object to the applications that let you sync data between SL1 and ServiceNow.

> **NOTE:** Depending on your SL1 and ServiceNow environments, you might be able to use the same configuration object with other ServiceNow SyncPacks.

> **NOTE:** When either multiple SL1 stacks or multiple ServiceNow systems are involved with PowerFlow, you should create an individual configuration object for each SL1 stack or ServiceNow system. Next, create an individual schedule for each configuration object. Each schedule should use a configuration object that is specific to that single SL1 stack or ServiceNow system. Creating copies of a PowerFlow application from a SyncPack for the purpose of distinguishing between domains is not supported, and will result in issues on upgrades.

Creating a Configuration Object

For this SyncPack, you can make a copy of the "ServiceNow SyncPack" configuration object, which is the sample configuration file that was installed with the "ServiceNow Base" SyncPack.

> **TIP:** The "ServiceNow SyncPack" configuration object contains all of the required variables. Make a copy of the configuration object and update the variables from that object to match your SL1 and ServiceNow settings.

To create a configuration object based on the "ServiceNow SyncPack" configuration object:

1. In the PowerFlow user interface, go to the **Configurations** page (⚙).

2. For the "ServiceNow SyncPack" configuration object, click the **[Actions]** button (⋮) and select *Edit*. The **Configuration** pane appears.

> **TIP:** Click **[Toggle JSON Editor]** to show the JSON code. Click the button again to see the fields.

3. Click **[Copy as]**. The **Create Configuration** pane appears.

> IMPORTANT: This step is required. Do *not* use the original configuration object to run PowerFlow
> applications.

4. Complete the following fields:
    - *Friendly Name*. Name of the configuration object that will display on the **Configurations** page.
    - *Description*. A brief description of the configuration object.
    - *Author*. User or organization that created the configuration object.
    - *Version*. Version of the configuration object.

5. In the **Configuration Data** field, include the required block of code to ensure that the applications aligned to this configuration object do not fail:

```
{
  "encrypted": false,
  "name": "sl1_db_host",
  "value": "${config.sl1_host}"
},
```

For example:

```
{
  "encrypted": false,
  "name": "sl1_db_host",
  "value": "10.2.11.42"
},
```

6. In the **Configuration Data Values** field, update the default variable definitions to match your PowerFlow configuration.

> NOTE: The *region* value is a user-defined variable that identifies your SL1 instance within ServiceNow.

7. To create a configuration variable in the JSON Editor, define the following keys:
    - *encrypted*. Specifies whether the value will appear in plain text or encrypted in this JSON file. If you set this to "true", when the value is uploaded, PowerFlow encrypts the value of the variable. The plain text value cannot be retrieved again by an end user. The encryption key is unique to each PowerFlow system. The value is followed by a comma.
    - *name*. Specifies the name of the configuration file, without the JSON suffix. This value appears in the user interface. The value is surrounded by double-quotes and followed by a comma.

- **value**. Specifies the value to assign to the variable. The value is surrounded by double-quotes and followed by a comma.

8. If you want to use OAuth2 for authentication with ServiceNow, complete the following Configuration Data Values fields:

   - **snow_oauth_client_id**. Enter the OAuth2 Client ID from ServiceNow.

   - **snow_oauth_client_secret**. Enter the OAuth2 Client secret from ServiceNow.

   - **snow_oauth_token_url**. Enter the full authentication URL, including host and protocol from ServiceNow. For example, "https://<test-instance-name>.service-now.com/oauth_token.do"

   - **snow_auth_method**: Enter *oauth* or *http_basic*. If no value is provided, *http_basic* will be used for connection.

> **NOTE:** The configuration options listed above are included by default with the sample configuration object provided in the "ServiceNow Base" SyncPack. The configuration options are only required in the configuration object if you plan to use OAuth2 to authenticate. If the values are not present in the configuration object, normal "http_basic" authentication will be used.

9. If you want to adjust the version of the ServiceNow IRE endpoint to send CI payloads to, complete the following Configuration Data Values field:

   - **snow_api_version**. If no value is entered, [[[Undefined variable ApplianceNames.IS4]]] will default to use *v2*. For proper interface handling, ScienceLogic recommends entering *v3*. The only possible values are *v2* or *v3*.

> **NOTE:** In order to use "v3" for the snow_api_version, you must install the latest version of the "ScienceLogic ServiceNow Integration (v1.0.81)+ Interface Hotfix" update set in ServiceNow. Ask your ScienceLogic contact for access to this update set.

10. Click **[Save]**. You can now align this configuration object with one or more applications.

Aligning a Configuration Object

Before you can run the applications in this SyncPack, you must first "align" a configuration object with the application you want to use.

To align a configuration object with an application:

1. From the **Applications** page of the PowerFlow user interface, open the relevant application and click **[Configure]**. The **Configurations** pane for that application appears.

2. From the **Configurations** drop-down, select the configuration object you want to use.

3. Click **[Save]** to align that configuration with the application.

4. Repeat this process for every other application you want to use.

Creating and Aligning a Configuration Object

> **NOTE:** The values for the parameters that display in the **Configuration** pane with a padlock icon (🔒) are populated by the values in the configuration object.

Creating an OAuth2 Credential Record in ServiceNow

In order to use OAuth2 for authentication with ServiceNow, you must create an OAuth2 credential record in ServiceNow. To configure the ServiceNow credential that will be used by the Connector Instance:

1. Navigate to **System OAuth > Application Registry**. The **Application Registries** page appears.
2. Click **[New]**.
3. Select **Create an OAuth API endpoint for external clients**. A new record appears.
4. Complete the following fields on the new record:

    - *Name*. Type a unique name for the credential. Required.
    - *Client ID*. The Client ID is automatically generated by the ServiceNow OAuth server.
    - *Client Secret*. Leave this empty. ServiceNow will auto-generate this when the record is saved.
    - *Refresh Token Lifespan*. Enter the length of time in seconds the Refresh Token will be valid.
    - *Access Token Lifespan*. Type the length of time in seconds that the Access Token will be valid. ScienceLogic recommends setting this to 3,600 seconds to avoid known issues for longer ServiceNow REST interactions.

> **NOTE:** In a scenario where the **[Access Token Lifespan]** value is shorter than the duration of a PowerFlow step that makes multiple REST interactions with ServiceNow, the access token will expire and need to be refreshed. As a result, retries were added to several PowerFlow steps where this issue may occur. This issue will hopefully be addressed in future versions of the Base Steps SyncPack.

5. Under the **Auth Scope** section at the bottom of the page, double click *Insert a new row*.
6. In the search box that appears, click the magnifying glass icon, select the *useraccount* record, and click the checkmark icon to save.
7. Click **[Submit]** to save the new record.

Syncing Organizations

An **Organization Sync** uses the "Sync Organizations from SL1 to ServiceNow" application to sync *organizations* from SL1 with ServiceNow *companies*. In this context, **sync** means that if you update a company in ServiceNow, the Organization Sync process will update the SL1 organization with that information, and vice versa.

If your ServiceNow environment is *domain-separated*, where the data, processes, and administrative tasks have been organized into logical groupings called *domains*, then the first sync you should run on a new PowerFlow system is an **Organization Sync**. For more information, see *For Domain-separated ServiceNow Environments Only*.

If you are using SL1 as your "source of truth", you are not required to configure and run an Organization Sync. However, if you want to filter by organization or company (using the use the **org_filters** option) with any of the other syncs in this SyncPack, such as Device Sync, File System Sync, and Interface Sync, you will need to run the Organization Sync as documented in the steps below.

For Domain-separated ServiceNow Environments Only

If your ServiceNow environment is *domain-separated*, where the data, processes, and administrative tasks have been organized into logical groupings called *domains*, then the first sync you need to run on a new PowerFlow system is an **Organization Sync**.

Starting with version 3.5.0 of this SyncPack, the following fields are no longer being used on the Company table (**core_company**) in ServiceNow:

- *SL1 Monitored*
- *SL1 Region*
- *SL1 ID*

The ScienceLogic-specific information was moved to an application-specific table (**x_sclo_scilogic_organizations**) to track which Organization goes with any specific Company.

> **NOTE:** Creating new records is currently only available with the **x_sclo_scilogic.admin** role.

> **NOTE:** If you get an error message stating that a device or CI has no **domain_sys_id** value, that means that the company for that CI has no domain.

Considerations for Version 3.5.x of the SyncPack

> **IMPORTANT:** If ServiceNow is set as the *Source of Truth* in the PowerFlow Organization Sync application, you must manually create the linking records in ServiceNow. If ScienceLogic is set as the *Source of Truth,* then you can skip these sections and go to *Configuring Organization Sync*.

Version 3.5 introduced linking records (using the **x_sclo_scilogic_organizations** table), which enabled a many-to-one relationship between **multiple** SL1 organizations to a single ServiceNow company.

A **linking record** in ServiceNow lets you set up the following configurations:

- Tracking the relationship of a single ServiceNow company with a single SL1 organization in multiple SL1 databases.

- Matching multiple SL1 organizations (in multiple SL1 systems) to the same ServiceNow company.

- Importing a ServiceNow company into SL1 as an SL1 organization.

> **NOTE:** Aligning multiple SL1 organizations from the *same* SL1 system to a single ServiceNow company is not supported at this time.

For more information about creating linking records, see *Creating a Linking Record when ServiceNow is the Source of Truth*.

Upgrading to Version 3.5.x

If you are *upgrading* to version 3.5.x from an older version and you want to sync multiple SL1 organizations to a single ServiceNow company, you need to determine if the organizations already exist in the multiple SL1 systems that you want to sync to the single company in ServiceNow:

- If the SL1 organizations already exist, you will need to create a linking record tied to the company with the **region** value for each SL1 stack, plus the native key, the SL1 id, and **monitored = True**.

- If the SL1 organizations do *not* currently exist and you want the Organization Sync to create the organizations in SL1 (using the **Create_Missing** option in the **Configuration** pane), the linking records tied to the companies only need the **region** and **monitored = True**.

> **IMPORTANT:** ScienceLogic highly recommends that you complete the upgrade steps on a development instance and test the syncs first before you implement the updates in a production environment.

To upgrade to version 3.5.x of the "ServiceNow CMDB" SyncPack:

1. In PowerFlow, *install* the latest version of the following SyncPacks:

    - ServiceNow CMDB SyncPack

    - ServiceNow Base SyncPack

    - Base Steps SyncPack

2. After you install the SyncPacks, click the Actions button (⋮), select *Change active version*, and select the most recent version of each SyncPack.

3. Download the most recent version of the "ScienceLogic SL1: CMDB & Incident Automation ServiceNow" scoped application from the ServiceNow Store at https://store.servicenow.com. For more information, see *Installing the "ScienceLogic SL1: CMDB & Incident Automation" Application in ServiceNow*.

> **TIP:** You can verify the version number on the *Dependencies for SL1 PowerFlow SyncPacks* page, in the **Supported "Scoped Application" Version** column for the SyncPack.

4. If your ServiceNow instance is domain-separated, download and install the domain separation update set, "ScienceLogic Domain Separation (Global) 1.0.67.xml" from ServiceNow. For more information, see *Installing the ScienceLogic Domain Separation (Global) Update Set in ServiceNow.*

5. Create linking records on the Organization Source table (**x_sclo_scilogic_organizations**) corresponding to existing records on the **core_company** table. For more information, see *Creating a Linking Record*.

> **TIP:** To assist in the migration, you can use the "ScienceLogic ServiceNow Integration (v1.0.76)+ (Organization migration)" fix script (Org Sync migration.xml). You shold only run this script once, upon initial upgrade. Ask your ScienceLogic contact for access to this script.

6. In the "Sync Organizations from SL1 to ServiceNow" application, make sure that the **Source_of_Truth** option on the **Configuration** pane is set to *ServiceNow*.

Creating a Linking Record when ServiceNow is the Source of Truth

Before running the "Sync Organizations from SL1 to ServiceNow" application for the first time in a domain-separated environment, you will need to create a **linking record** in ServiceNow.

You should also create a linking record if you select ServiceNow as the "source of truth" on the **Configuration** page of the "Sync Organizations from SL1 to ServiceNow" application, even if your ServiceNow instance is not domain-separated.

A linking record in ServiceNow lets you set up the following configurations:

- Tracking the relationship of a single SL1 organization with a single ServiceNow company in multiple SL1 databases.
- Matching multiple SL1 organizations (in multiple SL1 systems) to the same ServiceNow company.
- Importing a ServiceNow company into SL1 as an SL1 organization.

ScienceLogic highly recommends that you try out this feature in a test environment before pushing to production.

> **IMPORTANT:** If you set up an Organization Sync with a version of this SyncPack earlier than version 3.5.0, you will need to create a linking record when you upgrade to version 3.5.0 or later. The linking record is only required when you upgrade if you use ServiceNow as the source of truth.

To create a linking record:

1. In ServiceNow, navigate to **ScienceLogic > Supporting Imports > Organization Source**.
2. Click **[New]**. A new record appears.
3. Complete the following required fields:
   - *Region*. Set this value to match the *region* value in the configuration object aligned with the "Sync Organizations from SL1 to ServiceNow" application in the PowerFlow user interface.
   - *Company*. You would only need to use this field if you either want SL1 to create the organization or you want to link a specific SL1 organization with a specific ServiceNow company. To do this, search for the name of the ServiceNow company you want to link to (or have SL1 create). Otherwise, you can leave this field blank.

     Because this a document field, you must click the magnifying class icon to add the **Company [core_company]** table in the *Table name* field, and then click the magnifying class icon to locate a company from the *Document* field:
   - *Monitored*. Set to *True* (checked).

4. On the new record, the following fields are not required, or they are required under specific conditions:

- *native key*. This key is a unique identifier for an SL1 organization, from a specific SL1 database. Use the following format:

  <region>+ORG+<roa_id>

  where:

  - *<region>* is the *region* value in the configuration object aligned with the "Sync Organizations from SL1 to ServiceNow" application. This value should match the value in the *Region* field, above.

  - *<roa_id>* is the value for that organization in the **ID** column of the **Organizations** page (Registry > Accounts > Organizations) in SL1. This value should match the value in the *ID* field, above.

  For example: *ScienceLogic+ORG+2*.

- *ID*. Use this field only if you want to link a specific SL1 organization with a specific ServiceNow company. To do this, locate the value for that organization in the ID column of the **Organizations** page (Registry > Accounts > Organizations) in SL1. Otherwise, you can leave this field blank.

5. Repeat steps 2-4 to create a linking record for every instance where you want to track the relationship of a single organization with a single company in multiple SL1 databases.

6. Go to the *Configuring Organization Sync* topic to run the "Sync Organizations from SL1 to ServiceNow" application. Make sure that the *Source_of _Truth* field on the **Configuration** pane is set to *ServiceNow*.

For Case Integration ServiceNow Environments Only

Add the following roles to the Integration user so that user can interact with the "ScienceLogic SL1: Customer Service Management Integration" Application in ServiceNow.

- **x_sclo_case_mgmt.admin**. Provides user rights to interact with the scoped application tables and modules in ServiceNow.

- **import_transformer**. Provides user rights to manage import set transform maps, run transforms, and access responses.

If your ServiceNow environment is using the Case Integration module and you intend to use **customer_account** records, you will need to add additional rights to the Integration user. These rights allow the Integration user to read the table fields:

- **sn_customerservice.customer_data_viewer**

You will need to add cross-scoped access for read-only to the **customer_account** table as well. ScienceLogic recommends that you use ServiceNow as the source of truth for Organizations (Companies). For more information, see the "Allowing Cross-Scoped Access in ServiceNow" topic in the *ServiceNow CMDB SyncPack* manual.

Configuring Organization Sync

Organization Sync uses the "Sync Organizations from SL1 to ServiceNow" application to pull *organizations* from SL1 and sync them with ServiceNow *companies*.

> **NOTE**: If you are using SL1 as your "source of truth", you are not required to configure and run an Organization Sync. However, if you want to filter by organization or company (using the use the ***org_filters*** option) with any of the other syncs in this SyncPack, such as Device Sync, File System Sync, and Interface Sync, you will need to run the Organization Sync as documented in the steps below.

To sync SL1 organizations with ServiceNow companies:

1. In the PowerFlow user interface, go to the **Applications** page and select the "Sync Organizations from SL1 to ServiceNow" application. The **Application** page for that application appears.

2. Click **[Configure]**. The **Configuration** pane appears:

3. Complete the following fields, as needed:

- **Configuration**. Select the configuration object with the relevant SL1 and ServiceNow credentials to align with this application. You cannot edit fields that are populated by the configuration object. Required.

> **NOTE:** The *region* field is populated by the configuration object you aligned with this application. The region value must match the value in the **SL1 Region** field in ServiceNow. If you need to update this value, you will need to define the *region* variable in the configuration object that is aligned with this application, or align a different configuration object that has the correct *region* value.

- **read_timeout**. Specify the maximum amount of time in seconds that the application should wait for a response before timing out.

- **snow_chunk_size**. Specify the number of organizations to include in each chunk sent to ServiceNow when you run this application.

- **verify_snow_ssl**. Toggle on (blue) this option to enable verification of the SSL certification when you run this application.

- **Update_Name**. This option addresses the situation where PowerFlow finds a match with an organization and a company, but the names do not match. This option updates a company or organization name based on your selection in the Source_of_Truth field, below. For example, if you selected ScienceLogic as the source of truth, PowerFlow uses the company name from ScienceLogic as the updated name. By default, this option is not selected.

- **Create_Missing**. Select this option if you want PowerFlow to create a new organization or company if that record is missing, based on your selection in the *Source_of_Truth* field. By default, this option is not selected.

> **NOTE:** Starting with version 3.5.0 of this SyncPack, the **Domain Separation** parameter was removed from the **Configuration** pane. Domain separation is still supported, but this parameter no longer needs to be selected.

- **Source_of_Truth**. Select whether you want to use data from ServiceNow or ScienceLogic as the "source of truth" when this application encounters duplicate data or data collisions. In most situations, the source of truth would be the application where you initially configured and created the companies or organizations:

  - If you select *ServiceNow*, you must specify the values in the **SL1 Monitored** and **SL1 Region** fields in ServiceNow. Because these fields do not display by default on the **Companies** page in ServiceNow, navigate to the **Companies** page in ServiceNow, click the *Update Personalized List* icon ( ), and add the **SL1 Monitored** and **SL1 Region** columns to that page. Also, if your ServiceNow configuration uses domain separation, you must select *ServiceNow* as the source of truth.

  - If you select *ScienceLogic*, you do not need to do anything else related to this field.

4. In the **Attribute Mappings** section, click the Edit button to view and edit the mappings for any other organization attributes, such as additional address or contact information. These mappings will sync between SL1 (the first column) and ServiceNow (the second column). A set of organization attributes are already mapped by default.

> **TIP:** You can use Jinja2 Templates in fields that are aligned with the "Source of Truth" you selected (SL1 or ServiceNow). For more information, see *Using a Jinja2 Template*.

> **NOTE:** When an attribute value is "0" in SL1, the corresponding field in ServiceNow might display as empty.

5. Click **[Save]**. The **Configuration** pane automatically closes after this message appears.
6. Click **[Run]** to run the application.
7. When the application completes, open the **Step Log** and review the log messages for the "Process Organizations" step to see if any company or organization records were created. As needed, select the other steps to review the logs on the **Step Log** for those steps.

> **TIP:** Any SL1 organization that is synced to a ServiceNow Company will have the *crm_id* field on the **[Properties]** tab for that organization populated with the ServiceNow Company *sys_id* variable.

ScienceLogic recommends that you schedule an Organization Sync to run at least once a week. For more information, see *Scheduling PowerFlow Applications*.

Syncing Devices from SL1 to ServiceNow

The "Sync Devices from SL1 to ServiceNow" application syncs devices and virtual device relationships from SL1 to ServiceNow. You can also sync devices based on organization and collector group.

The Device Sync process use rules or "mappings" that you can define in the "Sync Devices from SL1 to ServiceNow" application. These mappings connect an **SL1 device class** to a **ServiceNow CI class**, which determines the CI class that ServiceNow uses when creating the CI in ServiceNow.

> **NOTE:** For more information about building *service rules* (containment rules and hosting rules) for devices and CIs, see *Configuring Service Rules for Device Sync*.

The "Sync Devices from SL1 to ServiceNow" application can also collect manufacturer and model attributes from asset records aligned with devices in SL1 and sync that information with ServiceNow. PowerFlow only populates the manufacturer and model attributes if the values exist in ServiceNow CIs; PowerFlow does not create new manufacturer values in ServiceNow. The "Sync Devices from SL1 to ServiceNow" application uses the *sys_id* field

as a reference when syncing manufacturer and model information between SL1 and ServiceNow. For more information, see *Device Attribute Mappings*.

Merged Devices in SL1

When the "Sync Devices from SL1 to ServiceNow" application encounters a merged device in SL1, it splits the record into two objects to allow for correct default relationships in ServiceNow.

Starting with version 3.2.0 of this SyncPack, the "Sync Devices from SL1 to ServiceNow" application syncs *both* the physical and the component device in SL1 to ServiceNow. When a merged device is encountered in SL1, the "Sync Devices" application splits the device in PowerFlow and creates two CIs in ServiceNow. This action does not impact the source device record in SL1.

In ServiceNow, the physical CI includes the relevant asset information. A relationship also exists between the physical CI and the virtual CI. The asset information is directly copied between both CIs, so the data will essentially be duplicated across both devices, and the data will be submitted to two separate tables. The **sl1_url** will also be the same on both devices, so that both CIs will point to the same device in SL1.

Using Other Data Sources with Merged Devices

If you have other data sources syncing into the ServiceNow CMDB and you have merged devices in SL1, ScienceLogic recommends caution when integrating to the CMDB.

Also, ScienceLogic recommends that you ensure that configuration of the Identification and Reconciliation (IRE) within ServiceNow affects all data sources that are integrating into it. In the case of ScienceLogic, this is most apparent when syncing merged devices. Modifications to the IRE to handle merged devices affects all other data sources that sync to those specific class tables. It is your responsibility to understand each data source, how that data source integrates with the ServiceNow CMDB, and how to leverage that knowledge to understand the impact IRE changes may have.

> **WARNING:** ScienceLogic cannot be held responsible for any duplicate, lost, or incorrect CI information as a result of merged devices when multiple data sources are involved. This scenario will also affect your Support SLAs, as this practice deviates from recommended best practices.

For more information about the ServiceNow Identification and Reconciliation module, see the ServiceNow documentation: https://docs.servicenow.com/bundle/orlando-servicenow-platform/page/product/configuration-management/concept/c_CompsandProcessIDandReconcil.html.

Common Fields Used by Device Sync

The "Sync Devices from SL1 to ServiceNow" application uses the following ServiceNow fields to determine which devices to sync from SL1 to ServiceNow:

- **SL1 Monitored**. This field displays a Boolean (true or false) value that is impacted by whether the device is in SL1 or not. The device being found in ServiceNow depends on the **SL1 Monitored** field. The device being found in SL1 depends on the class mappings defined in the "Sync Devices from SL1 to ServiceNow" application.

    ◦ If the CI is in ServiceNow *and* the device is in SL1 , the **SL1 Monitored** flag is set to *true*.

    ◦ If the CI is in ServiceNow but the device is *not* in SL1, the **SL1 Monitored** field is set to *false*. Anything pulled from ServiceNow (everything that is *monitored: true* and matches the *region*) that does *not* have a matching device pulled from SL1 gets marked as *monitored: false*.

- **SL1 Region**. This field represents an ID for the SL1 instance or instances being synced to the ServiceNow instance. The **SL1 Region** field is determined by the user when configuring the IS applications. In a multi-SL1 environment, ScienceLogic recommends that you make the **SL1 Region** field descriptive so the ServiceNow user knows from which SL1 stack the CI originated.

    ◦ If the **SL1 Region** field is defined as an identifier by the CI class, ServiceNow will create new CI records with the new **SL1 Region** value, and the user must manually delete the duplicate CIs in the old **SL1 Region** field.

    ◦ If the **SL1 Region** field is *not* defined as an identifier by the CI class, ServiceNow will not treat these devices as new CIs, and the **SL1 Region** field will be automatically updated.

> NOTE: Changing the **SL1 Region** value after an initial run of the "Sync Devices from SL1 to ServiceNow" application will have differing results depending on the service rules defined in ServiceNow that dictate reconciliation of the CI. If you change the **SL1 Region** value, you will ll need to run "Sync Devices from SL1 to ServiceNow" twice: once to align the CIs with the new region, and a second time to enable PowerFlow to re-cache the newly updated CIs in the region.

- **SL1 ID** or **sl1_id**. This field represents the value of the object from SL1 that is being synced. Do not use **sl1_id** as a CI Identifier.

Running a Device Sync

To perform a Device Sync between SL1 and ServiceNow, run the following applications in the PowerFlow user interface, in the specified order:

- **Cache ServiceNow Companies, CIs and SL1 Orgs, Device Classes**. Reads all existing SL1 device classes and ServiceNow CI classes and caches them for the Device Sync. This application uses this data to populate the **mappings** drop-down values in the "Sync Devices from SL1 to ServiceNow" application. Before version 3.5.0 of this SyncPack, this application was named "Cache ServiceNow CIs and SL1 Device Classes".

- **Generate Required CI Relations for ServiceNow**. Determines if you are missing any class mappings or service rules that might be required in ServiceNow.

- **Sync Devices from SL1 to ServiceNow**. Syncs devices and virtual device relationships from SL1 to ServiceNow.

> **IMPORTANT:** The speed of ServiceNow processing is reduced by ServiceNow errors generated when consuming payloads. If you experience slow processing or "maximum execution time exceeded" messages, you must review and address any ServiceNow errors reported to resolve. If there are significant ServiceNow errors in Device Sync, those errors will also impact Interface sync processing. When running dependent syncs at large scale, such as Interface Sync and Device Sync, ScienceLogic recommends that you run them serially, not at the same time. Running both syncs at the same time will greatly hinder ServiceNow processing and scalability.

To sync SL1 devices with ServiceNow:

1. In the PowerFlow user interface, select the "Cache ServiceNow CIs and SL1 Orgs, Device Classes" application from the **Applications** page, click **[Configure]**, align a configuration object, and then click **[Run]**.

   If you change any of the containment rules or hosting rules in ServiceNow, you will need to run "Cache ServiceNow CIs and SL1 Orgs, Device Classes" again. For more information, see *Configuring Service Rules for Device Sync*.

2. Select the "Generate Required CI Relations for ServiceNow" application from the **Applications** page, click **[Configure]**, align a configuration object, and then click **[Run]**.

   PowerFlow uses the Device Class mappings you are going to configure in step 6, so you do not need to set up any mappings on the **Configuration** pane for the "Generate Required CI Relations for ServiceNow" application. Any mappings you add to this application will *overwrite* mappings in the "Sync Devices from SL1 to ServiceNow" application.

3. When the "Generate Required CI Relations for ServiceNow" application completes, review the log information in the **Step Log** for the "Pull and Process Relations" step. You should see a log message stating that no missing relations were found. For more information, see *Log Messages for the "Generate Required CI Relations for ServiceNow" Application*.

> **NOTE:** If needed, address any missing class mappings or service rules. For more information on service rules, see *Creating a ServiceNow Update Set*.

4. Select the "Sync Devices from SL1 to ServiceNow" application from the **Applications** page and click **[Configure]**. The **Configuration** pane appears:

5. Complete the following fields, as needed:

- *Configuration*. Select the configuration object with the relevant SL1 and ServiceNow credentials to align with this application. You cannot edit fields that are populated by the configuration object. Required.

> NOTE: The *region* field (along with other fields related to user names and passwords) is populated by the configuration object you aligned with this application, using the *Configuration* field. The *region* value on this pane must match the value in the *SL1 Region* field in ServiceNow. If you need to update this value, you will need to define the *region* variable in the configuration object aligned with this application, or align a different configuration object that has the correct *region* value.

- **read_timeout**. Specify the maximum amount of time in seconds that the application should wait for a HTTP Read response before timing out. The default is 300 seconds.

- **Include_Orgs**. If you want to include a specific set of SL1 Organizations in the device sync, add the Organization IDs from the SL1 **Organizations** page (Registry > Accounts > Organizations) in this field, separated by commas. If this field is enabled, PowerFlow will pull only the Organizations listed in this field; PowerFlow does not pull all Organizations and then drop those not on the list. Leave this field empty to sync *all* SL1 Organizations. Optional.

- **Include_CUGs**. If you want to include SL1 Collector Groups (CUGs) in the device sync, add the Collector Group IDs from SL1 in this field, separated by commas. Leave this field empty to sync *all* SL1 Collector Groups. Optional.

> **CAUTION:** A misconfiguration in the **Include Orgs** field or the **Include_CUGs** fields might change the monitoring flag for one or more devices. In other words, a misconfiguration in this field could switch the SL1 Monitored flag from "true" to "false", removing that device or devices from the device sync.

- **sl1_chunk_size**. Specify the number of devices to pull from SL1 in each chunk. The default is 500 devices per chunk.

- **snow_request_limit**. Specify the number of CIs fetched from ServiceNow in each request. The default is 500 objects per chunk.

- **selected_devices**. If you want to sync a sub-set of all discovered devices, type a comma-separated list of the Device IDs from SL1 for only the devices that you want to sync. Leave this field empty to sync all SL1 devices.

- **sl1_url_override**. Update this field if you want to use an URL that is different from the standard SL1 URL that gets sent to the ServiceNow CI record. Optional.

- **excluded_devices**. Type a list of comma-separated device names or device IDs for any devices that you want to exclude from the device sync. A device on the excluded list will still be queried, but it will be dropped during the PowerFlow sync process. Optional.

- **cache_lookup_chunk_size**. Specify the number of CI correlation documents to pull from the PowerFlow cache in a chunk. The default is 1000 documents per chunk.

- **discovery_source**. Specify the ServiceNow Discovery source. The default is "Other Automated".

- **snow_batch_size**. Specify the total number of CIs being sent to ServiceNow in each triggered application. The default is 150 objects per batch.

- **snow_chunk_size**. Specify the number of CI objects to send in each chunk or in each sub-payload with a batch (specifed in the **snow_batch_size** field). The default is 150 objects per chunk or sub-payload.

- **max_count_for_relationships**. Specify the maximum number of device IDs to query for sets of relationships in a single chunk from SL1.

- **verify_snow_ssl**. Toggle on (blue) this option to enable verification of the SSL certification when you run this application.

- *exclude_inactive*. Select this option to prevent syncing devices to ServiceNow that are not enabled, unavailable, or in maintenance. By default, this option is not selected.

- *enable_device_active*. Select this option to enable the **Device Active** block in the device GraphQL query, which contains information about the active state of the SL1 device. By default, this option is not selected. Accessing this data in the attribute mappings requires a Jinja2 Template. For more information, see *Using a Jinja2 Template*.

- *enable_asset_networks*. Select this option to enable the **assetNetworks** block in the device GraphQL query, which returns a list of asset networks. By default, this option is not selected. Accessing this data in the attribute mappings requires a Jinja2 Template. For more information, see *Using a Jinja2 Template*.

> **WARNING**: Please note that enabling this option might cause performance issues on the SL1 side.

- *use_ap2_url*. Select this option if you want the device URLs in the *SL1 URL* field in ServiceNow CIs to point to the corresponding **Device Investigator** page in the SL1 user interface: `/inventory/devices/<SL1_device_ID>/investigator`. This option is enabled by default. If you turn off this option, the device URL points to the classic EM7 user interface: `/em7/index.em7?exec=device_summary&did=<SL1_device_ID>`.

- *unmerge_devices*. De-select this option if you want to turn off the "unmerge" behavior added in version 3.2.0 of this SyncPack, where PowerFlow splits a merged device record into two objects to allow for correct default relationships in ServiceNow. This option is enabled by default.

- *Domain_Separation*. Select this option if your ServiceNow environment is *domain-separated*, where the data, processes, and administrative tasks have been organized into logical groupings called *domains*. If your ServiceNow instance is domain-separated, the user listed in the *snow_user* field must be a member of the top domain and have access to *all* of the domains you intend to integrate. Also, ServiceNow should be the "source of truth" for organizations if your environment is domain-separated. If this option is selected, PowerFlow syncs the ServiceNow company *sys_id* with the corresponding SL1 organization.

- *drop_sys_id*. Select this option if you want to remove the *sys_id* in existing CIs from the sync. If you set *drop_sys_id* to *true*, make sure that ServiceNow can correctly identify and correlate your existing CIs with the properties that are available. By default, this option is not selected.

- **drop_company**. Select this option if you want to remove the **sys_id** in existing Companies from the sync, which prevents the "company" field from being sent to ServiceNow. Selecting this option has no effect if you selected the **Domain_Separation** option for this application. By default, this option is not selected.

> NOTE: If you select this option, the ServiceNow company **sys_id** for the CI and the SL1 organization **crm_id** for the SL1 device will not be compared when determining if an update has occurred. Do not select this option if you selected the **Domain_Separation** option, as ServiceNow requires PowerFlow to send the "company" field in domain-separated environments. Also, do not select this option if you selected the **change_device_organizations** option, as changing the company for a CI in ServiceNow requires the "company" field to be sent.

- **change_device_organizations**. When enabled, this parameter allows devices to change companies with a Device Sync if the companies are within the same domain in ServiceNow. If you change the Organization/Company and another field or fields, but do not enable this parameter, PowerFlow will not apply any changes to the device. By default, this option is not selected.

- **generate_report**. Select this option to create a report about the devices that you sync with ServiceNow. PowerFlow generates a report every time you run the device sync application, and the reports are available on the **Reports** page of the PowerFlow user interface.

- **enable_advanced_topology**. Select this option to allow the pull and process advanced topology step to be executed when the sync is run. If disabled, this step will be skipped. This toggle defaults to a value of *True*.

- **Simulation_Mode**. Select this option if you want to perform a simulated run of this application to show you the potential results of that run. By default, this option is not selected.

- **retry_jitter**. When selected, instead of using a defined interval between retries, the PowerFlow system will retry the step execution at random intervals. By default, this option is not selected.

- **retry_backoff**. When selected, instead of using a defined interval between retries, PowerFlow will incrementally increase the interval between retries. By default, this option is not selected.

- *gql_filter*. Using JSON, you can optionally define a custom GraphQL filter to apply to the devices in the sync. To test out a GraphQL filter, go to the GraphiQL interface in SL1 by typing the URL or IP address for SL1 in a browser, add **/gql** to the end of the URL or IP address, and press **[Enter]**. Search the built in GraphQL docs for **DeviceSearch** and determine how to set up your custom filter. An example of a Device Search query using JSON: `{"name": {"doesNotBeginWith": "jc-is-ma"}}`

> **NOTE:** You can use this field to filter devices by Device Group. For more information, see *Filtering Device Sync by Device Group*.

> **CAUTION:** A misconfiguration in this field might change the monitoring flag for one or more devices. For example, a misconfiguration in this field could switch the SL1 Monitored flag from "true" to "false", removing that device or devices from the device sync. This field is an advanced feature that requires a basic knowledge of the SL1 GraphQL implementation. For more information, ask your ScienceLogic contact for access to the ScienceLogic GraphQL documentation.

- *customer_ci_relation_overrides*. To override existing relationship linking and directly control the link between Device Classes and attributes, add JSON code to this field. The JSON for this field includes default relationship overrides for VMware instead of direct parent/child relations. For more information, see *Configuring Customer CI Relation Overrides* and *Mappings between SL1, ServiceNow, and Other Applications.*

6. In the **Mappings** section, click the **[Edit]** button (PowerFlow 2.7.0 or later) or the expand button (⌄) to view and edit the mappings between ServiceNow CI classes and SL1 device classes. This section is pre-loaded with a large number of default device class mappings. You can map a single ServiceNow CI class with multiple SL1 device classes.

> **NOTE:** The "Sync Devices from SL1 to ServiceNow" application will *only* sync a device from SL1 if the device class for that device is mapped to a ServiceNow CI class in the **Mappings** section. The default mappings in this section do not cover all technologies, however, and syncing additional technologies from SL1 to ServiceNow might require additional research to understand the class structure.

For more information about how to view, edit, and create mappings, see *Editing Mappings in a PowerFlow Application*. This topic also covers how to use the **Company Mapping Override** and the **Attribute Mappings** sections, below.

7. In the **Company Mapping Override** section, you can create a new mapping for the ServiceNow *company* field. You can override the *company* field with a different ServiceNow field where you can read and write the **company_sys_id** for CIs.

8. In the **Attribute Mappings** section, click the **[Edit]** button (PowerFlow 2.7.0 or later) or the expand button
    (⌄) to view and edit the mappings for any other custom device attributes you want to sync between SL1
    (the first column) and ServiceNow (the second column). For more information, including lists of default
    and available device attribute mappings, see *Device Attribute Mappings*.

    To sync Device Notes between SL1 and ServiceNow, create a custom attribute for the device note in this
    section. This option works best if you only want a single value synced over so that value can remain in the
    notes. All custom attributes for each SL1 device are automatically synced.

    > **TIP:** For the **Attribute Mappings** section, you can use a Jinja2 Template for device attribute fields on
    > the SL1 side (the left column). For more information, see *Using a Jinja2 Template*.

    > **NOTE:** When an attribute value is "0" in SL1, the corresponding field in ServiceNow might display as
    > empty.

9. Click **[Save]**. The **Configuration** pane closes.

10. Run the "Sync Devices from SL1 to ServiceNow" application.

11. When the application completes, open the **Step Log** and review the log messages for the "Compare
    SL1 Devices and ServiceNow CIs" step to see if any Device or CI records were added, updated, or
    disconnected from the sync. As needed, select the other steps to review the logs on the **Step Log** for those
    steps.

> **NOTE:** Depending on the number of devices you are syncing to ServiceNow, it might take a few minutes for
> all devices to get fully synced to the CMDB. You might notice after running device sync that the
> number of SL1 Monitored CIs continues to increase after each refresh. This is expected behavior due
> to payload chunking in ServiceNow. ServiceNow processes each payload as an individual chunk.

ScienceLogic recommends that you schedule a Device Sync to run every 24 hours. For more information, see
Scheduling PowerFlow Applications.

Using a Jinja2 Template

The attribute mappings in Device Sync applications now support Jinja2 Templates, which let you sync complex,
concatenated (linked) fields from SL1 to ServiceNow. For example, you can add these complex values in the SL1
side of the **attribute_mappings** section of the **Configuration** pane for the "Sync Devices from SL1 to
ServiceNow"application, and that value is mapped to one or many fields in ServiceNow. For more information
about Jinja2 Templates, see the Template Designer Documentation.

In the "Sync Devices from SL1 to ServiceNow" application, the SL1 side can be a Template. In the "Sync CI
Attributes from ServiceNow to SL1" application, the ServiceNow side can be a Template.

Example: A Basic Template for Device Attributes

This example is included in the "Sync Devices from SL1 to ServiceNow" application as the first default value in the **attribute_mappings** section of the **Configuration** pane:

This template, when used on the SL1 side of the **attribute_mappings** section, populates the **short_description** field in ServiceNow:

```
"Description: {{device.device_category}}, Device Class: {{device.device_
class}}": [ "short_description" ]
```

In the above example, for a device with a **category: Testing** and a Device Class of **Testing | Testing**, the end result would be **Description: Testing, Device Class: Testing | Testing**, which will be posted to the **short_description** field in ServiceNow.

The Jinja2 Templates will have access to all properties on the Device.

> **NOTE**: Any item that is generated by a template is always a string.

Example: An Advanced Template for Device Sync

The following example lets you get the active status of SL1 Devices:

```
{%- set output = [] -%}
{%- if device.active.unavailable == True -%}{%- set output = output +
['Unavailable'] -%}
{%- endif -%}
{%- if device.active.userDisabled == True -%}{%- set output = output +
['User Disabled'] -%}
{%- endif -%}
{%- if device.active.userInitiatedMaintenance == True -%}{%- set output =
output +
['User Initiated Maintenance'] -%}{%- endif -%}
```

```
{%- if device.active.userMaintenance == True -%}{%- set output = output +
['User Initiated Maintenance'] -%}{%- endif -%}
{%- if output|length > 0 -%}
{{ ", ".join(output) }}
{%- else -%}
{{ "Active" }}
{%- endif -%}
```

Example: Another Advanced Template for Device Sync

The following example shows another way to get the active status of SL1 Devices:

```
{%- set prettify = {"userInitiatedMaintenance": "User Initiated Main-
tenance", "systemDisabled":
"System Disabled", "maintenance": "System Maintenance", "unavailable":
"Unavailable",
"userDisabled": "User Disabled"} -%}
{%- set ns = namespace(maint=[]) -%}
{%- for k,v in device.active.items() -%}
{%- if v -%}
{%- set ns.maint = ns.maint + [prettify[k]] -%}
{%- endif -%}
{%- endfor -%}
{{", ".join(ns.maint) if ns.maint else "Active"}}
```

Example: Advanced Template for Device Class, Sub Category, and Model

The following example shows how to gather device data based on Device Class and Sub Category if available, or Model if Device Class and Sub Category are not available:

```
{%- set output = [] -%}
{%- set output = device.device_class.split('|') -%}
{%- if (output[0]|trim) in ['Checkpoint', 'ScienceLogic', 'Microsoft'] -%}
{%- set output = output[1]|trim -%}
{% else %}
{%- set output = device.model -%}
{%- endif -%}
{{output}}
```

Filtering Device Sync by Device Group

You can use the *gql_filter* field on the **Configuration** pane of the "Sync Devices from SL1 to ServiceNow" application to add a block of JSON code that lets you define a custom filter to apply to the devices in the sync.

> **NOTE:** You cannot enter GraphQL code in the **gql_filter** field in PowerFlow. The field *only* accepts JSON.

Using GraphQL to Create the Custom Filter

In this example, you will create a custom GraphQL filter in SL1 based on Device Groups. This filter will prevent Devices that belong to specific Device Groups from creating CIs in the ServiceNow CMDB.

To create a JSON filter for the **gql_filter** field:

1. Go to the GraphiQL interface in SL1 by typing the URL or IP address for SL1 in a browser, adding **/gql** to the end of the URL or IP address, and pressing **[Enter]**.

2. Create a GQL query that uses a search variable, such as the following example:

```
query DeviceFetch($search: DeviceSearch, $order: [ConnectionOrder],
$first: Int) {
  devices(first: $first, search: $search, after: "", order: $order) {
    pageInfo {
      hasNextPage
      matchCount
    }
    edges {
      cursor
      device: node {
        ...
      }
    }
  }
}
```

3. After the query gives you the results you want, copy the resulting JSON variable definitions from the **Query Variables** section:

4. Paste that JSON code into the **gql_filter** portion of PowerFlow and save the PowerFlow application.

JSON Code Template

Instead of using the GraphiQL user interface in SL1, you can use the following JSON code as a template for filtering based on Device Group; simply paste the code into the *gql_filter* field on the **Configuration** pane and edit the search criteria and group names as needed:

```
{
  "deviceGroup": {
    "has": {
      "or": [
        {
          "name": {
            "contains": "group1"
          }
        },
        {
          "name": {
            "contains": "group2"
          }
        }
      ]
    }
  }
}
```

Tips for customizing the JSON code:

- To add or remove additional Device Groups, copy or delete the following block of code and edit the group name as needed:

```
{
  "name": {
    "contains": "group2"
  }
}
```

- If you add additional name blocks, add a comma after each name block except for the last name block. If you delete a block, delete the preceding comma.

- You can change the "or" to an "and" if you want to block syncing for devices that are in more than one Device Group.

- The "contains" can be changed to "eq" to make the name an exact match. For example, "contains": "office" would match "The Office," "HQ Office," and "Office235."

Adding Device Class Mappings

You can dynamically set the device mappings on a per-run basis using the API. You can also *persistently save device mappings with the API*. You can find these mappings in the **mappings** section of the **Configuration** pane for the "Sync Devices from SL1 to ServiceNow" application.

The following image displays an example of using Postman to send the mapping data to PowerFlow:
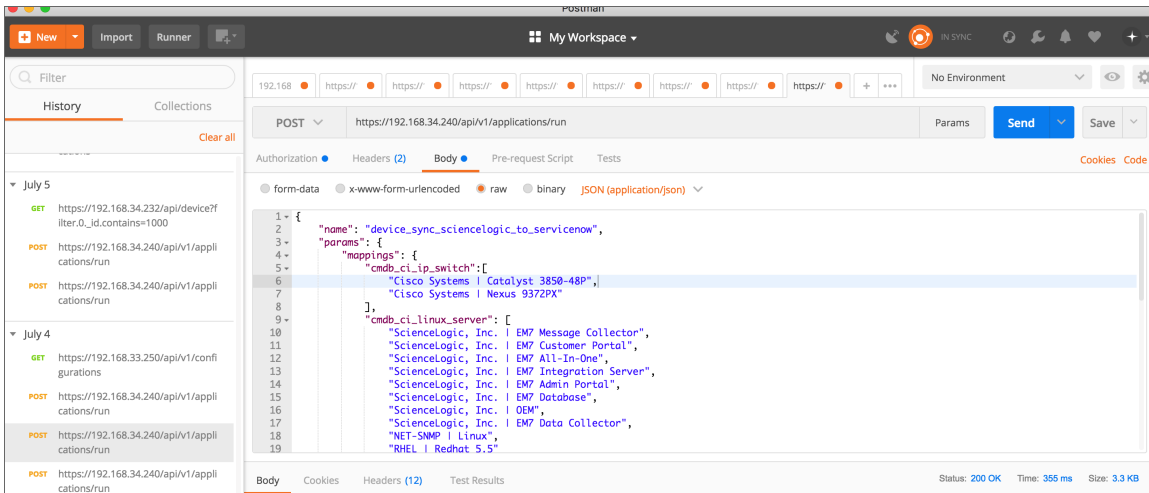
NOTE: This example only maps device classes to ServiceNow for VMware, SL1 devices, and a few Cisco devices. If the your environment contains other device classes, you must manually create the mappings.

To add device class mappings using Postman, POST the following JSON file to trigger the required applications in the PowerFlow user interface to model SL1 devices to ServiceNow:

```
{

    "name": "device_sync_sciencelogic_to_servicenow",

    "params": {

        "mappings": {

            "cmdb_ci_ip_switch":[

                "Cisco Systems | Catalyst 3850-48P",

                "Cisco Systems | Nexus 9372PX"

            ],
```

```
"cmdb_ci_linux_server": [

  "ScienceLogic, Inc. | EM7 Message Collector",

  "ScienceLogic, Inc. | EM7 Customer Portal",

  "ScienceLogic, Inc. | EM7 All-In-One",

  "ScienceLogic, Inc. | EM7 Integration Server",

  "ScienceLogic, Inc. | EM7 Admin Portal",

  "ScienceLogic, Inc. | EM7 Database",

  "ScienceLogic, Inc. | OEM",

  "ScienceLogic, Inc. | EM7 Data Collector",

  "NET-SNMP | Linux",

  "RHEL | Redhat 5.5"

],

"cmdb_ci_esx_resource_pool": ["VMware | Resource Pool"],

"cmdb_ci_esx_server": [

  "VMware | ESXi 5.1 w/HR",

  "VMware | Host Server",

  "VMware | ESX(i) 4.0",

  "VMware | ESX(i) w/HR",

  "VMware | ESX(i) 4.0 w/HR",

  "VMware | ESX(i)",

  "VMware | ESX(i) 4.1 w/HR",

  "VMware | ESXi 5.1 w/HR",

  "VMware | ESXi 5.0 w/HR",

  "VMware | ESX(i) 4.1",
```

```
      "VMware | ESXi 5.1",

      "VMware | ESXi 5.0"

    ],

    "cmdb_ci_vcenter_datacenter": ["VMware | Datacenter"],

    "cmdb_ci_vcenter_datastore": ["VMware | Datastore", "VMware |
    Datastore Cluser"],

    "cmdb_ci_vcenter_dv_port_group": ["VMware | Distributed Virtual
    Portgroup"],

    "cmdb_ci_vcenter_dvs": ["VMware | Distributed Virtual Switch"],

    "cmdb_ci_vcenter_folder": ["VMware | Folder"],

    "cmdb_ci_vcenter_network": ["VMware | Network"],

    "cmdb_ci_vmware_instance": ["VMware | Virtual Machine"],

    "cmdb_ci_vcenter": ["VMware | vCenter", "Virtual Device | Windows
    Services"],

    "cmdb_ci_vcenter_cluster": ["VMware | Cluster"]

  },

  "configuration": "template_snow_integration" #name your configuration
  file

  }

}
```

Persistently Saving Device Class Mappings with the API

You can persistently save device class mappings using the API.

1. Use Postman or cURL to do a GET to load the "Sync Devices from SL1 to ServiceNow" application:

   ```
   GET PowerFlow_hostname/api/v1/applications/device_sync_sciencelogic_
   to_servicenow
   ```

   where *PowerFlow_hostname* is the IP address or URL for your PowerFlow system.

   > NOTE: The response should contain the entire JSON output for the application.

2. Copy the entire JSON code and save it to a file named: "device_sync_sciencelogic_to_servicenow".

3. Open the file you created and locate the object with the `"name":"mappings"` property in the **"app_variables"** list. The `"value"` property in this object specifies the mappings to use throughout the PowerFlow applications:

```
"value": {
    "cmdb_ci_appl_sharepoint": [
        "VMware | Resource Pool"
    ],
    "cmdb_ci_esx_resource_pool": [
        "VMware | Resource Pool"
    ],
    "cmdb_ci_esx_server": [
        "VMware | ESXi 5.1 w/HR",
        "VMware | Host Server",
        "VMware | ESX(i) 4.0",
        "VMware | ESX(i) w/HR",
        "VMware | ESX(i) 4.0 w/HR",
        "VMware | ESX(i)",
        "VMware | ESX(i) 4.1 w/HR",
        "VMware | ESXi 5.1 w/HR",
        "VMware | ESXi 5.0 w/HR",
        "VMware | ESX(i) 4.1",
        "VMware | ESXi 5.1",
        "VMware | ESXi 5.0"
    ],
    "cmdb_ci_hyper_v_network": [
        "VMware | Resource Pool"
    ],
```

Syncing Devices from SL1 to ServiceNow

4. Modify the `"value"` property of the object to use the mappings you want to use.

5. Ensure that the mappings follow the same JSON data structure, or else the sync will not work:

```
{

    "cmdb_ci_class": [

        "ScienceLogic Dev Class| ScienceLogic subclass",

        "Another Silo Dev Class | Another Silo subclass"

    ]

}
```

6. After you update the mappings, use the iscli tool to upload the updated application with your new settings. Type the following command at the command line:

```
iscli -uaf device_sync_sciencelogic_to_servicenow -H PowerFlow_
hostname -p password
```

where:

- *PowerFlow_hostname* is the hostname or IP address of the PowerFlow system.
- *password* is the password you use to log in to the PowerFlow system.

Checking for Missing Device Mappings

You can use the "Report: Identify Unmapped Devices Classes" PowerFlow application to check whether any device mappings are missing in your PowerFlow server.

This application pulls the class mappings from Device Sync and Attribute Sync and compares the mappings with the full list of device classes of discovered devices in SL1. The application generates a report on the **Reports** page that lists missing mappings, and if any device classes are unmapped, the application generates an event in the target SL1 system.

To configure the "Report: Identify Unmapped Devices Classes" application:

1. If you do not have the "ServiceNow Base" PowerPack version 104 or later on your SL1 system, search for and download the PowerPack from the **PowerPacks** page on the ScienceLogic Support Site at https://support.sciencelogic.com/s/powerpacks. Install the PowerPack in SL1.

2. In SL1, enable the "ServiceNow CMDB: Un-Mapped Device Classes" Event Policy from the "ServiceNow Base" PowerPack version 104 or later to trap the alert generated by this application.

3. In the PowerFlow user interface, go to the **Configurations** page.

4. Click the **[Actions]** button (⋮) for the configuration object you want to use with this application and select *Edit*. The **Configuration** pane for the object appears.

5. In the **Configuration Data Values** section, click **[Add Value]**. A new name-value line appears.

6. Add a configuration variable named *pf_host* that has a value of the externally addressable IP or fully qualified domain name (FQDN) of the PowerFlow cluster or instance.

7. Click **[Save]**.

8. Go to the **Applications** page and select the "Report: Identify Unmapped Devices Classes" application.

9. Click **[Configure]** to open the **Configuration** pane:

10. Complete the following fields, as needed:

    - *Configuration*. Select the configuration object you updated in steps 1-5 to align with this application. Make sure that the *pf_host* value that is updated on the **Configuration** pane is accurate. You cannot edit fields that are populated by the configuration object. Required.

    - *read_timeout*. Specify the maximum amount of time in seconds the application should wait for a response before timing out. The default is 20 seconds.

    - *query_retries*. Specify the number of times to retry this query if needed. The default chunk size is 0.

11. Click **[Save]**. The **Configuration** pane automatically closes after this message appears.

12. Click **[Run]** to run the application.

13. After the application runs, click the **[Reports]** button and select the relevant *report-missing-classes* report from the **Reports** page. The "missing_classes" report appears:

> **TIP:** The report displays missing mappings for Device Sync (SL1 to ServiceNow) in the first column, and missing mappings for Attribute Sync (ServiceNow to SL1) in the second column.

Also, if any device classes are unmapped and you enabled the "ServiceNow CMDB: Un-Mapped Device Classes" Event Policy in step 2, SL1 generates an event on the **Events** page in SL1:
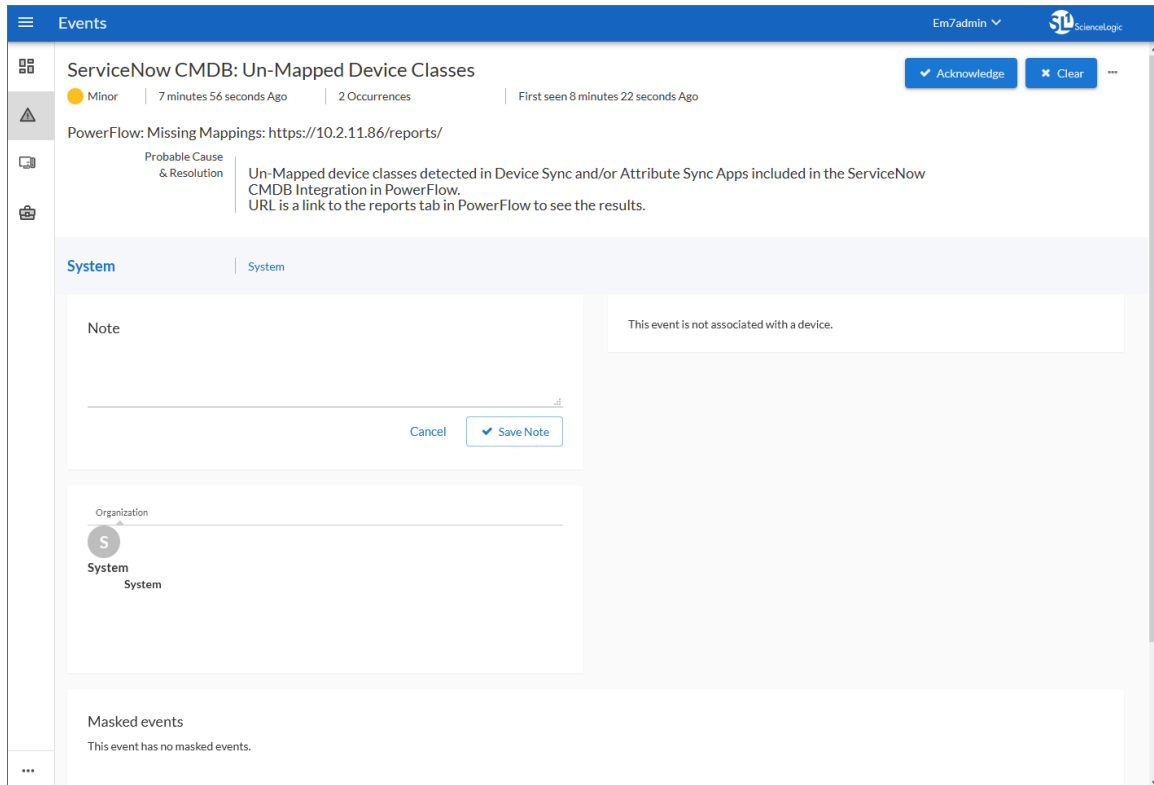
Device Attribute Mappings

The "Sync Devices from SL1 to ServiceNow" application can also collect manufacturer and model attributes from asset records aligned with devices in SL1 and sync that information with ServiceNow. You can access these mappings in the **Attribute Mappings** section of the **Configuration** pane for that application.

> **NOTE:** These manufacturer and model attributes are different from custom attributes in SL1.

PowerFlow only populates the manufacturer and model attributes if the values exist in ServiceNow CIs; PowerFlow does not create new manufacturer values in ServiceNow. "Sync Devices from SL1 to ServiceNow" application uses the *sys_id* field as a reference when syncing manufacturer and model information between SL1 and ServiceNow.

> **WARNING:** Integer fields in ServiceNow have a maximum value of 2147483647. If a value exceeds that value, ServiceNow stores it as 2147483647. This is a MySQL limitation on the maximum value that can be stored in a signed integer variable.

> **NOTE:** The "Sync Devices from SL1 to ServiceNow" application only syncs assets that have a value populated in SL1. SL1 automatically populates fields in an asset record when the record is properly configured. For more information on configuring assets in SL1, see the *Asset Management* manual.

Try one of the following options for the best approach for syncing manufacturer and model attributes, especially if your ServiceNow instance uses Domain Separation:

- Add the ServiceNow user that is listed in the *snow_user* field on the **Configuration** pane of the "Sync Devices from SL1 to ServiceNow" application to the Global Domain in ServiceNow.

- Schedule the "Sync Devices from SL1 to ServiceNow" application to run with different ServiceNow users that belong to each ServiceNow domain, using the *Custom Parameters* field on the schedule window to overwrite the *snow_user* value.

Default Device Attribute Mappings

The "Sync Devices from SL1 to ServiceNow" application contains a set of default device attribute mappings between SL1 and ServiceNow. You can access these mappings in the **Attribute Mappings** section of the **Configuration** pane for that application.

The following table describes the default device attribute mappings:

SL1 Device Attribute	ServiceNow CI Attribute
Description	`"Description: {{device.device_category}}, Device Class: {{device.device_class}}": [ "short_description"` **NOTE**: This field requires a Jinja2 Template. For more information, see *Using a Jinja2 Template*.
asset_tag	asset_tag
cpuCount	cpu_count
cpuMake	cpu_type
cpuSpeed	cpu_speed
dateAdded	first_discovered
diskSize	disk_space
dnsDomain	dns_domain
function	justification
hostname	fqdn, host_name
instance_uuid	account_id
ip	ip_address
manufacturer_sys_id	manufacturer
memory	ram
model_sys_id	model_id
name	name

SL1 Device Attribute	ServiceNow CI Attribute
operatingSystem	os
purchaseDate	order_date
serial	serial_number
status	hardware_substatus
warrantyExpirationDate	warranty_expiration

SL1 Device Attributes Available for Syncing

In the **Attribute Mappings** section of the **Configuration** pane for the "Sync Devices from SL1 to ServiceNow" application, you can also use the following SL1 device attributes from SL1 when syncing attributes with ServiceNow:

- arraySize
- asset_id
- asset_tag
- company_sys_id
- component_unique_id
- cpuCount
- cpuMake
- cpuSpeed
- dateAdded
- depreciationMethod
- depreciationSchedule
- device_category
- device_class
- diskCount
- diskSize
- dnsDomain
- dnsName
- domain_sys_id
- firmwareVersion
- floor
- function
- hostId
- hostname
- ip
- location

- make
- manufacturer_sys_id
- memory
- model
- modelNumber
- model_sys_id
- name
- operatingSystem
- org_id
- org_name
- owner
- panel
- parent_device
- parent_did
- plate
- punch
- purchaseCheck
- purchaseCost
- purchaseDate
- purchaseOrderNumber
- rack
- region
- rfid
- room
- serial
- serviceCheck
- serviceCost
- serviceDate
- serviceDescription
- serviceExpirationDate
- serviceOrderNumber
- servicePolicyNumber
- shelf
- sl1_id
- sl1_url
- snow_ci_class

- snow_sys_id

- status

- vitalAssetInformation

- vitalServiceInformation

- warrantyCheck

- warrantyCost

- warrantyDate

- warrantyDescription

- warrantyExpirationDate

- warrantyOrderNumber

- warrantyPolicyNumber

- zone

Adding New Device Attributes to ServiceNow

You can also add one or more new attributes to ServiceNow that you can then sync with SL1.

To add an attribute in ServiceNow:

1. In ServiceNow, go to the **Tables** page (System Definition > Tables) and select the table to which you want to add a field for a new attribute.

2. From the **Table** page, click the **[New]** button to add a new field on the table. A new record appears.

3. From the **Type** drop-down list, select the data type you want to store, such as *String*. Depending on your selection, additional required fields display. For example, If you selected *String*, then **Column label** should contain the text you want to display in ServiceNow, and **Column name** is the exact column name used by PowerFlow or the API.

4. Complete the required fields and any other fields as needed, and then click the **Submit** button. The field is added to ServiceNow.

Configuring Customer CI Relation Overrides

When you are mapping Device Classes and attributes, you might find that SL1 creates relationship mappings very differently than the way that ServiceNow creates relationships. As a result, ScienceLogic strongly recommends that you use the **customer_ci_relation_overrides** field instead of using ServiceNow to set up those relationships.
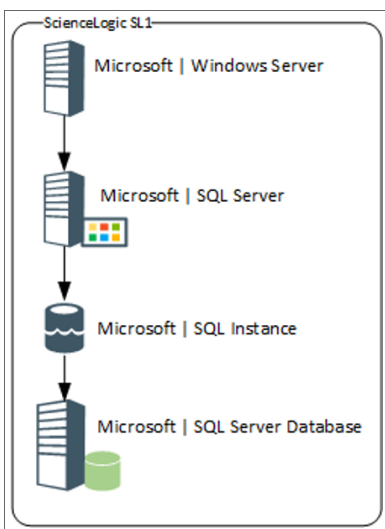
In the "Sync Devices from SL1 to ServiceNow" PowerFlow application, you can use the **customer_ci_relation_overrides** field to override the existing relationship linking and directly control the link between Device Classes and attributes. The **customer_ci_relation_overrides** field lets you build dynamic relationships rather than statically setting up relationships within ServiceNow.

> **WARNING:** ScienceLogic does not support using ServiceNow to control and set up your device relationships.

In addition, ScienceLogic strongly recommends that you use the default relationship overrides for VMware, which you can view by clicking **[Show JSON Configs]** from the *Configuration* pane for the "Sync Devices from SL1 to ServiceNow" PowerFlow application.

> **CAUTION:** This mapping process is intended for advanced users that are familiar with how SL1 and ServiceNow construct device relationships.
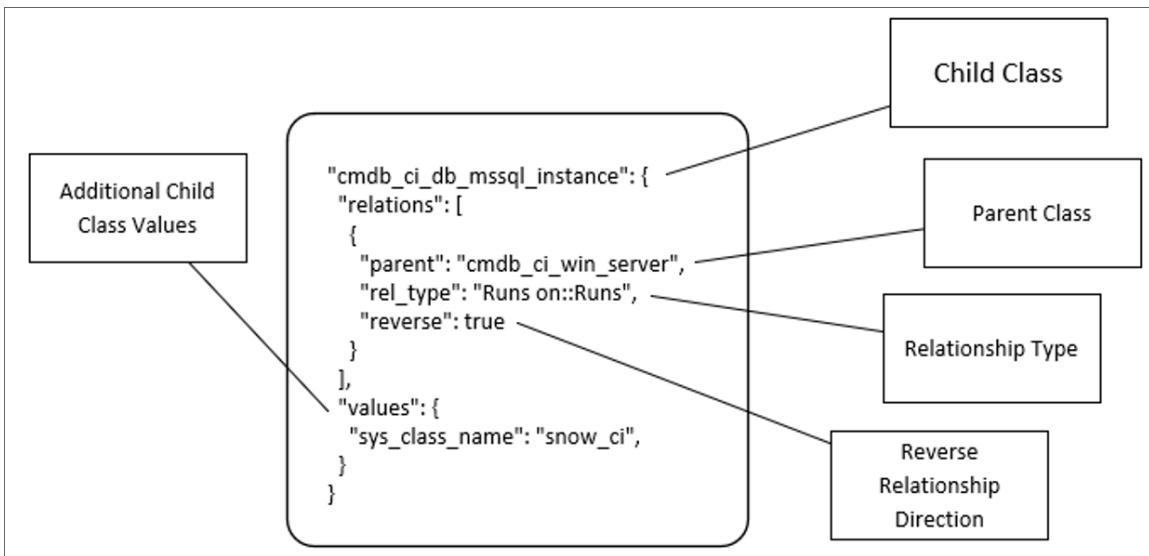
In the following example, the relationship structure in SL1 is linear :

In ServiceNow, however, the structure is not as linear, and it requires an override (a manual link) between classes to make the relationship link required:
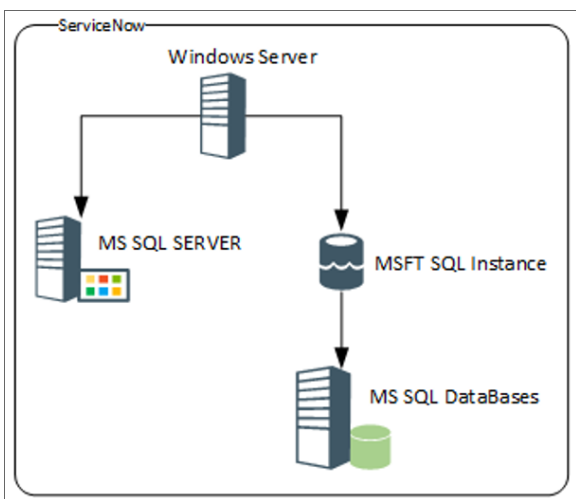
The following image shows the JSON structure formatting that is required for the ***customer_ci_relation_overrides*** field:

The values in the *customer_ci_relation_overrides* field supersede any of the values configured in the **mappings** section in the **Configuration** pane for the "Sync Device Classes from SL1 to ServiceNow" PowerFlow application.

> **WARNING:** You must ensure that all classes in the relationship chain in SL1 are mapped to classes in ServiceNow, or else the chain will break, and PowerFlow will not correctly apply the overrides.
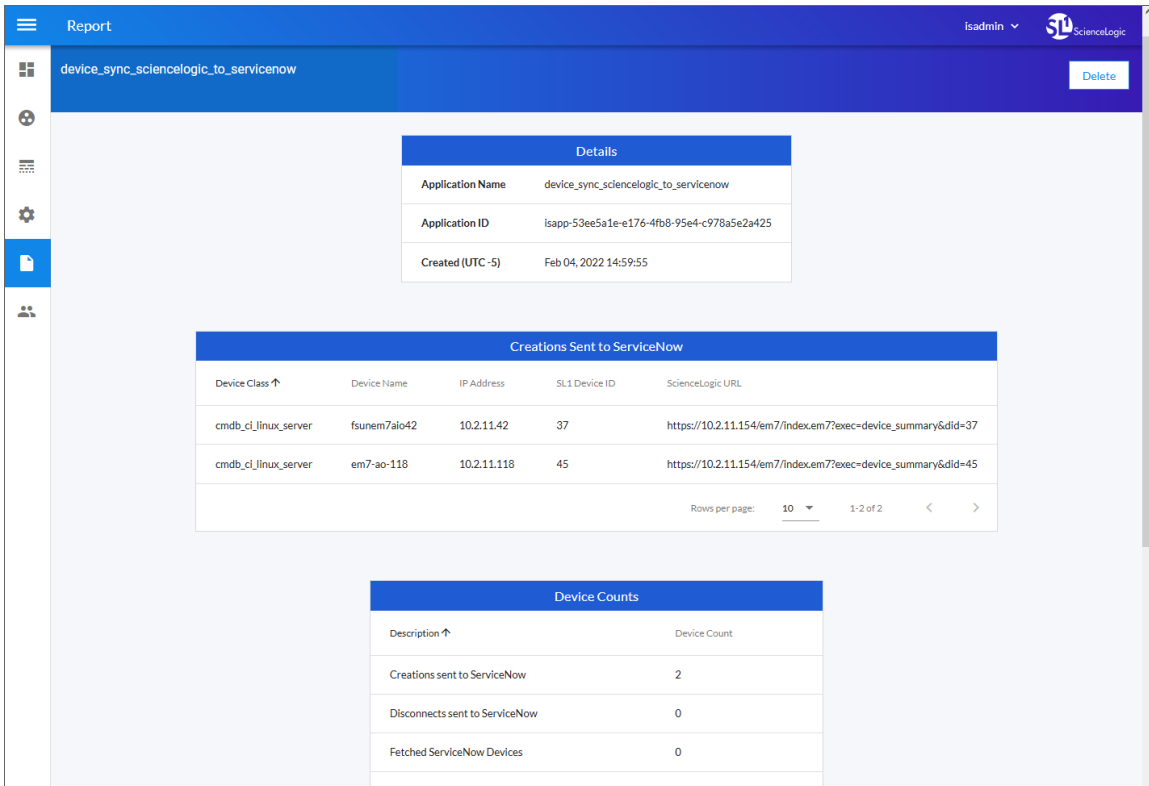
In the *customer_ci_relation_overrides* field, you can string together multiple relationships as in the following example:

```
{
    "cmdb_ci_db_mssql_instance": {
        "relations": [
            {
                "parent": "cmdb_ci_win_server",
                "rel_type": "Runs on::Runs",
                "reverse": true
            }
        ],
        "values": {"sys_class_name": "snow_ci_class", "instance_name":
"name"}
    },
    "cmdb_ci_db_mssql_database": {
        "relations": [
            {
                "parent": "cmdb_ci_db_mssql_instance",
                "rel_type": "Contains::Contained by",
                "reverse": false
            }
        ],
        "values": {"sys_class_name": "snow_ci_class", "database": "name"}
    },
    "cmdb_ci_db_mssql_server": {
        "relations": [
            {
                "parent": "cmdb_ci_win_server",
                "rel_type": "Runs on::Runs",
                "reverse": true
            }
        ],
        "values": {"sys_class_name": "snow_ci_class", "instance_name":
```

```
"name"}
    }
}
```

Viewing Reports for Device Sync

The "Sync Devices from SL1 to ServiceNow" application lets you create a report about the devices that you sync with ServiceNow. The report displays on the **Reports** page (📄):

> **TIP:** Enable the report by selecting the *generate_report* option on the **Configuration** pane for the application.

When the relevant data is present in a device sync, the report displays the following data under the **Details** pane:

- **Creations Sent to ServiceNow**. Device information for devices in SL1 that will be created in ServiceNow.
- **Device Counts**. Lists the number of created, disconnected, and updated devices, as well as the number of devices pulled from SL1 and ServiceNow during the sync. If domain separation is on and there are devices removed due to domain separation errors, that count will be added to the table.
- **Deletions Sent to ServiceNow**. Device information for devices in SL1 that will be deleted in ServiceNow.

Syncing Devices from SL1 to ServiceNow

- **Post CIs to ServiceNow**. Each instance of the "Post CIs to ServiceNow" application, listing creations, disconnects, errors, skips, and updates made in ServiceNow.

- **Updates Sent to ServiceNow**. Device information for devices in SL1 to be updated in ServiceNow.

Log Messages for the "Generate Required CI Relations for ServiceNow" Application

This section describes the different types of log messages you might see in the Step Log when you run the "Generate Required CI Relations for ServiceNow" application. Please note that this application is a report used by PowerFlow, and it does not send any data to ServiceNow.

The following message displays if there are devices in a device tree that do not currently have a CI class mapping assigned.

```
Warning: 2751 Relations with missing mappings detected. Please re-run app
    with log level 10 to troubleshoot.
```

In this situation, the device tree cannot be built in ServiceNow. To address this issue, make sure that you have your entire technology tree mapped out in the *mappings* section of the "Sync Devices from SL1 to ServiceNow" application or in the *mappings* section of the "Generate Required CI Relations for ServiceNow" application.

If you do a Custom Run of the "Generate Required CI Relations for ServiceNow" application in Debug mode (log level 10), the application will create a log that displays the parent and child class, CI, and device ID. For example:

```
Debug: Missing Mapping for Device. Parent: {"class": "VMware | Cluster",
    "ci": None, "id": 76}, Child: {"class": "VMware | Host Server", ci:
    "cmdb_ci_esx_server", id: 363 }
```

The following message appears if the GraphQL payloads had bad data for parent and or child devices:

```
Warning: 10 bad payloads received from SL1. Re-run app in debug to
    troubleshoot.
```

If you do a Custom Run the application in Debug mode, the application will create a log that displays these payloads.

The following message appears if all relations are mapped:

```
Flow: No missing relations found!
```

The following message appears if there is a parent/child relation between ServiceNow CI classes that does not currently exist in ServiceNow and is required to sync those devices:

```
Flow: Missing Relations: [{"parent": "cmdb_ci_vcenter_folder", "child":
    "cmdb_ci_esx_server"}, {"parent": "cmdb_ci_vcenter", "child": "cmdb_ci_
    vcenter_datacenter"}]
```

Refer to the labels in the log (above) to determine which CI class is the parent type and which is the child type. To address this issue, navigate to your ServiceNow instance and create the required service rules based on the recommendations in the **Step Log**.

The following message appears if the application encounters a list of relations that are required, but were successfully found in ServiceNow:

```
Info: Found Relations: [{"parent": "cmdb_ci_vcenter_folder", "child":
    "cmdb_ci_esx_server"}, {"parent": "cmdb_ci_vcenter", "child": "cmdb_ci_
    vcenter_datacenter"}]
```

This message lets you verify that your mappings and relations are configured correctly.

Chapter

4

Configuring Additional Syncs for the CMDB SyncPack

Overview

This chapter describes the how to configure and run the various PowerFlow applications contained in the "ServiceNow CMDB" SyncPack.

For more information about Organization and Device Sync, see *Configuring Organization Sync and Device Sync for the CMDB SyncPack*.

For more information about using configuration objects with this SyncPack, see Creating and Aligning a Configuration Object.

This chapter covers the following topics:

Syncing Business Services

Depending on where you have your business services configured initially, you can use one of the two following methods to sync business services between SL1 and ServiceNow:

- *Syncing Business Services from SL1 to ServiceNow*

- *Syncing Business Services from ServiceNow to SL1*

To avoid duplicate services, you should only use *one* of the above processes in the PowerFlow user interface. The PowerFlow application you use depends on in which application (SL1 or ServiceNow) you initially configured your services, before syncing.

For example, if you have Business Services, IT Services, and Devices Services set up in SL1, you would use the "Sync Business Services from SL1 to ServiceNow" application, and SL1 would be the "source of truth" after the sync.

Syncing Business Services from SL1 to ServiceNow

The *Sync Business Services from SL1 to ServiceNow* application reads Business Services, IT Services, and Device Services from SL1 and syncs them with business services in ServiceNow. This application creates and updates services, but it does not delete services.

The "Sync Business Services from SL1 to ServiceNow" application does not currently support deleting or disconnecting services within ServiceNow. The application logs for the "Post to ServiceNow" step might contain information regarding disconnects, but you can ignore that information.

If a Device Service does not have a parent IT Service, the CMDB Group will be created, but the CI will not be created because of the way PowerFlow pulls each of those items. At the time of the Group creation, PowerFlow does not know if a service has a parent or not. At the time of the CI creation, PowerFlow does not directly pull Device Services; it only pulls Business Services and IT Services and their children.

> **WARNING:** PowerFlow only syncs business services that are aligned with devices that are already synced with ServiceNow. Before setting up business service sync, you must first *sync devices between SL1 and ServiceNow*.

To sync SL1 business services with ServiceNow:

1. In ServiceNow, create an identifier rule for syncing services by typing "CI Identifiers" in the filter navigator and clicking **[New]** on the **Identifiers** page.

2. Complete the following fields:

   - *Name*. Type a relevant name for this rule, such as "Business Service".

   - *Applies to.* Select *cmdb_ci_service*.

   - *Independent*. Select this option.

3. Right-click the gray header and click *Save* to save the record.

4. On the **[Identifier Entries]** tab, click **[New]** and add the relevant values from the *Criterion attributes* field for this business service, such as *name, service_classification* and *correlation_id*.

5. Click **[Submit]**.

6. Repeat steps 4-5 for each identifier you want to add.

7. Go to the **Applications** page of the PowerFlow user interface and run the "Cache ServiceNow Companies, CIs and SL1 Orgs, Device Classes" application. This application reads all existing SL1 Device Classes, Organizations, ServiceNow CIs, and Companies and writes them to a cache.

8. When that application completes, go to the **Applications** page and run the "Sync Devices from SL1 to ServiceNow" application to sync devices and their properties and relationships from SL1 to ServiceNow.

9. When that application completes, go to the **Applications** page and select the "Sync Business Services from SL1 to ServiceNow" application.

10. Click **[Configure]** to open the **Configuration** pane:

11. Complete the following fields, as needed:

- **Configuration**. Select the relevant configuration object to align with this application. You cannot edit fields that are populated by the configuration object. Required.

> NOTE: The *region* field is populated by the configuration object you aligned with this application. The region value must match the value in the **SL1 Region** field in ServiceNow. If you need to update this value, you will need to define the *region* variable in the configuration object that is aligned with this application, or align a different configuration object that has the correct *region* value.

- **read_timeout**. Specify the maximum amount of time in seconds the application should wait for a response before timing out. The default is 20 seconds.
- **sl1_chunk_size**. Specify the number of services to include in each chunk sent to ServiceNow when you run this application. The default chunk size is 500.
- **snow_request_limit**. Specify the number of CIs fetched from ServiceNow in each request. The default is 500 objects per chunk.
- **sl1_url_override**. Specify a URL that is different from the standard SL1 URL that gets sent to the ServiceNow CI record. Optional.
- **business_service_ci_class**. Specify the ServiceNow CI Class for Business Services. The default is **cmdb_ci_service**.
- **it_service_ci_class**. Specify the ServiceNow CI Class for IT Services. The default is **cmdb_ci_service_technical**.
- **business_service_classification**, **it_service_classification**, and **device_service_classification**. Use these fields to update the default service classifications. Optional.
- **relationship_type**. Specify the relationship type string to use between services. The default is *Depends on::Used by*.
- **discovery_source**. Specify the ServiceNow Discovery source. The default is "Other Automated".
- **verify_snow_ssl**. Toggle on (blue) this option to enable verification of the SSL certification when you run this application.
- **snow_batch_size**. Specify the total number of CIs being sent to ServiceNow in each triggered application. The default is 150 objects per batch.
- **snow_chunk_size**. Specify the number of CI objects to send in each chunk or in each sub-payload with a batch (specifed in the **snow_batch_size** field). The default is 150 objects per chunk or sub-payload.
- **retry_max**. The maximum number of times PowerFlow will retry to execute the step before it stops retrying and logs a step failure. For example, if retry_max is 3, PowerFlow will retry after 1 second, then 2 seconds, then 4 seconds, and stop if the last retry fails. The default is 0.
- **retry_backoff_max**. The maximum time interval for the **retry_backoff** option, in seconds. For example, if you have **retry_max** set to 15, the delays will be 1, 2, 4, 8, 16, 32, 64, 120, 240, 480, 600, 600, 600, 600, and 600. The default is 600.

- **Domain_Separation**. Select this option if your ServiceNow environment is *domain-separated*, where the data, processes, and administrative tasks have been organized into logical groupings called *domains*. If your ServiceNow instance is domain-separated, the user listed in the ***snow_user*** field must be a member of the top domain and have access to *all* of the domains you intend to integrate. Also, ServiceNow should be the "source of truth" for organizations if your environment is domain-separated.

- **Simulation_Mode**. Select this option if you want to perform a simulated run of this application to show you the potential results of that run.

12. In the **Company Mapping Override** section, you can create a new mapping for the ServiceNow *company* field. You can override the *company* field with a different ServiceNow field where you can read and write the **company_sys_id** for CIs.

13. Click **[Save]**. The **Configuration** pane automatically closes.

14. Click **[Run]** to run the "Sync Business Services from SL1 to ServiceNow" application.

Syncing Business Services from ServiceNow to SL1

The **Sync Business Services from ServiceNow to SL1** application syncs services that were defined in ServiceNow with Business Services in SL1. When you sync services from Service Now to SL1, PowerFlow recreates the service structure in SL1, where you can see the relationships between the service components, the application components, and the infrastructure components. You can also sync any asset fields on the service using an attribute mapping on the **Configuration** pane.

Aspects of Syncing Business Services from ServiceNow to SL1

- This process syncs the business service structure or map from ServiceNow, and the device services created in SL1 as the result of the sync can contain multiple devices.

- An aggregate service in SL1 will only be created with services as children and a device service in SL1 will only be created with devices as children.

- Depending on the actions taken in ServiceNow, an aggregate service can be inserted above or a device service may be inserted below certain services when the aggregate service is created in SL1.

- This process creates device services in SL1, but it does not create devices in SL1.

- Services are not merged in this process, so when you use the "Sync Business Services from ServiceNow to SL1" application, the "source of truth" will be set to ServiceNow, not SL1.

- If a ServiceNow service has no child services or child devices, that service will not be created in SL1.

- If a ServiceNow service was previously synced to SL1 when it was not empty, but it is now an empty service ServiceNow, the service is deleted in SL1.

The following tables in ServiceNow are treated as services in a sync to SL1:

- cmdb_ci_service
- cmdb_ci_service_technical
- cmdb_ci_query_based_service
- cmdb_ci_service_discovered
- cmdb_ci_service_auto

- cmdb_ci_service_group
- service_offering
- cmdb_ci_service_manual
- cmdb_ci_service_calculated

All other tables will be treated as devices.

> **NOTE:** The **cmdb_ci_query_based_service** table contains the device services within ServiceNow. For any other service with a CI as a direct child, a device service will be inserted in between this service and the child CI in SL1.

> **WARNING:** Syncing services from ServiceNow to SL1 requires SL1 10.2.0 or later.

To sync ServiceNow services with SL1 Business Services:

1. In ServiceNow, configure your business service structure or map so it is ready to by synced to SL1:
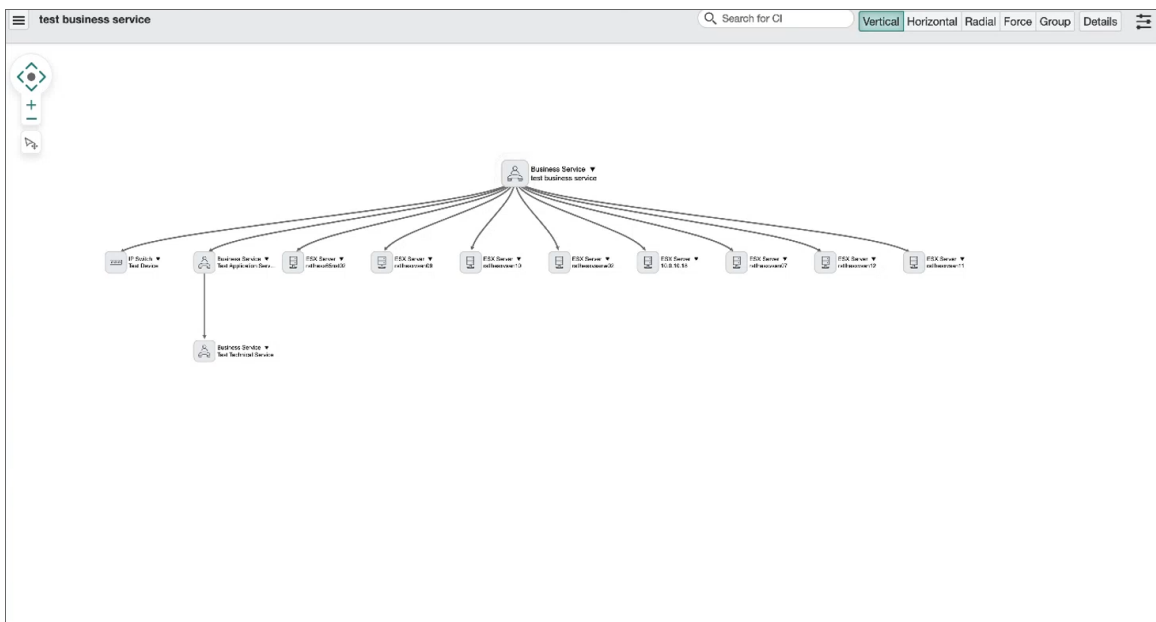
2. For each service that you want to sync with SL1:

   - Ensure that the *SL1 Region* field in ServiceNow matches the **region** field in PowerFlow.

   - Set the **SL1 Monitored** flag to *True*.

3. Go to the **Applications** page of the PowerFlow user interface and run the "Cache ServiceNow Companies, CIs and SL1 Orgs, Device Classes" application. This application reads all existing SL1 Device Classes, Organizations, ServiceNow CIs, and Companies and writes them to a cache.

4. When that application completes, go to the **Applications** page and run the "Sync Devices from SL1 to ServiceNow" application to enable PowerFlow to use the mappings and additional attribute options from Device Sync.

5. When that application completes, go to the **Applications** page and select the "Sync Business Services from ServiceNow to SL1" application.

6. Click **[Configure]** (⚙) to open the **Configuration** pane:

7. Complete the following fields, as needed:

    - **Configuration**. Select the relevant configuration object to align with this application. You cannot edit fields that are populated by the configuration object. Required.

    > **NOTE**: The *region* field is populated by the configuration object you aligned with this application. The region value must match the value in the *SL1 Region* field in ServiceNow. If you need to update this value, you will need to define the *region* variable in the configuration object that is aligned with this application, or align a different configuration object that has the correct *region* value.

    - **read_timeout**. Specify the maximum amount of time in seconds the application should wait for a response before timing out. The default is 20 seconds.

    - **verify_snow_ssl**. Toggle on (blue) this option to enable verification of the SSL certification when you run this application.

    - **attribute_map**. In this section, click **[Add Mapping]** if you want to create a mapping for any other asset fields that you want to sync between SL1 (the first column) and ServiceNow (the second column).

8. In the **Company Mapping Override** section, you can create a new mapping for the ServiceNow *company* field. You can override the *company* field with a different ServiceNow field where you can read and write the **company_sys_id** for CIs.

9. Click **[Save]**. The **Configuration** pane automatically closes.

10. Click **[Run]** to run the application. When the application completes, the ServiceNow business services will be available on the **Business Services** page of SL1:
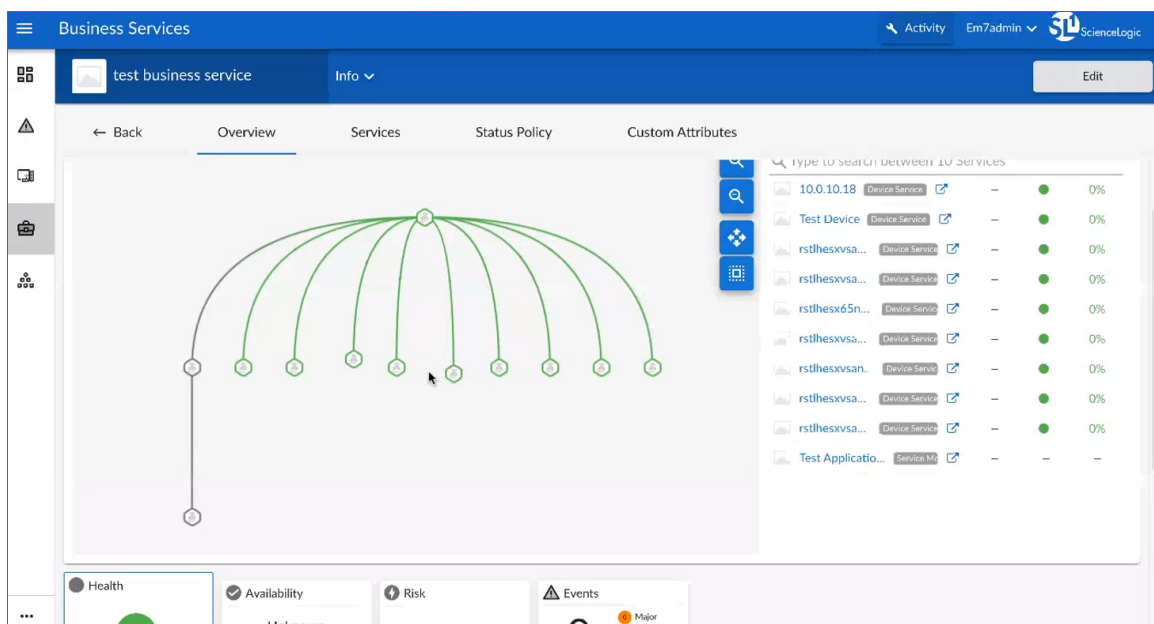
Syncing CI Attributes from ServiceNow to SL1

The "Sync CI Attributes from ServiceNow to SL1" application imports CI attributes from ServiceNow to the relevant asset and attribute fields in SL1. The CI Sync supports assets, asset configuration, asset maintenance, location, production statuses, and custom attributes.

The "Sync CI Attributes from ServiceNow to SL1" application can sync the display value and **sys_id** of **Reference** fields, such as location, as well as the value and label of **Choice List** fields, such as operational_status. These values can be accessed by appending **_label** to the desired field name.

Reference Example:

```
"location": "240f6630db993300dc44f00fbf96196f"
```

```
"location_label": "Corporate Headquarters"
```

Choice List Example:

```
"operational_status": "1",
```

```
"operational_status_label": "Operational",
```

The following image shows the **Location** table, and the **Display** column shows the **Name** marked as *true*. Only one field on the table can be marked as *true*, and that is the field that will be returned to PowerFlow :

> **NOTE**: When this application runs, if no mappings are provided, PowerFlow queries the "Sync Devices from SL1 to ServiceNow" application and uses the mappings from that application.

To sync CI attributes from ServiceNow to SL1:

1. Go to the **Applications** page of the PowerFlow user interface and run the "Cache ServiceNow Companies, CIs and SL1 Orgs, Device Classes" application. This application reads all existing SL1 Device Classes, Organizations, ServiceNow CIs, and Companies and writes them to a cache.

2. When that application completes, go to the **Applications** page and run the "Sync Devices from SL1 to ServiceNow" application to enable PowerFlow to use the mappings and additional attribute options from Device Sync.

3. When that application completes, go to the **Applications** page and run the "Sync CI Attributes from ServiceNow to SL1" application.

4. Click **[Configure]**. The **Configuration** pane appears:

5. Complete the following fields, as needed:

- **Configuration**. Select the configuration object with the relevant SL1 and ServiceNow credentials to align with this application. You cannot edit fields that are populated by the configuration object. Required.

> NOTE: The **region** field (along with other fields related to user names and passwords) is populated by the configuration object you aligned with this application. The region value must match the value in the **SL1 Region** field in ServiceNow. If you need to update this value, you will need to define the **region** variable in the configuration object that is aligned with this application, or align a different configuration object that has the correct **region** value.

- **read_timeout**. Specify the maximum amount of time in seconds that the application should wait for a response before timing out. The default is 20 seconds

- **chunk_size**. Specify the number of devices to include in each chunk sent to ServiceNow when you run this application. The default chunk size is 500 devices.

- **Include_Orgs**. If you want to include SL1 Organizations in the sync, add the Organization IDs from SL1 in this field, separated by commas. Leave this field empty to sync all SL1 Organizations. This option filters on the ServiceNow CI sync as well as the SL1 Device sync.

- **Include_CUGs**. If you want to include SL1 Collector Groups (CUGs) in the sync, add the Collector Group IDs from SL1 in this field, separated by commas. Leave this field empty to sync all SL1 Collector Groups.

- **snow_request_limit**. Specify the number of CIs fetched from ServiceNow in each request. The default is 500 objects per chunk.

- **retry_max**. The maximum number of times PowerFlow will retry to execute the step before it stops retrying and logs a step failure. For example, if retry_max is 3, PowerFlow will retry after 1 second, then 2 seconds, then 4 seconds, and stop if the last retry fails. The default is 0.

- **retry_backoff_max**. The maximum time interval for the **retry_backoff** option, in seconds. For example, if you have **retry_max** set to 15, the delays will be 1, 2, 4, 8, 16, 32, 64, 120, 240, 480, 600, 600, 600, 600, and 600. The default is 600.

- **verify_snow_ssl**. Toggle on (blue) this option to enable verification of the SSL certification when you run this application.

- **sync_empty_fields**. Select this option if you want to include empty fields on ServiceNow CIs when you run this application. This option is disabled by default.

> NOTE: SL1 does not return extended custom attributes for devices if no value is present. As a result, if the value for the custom attribute is empty in ServiceNow, PowerFlow will continually send updates for those devices, because the empty value written to is not returned in subsequent runs of the application. When the sync_empty_fields option is selected, the application sends a 0 or ", depending on the attribute type for the custom attributes in the mappings for all devices.

- *snow_attributes_created*. Select this option to show that custom attributes for ServiceNow **sys_id** and **ci_class** already existing in SL1.

- *retry_jitter*. When selected, instead of using a defined interval between retries, PowerFlow will retry the step execution at random intervals. The default is unselected.

- *retry_backoff*. When selected, instead of using a defined interval between retries, PowerFlow will incrementally increase the interval between retries. The default is unselected.

- *gql_filter*. Specify a custom GraphQL filter to apply to the devices in the sync. An example of a Device Search GraphQL query is: `{"name": {"doesNotBeginWith": "jc-is-ma"}}`

6. In the **Mappings** section, click the **[Edit]** button (PowerFlow 2.7.0 or later) or the expand button (∨) to view and edit the mappings between ServiceNow CI classes and SL1 device classes. You can map a single ServiceNow CI class with multiple SL1 device classes.

   If no mappings are provided in this section, the application will only pull CI classes provided in the "Sync Devices from SL1 to ServiceNow" application. If you add mappings in this section, only CI classes that were included in this section will be returned from ServiceNow.

   For more information about how to view, edit, and create mappings, see *Editing Mappings in a PowerFlow Application*. This topic also covers how to use the **Company Mapping Override** and the **Attribute Mappings** sections, below.

7. In the **Company Mapping Override** section, you can create a new mapping for the ServiceNow *company* field. You can override the *company* field with a different ServiceNow field where you can read and write the **company_sys_id** for CIs.

8. In the **Attribute Mappings** section, click the **[Edit]** button (PowerFlow 2.7.0 or later) or the expand button (∨) to view and edit mappings for any other custom attributes you want to sync between SL1 (the first column) and ServiceNow (the second column). For more information, including lists of default and available device attribute mappings, see *Device Attribute Mappings*.

   To sync a user-defined field on the ServiceNow CI record to the device notes table in SL1, create a custom attribute mapping for that field in this section. This option works best if you only want a single value synced over so that value can remain in the notes.

   > **TIP:** For the **Attribute Mappings** section, you can use a Jinja2 Template for device attribute fields on the SL1 side (the left column). For more information, see *Using a Jinja2 Template*.

   > **NOTE:** When an attribute value is "0" in SL1, the corresponding field in ServiceNow might display as empty.

9. Click **[Save]**. The **Configuration** pane automatically closes.

10. Click **[Run]** to run the "Sync CI Attributes from ServiceNow to SL1" application.

Creating a Link in an SL1 Device to a ServiceNow CI

To help you mange your SL1 devices and their corresponding ServiceNow CIs, you can create a link in an SL1 Device that goes directly to the synced CI in ServiceNow.

The "Create Custom Attributes and ServiceNow Custom Link in SL1" PowerFlow application creates **servicenow_sys_id** and **servicenow_ci_class** custom device attributes in SL1, and then SL1 uses those two attributes to create a custom link in SL1 titled *ServiceNow* that redirects to the ServiceNow CI.

In SL1, the link appears as a custom link on the **Tools** menu for the corresponding device:
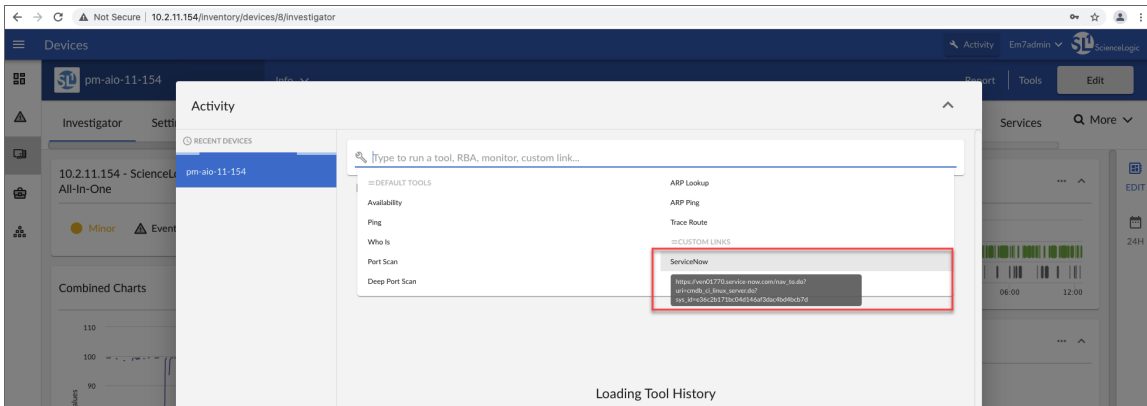
NOTE: The application only creates the custom attributes if the attributes do not already exist. Additionally, if the custom link functionality is disabled by default on SL1, the application will enable the feature before creating the necessary attributes and link.

To create a link for the SL1 device:

1. In the PowerFlow user interface, go to the **Applications** page and open the "Sync CI Attributes from ServiceNow to SL1 to ServiceNow" application.

2. Click the **[Configure]** button. The **Configuration** pane appears.

3. Enable the *snow_attributes_created* toggle and click **[Save]**.

4. Click **[Run]** to run the application.

5. On the **Applications** page, open the "Create Custom Attributes and ServiceNow Custom Link in SL1" PowerFlow application, align a configuration object if needed, and click **[Run]** to run the application.

NOTE: You only need to run the "Create Custom Attributes and ServiceNow Custom Link in SL1" application once.

Adding Related Items and Lists to the CI Record in ServiceNow

In ServiceNow you can add Related Items and Related Lists to CI records so you can see all related records in ServiceNow. The following image shows the **Related Items** pane on a CI record for a VMware Virtual Machine Instance:
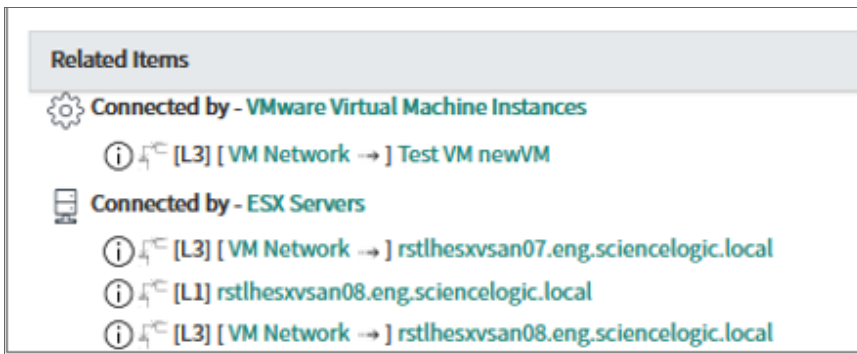
The following image shows how to add Related Items for a VMware Virtual Machine instance by selecting the *CI relations* option.
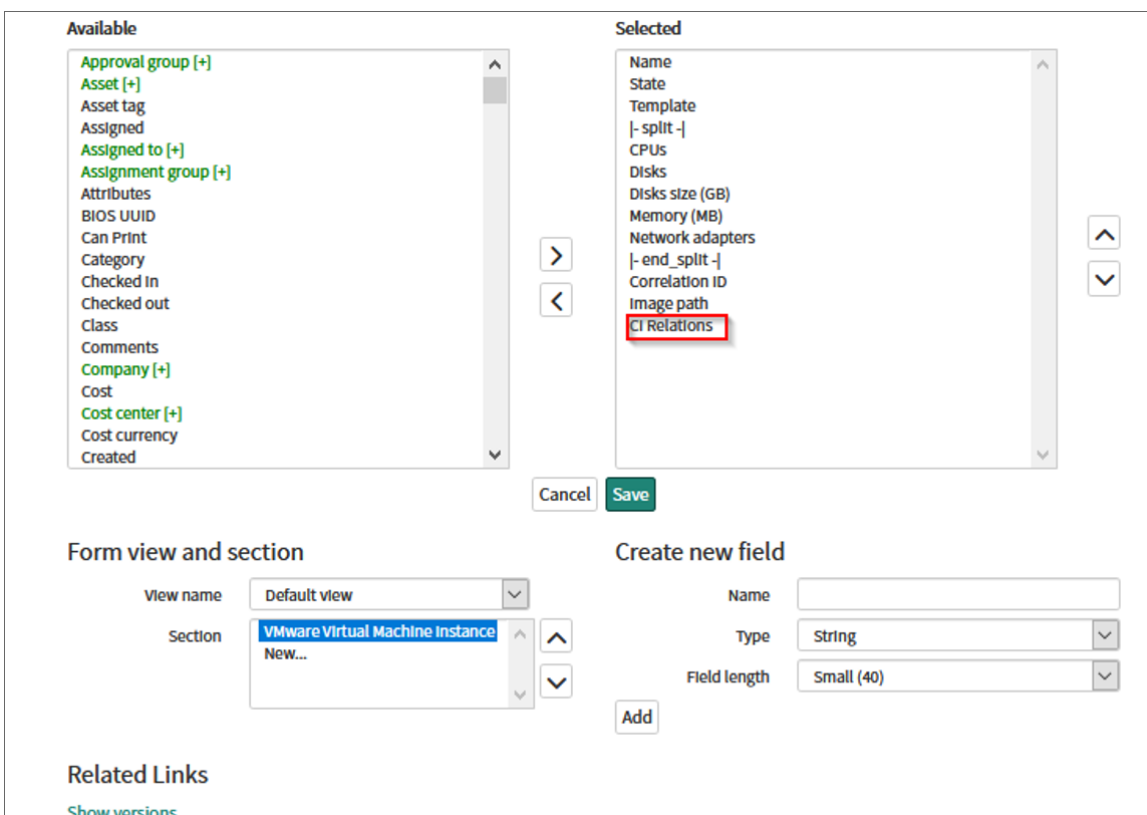
The following image shows the related lists for network adapters and files systems:
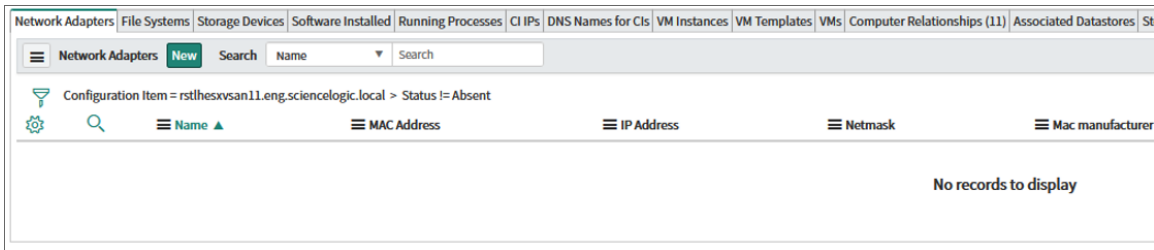
The items highlighted by the red boxes below are the most commonly used items:
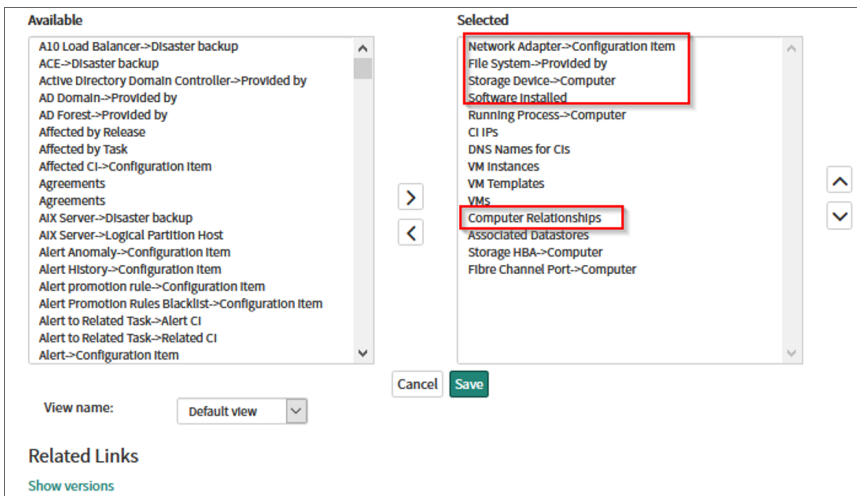
For more information, see the ServiceNow Documentation at https://docs.servicenow.com/bundle/paris-platform-user-interface/page/use/using-forms/concept/c_RelatedLists.html.

Discovery Sync

The Discovery Sync integration lets you use SL1 for discovering and syncing ServiceNow devices. With Discovery Sync, you start an SL1 discovery session from ServiceNow and then sync the newly discovered SL1 devices or virtual devices and their data with ServiceNow.

Before running a Discovery Sync session, you must complete the following steps first:

1. For domain-separated ServiceNow instances, perform a company sync by running the "Sync Organizations from SL1 to ServiceNow" application in the PowerFlow user interface. For more information, see *Syncing Organizations from SL1 to ServiceNow*.

2. In ServiceNow, configure a service request for Discovery Sync. For more information, see *Configuring a ServiceNow Service Request for Discovery Sync*.

3. In the PowerFlow user interface, run the applications listed in the *Discovery Sync Workflow*.

Configuring a ServiceNow Service Request for Discovery Sync

Before you can run a Discovery Sync, you need to configure the catalog and category values in the ServiceNow service request forms. You also need to activate the "Device Discovery" service request in ServiceNow.

> **NOTE**: Because some of the fields in the service request form will only populate if you have completed the previous fields in the form, you need to complete the fields in the service request form in sequential order.

To configure the ServiceNow service requests for Discovery Sync:

1. In ServiceNow, go to the **Catalog Items** page (Service Catalog > Catalog Definitions > Maintain Items).

2. Type "ScienceLogic" in the *Category* column. The **Device Discovery** and **Monitoring Removal** service requests appear in the search results.

3. Open the **Device Discovery** service request and ensure that the *Catalogs* and *Category* fields are accurate. For example:

> **NOTE**: Do not set the *Category* to a *Change Request*.

4. If you need to update these fields, click the **[Edit in Catalog Builder]** button at the top of the detail page.

5. Update the fields and click the **[Update]** button to save your changes.

6. From the **Catalog Items** page, click the check box for the **Device Discovery** service request and click the **[Activate]** button at the bottom of the **Catalog Items** window.

> **NOTE**: This service request is instance-specific, which means that the service request will appear in the same location as the catalogs you specified for that request. In the example, above, the **Catalog** was set to *Service Catalog*.

7. Navigate to the relevant catalog for the service request. For example, if you selected *Service Catalog* for one or both requests, then type "Service Catalog" in the filter navigator, or select **Self-Service > Service Catalog** to view the new service requests. Type "device discovery" in the *Search catalog* field to quickly locate the request.

8. Run the applications listed in the *Discovery Sync Workflow* before creating the Device Discovery service request in ServiceNow.

Discovery Sync Workflow

To prepare SL1 and ServiceNow for a Discovery Sync, run the following applications in the PowerFlow user interface, in the following order:

1. *Sync Discovery Requirements*. This application exports information from SL1 to populate the information in the ServiceNow request form. You must run this application before you can create the discovery sync session in ServiceNow. This application uses one or more of the following options from the **Configuration** pane:

   - *Configuration*. Select the relevant configuration object to align with this application. You cannot edit fields that are populated by the configuration object. Required.

   > NOTE: The *region* field is populated by the configuration object you aligned with this application. The region value must match the value in the *SL1 Region* field in ServiceNow. If you need to update this value, you will need to define the *region* variable in the configuration object that is aligned with this application, or align a different configuration object that has the correct *region* value.

   - *read_timeout*. Specify the maximum amount of time in seconds the application should wait for a response before timing out. The default is 30 seconds.
   - *sl1_chunk_size*. Specify the number of object to include in each chunk sent from SL1 to ServiceNow. The default is 500.
   - *snow_chunk_size*. Specify the number of objects to send in each chunk from ServiceNow. The default is 150 objects per chunk.
   - *Create_Missing*. Select this option if you want PowerFlow to create a new device or CI if that record is missing, based on your selection in the *Source_of_Truth* field.
   - *Update_Name*. This option addresses the situation where PowerFlow finds a match with a device or CI, but the names do not match. This option updates a device or CI name based on your selection in the *Source_of_Truth* field, below. For example, if you selected *ScienceLogic* as the source of truth, PowerFlow uses the device name from ScienceLogic as the updated name.
   - *Sync_Empty_Groups*. Select this option if you want to sync device groups that have no devices, or device groups that have devices but no matching CIs.
   - *Source_of_Truth*. Select whether you want to use data from ServiceNow or ScienceLogic as the "source of truth" when this application encounters duplicate data or data collisions.
   - In the **Company Mapping Override** section, you can create a new mapping for the ServiceNow *company* field. You can override the *company* field with a different ServiceNow field where you can read and write the **company_sys_id** for CIs.

2. *Sync Service Requests from ServiceNow to SL1*. This application sends the request forms to SL1. This application was called "Sync Discovery Session Requests from ServiceNow to SL1" in previous versions of the SyncPack. This application uses one or more of the following options from the **Configuration** pane:

   - *Configuration*. Select the relevant configuration object to align with this application. You cannot edit fields that are populated by the configuration object. Required.

- **read_timeout**. Specify the maximum amount of time in seconds the application should wait for a response before timing out. The default is 20 seconds.
- **Open_State**. The State ID from ServiceNow that specifies which Requested Items (RITMs) to pull and process. The default is 1.
- **Closed_Success_State**. The State ID for a successfully created virtual device. The State ID for a successful run changes from 1 to 2 and then ends with 4. The default is 3.
- **Closed_Failed_State**. The State ID for failed discoveries or failed virtual device creation, usually caused by invalid payloads. The State ID for a failed run changes from 1 to 2 and then ends with 4. The default is 4.
- **In_Progress_State**. The State ID for RITMs for a running discovery. The default is 2.
- **target_vcug**. Leave this field blank.
- **recursively_disable_children**. Leave this field blank.

3. **Sync Discovery Session Status from SL1 to ServiceNow**. This application populates the discovery session logs back to ServiceNow. This application uses the following options from the **Configuration** pane:

- **Configuration**. Select the relevant configuration object to align with this application. You cannot edit fields that are populated by the configuration object. Required.
- **read_timeout**. Specify the maximum amount of time in seconds the application should wait for a response before timing out. The default is 20 seconds.
- **retry_max**. The maximum number of times PowerFlow will retry to execute the step before it stops retrying and logs a step failure. For example, if retry_max is 3, PowerFlow will retry after 1 second, then 2 seconds, then 4 seconds, and stop if the last retry fails. The default is 0.
- **retry_backoff_max**.The maximum time interval for the **retry_backoff** option, in seconds. For example, if you have **retry_max** set to 15, the delays will be 1, 2, 4, 8, 16, 32, 64, 120, 240, 480, 600, 600, 600, 600, and 600. The default is 600.
- **Closed_Success_State**. The State ID for a successfully created discovery. The State ID for a successful run changes from 1 to 2 and then ends with 4. The default is 3.
- **Closed_Failed_State**. The State ID for failed discoveries, usually caused by invalid payloads. The State ID for a failed run changes from 1 to 2 and then ends with 4. The default is 4.
- **sys_id_target**. Takes the **sys_id** value from the CI in the ServiceNow Service Request and populates it in the relevant field in SL1, such as **c-sys_id**.
- **ci_class_target**. Takes the **ci_class** value from the CI in the ServiceNow Service Request and populates it in the relevant field in SL1, such as **c-ci_class_target**.

> NOTE: If the **sys_id_target** field and the **ci_class_target** field are not populated, PowerFlow will skip the process of consuming cached data and populating custom attribute fields in SL1 with the **sys_id** and **ci_class** values of newly discovered devices.

- **snow_attributes_created**. Select this option if custom attributes for ServiceNow **sys_id** and **ci_class** already exist in SL1.

- *retry_jitter*. When selected, instead of using a defined interval between retries, the PowerFlow system will retry the step execution at random intervals. By default, this option is not selected.

- *retry_backoff*. When selected, instead of using a defined interval between retries, PowerFlow will incrementally increase the interval between retries. By default, this option is not selected.

- *verify_snow_ssl*. Toggle on (blue) this option to enable verification of the SSL certification when you run this application.

4. *Sync Discovery Templates from SL1 to ServiceNow*. This application creates Service Catalog templates in ServiceNow based on Discovery Sessions that were created in SL1. This option lets you use any existing SL1 Discovery Sessions as a template for discovering or monitoring a CI with SL1. This application uses the following options from the **Configuration** pane:

- *Configuration*. Select the relevant configuration object to align with this application. You cannot edit fields that are populated by the configuration object. Required.

- *chunk_size*. Specify the number of devices to include in each chunk sent to ServiceNow when you run this application. The default is 0.

- *template_prefix*. Specify the prefix string that PowerFlow will search for in SL1. Any Discovery Sessions that contain that string will be used in ServiceNow to create a service catalog template. The default string is *ServiceNow Template:*, but you can configure this as needed. In SL1, go to the **Discovery Sessions** page (Devices > Discovery Sessions) and search for the discovery session or sessions that you want to use as a template. The start of the name in the **Name** field should match the value in the *template_prefix* field, above.

- *read_timeout*. Specify the maximum amount of time in seconds the application should wait for a response before timing out. The default is 30 seconds.

5. *Sync Devices from SL1 to ServiceNow*. Running this application ensures that the devices discovered by SL1 get synced to ServiceNow.

6. When the applications finish running, PowerFlow sends the status of those applications to ServiceNow, and you can *run a Discovery Sync in ServiceNow*.

Running a Discovery Sync in ServiceNow

The Discovery Sync process starts an SL1 discovery session from ServiceNow and syncs the newly discovered SL1 devices and their data with ServiceNow. You can choose to discover standard devices or virtual devices.

To run a Discovery Sync from the Service Catalog page:

1. In ServiceNow, go to the **Service Catalog** page (Self-Service > Service Catalog).

2. Type "device discovery" in the *Search catalog* field at the top right, and press **[Enter]**. The **Device Discovery** catalog entry appears.

> NOTE: Previous versions of the "ScienceLogic SL1: CMDB & Incident Automation Application" (also called the Certified or Scoped Application) created *two* separate service requests: **Create Virtual Device** and **Device Discovery**. Both features have been combined into the **Device Discovery** service request.

3. Click **Device Discovery**. The **Device Discovery** service request appears.

4. In the *What company is this for?* field, specify the company. The *Region* field updates automatically based on the company you select.

5. In the *Request Type* field, select *Discover Device(s)* or *Create Virtual Device*, depending on the type of device you want to discover.

   ○ If you selected *Discover Device(s)*, go to step 6.

   ○ If you selected *Create Virtual Device*, go to step 7.

6. If you selected *Discover Device(s)* in the *Request Type* field, complete the following fields:

   - *Log All*. Select this option if you want the discovery session to use verbose logging. When you select this option, SL1 logs details about each IP address or hostname specified in the *IP Address/Hostname Discovery List* field, even if the results are "No device found at this address."

   - *Select Template*. To use a template that contains your device discovery information, select the template from the drop-down.

> **TIP:** You can save the current device discovery as a template by checking *Save as Template*. A template saves all of the discovery settings except for the IP addresses. You can access existing templates on the **Catalog Template** page in ServiceNow (ScienceLogic > Automations > Catalog Templates).

   - *IP Address/Hostname Discovery List*. Provide a list of IP addresses, hostnames, or fully-qualified domain names for SL1 to scan during discovery:

     - One or more *single IPv4 addresses* separated by commas and a new line. Each IP address must be in standard IP notation and cannot exceed 15 characters. For example, "10.20.30.1, 10.20.30.2, 10.20."

     - One or more *ranges of IPv4 addresses* with "-" (dash) characters between the beginning of the range and the end of the range. Separate each range with a comma. For example, "10.20.30.1 – 10.20.30.254".

     - One or more IP address ranges in *IPv4 CIDR notation*. Separate each item in the list with a comma. For example, "192.168.168.0/24".

     - One or more hostnames (fully-qualified domain names). Separate each item in the list with a comma.

   - *Credentials*. Select one or more SNMP credentials to allow SL1 to access a device's SNMP data.

   - *Discover Non-SNMP*. Specifies whether or not SL1 should discover devices that don't respond to SNMP requests.

   - *Model Devices*. Determines whether or not the devices that are discovered with this discovery session can be managed through SL1.

   - *DHCP*. Specifies whether or not the specified range of IPs and hostnames use DHCP. If you select this option, SL1 performs a DNS lookup for the device during discovery and each time SL1 retrieves information from the device.

- **Device Model Cache TTL (h)**. Amount of time, in hours, that SL1 stores information about devices that are discovered but not modeled, either because the **Model Devices** option is not enabled or because SL1 cannot determine whether a duplicate device already exists. The cached data can be used to manually model the device from the **Discovery Session** window.

- **Collection Server**. Select an existing collector to monitor the discovered devices. Required.

- **What company is this for?**. Specify the company that will use this discovery data. Click the magnifying glass icon to locate a company.

- **Add Devices to Device Groups**. Select one or more existing device groups to which you want to add the discovered devices.

- **Apply Device Template** . Select an existing device template if needed. As SL1 discovers a device in the IP discovery list, that device is configured with the selected device template.

- **Initial Scan Level**. For this discovery session only, specifies the data to be gathered during the initial discovery session.

- **Scan Throttle**. Specifies the amount of time a discovery process should pause between each specified IP address (specified in the **IP Address/Hostname Discovery List** field). Pausing discovery processes between IP addresses spreads the amount of network traffic generated by discovery over a longer period of time.

- **Scan Default Ports**. Select this option to scan the default ports: 21,22,23,25,80. If you de-select this option, you can specify a different list of ports in the **Custom Port Scan** field that appears.

- **Port Scan All IPs**. For the initial discovery session only, specifies whether SL1 should scan all IP addresses on a device for open ports.

- **Port Scan Timeout**. For the initial discovery session only, specifies the length of time, in milliseconds, after which SL1 should stop trying to scan an IP address for open ports and begin scanning the next IP address (if applicable).

- **Interface Inventory Timeout (ms)**. Specifies the maximum amount of time that the discovery processes will spend polling a device for the list of interfaces. After the specified time, SL1 will stop polling the device, will not model the device, and will continue with discovery. The default value is 600,000 ms (10 minutes).

- **Maximum Allowed Interfaces**. Specifies the maximum number of interfaces per devices. If a device exceeds this number of interfaces, SL1 stops scanning the device, will not model the device, and will continue with discovery. The default value is 10,000.

- **Bypass Interface Inventory**. Select this option if you do *not* want SL1 to attempt to discover interfaces for each device in the discovery session.

7. If you selected *Create Virtual Device* in the **Request Type** field, complete the following fields:

   ○ **Name**. Type a name for the virtual device.

   ○ **Virtual Device Class**. Specify the device class of the virtual device. Click the magnifying glass icon to locate any classes aligned with your organization.

   ○ **Collector Group**. Specify the SL1 collector group to use for the Discovery Sync. Click the magnifying glass icon to locate any collector groups aligned with your organization.

8. Click **[Order Now]**. On the **Order Status** page that appears, make a note of value in the **Request Number** field.

9. In the PowerFlow user interface, go to the **Applications** page and run the "Sync Service Requests from ServiceNow to SL1" application.

10. When the application completes, go to **Self-Service > My Requests** in ServiceNow.

11. Click the **RITM** record link to go to the **Requested Item** page. The **State** field should update to *Closed Complete* and the request should be assigned to itself.

12. In the PowerFlow user interface, go to the **Applications** page and run the "Sync Devices from SL1 to ServiceNow" application to make sure that the device or devices were discovered.

13. For a standard device discovery, go to ServiceNow and perform the following:

    • Scroll down to the **Activities** pane to verify that you have a comment stating the discovery completed.

    • In SL1, navigate to the **Discovery Control Panel** page (Registry > Manage > Discovery) and verify that SL1 created a new discovery session with that ID.

14. For a virtual device discovery, go to ServiceNow and perform the following:

    • Scroll down to the **Activities** pane to verify that you have a comment stating "Virtual Device <name> Created with SLID: <new id>":

    • In SL1, navigate to the **Device Manager** page (Registry > Device Manager) and verify that SL1 created a new device with that device ID.

Discovering One or More Devices from ServiceNow to SL1

If you want to quickly select one or more CIs in ServiceNow for monitoring in SL1, you can use the *Monitor Device List* option from the **Configuration Items** list view, or the *Monitor Device* option from the Configuration Item detail view.

This feature uses registered events in ServiceNow that are queued to ServiceNow Event Management to trigger actions in PowerFlow. Also, this method is just an example of one of many ways to trigger a registered event. For more information about registered events, including examples of other triggering events you can define in ServiceNow, see the *ServiceNow Registered Events* appendix.

You will need to create a discovery template for a discovery process created on the **Service Catalog** page before you can discover devices using that template on the **Configuration Items** page. A template saves all of the discovery settings except for the IP addresses. You can access existing templates on the **Catalog Template** page in ServiceNow (ScienceLogic > Automations > Catalog Templates).

To discover one or more devices from ServiceNow:

1. In ServiceNow, navigate to the **Configuration Items** page.

2. From the list view, select the CI or CIs (devices) that you want to discover.

NOTE: A CI in ServiceNow must be aligned with a company in ServiceNow, or the service request will be canceled. Also, that company must be associated with a ScienceLogic Region.

---

3. Right-click anywhere in the window and select *Monitor Device List* from the pop-up menu. A **Select Discovery Template** dialog box appears.

> **TIP:** You can also select a specific CI from the list view and click the *Monitor Device* option from the Configuration Item detail view. You will also need to use an existing template for this process.

4. Select a discovery template to use for the current discovery.

5. Click **[OK]** to use the template. ServiceNow generates a new service request for **Device Discovery** for each CI.

6. In the PowerFlow user interface, select the "Sync Service Requests from ServiceNow to SL1" application from the **Applications** page and click **[Configure]**. The **Configuration** pane appears:

7. Complete the following fields, as needed:

- *Configuration*. Select the relevant configuration object to align with this application. You cannot edit fields that are populated by the configuration object. Required.

- *read_timeout*. Specify the maximum amount of time in seconds the application should wait for a response before timing out. The default is 20 seconds.

- *Open_State*. The State ID from ServiceNow that specifies which Requested Items (RITMs) to pull and process. The default is 1.

- *Closed_Success_State*. The State ID for a successfully created virtual device. The State ID for a successful run changes from 1 to 2 and then ends with 4. The default is 3.

- *Closed_Failed_State* The State ID for failed discoveries or failed virtual device creation, usually caused by invalid payloads. The State ID for a failed run changes from 1 to 2 and then ends with 4. The default is 4.

- *In_Progress_State*. The State ID for RITMs for a running discovery. The default is 2.

- *target_vcug*. Leave this field blank.

- *verify_snow_ssl*. Toggle on (blue) this option to enable verification of the SSL certification when you run this application.

- *recursively_disable_children*. Leave this field blank.

8. Click **[Save]**. The **Configuration** pane automatically closes.

9. Click **[Run]** to run the application.

10. Go to the **Applications** page and run the "Sync Devices from SL1 to ServiceNow" application to make sure that the device or devices were discovered.

Decommissioning Devices

If you want to quickly select one or more CIs in ServiceNow for to remove from monitoring (or "decommission") in SL1, you can use the *Device Monitoring Removal list* option from the **Configuration Items** list view, or the *Monitoring Removal* option from the Configuration Item detail view.

You then use the "Sync Service Requests from ServiceNow to SL1" application to decommission the devices that you no longer want to monitor. Running this application takes the list of synced devices in the service request and moves them to an SL1 Virtual Collector Group (VCUG). The "Sync Service Requests from ServiceNow to SL1" application was formerly named "Sync Discovery Session Requests from ServiceNow to SL1".

> **WARNING:** If you move a parent device to a new VCUG, then all of its children move as well. If you move a child directly, only the child moves.

This feature uses registered events in ServiceNow that are queued to ServiceNow Event Management to trigger actions in PowerFlow. Also, this method is just an example of one of many ways to trigger a registered event. For more information about registered events, including examples of other triggering events you can define in ServiceNow, see the *ServiceNow Registered Events* appendix.

Activating the ServiceNow Service Request for Monitoring Removal

To activate the ServiceNow service request for Device Decommission:

1. In ServiceNow, go to the **Catalog Items** page (Service Catalog > Catalog Definitions > Maintain Items).

2. Type "ScienceLogic" in the *Category* column. The **Device Discovery** and **Monitoring Removal** service requests appear in the search results.

3. Open the **Monitoring Removal** service request and ensure that the *Catalogs* and *Category* fields are complete. Add the relevant information if the fields are blank. Do not set the **Category** to a *Change Request*.

4. If you need to update these fields, click the "To edit this record click *here*" link at the top of the detail page.

5. Update the fields and click the **[Update]** button to save your changes.

6. From the **Catalog Items** page, click the check box for the **Monitoring Removal** service request and click the **[Activate]** button at the bottom of the **Catalog Items** window.

7. Navigate to the relevant catalog for the service request. For example, if you selected *Service Catalog*, then type "Service Catalog" in the filter navigator, or select **Self-Service > Service Catalog** to view the new service requests.

Removing Devices from Monitoring

To decommission Configuration Items (devices) in ServiceNow that you no longer want to monitor:

1. In ServiceNow, go to the **Configuration Items** window.

2. From the list view, select the CI or CIs (devices) that you want to decommission.

> **NOTE:** A CI in ServiceNow must be aligned with a company in ServiceNow, or the service request will be canceled. Also, a company must be associated with a ScienceLogic Region.

3. Right-click anywhere on the window and select *Device Monitoring Removal list* from the pop-up menu. A dialog box appears.

4. Click **[OK]** to remove the CI or CIs from monitoring. ServiceNow generates a new service request for **Monitoring Removal** for each CI.

> **TIP:** You can also select a specific CI from the list view and click the *Monitoring Removal* option from the Configuration Item detail view.

5. In the PowerFlow user interface, select the "Sync Service Requests from ServiceNow to SL1" application from the **Applications** page and click **[Configure]** on the application detail page. The **Configuration** pane appears:

6. Complete the following fields:

   - **Configuration**. Select the relevant configuration object to align with this application. You cannot edit fields that are populated by the configuration object. Required.

   - **read_timeout**. Specify the maximum amount of time in seconds the application should wait for a response before timing out. The default is 20 seconds.

   - **recursively_disable_children**. Check this option to move all child devices of the devices you are decommissioning to the VCUG. If this option is not checked and a parent device is in the disable request, the parent device will be skipped with a warning message.

   - **target_vcug**. Specify the ID of the SL1 Virtual Collection Group (VCUG) you created to hold the devices on the **Collector Group Settings** page (System > Settings > Collector Groups). If this value is null, the application will attempt to pull the value from the **target_vcug** field in the "Delete Devices from SL1" application.

7. Click **[Save]**. The **Configuration** pane automatically closes after this message appears.

8. Click **[Run]** to run the application.

Deleting Devices

The "Delete Devices from SL1" application lets you delete devices in a specific Virtual Collector Group (VCUG) if those devices have not been modified in SL1 for a specified time, such as one day or five days. You can update this time in the **max_age** configuration value, which is described below.

To delete devices from an SL1 Virtual Collector Group:

1. In the PowerFlow user interface, run the "Sync Service Requests from ServiceNow to SL1" application to pull a list of decommissioned devices that you no longer want to monitor. For more information, see *Decommissioning Devices*.

2. On the **Applications** page, select the "Delete Devices from SL1" application and click **[Configure]** on the application detail page. The **Configuration** page appears:

3. Complete the following fields, as needed:

    *   *Configuration*. Select the relevant configuration object to align with this application. You cannot edit fields that are populated by the configuration object. Required.

    *   *read_timeout*. Specify the maximum amount of time in seconds the application should wait for a response before timing out. The default is 20 seconds.

    *   *max_age*. Specify how long (in days) that you want to keep the devices in the VCUG before deleting the devices. The default is 0 days. If this setting is 0, all devices in the VCUG will be deleted as soon as this application runs. If this setting is null, the application will fail. If all device children are in the same VCUG, the application will delete the target device and all of its children.

    *   *target_vcug*. Specify the ID of the SL1 Virtual Collection Group (VCUG) you created to hold the devices on the **Collector Group Settings** page (System > Settings > Collector Groups). Set this value to **-1** if you want this applications to use the *target_vcug* value from the "Sync Service Requests from ServiceNow to SL1" application.

    > **WARNING:** If you specify a value to *target_vcug* here, the "Delete Devices from SL1" application will use that value instead of the *target_vcug* value from the "Sync Service Requests from ServiceNow to SL1" application.

4. Click **[Save]**. The **Configuration** pane automatically closes.

5. Click **[Run]** to run the application.

Syncing File Systems from SL1 to ServiceNow

You can map and sync file systems in much the same way you sync devices between SL1 and ServiceNow. The "Sync File Systems from SL1 to ServiceNow" application reads file systems discovered in SL1 and then maps them to a parent CI record in ServiceNow.

For a file system to be synced from SL1 to ServiceNow, the class of the file system's parent device must be included in the mappings parameters for the "Sync Devices from SL1 to ServiceNow" application. File systems with parent CIs that have **className** values of **cmdb_ci_ip_switch**" or **cmdb_ci_ip_router**" will be sent to ServiceNow, but by default they will not be created in ServiceNow due to missing relationship information.

> **WARNING:** PowerFlow only syncs file systems that are aligned with devices that are already synced with ServiceNow. Before setting up file system sync, you must first *sync devices between SL1 and ServiceNow*.

To sync SL1 file systems with ServiceNow:

1. In the PowerFlow user interface, go to the **Applications** page and run the "Cache ServiceNow Companies, CIs and SL1 Orgs, Device Classes" application to read all existing SL1 Device Classes, Organizations, ServiceNow CIs, and Companies and write them to a cache.

2. When that application completes, go to the **Applications** page and select the "Sync File Systems from SL1 to ServiceNow" application.

3. Click **[Configure]** to open the **Configuration** pane:

4. Complete the following fields, as needed:

- **Configuration**. Select the relevant configuration object to align with this application. You cannot edit fields that are populated by the configuration object. Required.

> NOTE: The *region* field (along with other fields related to user names and passwords) is populated by the configuration object you aligned with this application, using the *Configuration* field. The *region* value on this pane must match the value in the *SL1 Region* field in ServiceNow. If you need to update this value, you will need to define the *region* variable in the configuration object aligned with this application, or align a different configuration object that has the correct *region* value.

- **Include_Orgs**. If you want to include a specific set of SL1 Organizations in the sync, add the Organization IDs from the SL1**Organizations** page (Registry > Accounts > Organizations) in this field, separated by commas. If this field is enabled, PowerFlow will pull only the Organizations listed in this field; PowerFlow does not pull all Organizations and then drop those not on the list. Leave this field empty to sync *all* SL1 Organizations. Optional.

- *snow_request_limit*. Specify the number of objects fetched from ServiceNow in each request. The default is 500 objects per chunk.

- *read_timeout*. Specify the maximum amount of time in seconds the application should wait for a response before timing out. The default is 30 seconds.

- *snow_batch_size*. Specify the total number of objects being sent to ServiceNow in each triggered application. The default is 150 objects per batch.

- *discovery_source*. Specify the ServiceNow Discovery source. The default is "Other Automated".

- *snow_chunk_size*. Specify the number of objects to send in each chunk or in each sub-payload with a batch (specifed in the *snow_batch_size* field). The default is 150 objects per chunk or sub-payload.

- *retry_max*. The maximum number of times PowerFlow will retry to execute the step before it stops retrying and logs a step failure. For example, if retry_max is 3, PowerFlow will retry after 1 second, then 2 seconds, then 4 seconds, and stop if the last retry fails. The default is 0.

- *retry_backoff_max*.The maximum time interval for the *retry_backoff* option, in seconds. For example, if you have *retry_max* set to 15, the delays will be 1, 2, 4, 8, 16, 32, 64, 120, 240, 480, 600, 600, 600, 600, and 600. The default is 600.

- *include_hidden*. Select this option to sync hidden file systems. This filter is only applied on the SL1 side of the sync. By default, this option is not selected.

- *include_stale*. Select this option to sync stale file systems. This filter is only applied on the SL1 side of the sync. By default, this option is not selected.

> **WARNING:** Be cautious when applying these *include* options, as enabling one or both options might cause unwanted disconnects if the File System Sync is run with these options enabled and then run with the options disabled.

- *sync_unsupported_file_systems*. Toggling this on will allow file systems that have a parent device class of **Network.Switch** or **Network.Router** to sync when the application is run.

- *Domain_Separation*. Select this option if your ServiceNow environment is *domain-separated*, where the data, processes, and administrative tasks have been organized into logical groupings called *domains*. If your ServiceNow instance is domain-separated, the user listed in the *snow_user* field must be a member of the top domain and have access to *all* of the domains you intend to integrate. Also, ServiceNow should be the "source of truth" for organizations if your environment is domain-separated.

- *change_fs_organizations*. Toggle on (blue) this option to allow file systems to change companies with an Interface Sync if the companies are within the same domain in ServiceNow. If you change the Organization/Company and another field, but do not enable this parameter, PowerFlow will not apply any changes to the file system. By default, this option is not selected.

- **drop_company**. Toggle on (blue) this option to remove the sys_id in existing companies from the sync. This will prevent the "Company" field from sending to ServiceNow. This option has no effect if the **domain_separation** parameter is enabled for this application. By default, this option is not selected.

> NOTE: If you select this option, the ServiceNow company **sys_id** for the file system and the SL1 organization **crm_id** for the SL1 device will not be compared when determining if an update has occurred. Do not select this option if you selected the **Domain_Separation** option, as ServiceNow requires PowerFlow to send the "company" field in domain-separated environments. Also, do not select this option if you selected the **change_interface_organizations** option, as changing the company for a file system in ServiceNow requires the "company" field to be sent.

- **verify_snow_ssl**. Toggle on (blue) this option to enable verification of the SSL certification when you run this application.

- **sl1_org_filter_only**. Toggle on (blue) this option to apply the organization filter to only the SL1 side of the data pull.

> CAUTION: Enabling this toggle will only apply organization filtering to the SL1 side of the data pulls, while all file systems for the provided region in ServiceNow will be returned. Be cautious using this option, as enabling it may cause unwanted disconnects in ServiceNow.

- **generate_report**. Select this option to create a "Sync File Systems from SL1 to ServiceNow" report. This report contains counts of the number of file systems gathered from SL1 and ServiceNow, the number of unchanged file systems, and the number of creations, updates, and disconnects sent to ServiceNow. Detailed file system information for the file systems sent to ServiceNow is also displayed. Finally, the report contains file systems that are in an unsupported format. The reports are available on the **Reports** page of the PowerFlow user interface.

> NOTE: Due to the differences between how SL1 and ServiceNow discover file systems, only certain formats of file systems contained in the SL1 database will be synced to ServiceNow. Only file systems where the name and the mount point can be determined from the name database field stored in the SL1 will be synced.

- **Simulation_Mode**. Select this option if you want to perform a simulated run of this application to show you the potential results of that run.

- **retry_jitter**. When selected, instead of using a defined interval between retries, the PowerFlow system will retry the step execution at random intervals. By default, this option is not selected.

- **retry_backoff**. When selected, instead of using a defined interval between retries, PowerFlow will incrementally increase the interval between retries. By default, this option is not selected.

5. In the **Company Mapping Override** section, you can create a new mapping for the ServiceNow *company* field. You can override the *company* field with a different ServiceNow field where you can read and write the **company_sys_id** for CIs.

6. In the **Attribute Mappings** section, click the **[Edit]** button (PowerFlow 2.7.0 or later) or the expand button (∨) to view and edit the mappings for any other custom device attributes you want to sync between SL1 (the first column) and ServiceNow (the second column).

> **NOTE:** When an attribute value is "0" in SL1, the corresponding field in ServiceNow might display as empty.

7. Click **[Save]**. The **Configuration** pane closes.

8. Click **[Run]** to run the application.

Syncing Installed Software between SL1 and ServiceNow

You can use the following applications to sync your installed software assets between and ServiceNow:

- "Sync Software Packages from SL1 to ServiceNow". Reads all software packages from SL1 and creates new CIs in ServiceNow. Run this application before running the "Sync Installed Software" application.
- "Sync Installed Software from SL1 to ServiceNow". Reads all available software packages from SL1 and the devices aligned to that software by region and syncs them with ServiceNow.
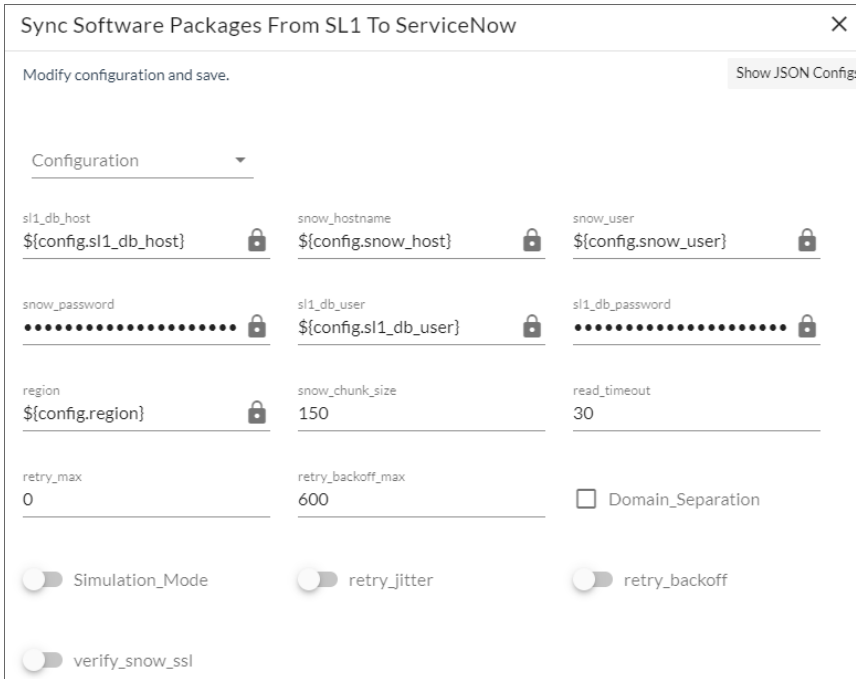
The applications do not currently support domain separation.

> **NOTE:** The Software Asset Management (SAM) application in ServiceNow is not supported with the current level of installed software data acquired with SL1. As a result, syncing installed software data with ServiceNow Discovery and other Software Asset Management software is not currently supported.

To sync installed software between SL1 and ServiceNow:

1. Make sure that you have recently run the "Sync Devices from SL1 to ServiceNow" application to populate the device cache.

2. In the PowerFlow user interface, go to the **Applications** page and select the "Sync Software Packages from SL1 to ServiceNow" application.

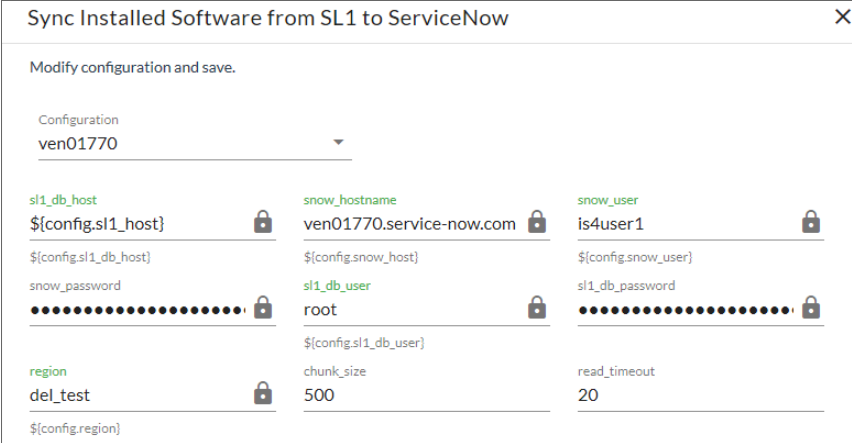3. Click **[Configure]** (⚙) to open the **Configuration** pane:

4. Complete the following fields, as needed:

    - *Configuration*. Select the relevant configuration object to align with this application. You cannot edit fields that are populated by the configuration object. Required.

    > NOTE: The *region* field is populated by the configuration object you aligned with this application. The region value must match the value in the *SL1 Region* field in ServiceNow. If you need to update this value, you will need to define the *region* variable in the configuration object that is aligned with this application, or align a different configuration object that has the correct *region* value.

    - *read_timeout*. Specify the maximum amount of time in seconds the application should wait for a response before timing out. The default is 20 seconds.

    - *snow_request_limit*. Specify the number of objects fetched from ServiceNow in each request. The default is 500 objects per chunk.

    - *snow_chunk_size*. Specify the number of objects to send in each chunk. The default is 150 objects per chunk.

    - *verify_snow_ssl*. Toggle on (blue) this option to enable verification of the SSL certification when you run this application.

5. Click **[Save]** and wait for the "App & Config modifications saved" pop-up message to appear. The **Configuration** pane automatically closes after this message appears.

6. Click **[Run]** (▶) to run the application.

7. After the "Sync Software Packages from SL1 to ServiceNow" application finishes running, go to the **Applications** page and select the "Sync Installed Software from SL1 to ServiceNow" application.

8. Click **[Configure]** (⚙) to open the **Configuration** pane:

9. Complete the following fields, as needed:

- **Configuration**. Select the relevant configuration object to align with this application. You cannot edit fields that are populated by the configuration object. Required.

- **region**. The region value is populated by the configuration object you selected. The region value must match the value in the **SL_Region** field in ServiceNow. If you need to update this value, you will need to define the **region** variable in the configuration that is aligned with this application, or align a different configuration that has the correct region value.

- **chunk_size**. Specify the number of services to include in each chunk sent to ServiceNow when you run this application. The default chunk size is 500.

- **read_timeout**. Specify the maximum amount of time in seconds the application should wait for a response before timing out. The default is 20 seconds.

10. Click **[Save]** and wait for the "App & Config modifications saved" pop-up message to appear. The **Configuration** pane automatically closes after this message appears.

11. Click **[Run]** (▶) to run the application.

Syncing Interfaces from SL1 to ServiceNow

You can map and sync network interfaces in much the same way you sync devices between SL1 and ServiceNow. The "Sync Interfaces from SL1 to ServiceNow" application collects interface data from ServiceNow and SL1 and runs multiple CI syncs for each interface to be synced.

Version 3.5.0 and later of this SyncPack requires SL1 version 11.2.0 or later.

IMPORTANT: The speed of ServiceNow processing is reduced by ServiceNow errors generated when consuming payloads. If you experience slow processing or "maximum execution time exceeded" messages, you must review and address any ServiceNow errors reported to resolve. If there are significant ServiceNow errors in Device Sync, those errors will also impact Interface sync processing. When running dependent syncs at large scale, such as Interface Sync and Device Sync, ScienceLogic recommends that you run them serially, not at the same time. Running both syncs at the same time will greatly hinder ServiceNow processing and scalability.

WARNING: PowerFlow only syncs network interfaces that are aligned with devices that are already synced with ServiceNow. Before setting up network interface sync, you must first *sync devices between SL1 and ServiceNow*.

To sync SL1 network interfaces with ServiceNow:

1. In the PowerFlow user interface, go to the **Applications** page and run the "Cache ServiceNow Companies, CIs and SL1 Orgs, Device Classes" application to read all existing SL1 Device Classes, Organizations, ServiceNow CIs, and Companies and write them to a cache.

2. When that application completes, go to the **Applications** page and select the "Sync Interfaces from SL1 to ServiceNow" application.

3. Click **[Configure]**. The **Configuration** pane appears:

4. Complete the following fields, as needed:

    - *Configuration*. Select the relevant configuration object to align with this application. You cannot edit fields that are populated by the configuration object. Required.

    > NOTE: The *region* field (along with other fields related to user names and passwords) is populated by the configuration object you aligned with this application, using the *Configuration* field. The *region* value on this pane must match the value in the *SL1 Region* field in ServiceNow. If you need to update this value, you will need to define the *region* variable in the configuration object aligned with this application, or align a different configuration object that has the correct *region* value.

    - *sl1_chunk_size*. Specify the number of interfaces to pull from SL1 in each chunk. The default is 500.

    - *read_timeout*. Specify the maximum amount of time in seconds the application should wait for a response before timing out.

- **Include_Orgs**. If you want to include a specific set of SL1 Organizations in the interface sync, add the Organization IDs from the SL1**Organizations** page (Registry > Accounts > Organizations) in this field, separated by commas. If this field is enabled, PowerFlow will pull only the Organizations listed in this field; PowerFlow does not pull all Organizations and then drop those not on the list. Leave this field empty to sync *all* SL1 Organizations. Optional.

> **CAUTION:** A misconfiguration in the **Include Orgs** field field might change the monitoring flag for one or more devices. In other words, a misconfiguration in this field could switch the SL1 Monitored flag from "true" to "false", removing that interface or interfaces from the device sync.

- **snow_request_limit**. Specify the number of objects fetched from ServiceNow in each request. The default is 500 objects per chunk.
- **snow_batch_size**. Specify the total number of objects being sent to ServiceNow in each triggered application. The default is 150 objects per batch.
- **discovery_source**. Specify the ServiceNow Discovery source. The default is "Other Automated".
- **snow_chunk_size**. Specify the number of objects to send in each chunk or in each sub-payload with a batch (specifed in the **snow_batch_size** field). The default is 150 objects per chunk or sub-payload.
- **max_count_for_relationships**. Specify the maximum number of device IDs to query for sets of relationships in a single chunk from SL1.
- **retry_max**. The maximum number of times PowerFlow will retry to execute the step before it stops retrying and logs a step failure. For example, if retry_max is 3, PowerFlow will retry after 1 second, then 2 seconds, then 4 seconds, and stop if the last retry fails. The default is 0.
- **retry_backoff_max**.The maximum time interval for the **retry_backoff** option, in seconds. For example, if you have **retry_max** set to 15, the delays will be 1, 2, 4, 8, 16, 32, 64, 120, 240, 480, 600, 600, 600, 600, and 600. The default is 600.
- **enabled_only**. Select this option only sync enabled interfaces. When enabled, this filter is applied to both the SL1 and ServiceNow side pulls. By default, this option is not selected.
- **override_snow_mappings**. Toggle on to send all interfaces to the "cmdb_ci_network_adapter" table in ServiceNow, ignoring the user-entered IANA type mappings and skipping the hardcoded mappings. However, the interface will be ignored if it is missing the required fields to be correctly created and identified in ServiceNow on the cmdb_ci_network_adapter table. The required fields are "ip", "macAddress", and"hardwareDescription".

> **NOTE:** If the **override_snow_mappings toggle** is toggled off, the following IANA types have hard-coded mappings that cannot be overwritten. This is done to match ServiceNow discovery:

- "ethernetCsmacd"
  - If the parent type is "Servers", maps to the **cmdb_ci_network_adapter** table in ServiceNow
  - If the parent type is "Network.Firewall", maps to the **cmdb_ci_network_adapter** table in ServiceNow
  - If the parent type is "Network.Switches", maps to the **cmdb_ci_network_adapter** and **dscy_switchport** tables in ServiceNow
  - If the parent type is "Network.Router", maps to the **cmdb_ci_network_adapter** and **dscy_router_interface** tables in ServiceNow
  - If the parent type is "Network.Balancers", maps to the **cmdb_ci_network_adapter** and **cmdb_ci_lb_vlan** tables in ServiceNow
- "slip"
  - If the parent type is "Network.Switches", maps to the **dscy_switchport** table in ServiceNow
- "propVirtual"
  - If the parent type is "Network.Switches", maps to the **cmdb_ci_network_adapter** and **dscy_swtch_partition** tables in ServiceNow
  - If the parent type is "Network.Router", maps to the **cmdb_ci_network_adapter** and *dscy_swtch_partition* tables in ServiceNow
- "tunnel"
  - If the parent type is "Network.Router", maps to the **cmdb_ci_network_adapter** table in ServiceNow

- ***Domain_Separation***. Select this option if your ServiceNow environment is *domain-separated*, where the data, processes, and administrative tasks have been organized into logical groupings called *domains*. If your ServiceNow instance is domain-separated, the user listed in the ***snow_user*** field must be a member of the top domain and have access to *all* of the domains you intend to integrate. Also, ServiceNow should be the "source of truth" for organizations if your environment is domain-separated. By default, this option is not selected.

- ***change_interface_organizations***. Toggle on (blue) this option to allow interfaces to change companies with an Interface Sync if the companies are within the same domain in ServiceNow. If you change the Organization/Company and another field, but do not enable this parameter, PowerFlow will not apply any changes to the interface. By default, this option is not selected.

- ***drop_company***. Toggle on (blue) this option to remove the sys_id in existing companies from the sync. This will prevent the "Company" field from sending to ServiceNow. This option has no effect if the ***domain_separation*** parameter is enabled for this application. By default, this option is not selected. If you select this option, the ServiceNow company ***sys_id*** for the interface and the SL1 organization ***crm_id*** for the SL1 device will not be compared when determining if an update has occurred. Do not select this option if you selected the ***Domain_Separation*** option, as ServiceNow requires PowerFlow to send the "company" field in domain-separated environments. Also, do not select this option if you selected the ***change_interface_organizations*** option, as changing the company for an interface in ServiceNow requires the "company" field to be sent.

- *verify_snow_ssl*. Toggle on (blue) this option to enable verification of the SSL certification when you run this application.

- *generate_report*. Select this option to create a "Sync Interfaces from SL1 to ServiceNow" report. This report contains counts of the number of interfaces gathered from SL1 and ServiceNow, the number of unchanged interfaces, as well as the number of creations, updates, and disconnects sent to ServiceNow. Detailed interface information for the interfaces sent to ServiceNow is also displayed. The reports are available on the **Reports** page of the PowerFlow user interface.

- *enable_advanced_topology*. Select this option to allow the pull and process advanced topology step to be executed when the sync is run. If disabled, this step will be skipped. This toggle defaults to a value of *False*.

- *Simulation_Mode*. Select this option if you want to perform a simulated run of this application to show you the potential results of that run. This option is not selected by default.

- *retry_jitter*. When selected, instead of using a defined interval between retries, the PowerFlow system will retry the step execution at random intervals. By default, this option is not selected.

- *retry_backoff*. When selected, instead of using a defined interval between retries, PowerFlow will incrementally increase the interval between retries. By default, this option is not selected.

- *gql_filter*. Using JSON, you can optionally define a custom GraphQL filter to apply to the interfaces in the sync. To test out a GraphQL filter, go to the GraphiQL interface in SL1 by typing the URL or IP address for SL1 in a browser, add **/gql** to the end of the URL or IP address, and press **[Enter]**. Search the built in GraphQL docs for **InterfaceSearch** and determine how to set up your custom filter.

- *interface_table_relations*. Using JSON, define the relationship for any ServiceNow tables used in the *mappings* parameter, below. If a table is included in the mappings, but the relationship is not defined in the **interface_table_relations**, the Interface Sync application will fail.

- *interface_advanced_mappings*. Using JSON, define the mappings for the combination of interface IANA type and parent device type to a specific ServiceNow table. This configuration option is blank by default, and must be configured to define the combination of specific mappings for splitting relationships.

> **NOTE:** An example value for the *interface_advanced_mappings* field is as follows: `{ "ethernetCsmacd": { "Servers": [ "dscy_router_interface" ], "System.SL1": [ "cmdb_ci_network_adapter", "dscy_router_interface" ] }, "l2vlan": { "Network.Router": [ "dscy_router_interface" ] } }`. In this example, an additional split is added to the ethernetCsmacd / Servers mapping. This will cause the two server splits (even if one is thrown out) to be tagged as "duplicates", which will affect the relationships they defined. Additionally, a new parent device type mapping of System.SL1 to ethernetCsmacd is added. Also, a new IANA/parent device type mapping is added for 12vlan.

> **TIP:** The hardcoded mappings will take priority and cannot be overwritten. If accidentally added to the user-inputted mappings, it will be ignored. The tables defined in the advanced mappings must be one of the following:
>
> [

```
"cmdb_ci_network_adapter",

"dscy_switchport",

"dscy_router_interface",

"dscy_swtch_partition",

"cmdb_ci_lb_vlan",

]
```

The tables defined in the mappings have required fields (field names are SL1 interface fields) which cannot be ignored. Any interface missing these fields will be thrown out:

```
{

"cmdb_ci_network_adapter": ["ip", "macAddress",
"hardwareDescription"],

"dscy_switchport": ["index"],

"dscy_router_interface": ["macAddress", "hardwareDescription"],

"dscy_swtch_partition": ["index"],

"cmdb_ci_lb_vlan": ["hardwareDescription"],

}
```

- **interface_relation_mappings**. Using JSON, define which specific interface relationships to send to ServiceNow. This configuration option is blank by default, and must be configured to define the relationships you want to send directly.

> **NOTE:** An example value for the *interface_relation_mappings* field is as follows: `{ "cmdb_ci_network_adapter": { "relations": [ { "parent": "cmdb_ci_network_adapter", "rel_type": "Connects to::Connected by", "create_to_duplicate": False, "create_from_duplicate": False } ] } }`. In this example, the sync creates relationships between interfaces where the child interface is on the "cmdb_ci_network_adaptor" table and the parent interface is on the "cmdb_ci_network_adaptor" table, and neither is a "duplicate". The "create_to_duplicate" parameter being false means that relationships where the child interface is a duplicate will not be created. The "create_from_duplicate" parameter being false means that relationships where the parent interface is a duplicate will not be created. Determining if an interface is a "duplicate" depends on the number of destination tables in the hard-coded or user provided mappings. For example, in the hard-coded mappings, if the IANA type of the interface is "ethernetcsmacd" and the parent device type is "network.switches", the interface is sent to two tables: "dscy_switchport" and "cmdb_ci_network_adapter". Even if one of the interfaces is discarded due to missing required fields for creation, both interfaces are tagged as "duplicates".

5. In the **Mappings** section, click the **[Edit]** button (PowerFlow 2.7.0 or later) or the expand button (⌄) to view and edit the mappings between specific SL1 interface types to different tables in ServiceNow. These mappings can be used to send an IANA type to a specific ServiceNow table if its parent is not included in the hardcoded list above. The hard-coded mappings will supersede any user-input mappings.

   For more information about how to view, edit, and create mappings, see *Editing Mappings in a PowerFlow Application*. This topic also covers how to use the **Company Mapping Override** and the **Attribute Mappings** sections, below.

6. In the **Company Mapping Override** section, you can create a new mapping for the ServiceNow *company* field. You can override the *company* field with a different ServiceNow field where you can read and write the **company_sys_id** for CIs.

7. In the **Attribute Mappings** section, click the Edit button to view and edit the mappings for any other custom device attributes you want to sync between SL1 (the first column) and ServiceNow (the second column).

> **NOTE:** When an attribute value is "0" in SL1, the corresponding field in ServiceNow might display as empty.

8. Click **[Save]**. The **Configuration** pane closes.

9. Click **[Run]** to run the application.

10. When the application completes, go to ServiceNow and type "cmdb_ci_network_adapter.list". The **Network Adapters** page appears with a list of synced interfaces.

11. You can add additional columns to this page using the Update Personalized List button (⚙) to add columns like *Operational status*, which is synced from SL1. The *Operational status* value is different from the **[SL1 Monitored]** value, but PowerFlow tracks both values.

12. Select a network interface from the list tab to see more information about the interface.

Scheduling PowerFlow Applications

Using the PowerFlow user interface, you can configure PowerFlow applications to run on a schedule instead of manually running the applications. As a best practice, if you use any of these applications, ScienceLogic recommends that you schedule those applications, in the following order:
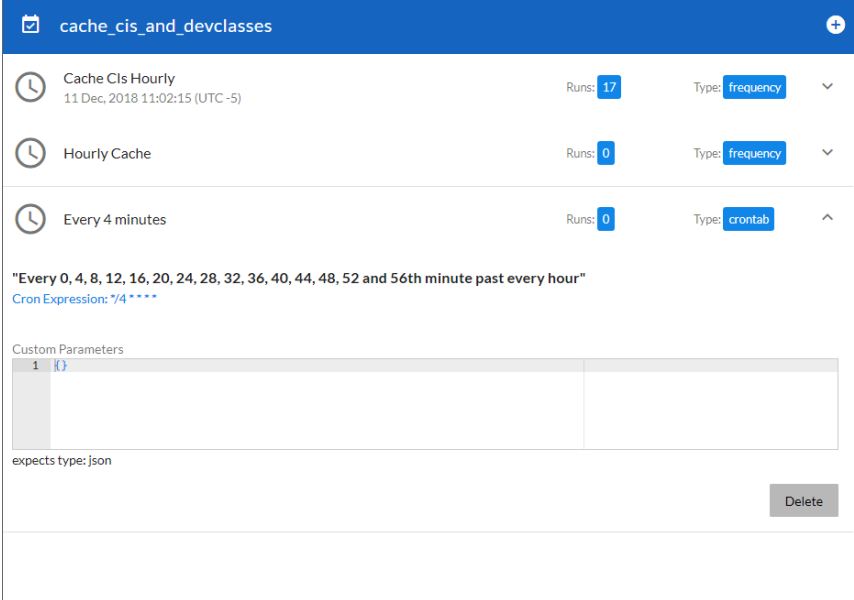
- "Cache ServiceNow CIs and SL1 Device Classes"

- "Sync Devices from SL1 to ServiceNow"

- "Sync Interfaces from SL1 to ServiceNow"

> **TIP:** ScienceLogic recommends that you schedule these applications to run at least every 23 hours. You can also schedule additional applications as needed.

You can create one or more schedules for a single application in the PowerFlow user interface. When creating each schedule, you can specify the queue and the configuration file for that application.

To schedule an application:

1. On the **Applications** page (▦), click the **[Schedule]** button for the application you want to schedule. The **Schedule** window appears, displaying any existing schedules for that application:

> **NOTE:** If you set up a schedule using a cron expression, the details of that schedule display in a more readable format in this list. For example, if you set up a cron expression of ***/4 * * * ***, the schedule on this window includes the cron expression along with an explanation of that expression: *"Every 0, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48, 52, and 56th minute past every hour"*.

2. Select a schedule from the list to view the details for that schedule.

3. Click the + icon to create a schedule. A blank **Schedule** window appears:

4. In the **Schedule** window, complete the following fields:

    - *Schedule Name*. Type a name for the schedule.

    - *Switch to*. Use this toggle to switch between a cron expression and setting the frequency in seconds.

        ◦ *Cron expression*. Select this option to schedule the application using a cron expression. If you select this option, you can create complicated schedules based on minutes, hours, the day of the month, the month, and the day of the week. As you update the cron expression, the **Schedule** window displays the results of the expression in more readable language, such as *Expression: "Every 0 and 30th minute past every hour on the 1 and 31st of every month"*, based on ***/30 * * /30 * ***.

        ◦ *Frequency in seconds*. Type the number of seconds per interval that you want to run the application.

    - *Custom Parameters*. Type any JSON parameters you want to use for this schedule, such as information about a configuration file or mappings. The values in this field will override any application variables that exist in the PowerFlow application you are scheduling.

5. Click **[Save Schedule]**. The schedule is added to the list of schedules on the initial **Schedule** window. Also, on the **Applications** page, the word "Scheduled" appears in the **Scheduled** column for this application, and the **[Schedule]** button contains a check mark:
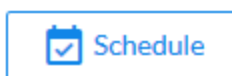
> **NOTE**: After you create a schedule, it continues to run until you delete it. Also, you cannot edit an existing schedule, but you can delete it and create a similar schedule if needed.

To view or delete an existing schedule:

1. On the **Applications** page, click the **[Schedule]** button for the application that contains a schedule you want to delete. The **Schedule** window appears.

2. Click the down arrow icon (∨) to view the details of an existing schedule:

3. To delete the selected schedule, click **[Delete]**.

> **NOTE**: When either multiple SL1 stacks or multiple ServiceNow systems are involved with PowerFlow, you should create an individual configuration object for each SL1 stack or ServiceNow system. Next, create an individual schedule for each configuration object. Each schedule should use a configuration object that is specific to that single SL1 stack or ServiceNow system. Creating copies of a PowerFlow application from a SyncPack for the purpose of distinguishing between domains is not supported, and will result in issues on upgrades.

The following image shows how you can schedule PowerFlow applications for multiple ServiceNow domains:

Log Messages for the "Generate Required CI Relations for ServiceNow" Application

This section describes the different types of log messages you might see in the Step Log when you run the "Generate Required CI Relations for ServiceNow" application. Please note that this application is a report used by PowerFlow, and it does not send any data to ServiceNow.

The following message displays if there are devices in a device tree that do not currently have a CI class mapping assigned.

```
Warning: 2751 Relations with missing mappings detected. Please re-run app
    with log level 10 to troubleshoot.
```

In this situation, the device tree cannot be built in ServiceNow. To address this issue, make sure that you have your entire technology tree mapped out in the *mappings* section of the "Sync Devices from SL1 to ServiceNow" application or in the *mappings* section of the "Generate Required CI Relations for ServiceNow" application.

If you do a Custom Run of the "Generate Required CI Relations for ServiceNow" application in Debug mode (log level 10), the application will create a log that displays the parent and child class, CI, and device ID. For example:

```
Debug: Missing Mapping for Device. Parent: {"class": "VMware | Cluster",
    "ci": None, "id": 76}, Child: {"class": "VMware | Host Server", ci:
    "cmdb_ci_esx_server", id: 363 }
```

The following message appears if the GraphQL payloads had bad data for parent and or child devices:

```
Warning: 10 bad payloads received from SL1. Re-run app in debug to
    troubleshoot.
```

If you do a Custom Run the application in Debug mode, the application will create a log that displays these payloads.

The following message appears if all relations are mapped:

```
Flow: No missing relations found!
```

The following message appears if there is a parent/child relation between ServiceNow CI classes that does not currently exist in ServiceNow and is required to sync those devices:

```
Flow: Missing Relations: [{"parent": "cmdb_ci_vcenter_folder", "child":
    "cmdb_ci_esx_server"}, {"parent": "cmdb_ci_vcenter", "child": "cmdb_ci_
    vcenter_datacenter"}]
```

Refer to the labels in the log (above) to determine which CI class is the parent type and which is the child type. To address this issue, navigate to your ServiceNow instance and create the required service rules based on the recommendations in the **Step Log**.

The following message appears if the application encounters a list of relations that are required, but were successfully found in ServiceNow:

```
Info: Found Relations: [{"parent": "cmdb_ci_vcenter_folder", "child":
    "cmdb_ci_esx_server"}, {"parent": "cmdb_ci_vcenter", "child": "cmdb_ci_
    vcenter_datacenter"}]
```

This message lets you verify that your mappings and relations are configured correctly.

Chapter

5

Troubleshooting the CMDB SyncPack

Overview

This chapter includes troubleshooting resources and procedures to use with the "ServiceNow CMDB" SyncPack.

This chapter covers the following topics:

Initial Troubleshooting Steps

PowerFlow acts as a middle server between data platforms. For this reason, the first steps should always be to ensure that there are no issues with the data platforms with which PowerFlow is talking. There might be additional configurations or actions enabled on ServiceNow or SL1 that result in unexpected behavior. For detailed information about how to perform the steps below, see *Resources for Troubleshooting*.

SL1 PowerFlow

1. Run `docker service ls` on the PowerFlow server:
    - Note the Docker container version.
    - Verify that the Docker services are running.
2. If a certain service is failing, make a note of the service name and version.
3. If a certain service is failing, run `docker service ps <service_name>` to see the historical state of the service and make a note of this information. For example: `docker service ps iservices_contentapi`.
4. Make a note of any logs impacting the service by running `docker service logs <service_name>`. For example: `docker service logs iservices_couchbase`.

ServiceNow

1. Make a note of the ServiceNow version and SyncPack version, if applicable.
2. Make a note if you are running a ServiceNow certified application or a Service Graph SyncPack.
3. Make a note of the SyncPack application that is failing in PowerFlow.
4. Make a note of what step is failing in the application, try running the application in debug mode, and capture any traceback or error messages that occur in the step log.

Resources for Troubleshooting

This section contains port information for PowerFlow and troubleshooting commands for Docker, Couchbase, and the PowerFlow API.

Useful PowerFlow Ports

- **https://<IP of PowerFlow>:8091**. Provides access to Couchbase, a NoSQL database for storage and data retrieval.
- **https://<IP of PowerFlow>:15672**. Provides access to the RabbitMQ Dashboard, which you can use to monitor the service that distributes tasks to be executed by PowerFlow workers.
- **https://<IP of PowerFlow>/flower/workers**. Provides access to Flower, a tool for monitoring and administrating Celery clusters.
- **https://<IP of PowerFlow>:3141**. Provides access to the pypiserver service. which you can use to see if SyncPacks have been correctly uploaded to Devpi container.

> **IMPORTANT:** Port 5556 must be open for both PowerFlow and the client.

Helpful Docker Commands

PowerFlow is a set of services that are containerized using Docker. For more information about Docker, see the [Docker tutorial](#).

Use the following Docker commands for troubleshooting and diagnosing issues with PowerFlow:

Viewing Container Versions and Status

To view the PowerFlow version, SSH to your instance and run the following command:

```
rpm -qa | grep powerflow
```

To view the individual services with their respective image versions, SSH to your PowerFlow instance and run the following command:

```
docker service ls
```

In the results, you can see the container ID, name, mode, status (see the *replicas* column), and version (see the *image* column) for all the services that make up PowerFlow:

Restarting a Service

Run the following command to restart a single service:

```
docker service update --force <service_name>
```

Stopping all PowerFlow Services

Run the following command to stop all PowerFlow services:

```
docker stack rm iservices
```

Restarting Docker

Run the following command to restart Docker:

```
systemctl restart docker
```

> **NOTE:** Restarting Docker does not clear the queue.

Diagnosis Tools

Multiple diagnosis tools exist to assist in troubleshooting issues with the PowerFlow platform:

- *Docker PowerPack*. This PowerPack monitors your Linux-based PowerFlow server with SSH (the PowerFlow ISO is built on top of an Oracle Linux Operating System). This PowerPack provides key performance indicators about how your PowerFlow server is performing. For more information on the Docker PowerPack and other PowerPacks that you can use to monitor PowerFlow, see the "Using SL1 to Monitor SL1 PowerFlow" chapter in the *SL1 PowerFlow Platform* manual.

- *Flower.* This web interface tool can be found at the /flower endpoint. It provides a dashboard displaying the number of tasks in various states as well as an overview of the state of each worker. This tool shows the current number of active, processed, failed, succeeded, and retried tasks on the PowerFlow platform. This tool also shows detailed information about each of the tasks that have been executed on the platform. This data includes the UUID, the state, the arguments that were passed to it, as well as the worker and the time of execution. Flower also provides a performance chart that shows the number of tasks running on each individual worker.

- *Debug Mode*. All applications can be run in "debug" mode via the PowerFlow API. Running applications in debug mode may slow down the platform, but they will result in much more detailed logging information that is helpful for troubleshooting issues. For more information on running applications in Debug Mode, see *Retrieving Additional Debug Information*.

- *Application Logs*. All applications generate a log file specific to that application. These log files can be found at **/var/log/iservices** and each log file will match the ID of the application. These log files combine all the log messages of all previous runs of an application up to a certain point. These log files roll over and will get auto-cleared after a certain point.

- *Step Logs*. Step logs display the log output for a specific step in the application. These step logs can be accessed via the PowerFlow user interface by clicking on a step in an application and bringing up the **Step Log** tab. These step logs display just the log output for the latest run of that step.

- *Service Logs*. Each Docker service has its own log. These can be accessed via SSH by running the following command:

```
docker service logs -f <service_name>
```

Retrieving Additional Debug Information (Debug Mode)

The logs in PowerFlow use the following **loglevel** settings, from most verbose to least verbose:

- *10*. Debug Mode.
- *20*. Informational.
- *30*. Warning. This is the default settings if you do not specify a loglevel.
- *40*. Error.

> **WARNING**: If you run applications in Debug Mode ("loglevel": 10), those applications will take longer to run because of increased I/O requirements. Enabling debug logging using the following process is the only recommended method. ScienceLogic does not recommend setting "loglevel": 10 for the whole stack with the **docker-compose** file.

To run an application in Debug Mode using the PowerFlow user interface:

1. Select the PowerFlow application from the **Applications** page.

2. Hover over the **[Run]** button and select *Custom Run* from the pop-up menu. The **Custom Run** window appears.

3. Select the Logging Level. *Debug* is the most verbose and will take longer to run.

4. Specify the configuration object for the custom run in the **Configuration** field, and add any JSON parameters in the **Custom Parameters** field, if needed.

5. Click **[Run]**.

To run an application in Debug Mode using the API:

1. POST the following to the API endpoint:

   ```
   https://<PowerFlow_IP>/api/v1/applications/run
   ```

2. Include the following in the request body:

   ```
   {
     "name": "<application_name>",
     "params": {
     "loglevel": 10
     }
   }
   ```

After running the application in Debug Mode, review the step logs in the PowerFlow user interface to see detailed debug output for each step in the application. This information is especially helpful when trying to understand why an application or step failed:

You can also run an application in debug using curl via SSH:

1. SSH to the PowerFlow instance.

2. Run the following command:

```
curl -v -k -u isadmin:<password> -X POST "https://<your_
hostname>/api/v1/applications/run"
 -H 'Content-Type: application/json' -H 'cache-control: no-cache' -d
'{"name":
"interface_sync_sciencelogic_to_servicenow","params": {"loglevel":
10}}'
```

Troubleshooting CMDB Sync

This section contains specific troubleshooting steps for the CMDB SyncPack.

Issues Creating CIs in ServiceNow

If you can successfully send data to your ServiceNow system, but you encounter issues with creating CIs in the ServiceNow CMDB, this section provides troubleshooting steps to help you test the payload and identify possible issues. These steps might be helpful if you have set up datasource precedence rules.

1. In ServiceNow, search for "import" in the filter navigator.

2. Select **ScienceLogic > Device > Imports**. The **Device Import** window appears.

3. From the list, select the Device Import log entry you want to view.

4. Copy the data from the **Payload** field in the log entry and decode the data from its Base64 encoding.

5. In the decoded string of data, remove the square brackets from the first and last line: ("**[**", "**]**")

6. Copy this modified JSON payload, and then use the filter navigator to search for "Identification Simulation" or select **Configuration > Identification Simulation**.

7. On the **Identification Simulation** page, click the **[Start]** button in the **Start with Existing Payload** section. The **Insert JSON Payload** page appears.

8. In the *Source* field, select *ScienceLogic* as the data source.

9. In the *Please insert payload below* field, paste the JSON payload you edited in step 5.

10. Click the **[Execute]** button and review the payload to identify any potential issues.

Enabling Debugging of the CI Payload

You must have administrator-level permissions in ServiceNow to access the system properties and enable debugging of the Configuration Item (CI) payload in the *ServiceNow Identification and Reconciliation module*.

To enable debugging of the CI payload in ServiceNow:

1. On the ServiceNow system, check to see if the `glide.cmdb.logger.source.identification_engine` record exists in `sys_properties.list`.

    - If the record exists, set this value to `(* or debugVerbose)`

    - If the record does not exist, you will need to create the record.

2. To create the record, complete the following fields:

    - **Name**. glide.cmdb.logger.source.identification_engine

    - *Description*. Enable and configure the type of details the system logs when using the Identification and Reconciliation module outside the scope of identification simulation, such as when using an API, an ECC queue, or scheduled jobs (info, warn, error, debug, or debugVerbose).

    - *Type*. String.

    - *Value*: `* or debugVerbose`

    > **NOTE:** Set the system property of *Value* back to `error` when troubleshooting is complete.

3. Run the "Sync Devices from SL1 to ServiceNow" application. The system logs will have "identfication_engine" as the source, and the log messages will contain `identification_engine : Input`.

4. Copy the payload beginning from `{"items"` to the end of the message. For example:

```
Message: {"items":[{"className":"","values":{"discovery_
source":"ScienceLogic","mac_address":"9E:0F:04:0A:12:C7",
"name":"Postman Test Server 1","x_sclo_scilogic_id":"1","serial_
number":"gJ3Bwkzc8r","model_id":"",
"ip_address":"10.10.10.102","manufacturer":"ScienceLogic,
Inc.","ram":"16000",
"x_sclo_scilogic_region":"Postman"},"lookup":[],"related":
[]}],"relations":[]}
```

5. You can run this message through the ScienceLogic endpoint by putting the `{"items"}` bracket within `[]`. For example, send the following message to the endpoint `/api/x_sclo_scilogic/v1/sciencelogic/IdentificationEngine`:

```
Message: [{"items":[{"className":"","values":{"discovery_
source":"ScienceLogic","mac_address":"9E:0F:04:0A:12:C7",
"name":"Postman Test Server 1","x_sclo_scilogic_id":"1","serial_
number":"gJ3Bwkzc8r",
"model_id":"","ip_
address":"10.10.10.102","manufacturer":"ScienceLogic,
Inc.","ram":"16000",
"x_sclo_scilogic_region":"Postman"},"lookup":[],"related":
[]}],"relations":[]}]
```

> **NOTE**: The endpoint is different in a domain-separated environment.

After the identification run is complete, the ServiceNow logs contain additional data about the run.

Appendix

A

Certified Application Objects

Overview

This appendix describes the tables, endpoints, and roles that were created in ServiceNow as part of the "ScienceLogic SL1: CMDB & Incident Automation" application. This application is also known as the "Certified Application" or the "Scoped Application".

This chapter covers the following topics:

Roles

Two Roles were added with the ScienceLogic update set, Admin (x_sclo_scilogic.Admin) and User (x_sclo_scilogic.User). Both give access to SL1.

Role	Inherited Roles	Other Inherited Roles	Role Definition
x_sclo_scilogic.Admin			Role for ScienceLogic Service Accounts.
	itil		Can perform standard actions for an ITIL help desk technician. This is the default "Technician" role. Can open, update, close incidents, problems, changes, config management items. By default, only users with the itil role can have tasks assigned to them
		Dependency_view	A special role to be applied both on the $ngbsm UI page and on the BSMProcessor. This role is required to access the dependency views module. By default, ITIL includes this role to avoid regressions.
		cmdb_query_builder	Can access the CMDB Query Builder application to create, run, and save queries on the CMDB.
		template_editor	
		view_changer	Can switch active views.
		app_service_user	Can view and retrieve information using API from application service maps (cmdb_ci_service_discovered).
		certification	Can work on Certification tasks.
	import_transformer		Can manage Import Set Transform Maps and run transforms.
x_sclo_scilogic.User			General user account that allows read-only access to SL1.

Tables

Name	Label	Extends	Comments
x_sclo_scilogic_catalog_item_templates	Catalog item Templates		Templates use to fill out catalog items

Name	Label	Extends	Comments
x_sclo_scilogic_discovery_ dependents	Discovery Dependent		Discovery Dependent Information
x_sclo_scilogic_event	Event		Event information
x_sclo_scilogic_catalog_event_ severity	Event Severity Look Rules	Data Lookup Matcher Rules	Lookup table for event Severity
x_sclo_scilogic_organizations	Organizations		Table to track Organizations between different ScienceLogic platforms
x_sclo_scilogic_import_ discovery_dependent	Import Discovery Dependent	Import Set Row	Import / staging events before transform to Discovery Dependent table
x_sclo_scilogic_discovery_ sessions	Import Discovery Sessions	Import Set Row	Import / staging events before transform to Event and Incident
x_sclo_scilogic_incident	Import Incident	Import Set Row	Import / staging events before transform to Event and Incident
x_sclo_scilogic_catalog_event_ severity			
x_sclo_scilogic_import_ installed_software	Import Installed Software	Import Set Row	Import / staging events before transform to Software Instance
x_sclo_scilogic_org_ven_mfg	Import ORG VEN MFG	Import Set Row	Import / staging events before transform to core_company

Table Columns (cmdb_ci)

Name	Label	Type	Comments
x_sclo_scilogic_id	SL1 ID	Integer	Unique ID
x_sclo_scilogic_region	SL1 Region	String	Unique String of SL1 Platform
x_sclo_scilogic_url	SL1 URL	URL	URL to SL1 Platform
x_sclo_scilogic_ monitored	SL1 Monitored	True/False	Device currently synced with SL1 Platform

Table Columns (core_company)

Name	Label	Type	Comments
x_sclo_scilogic_id	SL1 ID	String	Unique ID
x_sclo_scilogic_region	SL1 Region	String	Unique String of SL1 Platform
x_sclo_scilogic_ monitored	SL1 Monitored	True/False	Organization currently synced with SL1 Platform

Table Columns (cmdb_group)

Name	Label	Type	Comments
x_sclo_scilogic_id	SL1 ID	String	Unique ID
x_sclo_scilogic_region	SL1 Region	String	Unique String of SL1 Platform

Script Includes

Name	API Name	Comments
CatalogUtils	x_sclo_scilogic.CatalogUtils	Catalog Script include scripts
ChangeUtils	x_sclo_scilogic.ChangeUtils	
DeviceUtils	x_sclo_scilogic.DeviceUtils	
EventUtils	x_sclo_scilogic.EventUtils	
GeneralUtils	x_sclo_scilogic.GeneralUtils	
IntgSvcUtils	x_sclo_scilogic.IntgSvcUtils	

Event Registry

Suffix	Event name	Table
device_maintenance	x_sclo_scilogic.device_maintenance	Change Request [change_request]
device_maintenance_skd	x_sclo_scilogic.device_maintenance_skd	Change Request [change_request]
remove_monitoring	x_sclo_scilogic.remove_monitoring	Configuration Item [cmdb_ci]

Scripted Actions

Name	Event name
Device Maintenance	x_sclo_scilogic.device_maintenance
Device Monitoring Catalog item	x_sclo_scilogic.device_monitoring

Name	Event name
Device Removal Catalog item	x_sclo_scilogic.remove_monitoring
Device Schedule Maintenance	x_sclo_scilogic.device_maintenance_skd

Data Lookup Definitions

Name	Source Table	Matcher Table	Comments
Event Severity	Import Incident [x_sclo_scilogic_incident]	Event Severity Lookup Rules [x_sclo_scilogic_event_severity]	Lookup for ScienceLogic Severity to Impact and Urgency

System Properties

Suffix	Name
CatalogItemDiscovery	x_sclo_scilogic.CatalogItemDiscovery
CatalogItemRemove	x_sclo_scilogic.CatalogItemRemove
closeCode	x_sclo_scilogic.closeCode
Contact type	x_sclo_scilogic.Contact Type
discoverySource	x_sclo_scilogic.discoverySource
IdentificationEngineAPI	x_sclo_scilogic.IdentificationEngineAPI
IS_log_level	x_sclo_scilogic.IS_log_level
notResolved	x_sclo_scilogic.notResolved
SN_log_level	x_sclo_scilogic.SN_log_level
stateNew	x_sclo_scilogic.stateNew
StateResolved	x_sclo_scilogic.stateResolved

Catalog Item

Name	Comments
Device Discovery	Role for ScienceLogic Service Accounts.
Monitoring Removal	General user account that allows read only access to ScienceLogic Application.

Catalog UI Policies

Catalog item	Short description	Comments
Device Discovery	Catalog Template	Updates form based on Select template
Device Discovery	Create Virtual Device	Updates form based on Request type
Device Discovery	Create Virtual Device (Retired)	
Device Discovery	Device Discovery	Updates form based on Request type
Device Discovery	Device Discovery (Retired)	
Monitoring Removal	Hide Overview variables not required	Hide variables not required for the Monitoring Removal request
Device Discovery	Port Scan	Hide scan ports that are not default
Device Discovery	Port Scan (Retired)	
Device Discovery	Region	Updates form based on Organization
Device Discovery	Region (Retired)	
Monitoring Removal	Region via Organization	Updates form based on Organization
Device Discovery	Save as Template	Updates form based on Save as template

Variable Sets

Title	Internal name	Comments
Create_virtual_device	create_virtual_device	
Discovery Overview	discovery_overview	
Discovery Sesion - Basic Settings	discovery_sesion_basic_settings	
Discovery Session - Detection and Scanning	discovery_session_detection_and_scanning	
Discovery Session - IP & Credentials	discovery_session_ip_credentials	
Monitoring Removal	monitoring _removal	
Service Catalog item Template	service_catalog_item_template	

Catalog Client Scripts

Name	Catalog item	Type	Comments
Hide Request Type Options	Monitoring Removal	onLoad	Shared variable hide options that don't apply
Hide Request Type Options	Device Discovery	onLoad	Shared variable hide options that don't apply

Name	Catalog item	Type	Comments
Region	Monitoring Removal	onChange	Update Region field based on Company Region
Region	Monitoring Removal	onChange	Update Region field based on Company Region

Workflows

Name	Table	Comments
SL1 Monitoring Removal	Requested Item [sc_req_item]	Workflow for Removal of devices from SL1 process
SL1 Discovery Session	Requested Item [sc_req_item]	Workflow for Discovery session process

Scripted REST Resources

Name			Comments
Business Services	/api/x_sclo_scilogic/v1/sciencelogic/business_service	GET	This GET api will pull all ScienceLogic monitored Configuration items specific to Business Services class from the CMDB. It will be ordered via the sys_id field to ensure the same order every time.
Change Requests	/api/x_sclo_scilogic/v1/sciencelogic/change_requests	GET	This GET api will pull Active Change Requests or Change Tasks based on the record_type supplied that have ScienceLogic monitored CI attached. It will be ordered via the sys_id field to ensure the same order every time.
Classification	/api/x_sclo_scilogic/v1/sciencelogic/classification	GET	This GET api will pull all required CMDB information to build JSON payloads.
CMDB Group	/api/x_sclo_scilogic/v1/sciencelogic/cmdb_group	POST	Use this API to create cmdb_groups & add a CI to them.

Name			Comments
Companies	/api/x_sclo_scilogic/v1/sciencelogic/companies	GET	This GET api will pull all Active Companies that are ScienceLogic monitored. It will be ordered via the sys_id field to ensure the same order every time.
Configuration Items	/api/x_sclo_ scilogic/v1/sciencelogic/configuration_Items	GET	This GET api will pull all ScienceLogic monitored Configuration items from the CMDB. It will be ordered via the sys_id field to ensure the same order every time.
Device IdentificationEngine	/api/x_sclo_ scilogic/v1/sciencelogic/IdentificationEngine	POST	Use this API to create or update configuration items within the CMDB via ScienceLogic.
Discovery Dependent	/api/x_sclo_scilogic/v1/sciencelogic/discovery_ dependent	GET	
File Systems	/api/x_sclo_scilogic/v1/sciencelogic/file_systems	GET	This GET api will pull all ScienceLogic monitored Configuration items specific to File systems class from the CMDB. It will be ordered via the sys_id field to ensure the same order every time.
Import Set	/api/x_sclo_scilogic/v1/sciencelogic/import_set	POST	This POST API will post to the target import set table and create a record for each cmdb_ci.
Incidents	/api/x_sclo_scilogic/v1/sciencelogic/incidents	GET	This GET api will pull all incidents. It will be ordered via the sys_id field to ensure the same order every time.
Installed Software	/api/x_sclo_scilogic/v1/sciencelogic/installed_ software	GET	This GET api will pull all Servicenow Software packages and installed instances from the CMDB. It will be ordered via the sys_id field to ensure the same order every time.
Manufacture	/api/x_sclo_	POST	This POST API will pull all

Name			Comments
and Model	scilogic/v1/sciencelogic/ManufactureAndModel		Manufacturers and Models.
Network Adapters	/api/x_sclo_scilogic/v1/sciencelogic/network_adapters	GET	This GET api will pull all ScienceLogic monitored Configuration items specific to Network Adapter class from the CMDB. It will be ordered via the sys_id field to ensure the same order every time.
Service Request	/api/x_sclo_scilogic/v1/sciencelogic/service_request	GET	This GET api will pull all ServiceRequest items from the CMDB associated with Device Discovery Catalog item. It will be ordered via the sys_id field to ensure the same order every time.

Transform Maps

Name	Source Table	Target Table	Comments
ScienceLogic Catalog Item Templates	Import Discovery Sessions []	Catalog item Templates []	
ScienceLogic Change Request	Import Service Request []	Change Request []	
ScienceLogic Discovery Dependent	Import Discovery Dependent	Discovery Dependent	Import / staging table for Catalog Dependents
ScienceLogic Event	Import Incident	Event [x_sclo_scilogic_event]	Import / staging table for Events.
ScienceLogic Installed Software	Import Installed Software []	Software Instance	
ScienceLogic Organization	Import ORG VEN MFG []	Company [core_company]	Import / staging table for Organization
ScienceLogic Service Request	Import Service Request []	Request Item [sc_req_item]	Import / staging table for Request item
ScienceLogic Source Organization	Import ORG VEN MFG []	Organizations []	

Appendix

B

Mappings between SL1, ServiceNow, and Other Applications

Overview

This appendix contains information about how SL1 and ServiceNow discover and model out different technologies, using VMware as an example. The VMware example explains how the differences between the systems requires rules be configured to bridge the gap between SL1 and ServiceNow.

This chapter covers the following topics:

Overview

Different platforms do not model topology the same way. The default Customer Specific CI relation overrides delivered with the "ServiceNow CMDB" SyncPack provides several topology mappings by default. Existing customers upgrading to new versions of these mappings will not override any mappings that are already set.

The following topics explain how SL1 and ServiceNow model topology differently and the mappings that are provided in this SyncPack.

> **IMPORTANT:** The Discovery and Service Mapping Patterns application (sn_itom_pattern) plugin is required to align with your VMware mapping requirements.

VMware vCenter

The following table lists the Class mappings that are delivered by default. These are the same Class mappings that are used in Service Graph solution.

> **IMPORTANT:** The *VMware | vCenter* device class is mapped to the *Cloud Service Account* class and not the *vCenter* class directly. The default mapping adheres to ServiceNow Cloud Management Relationships.

Virtual Type	Device Category	Device Class | Sub Class	Class Name	Table Name
ScienceLogic			**ServiceNow**	
Component	Virtual.Infrastructure	VMware | Datacenter	VMware vCenter Datacenter	cmdb_ci_vcenter_datacenter
Component	Virtual.Infrastructure	VMware | Folder	VMware vCenter Folder	cmdb_ci_vcenter_folder
PowerFlow	Servers.VMware	VMware | vCenter	Cloud Service Account	cmdb_ci_cloud_service_account
Component	Virtual.Infrastructure	VMware | Datastore	VMware vCenter Datastore	cmdb_ci_vcenter_datastore
Component	Virtual.Guest	VMware | Virtual Machine	VMware Virtual Machine Instance	cmdb_ci_vmware_instance
Component	Servers.VMware	VMware | Cluster	VMware vCenter Cluster	cmdb_ci_cluster
Component	Virtual.Network	VMware | Network	VMware vCenter Network	cmdb_ci_network
Component	Virtual.Network	VMware | Distributed Virtual Portgroup	VMware Distributed Virtual Port Group	cmdb_ci_vcenter_dv_port_group
Component	Virtual.Network	VMware | Distributed	VMware Distributed	cmdb_ci_dvs

Virtual Type	Device Category	Device Class | Sub Class	Class Name	Table Name
		Virtual Switch	Virtual Switch	
Component	Virtual.Host	VMware | Host Server	ESX Server	cmdb_ci_esx_server
Component	Virtual.Infrastructure	VMware | Resource Pool	ESX Resource Pool	cmdb_ci_esx_resource_pool

Default Customer Relationship Overrides

The following table shows the relationship breakdown for the default Customer Relationship Overrides:

Relationships Mapped	Relationship	Child Class	Identifier
VMware vCenter Datacenter	Hosted on::Hosts	Cloud Service Account	Yes
VMware vCenter Datacenter	Contains::Contained by	VMware Distributed Virtual Port Group	
VMware vCenter Datacenter	Contains::Contained by	VMware vCenter Network	
VMware vCenter Datacenter	Contains::Contained by	VMware Distributed Virtual Switch	
VMware vCenter Datacenter	Contains::Contained by	VMware vCenter Folder	
VMware vCenter Datacenter	Contains::Contained by	VMware vCenter Datastore	
VMware vCenter Folder	Hosted on::Hosts	VMware vCenter Datacenter	
VMware vCenter Datastore	Hosted on::Hosts	VMware vCenter Datacenter	
VMware vCenter Datastore	Provides storage for::Stored on	VMware Virtual Machine Instance	
VMware vCenter Datastore	Used by::Uses	ESX Server	
VMware Virtual Machine Instance	Hosted on::Hosts	VMware vCenter Datacenter	
VMware Virtual Machine Instance	Connected by::Connects	VMware Distributed Virtual Port Group	
VMware vCenter Datacenter	Contains::Contained by	VMware vCenter Cluster	
VMware vCenter Cluster	Hosted on::Hosts	VMware vCenter Cluster	
VMware vCenter Cluster	Hosted on::Hosts	VMware vCenter Datacenter	
VMware vCenter Cluster	Members::Member of	ESX Resource Pool	
ESX Server	Hosted on::Hosts	VMware vCenter Datacenter	
ESX Resource Pool	Hosted on::Hosts	VMware vCenter Datacenter	
ESX Resource Pool	Defines resources for::Gets resources from	ESX Server	
ESX Resource Pool	Members::Member of	VMware Machine Instance	
ESX Resource Pool	Defines resources for::Gets	VMware vCenter Cluster	

Relationships Mapped	Relationship	Child Class	Identifier
	resources form		
VMware vCenter Network	Hosted on::Hosts	VMware vCenter Datacenter	
VMware vCenter Network	Provided by::Provides	ESX Server	
VMware Distributed Virtual Port Group	Hosted on::Hosts	VMware vCenter Datacenter	
VMware Distributed Virtual Switch	Hosted on::Hosts	VMware vCenter Datacenter	
VMware Distributed Virtual Switch	Connected by::Connects	VMware Distributed Virtual Port Group	
VMware Distributed Virtual Switch	Provided by::Provides	ESX Server	

Specific Class Breakdown

ScienceLogic Fields	ServiceNow Fields	Identifier	Notes
Instance UUID	Account ID	Yes	Verify that Instance UUID is a custom attribute on the vCenter device class, and is being populated with a GUID. Download the latest version of the VMware PowerPack.
Instance UUID	Object ID	Yes	
Name	name		
Datacenter type	Datacenter Type		
Instance URL	Instance URL		

ScienceLogic Fields	ServiceNow Fields	Identifier	Notes
Component Unique ID	Object ID	Yes	
Manufacture	Manufacturer		
Model	Model ID		
Instance UUID	vCenter Instance UUID		Service Graph Only
company	company		
	Location		
Component Unique ID	Managed object reference ID (MORID)		
Name	Name		

VMware vCenter Folder

ScienceLogic Fields	ServiceNow Fields	Notes
Component Unique ID	Object ID	
Component Unique ID	Managed object reference ID (MORID)	
Model	Model ID	
	Location	
Instance UUID	vCenter Instance UUID	Service Graph Only
Name	Name	
Company	Company	
Manufacturer	Manufacture	

VMware Datastore

ScienceLogic Fields	ServiceNow Fields	Notes
Instance UUID (Service Graph only), Component Unique ID	Morid, object_id, url	
Name	Name	
Component Unique ID	Object ID	
Instance UUID	vCenter Instance UUID	Service Graph Only
Host ID	URL	
Model	Model ID	
Company	Company	
Component Unique ID	Managed object reference ID (MORID)	
Manufacturer	Manufacturer	
	Location	

VMware Virtual Machine Instance

ScienceLogic Fields	ServiceNow Fields	Notes
Component Unique ID	Object ID	

ScienceLogic Fields	ServiceNow Fields	Notes
Instance UUID	vCenter Instance UUID	Service Graph Only
Manufacture	Manufacturer	
Model	Model ID	
	Location	Service Graph Only
Company	Company	
Host ID	Bios UUID	

VMware Cluster

ScienceLogic Fields	ServiceNow Fields	Notes
Component Unique ID	Object ID	
Name	Name	
Component Unique ID	Managed object reference ID (MORID)	
Manufacturer	Manufacturer	
	Location	
Model	Model ID	
Instance UUID	vCenter Instance UUID	Service Graph Only
Company	Company	

ESX Server

ScienceLogic Fields	ServiceNow Fields	Notes
Host ID	Correlation ID	
Component Unique ID	Managed object reference ID (MORID)	
Model	Model ID	
	Location	
Manufacturer	Manufacture	
Serial	Serial Number	
Company	Company	
Name	Name	
Component Unique	Object ID	

ScienceLogic Fields	ServiceNow Fields	Notes
ID		

ESX Resource Pool

ScienceLogic Fields	ServiceNow Fields	Notes
Name	Name	
Component Unique ID	Object ID	
	Location	
Instance UUID	vCenter Instance UUID	Service Graph Only
Manufacturer	Manufacturer	
Company	Company	
Component Unique ID	Managed object reference ID (MORID)	
Model	Model ID	

vCenter Network

ScienceLogic Fields	ServiceNow Fields	Notes
Component Unique ID	Object ID	
Instance UUID	vCenter Instance UUID	Service Graph Only
Manufacturer	Manufacturer	
	Location	Service Graph Only
Name	Name	
Company	Company	
Component Unique ID	Managed object reference ID (MORID)	

Distributed Virtual Switch

ScienceLogic Fields	ServiceNow Fields	Notes
Component Unique ID	Object ID	

ScienceLogic Fields	ServiceNow Fields	Notes
Company	Company	
Instance UUID	vCenter Instance UUID	Service Graph Only
Name	Name	
Component Unique ID	Managed object reference ID (MORID)	
Manufacturer	Manufacturer	
Model	Model	

Distributed Virtual Portgroup

ScienceLogic Fields	ServiceNow Fields	Notes
Component Unique ID	Object ID	
Instance UUID	vCenter Instance UUID	Service Graph Only
Model	Model ID	
Name	Name	
	Location	Service Graph Only
Manufacturer	Manufacturer	

Appendix

C

ServiceNow API Endpoints

Overview

This appendix describes the customized ServiceNow API Endpoints that were created for the ServiceNow SyncPacks. These scripted endpoints reduce the amount of REST calls that PowerFlow makes to ServiceNow.

Please note that for pagination, you can use the following Query parameters: `sysparm_offset` and `sysparm_limit`. The default settings are:

- `sysparm_offset=0`
- `sysparm_limit` = ServiceNow defines the default upper limits for data export. It will check the following properties at *System Properties > Import Export*: `glide.json.export.limit`, `glide.ui.export.limit`, and then `glide.ui.export.war.threshold`.

For example, if you have 200 total records and you want to pull the records in 100-record chunks, then the first pull would be `sysparm_offset=0 & sysparm_limit=100` and the second pull would be `sysparm_offset=100 & sysparm_limit=100`. For more information, see the ServiceNow documentation for Export Limits.

This chapter covers the following topics:

159

Classification

HTTP Method

GET

Resource Path

```
/api/x_sclo_scilogic/v2/sciencelogic/classification
```

Pagination

Enabled

Overview

To support the Identification and Reconciliation framework, SL1 requires a large amount of information to correctly fill out the JSON-formatted string defined by the Identification Engine documentation.

This operation uses the **getTableExtension()** function to find all of the tables extended from the **cmdb_ci** table, and then it goes through each table individually. This operation collects information about each class, such as which fields are required to identify and if it is considers another class to help find uniqueness. This operation then finds all the associated metadata.

Finally, the operation pulls a list of all field names from the table. By default the **criterion_attributes** and **attributes** are not included and require "action=attributes" as a parameter in the API call to be passed.

The class attributes have been disabled. The class attributesd require x_sclo_scilogic.Admin be added to **sys_ dictionary.*** (read) ACL to allow the API to access field names on each class table.

Headers	
Key	*Value*
Content-Type	application/json
Accept	application/json

Parameters	
Key	*Value*
action	attributes
sysparm_offset	0
sysparm_limit	glide.json.export.limit, glide.ui.export.limit, glide.ui.export.war.threshold

HTTP Status	
Code	*Value*
200	OK

Fixed Internal Query

```
new GlideTableHierarchy("cmdb_ci").getTableExtensions() //Returns an array
of all tables extended from the current table ('cmdb_ci').
```

Example

```
https://<your_instance>.service-now.com/api/x_sclo_
scilogic/v2/sciencelogic/classification
```

Example (Response)

```
{
    "results": [
        {
            "class_label": "Tomcat WAR",
            "class_table": "cmdb_ci_app_server_tomcat_war",
            "independent": "false",
            "search_on_table": "cmdb_ci_app_server_tomcat_war",
            "criterion_attributes": [
                {
                    "cmdb_ci_app_server_tomcat_war": [
                        "sys_class_name,name,install_directory"
                    ]
                }
            ],
            "containment_rule": [
                {
                    "configuration_item": "cmdb_ci_app_server_tomcat_war",
                    "parent": "cmdb_ci_app_server_tomcat",
                    "rel_type": "Contains::Contained by",
                    "is_reverse": "false"
                }
            ],
            "hosting_rule": [],
            "reference_rule": []
```

Classification

```
        }
    ],
    "sysparm_offset": 29,
    "sysparm_limit": 1,
    "return_count": 1,
    "total_count": 1
}
```

Companies

HTTP Method

GET

Resource Path

```
/api/x_sclo_scilogic/v2/sciencelogic/companies
```

Pagination

Enabled

Overview

This operation supports multiple SL1 platforms synchronizing organization data that may share companies across different SL1 systems. The results are ordered by **sys_id** of the **sys_object_source** table entry. The **core_company** and **x_sclo_scilogic_organizations** tables are joined on query to only pull companies associated with a specific SL1. ScienceLogic-specific data is stored on **x_sclo_scilogic_organizations**, and company data is stored on the **core_company** table.

Headers	
Key	*Value*
Content-Type	application/json
Accept	application/json

Parameters	
Key	*Value*
region (required)	ScienceLogic
domainSep	false
sysparm_offset	0
sysparm_limit	glide.json.export.limit, glide.ui.export.limit, glide.ui.export.war.threshold

HTTP Status	
Code	*Value*
200	OK
400	Query parameter \'region\' is not defined and is required.

Example

```
https://<your_instance>.service-now.com/api/x_sclo_
scilogic/v2/sciencelogic/companies with headers: {'content-type':
'application/json', 'accept': 'application/json'} and read timeout: 30 and
api parameters {'region': 'ScienceLogic', 'sysparm_limit': 1000}
```

Example (Response)

```
{
    "results": [
        {
            "country": "USA",
            "sys_updated_on": "2022-09-09 15:27:47",
            "sys_updated_by": "powerflow_user_cmdb_incident",
            "sys_created_on": "2022-09-09 15:27:47",
            "state": "VA",
            "sys_created_by": "powerflow_user_cmdb_incident",
            "zip": "27560",
            "profits": "0",
            "phone": "(703)-354-1010",
            "fax_phone": "(703)-336-8000",
            "name": "System",
            "city": "Reston",
            "sys_class_name": "core_company",
            "sys_class_name_label": "Company",
            "sys_id": "4d6f7f821bb159100efb206cbc4bcb65",
            "market_cap": "0",
            "street": "System Location",
            "revenue_per_year": "0",
            "customer": "true",
            "native_key": "ven55_sl151_pf63+ORG+0",
            "region": "ven55_sl151_pf63"
        }
    ],
    "sysparm_offset": 4,
    "sysparm_limit": 1,
    "return_count": 1
}
```

Change Requests

HTTP Method

GET

Resource Path

```
/api/x_sclo_scilogic/v2/sciencelogic/change_requests?record_type=change_
request&state=1&region=ScienceLogic
```

Pagination

Enabled

Overview

This scripted API was built for pulling Change Requests or Change Tasks and formatting a JSON object response with the required information to create a maintenance schedule in SL1. The GET queries the **task_ci** table to find configuration items that are monitored by SL1 and are the correct record type. The GET will *only* pull Change Requests and Change Tasks that are in the future.

Headers	
Key	*Value*
Content-Type	application/json
Accept	application/json

Parameters	
Key	*Value*
record_type (required)	change_request
state	-5
region (required)	ScienceLogic
sysparm_offset	0
sysparm_limit	glide.json.export.limit, glide.ui.export.limit, glide.ui.export.war.threshold

HTTP Status

Code	Value
200	OK
400	Query parameter \'region\' or \'record_type\' is not defined and is required.

Example (Request URL)

```
https://<your Instance>.service-now.com/api/x_sclo_
scilogic/v2/sciencelogic/change_requests?record_type=change_
request&state=-2&region=ven70_sl151_pf199&sysparm_offset=1&sysparm_limit=1
```

Examples (Response)

Change Request:

```
{
    "results": [
        {
            "sys_id": "129994931bf91110f219866fdc4bcbb6",
            "number": "CHG0030115",
            "type": "Normal",
            "state_value": "-2",
            "state": "Scheduled",
            "short_description": "Firmware upgrade Printer (x8)",
            "planned_start_date": "2022-09-30 17:00:00",
            "planned_end_date": "2022-09-30 17:30:00",
            "device": [
                {
                    "sys_id": "75febcd21b3d99106af3dac4bd4bcb95",
                    "name": "Printer: SIM-0019004",
                    "class": "cmdb_ci_printer",
                    "id": "488",
                    "region": "ven70_sl151_pf199"
                }
            ]
        }
    ],
    "sysparm_offset": 0,
    "sysparm_limit": 1,
    "return_count": 1,
```

```
        "total_count": 2
    }
```

Change Task:

```
{
    "results": [
        {
            "sys_id": "af2554eb1b7d5110f219866fdc4bcbcd",
            "number": "CTASK0010073",
            "state_value": "1",
            "state": "Open",
            "short_description": "firmware upgrade",
            "planned_start_date": "2022-10-01 14:30:00",
            "planned_end_date": "2022-10-01 15:00:00",
            "device": [
                {
                    "sys_id": "01fef8d21b3d99106af3dac4bd4bcb1f",
                    "name": "IP Router: SIM-0019217",
                    "class": "cmdb_ci_ip_router",
                    "id": "439",
                    "region": "ven70_sl151_pf199"
                }
            ]
        }
    ],
    "sysparm_offset": 0,
    "sysparm_limit": 1,
    "return_count": 1,
    "total_count": 1
}
```

Change Requests

Incidents

HTTP Method

GET

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/incidents
```

Pagination

Enabled

Overview

This operation pulls all records from the incident table that are created by a specific **user_id** and its related events. The results are ordered by the **sys_id** of the incident, so the results display in the same order every time. This operation is also based on the incident being in an active state. This operation returns a pre-set of data and does not return everything on the Incident and Event (**x_sclo_scilogic_event**) tables.

Headers	
Key	*Value*
Content-Type	application/json
Accept	application/json

> **NOTE:** If you are using version 3.5.0 or later of the ServiceNow CMDB SyncPack, the **user_id** for each SL1 Region must be different.

Parameters	
Key	*Value*
user_id (required)	is4user1 **NOTE:** If you are using version 3.5.0 or later of the ServiceNow CMDB SyncPack, the **user_id** for each SL1 Region must be different.
sysparm_offset	0
sysparm_limit	glide.json.export.limit, glide.ui.export.limit, glide.ui.export.war.threshold

HTTP Status	
Code	*Value*
200	OK
400	Query Parameter \'user_id\' is not defined and is required.

Fixed Internal Query

```
'sys_created_by=' +  user_id + 'active=true'
```

Example

```
https://<your Instance>.service-now.com/api/x_sclo_
scilogic/v1/sciencelogic/incidents?user_id=is4user1&sysparm_
offset=0&sysparm_limit=100
```

Example (Response)

```
{
    "results": [
        {
            "sys_id": "e28538241b4ed9106af3dac4bd4bcba8",
            "number": "INC0022814",
            "correlation": "pm_postman_test+DEV+130",
            "state": "1",
            "state_label": "New",
            "priority": "5",
            "events": [
                {
                    "event_id": "1337",
                    "device": {
                            "sys_id": "ccfef4d21b3d99106af3dac4bd4bcbcb",
                            "name": "Linux Server: en-sl1cu-11-148",
                            "class": "cmdb_ci_linux_server",
                            "id": "3",
                            "region": "pm_postman_test"
                    }
                }
            ]
        }
    ],
```

```
        "sysparm_offset": 0,
        "sysparm_limit": 5000,
        "return_count": 1,
        "total_count": 1
}
```

Service Requests

HTTP Method

GET

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/service_request
```

Pagination

Enabled

Overview

This operation pulls all service requests that are tied to specific catalog item. Based on the request type it returns a formatted JSON object. It pulls all the required information for an SL1 Discovery session and creating a virtual device in SL1. Both requests require different information and are formatted accordingly.

The basic catalog item Device Discovery is set up as information collection to support the process within SL1. The Service Catalog has been simplified to its most basic form. The workflow moves the request into the correct state to be picked up by the GET request and then waits for its return before completing the workflow.

Headers	
Key	*Value*
Content-Type	application/json
Accept	application/json

Parameters	
Key	*Value*
region (required)	ScienceLogic
state	2
sysparm_offset	0
sysparm_limit	glide.json.export.limit, glide.ui.export.limit, glide.ui.export.war.threshold

HTTP Status	
Code	*Value*

200	OK
400	Query Parameter \'region\' is not defined and is required.

Fixed Internal Query

State:

```
'request_item.active=true^request_item.cat_item=' + catalog + '^sc_item_
option.item_option_new.name=Region^sc_item_option.value=' + region
```

Non-State:

```
'request_item.active=true^request_item.cat_item=' + catalog + '^sc_item_
option.item_option_new.name=Region^sc_item_option.value=' + region +
'^request_item.state=' + state
```

Example

```
https://<your Instance>.service-now.com/api/x_sclo_
scilogic/v1/sciencelogic/ service_request?region=Cisco
```

Example (Response)

```
{
    "results": [
        {
            "number": "RITM0012892",
            "sysid": "b87564ad1bc6d910f219866fdc4bcb9a",
            "state": "1",
            "request_type": "remove_device",
            "organization": "3",
            "region": "ven70_sl151_pf199",
            "device_monitoring_removal": [
                {
                    "sys_id": "33f87f801b0299106af3dac4bd4bcbc7",
                    "name": "IP Router: SIM-0000225",
                    "class": "cmdb_ci_ip_router",
                    "id": "244",
                    "region": "ven70_sl151_pf199",
                    "ip_address": "10.140.1.0"
                }
            ]
        },
```

```
{
    "number": "RITM0012890",
    "sysid": "017e9f111b8251506af3dac4bd4bcb82",
    "state": "1",
    "request_type": "discover_device",
    "region": "ven70_sl151_pf199",
    "log_all": false,
    "ip_hostname_list": "10.140.150.60",
    "credentials": [
        {
            "Category": "dbCredentials",
            "ID": "20"
        }
    ],
    "discover_non_snmp": true,
    "model_devices": true,
    "dhcp": true,
    "device_model_cache_ttl_h": "",
    "collection_server": "2",
    "organization": "2",
    "add_devices_to_device_groups": [],
    "device_template": "7",
    "initial_scan_level": "System Default (Recommended)",
    "scan_throttle": "0",
    "scan_ports": "21,22,23,25,80",
    "port_scan_all": "0",
    "port_scan_timeout": "System Default (Recommended)",
    "interface_inventory_timeout": "",
    "maximum_allowed_interfaces": "",
    "bypass_interface_inventory": false,
    "affected": [
        {
            "sys_id": "71fefcd21b3d99106af3dac4bd4bcbab",
            "name": "IP Router: SIM-0038429",
            "class": "cmdb_ci_ip_router",
            "id": "9999",
            "region": "ven70_sl151_pf199",
            "ip_address": "10.140.150.60"
        }
    ]
```

```
            }
    ],
    "sysparm_offset": 0,
    "sysparm_limit": 10000,
    "return_count": 2,
    "total_count": 2
}
```

Configuration Items

HTTP Method

POST

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/configuration_item
```

Pagination

Enabled

Overview

This operation pulls all the fields for any configuration item or class that are not Null values by default. The results are ordered by **sys_id** of the record on the **sys_object_source** table. The **region** is required, and the POST will only return configuration items with an associated region. Business Services are handled in another GET request; otherwise all configuration items are pulled from this POST request.

Headers	
Key	*Value*
Content-Type	application/json
Accept	application/json

Parameters	
Key	*Value*
region (required)	ScienceLogic
sysparm_offset	0
sysparm_limit	glide.json.export.limit, glide.ui.export.limit, glide.ui.export.war.threshold
action	empty_fields

HTTP Status	
Code	*Value*
200	OK
400	Query Parameter \'region\' is not defined and is required.

Example

```
https://<your Instance>.service-now.com/api/x_sclo_
scilogic/v1/sciencelogic/configuration_Items?region=ven70_sl151_pf199&
```

Example (Body)

```
{
    "classes": [
        "cmdb_ci_linux_server"
    ],
    "organization": "c16f7f821bb159100efb206cbc4bcb61"
}
```

Example (Response)

```
{
    "results": [
        {
            "object_source_id": "ven70_sl151_pf199+DEV+3",
            "firewall_status": "Intranet",
            "firewall_status_label": "Intranet",
            "operational_status": "1",
            "operational_status_label": "Operational",
            "sys_updated_on": "2022-09-15 20:08:15",
            "discovery_source": "Other Automated",
            "discovery_source_label": "Other Automated",
            "first_discovered": "2022-07-28 15:44:45",
            "used_for": "Production",
            "used_for_label": "Production",
            "sys_created_by": "powerflow_user_cmdb_incident",
            "ram": "16431296",
            "cpu_speed": "2300",
            "sys_domain_path": "/",
            "classification": "Production",
            "classification_label": "Production",
            "short_description": "Description: System.EM7, Device Class:
ScienceLogic, Inc. | EM7 Data Collector",
            "last_discovered": "2022-09-15 20:08:15",
            "sys_class_name": "cmdb_ci_linux_server",
            "sys_class_name_label": "Linux Server",
```

```
            "manufacturer": "eafa349e1bf999106af3dac4bd4bcbf0",
            "manufacturer_label": "ScienceLogic, Inc.",
            "serial_number": "VMware-42 05 99 79 59 51 5b 82-05 80 01 86 7e
83 b4 82",
            "x_sclo_scilogic_monitored": "true",
            "asset": "04fef4d21b3d99106af3dac4bd4bcbcd",
            "asset_label": "ScienceLogic, Inc. EM7 Data Collector",
            "x_sclo_scilogic_region": "ven70_sl151_pf199",
            "sys_updated_by": "powerflow_user_cmdb_incident",
            "sys_created_on": "2022-09-09 20:05:25",
            "sys_domain": "global",
            "cpu_type": "Xeon(R)",
            "x_sclo_scilogic_url": "https://<SL1 instance>/in-
ventory/devices/3/investigator",
            "fqdn": "en-sl1cu-11-148",
            "internet_facing": "true",
            "hardware_status": "installed",
            "hardware_status_label": "Installed",
            "install_status": "1",
            "install_status_label": "Installed",
            "name": "en-sl1cu-11-148",
            "subcategory": "Computer",
            "virtual": "true",
            "sys_id": "ccfef4d21b3d99106af3dac4bd4bcbcb",
            "sys_class_path": "/!!/!2/!(/!!/!0",
            "company": "c16f7f821bb159100efb206cbc4bcb61",
            "company_label": "ScienceLogic",
            "attestation_status": "Not Yet Reviewed",
            "attestation_status_label": "Not Yet Reviewed",
            "sys_mod_count": "2",
            "x_sclo_scilogic_id": "3",
            "model_id": "e6fa349e1bf999106af3dac4bd4bcbf1",
            "model_id_label": "ScienceLogic, Inc. EM7 Data Collector",
            "ip_address": "10.2.11.148",
            "cost_cc": "USD",
            "cost_cc_label": "USD",
            "category": "Hardware",
            "host_name": "en-sl1cu-11-148"
        }
    ],
```

```
    "sysparm_offset": 0,
    "sysparm_limit": 100,
    "return_count": 1
}
```

Device Identification Engine

HTTP Method

POST

Resource Path

```
/api/x_sclo_scilogic/v2/sciencelogic/IdentificationEngine
```

Overview

This operation handles all creates and updates to the CMDB. This operation incorporates Identification Engine and uses the Identification and Reconciliation framework to properly import devices into the CMDB as a configurable discovery source. SL1 uses the classification GET to populate the JSON object.

All versions after version 1.0.45 of the IndentificationEngine endpoint use [IdentificationEngineScopedAPI](#) Enhanced versions.

Headers	
Key	**Value**
Content-Type	application/json
Accept	application/json

Parameters	
Key	**Value**
test	true
Source	Other Automated

Example (Request URL)

```
https://<your_Instance>.service-now.com/api/x_sclo_
scilogic/v2/sciencelogic/IdentificationEngine?test=false&source=Other%20A
utomated
```

Example (Body)

```
[
  {
    "items": [
      {
```

```
          "className": "cmdb_ci_ip_router",
          "values": {
              "x_sclo_scilogic_monitored": true,
              "x_sclo_scilogic_id": "9861",
              "x_sclo_scilogic_region": "ven70_sl151_pf199",
              "x_sclo_scilogic_url": "https://10.2.11.151/in-
ventory/devices/9861/investigator",
              "short_description": "Description: Network.Router, Device
Class: Cisco Systems | 1811 ISR G1",
              "cpu_count": 0,
              "cpu_speed": "",
              "first_discovered": "2022-07-30 04:43:03",
              "fqdn": "SIM-0038660",
              "host_name": "SIM-0038660",
              "ip_address": "10.140.151.35",
              "manufacturer": "cefa349e1bf999106af3dac4bd4bcb18",
              "model_id": "0afa349e1bf999106af3dac4bd4bcb1e",
              "name": "SIM-0038660"
          },
          "sys_object_source_info": {
            "source_feed": "ven70_sl151_pf199",
            "discovery_source": "Other Automated",
            "source_native_key": "ven70_sl151_pf199+DEV+9861",
            "source_recency_timestamp": "2022-09-14 17:10:33"
          }
        }
    ],
    "relations": [
    ]
  },
  {
    "items": [
        {
          "className": "cmdb_ci_ip_router",
          "values": {
              "x_sclo_scilogic_monitored": true,
              "x_sclo_scilogic_id": "10012",
              "x_sclo_scilogic_region": "ven70_sl151_pf199",
              "x_sclo_scilogic_url": "https://10.2.11.151/in-
ventory/devices/10012/investigator",
```

```
            "short_description": "Description: Network.Router, Device
Class: Cisco Systems | 1811 ISR G1",
            "cpu_count": 0,
            "cpu_speed": "",
            "first_discovered": "2022-07-30 04:50:47",
            "fqdn": "SIM-0039677",
            "host_name": "SIM-0039677",
            "ip_address": "10.140.155.28",
            "manufacturer": "cefa349e1bf999106af3dac4bd4bcb18",
            "model_id": "0afa349e1bf999106af3dac4bd4bcb1e",
            "name": "SIM-0039677"
          },
          "sys_object_source_info": {
            "source_feed": "ven70_sl151_pf199",
            "discovery_source": "Other Automated",
            "source_native_key": "ven70_sl151_pf199+DEV+10012",
            "source_recency_timestamp": "2022-09-14 17:10:33"
          }
        }
    ],
    "relations": [
    ]
  }
]
```

CMDB Group

HTTP Method

POST

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/cmdb_group
```

Overview

This operation handles the intake of groups of devices from SL1 and converts the device groups to CMDB groups. This operation uses a standard formatted JSON string, and it checks for a *sys_id* of the group first by searching for a matching group. This process creates a group if a group is not supplied or found, and then it passes the JSON object to the ServiceNow CMDBGroupAPI, which sets the manual CI list of the group.

Headers	
Key	*Value*
Accept	application/json
Content-Type	application/json

Example (Request URL

```
https://<your Instance>.service-now.com/api/x_sclo_
scilogic/v1/sciencelogic/cmdb_group
```

Example (Body)

```
{
    "items": [
        {
            "name": "pm_postman_test: Servers - 3",
            "manualCIList": [
                "pm_postman_test+DEV+2",
                "pm_postman_test+DEV+3",
                "pm_postman_test+DEV+4",
                "pm_postman_test+DEV+264",
                "pm_postman_test+DEV+280",
                "pm_postman_test+DEV+283",
                "pm_postman_test+DEV+285",
                "pm_postman_test+DEV+293",
```

```
            "pm_postman_test+DEV+294",
            "pm_postman_test+DEV+303"
        ],
        "native_key": "pm_postman_test+GRP+3"
    },
    {
        "name": "pm_postman_test: SL1 Appliances - 6",
        "manualCIList": [
            "pm_postman_test+DEV+551",
            "pm_postman_test+DEV+616",
            "pm_postman_test+DEV+632"
        ],
        "native_key": "pm_postman_test+GRP+6"
    }
  ]
}
```

Example (Response)

```
{
    "result": [
        {
            "name": "pm_postman_test: Servers - 3",
            "manualCIList": [
                "pm_postman_test+DEV+2",
                "pm_postman_test+DEV+3",
                "pm_postman_test+DEV+4",
                "pm_postman_test+DEV+264",
                "pm_postman_test+DEV+280",
                "pm_postman_test+DEV+283",
                "pm_postman_test+DEV+285",
                "pm_postman_test+DEV+293",
                "pm_postman_test+DEV+294",
                "pm_postman_test+DEV+303"
            ],
            "native_key": "pm_postman_test+GRP+3",
            "group": "aa1535f51b4691100efb206cbc4bcb22",
            "result": true,
            "nextBatchStart": 0,
            "idList": [],
            "partialCIListDueToACLFlag": false
```

```
        },
        {
            "name": "pm_postman_test: SL1 Appliances - 6",
            "manualCIList": [
                "pm_postman_test+DEV+551",
                "pm_postman_test+DEV+616",
                "pm_postman_test+DEV+632"
            ],
            "native_key": "pm_postman_test+GRP+6",
            "group": "aa1535f51b4691100efb206cbc4bcb23",
            "result": true,
            "nextBatchStart": 0,
            "idList": [],
            "partialCIListDueToACLFlag": false
        }
    ]
}
```

Manufacturer and Model

HTTP Method

POST

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/ManufactureAndModel
```

Overview

This operation takes an array of model names and attempts to line them up with Models already in ServiceNow, and then returns the **sys_id** of models that match by name. This function uses the following provided function (MakeAndModelJS.fromNames("VAR_MANFACTURER", "VAR_MODEL_OF_DEVICE", "VAR_MODEL_CLASS")).

Headers	
Key	*Value*
Accept	application/json
Content-Type	application/json

Parameters	
Key	*Value*
ModelCategory (required)	hardware

Example (Request URL)

```
https://<your Instance>.service-now.com/api/x_sclo_
scilogic/sciencelogic/ManufactureAndModel?ModelCategory=hardware
```

Example (Body)

```
{
    "Cisco Systems": [
        "1803 ISR G1",
        "1861 ISR G1",
        "2821 ISR G1",
        "826 Quad V",
        "888SRST ISR G2",
        "Catalyst 1912C",
        "Catalyst2912XL",
```

```
        "Catalyst2960S-48FPD-L",
        "TelePresenceEX60",
        "UCSC460M4RackServer",
        "WS-C2960S-24PS-L",
        "VG202XM"
    ],
    "OpenStack": [
        "Cloud Physical Service"
    ],
    "VMware, Inc.": [
        "VMware Virtual Platform"
    ],
    "Dell Inc.": [
        "PowerEdge R530",
        "PowerEdge R740xd"
    ]
}
```

Example (Response)

```
{
    "result": {
        "Cisco Systems": {
            "models": {
                "1803 ISR G1": "b36b2d711b025d10f219866fdc4bcbb7",
                "1861 ISR G1": "f76b2d711b025d10f219866fdc4bcbb8",
                "2821 ISR G1": "4efa349e1bf999106af3dac4bd4bcb28",
                "826 Quad V": "376b2d711b025d10f219866fdc4bcbb9",
                "888SRST ISR G2": "ff6b2d711b025d10f219866fdc4bcbb9",
                "Catalyst 1912C": "bb6b2d711b025d10f219866fdc4bcbba",
                "Catalyst2912XL": "776b2d711b025d10f219866fdc4bcbbb",
                "Catalyst2960S-48FPD-L": "336b2d711b025d10f219866f-
dc4bcbbc",
                "TelePresenceEX60": "fb6b2d711b025d10f219866fdc4bcbbc",
                "UCSC460M4RackServer": "b76b2d711b025d10f219866fdc4bcbbd",
                "WS-C2960S-24PS-L": "736b2d711b025d10f219866fdc4bcbbe",
                "VG202XM": "3f6b2d711b025d10f219866fdc4bcbbe"
            },
            "sys_id": "cefa349e1bf999106af3dac4bd4bcb18"
        },
        "OpenStack": {
```

```
            "models": {
                "Cloud Physical Service": "4c7b2d711b025d10f219866f-
dc4bcbc1"
            },
            "sys_id": "b76b2d711b025d10f219866fdc4bcbbf"
        },
        "VMware, Inc.": {
            "models": {
                "VMware Virtual Platform": "36fa749e1b-
f999106af3dac4bd4bcb66"
            },
            "sys_id": "3afa749e1bf999106af3dac4bd4bcb65"
        },
        "Dell Inc.": {
            "models": {
                "PowerEdge R530": "887b2d711b025d10f219866fdc4bcbc2",
                "PowerEdge R740xd": "447b2d711b025d10f219866fdc4bcbc3"
            },
            "sys_id": "16fa349e1bf999106af3dac4bd4bcb84"
        }
    }
}
```

Manufacturer and Model

Business Services

HTTP Method

GET

Resource Path

```
/api/x_sclo_scilogic/v2/sciencelogic/business_service
```

Pagination

Enabled

Overview

This operation pulls all the fields from just the Business Service (**cmdb_ci_service**) table. The return is ordered by the **sys_id** of the record in the **sys_object_source** table. This operation requires the region to be supplied by the requester, and it will only return region-supplied configuration items.

Headers	
Key	*Value*
Content-Type	application/json
Accept	application/json

Parameters	
Key	*Value*
region (required)	ScienceLogic
sysparm_offset	0
sysparm_limit	glide.json.export.limit, glide.ui.export.limit, glide.ui.export.war.threshold

Example (Request URL)

```
https://<your Instance>.service-now.com/api/x_sclo_
scilogic/v2/sciencelogic/business_service?region=ven70_sl151_pf199_
cmdb&action=relationships&sysparm_limit=1
```

Example (Response)

```
{
 "results": [
     {
         "operational_status": "1",
         "operational_status_label": "Operational",
         "consumer_type": "internal",
         "consumer_type_label": "Internal",
         "sys_updated_on": "2022-09-23 20:17:00",
         "number": "BSN0001552",
         "discovery_source": "Other Automated",
         "discovery_source_label": "Other Automated",
         "first_discovered": "2022-09-13 18:20:51",
         "used_for": "Production",
         "used_for_label": "Production",
         "state": "published",
         "state_label": "Published",
         "sys_created_by": "powerflow_user_cmdb_incident",
         "sys_domain_path": "/",
         "service_status": "requirements",
         "service_status_label": "Requirements",
         "portfolio_status": "pipeline",
         "portfolio_status_label": "Pipeline",
         "busines_criticality": "2 - somewhat critical",
         "busines_criticality_label": "2 - somewhat critical",
         "last_discovered": "2022-09-23 20:17:00",
         "sys_class_name": "cmdb_ci_service_technical",
         "sys_class_name_label": "Technical Service",
         "x_sclo_scilogic_monitored": "true",
         "correlation_id": "cl80ioa534zga9ov0b4wx6l15",
         "x_sclo_scilogic_region": "ven70_sl151_pf199_cmdb",
         "sys_updated_by": "powerflow_user_cmdb_incident",
         "sys_created_on": "2022-09-13 18:20:51",
         "sys_domain": "global",
         "x_sclo_scilogic_url": "https://<powerflow_instance>/in-
ventory/services/cl80ioa534zga9ov0b4wx6l15/overview",
         "install_status": "1",
         "install_status_label": "Installed",
         "name": "It test",
```

```
        "price_model": "per_unit",
        "price_model_label": "Per Unit",
        "sys_id": "1f5d0de31bb19110f219866fdc4bcbdd",
        "sys_class_path": "/!!/#C/!,",
        "company": "4d6f7f821bb159100efb206cbc4bcb65",
        "company_label": "System",
        "checkout": "true",
        "checkout_label": "True",
        "attestation_status": "Not Yet Reviewed",
        "attestation_status_label": "Not Yet Reviewed",
        "sys_mod_count": "2",
        "cost_cc": "USD",
        "cost_cc_label": "USD",
        "service_classification": "Technical Service",
        "service_classification_label": "Technical Service",
        "object_source_id": "ven70_sl151_pf199_cmd-
b+BIZITSRV+cl80ioa534zga9ov0b4wx6l15",
        "child_devices": [
              {
                  "sys_id": "575d0de31bb19110f219866fdc4bcbdf",
                  "name": "Dynamic CI Group: ds test",
                  "class": "cmdb_ci_query_based_service",
                  "id": "cl80iomgd4zol9iv0e0mu6pjb",
                  "region": "ven70_sl151_pf199_cmdb"
              }
          ]
      }
  ],
 "sysparm_offset": 0,
 "sysparm_limit": 1,
 "return_count": 1
}
```

Installed Software

HTTP Method

GET

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/installed_software
```

Pagination

Enabled

Overview

This operation pulls all the fields from the software (**cmdb_ci_spkg**) table. The return is ordered by **sys_id**, so the results display in the same order every time. The results are filtered by the *SL1 monitored* field on the ServiceNow side. This operation requires the *region* to filter the installed software on devices.

Headers	
Key	*Value*
Content-Type	application/json
Accept	application/json

Parameters	
Key	*Value*
region	ScienceLogic
sysparm_offset	0
sysparm_limit	glide.json.export.limit, glide.ui.export.limit, glide.ui.export.war.threshold

HTTP Status	
Code	*Value*
200	OK
400	Query parameter \'region\' is not defined and are required.

Fixed Internal Query

'x_sclo_scilogic_monitored=true'

Example (Request URL)

```
https://<your Instance>.service-now.com/api/x_sclo_
scilogic/v1/sciencelogic/installed_software?sysparm_offset=0&sysparm_
limit=100&region=ScienceLogic
```

Example (Response)

```
{
"results": [
{
"operational_status": "1",
"operational_status_label": "Operational",
"sys_updated_on": "2019-05-01 06:00:09",
"install_count": "2",
"sys_updated_by": "system",
"sys_created_on": "2019-03-29 19:42:58",
"sys_domain": "global",
"sys_created_by": "admin",
"sys_domain_path": "/",
"install_status": "1",
"install_status_label": "Installed",
"name": "Test_31",
"subcategory": "Package",
"sys_class_name": "cmdb_ci_spkg",
"sys_class_name_label": "Software",
"sys_id": "1e9608fcdb2cb740dc44f00fbf961949",
"sys_class_path": "/!!/#$",
"key": "Test_31_:::_NULL",
"license_available": "-2",
"sys_mod_count": "1",
"x_sclo_scilogic_id": "31",
"model_id": "2c146728dbe8b740dc44f00fbf9619c6",
"model_id_label": "Unknown",
"cost_cc": "USD",
"cost_cc_label": "USD",
"x_sclo_scilogic_monitored": "true",
"package_name": "Test_31",
```

```
"category": "Software",
"x_sclo_scilogic_region": "AutoGenerateClass",
"installed_on": [
{
"sys_id": "5a271407dbfe6300dc44f00fbf96190f",
"id": "10",
"region": "ScienceLogic",
"monitored": "true"
},
{
"sys_id": "5a271407dbfe6300dc44f00fbf96190f",
"id": "10",
"region": "ScienceLogic",
"monitored": "true"
}
]
}
],
"sysparm_offset": 0,
"sysparm_limit": 100,
"return_count": 4,
"total_count": 4
}
```

Import Set

HTTP Method

POST

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/import_set
```

Overview

This operation handles the custom intake of import sets before it reaches the transform map staging table, such as **x_sclo_scilogic_import_installed_software**. This operation is currently only used for importing installed software (x_**sclo_scilogic_import_installed_software**).

Headers	
Key	*Value*
Accept	application/json
Content-Type	application/json

Parameters	
Key	*Value*
record_type (required)	x_sclo_scilogic_import_installed_software

Example (Request URL)

```
https://<your Instance>.service-now.com/api/x_sclo_
scilogic/v1/sciencelogic/import_set
```

Example (Body)

```
[
  {
  "records": [
    {
      "name": "acl-2.2.51-12.el7",
      "software": "671bafd8dba13700dc44f00fbf961953",
      "cmdb_ci": [
        "ff01a81edb1df300dc44f00fbf961947",
        "4011a81edb1df300dc44f00fbf961958",
```

```
            "f301a81edb1df300dc44f00fbf96193d",
            "7b01a81edb1df300dc44f00fbf961942",
            "c411a81edb1df300dc44f00fbf96195d",
            "7701a81edb1df300dc44f00fbf961922",
            "7b01681edb1df300dc44f00fbf9619e7",
            "fb01a81edb1df300dc44f00fbf961927"
        ],
        "active": true
    }
    ]
  }
]
```

Import Set

Discovery Dependents

HTTP Method

GET

Resource Path

`/api/x_sclo_scilogic/v1/sciencelogic/discovery_dependent`

Pagination

Enabled

Overview

This operation pulls all Discovery-dependent records that are tied to the **region** value, which is used for the catalog request process. Based on the request type, this operation returns a formatted JSON object.

Headers	
Key	*Value*
Content-Type	application/json
Accept	application/json

Parameters	
Key	*Value*
region (required)	ScienceLogic
sysparm_offset	0
sysparm_limit	glide.json.export.limit, glide.ui.export.limit, glide.ui.export.war.threshold

HTTP Status	
Code	*Value*
200	OK
400	Query parameter \'region\' is not defined and is required.

Fixed Internal Query

Region Specific: 'region=' + region

Example

```
https://<your instance>.service-now.com/api/x_sclo_
scilogic/v1/sciencelogic/discovery_dependent?region=del_test&sysparm_
offset=0&sysparm_limit=100
```

Example (Response)

```
{
    "results": [
        {
            "sys_updated_on": "2022-02-25 18:50:23",
            "type": "device_template",
            "type_label": "Device Template",
            "sys_id": "02261f071b3d0950f219866fdc4bcbe6",
            "sys_updated_by": "powerflow_user_cmdb_incident",
            "sys_created_on": "2022-02-25 18:50:23",
            "name": "SL1 Agent for Microsoft: Windows Server Template",
            "id": "5",
            "region": "unique_id",
            "sys_created_by": "powerflow_user_cmdb_incident"
        }
    ],
    "sysparm_offset": 0,
    "sysparm_limit": 1,
    "return_count": 1,
    "total_count": 168
}
```

Appendix

D

ServiceNow Registered Events

Overview

This appendix describes the commands and data you can use to generate registered events in ServiceNow that are queued to ServiceNow Event Management. These events can trigger actions in PowerFlow, such as specifying one or more CIs for monitoring, or putting a CI into maintenance.

> **WARNING:** This appendix is recommended for advanced ServiceNow administrators.

These events use the `gs.eventQueue` command, using the following format:

```
eventQueue(String name, Object instance, String parm1, String parm2)
```

You can use examples found in the "ScienceLogic ServiceNow Integration (Catalog UI)" update set in ServiceNow to help you customize the `gs.eventQueue` command to specify which ServiceNow events can trigger PowerFlow actions. You will need to install this update set in ServiceNow.

> **NOTE:** Ask your ScienceLogic contact for access to this update set.

This chapter covers the following topics:

Catalog Item Events

The following events are available through the "ScienceLogic ServiceNow Integration (Catalog UI)" update set in ServiceNow.

x_sclo_scilogic.device_monitoring

This event takes the selected Configuration Items in ServiceNow, files a catalog request using the template selected by the user, and submits the catalog request.

Trigger

Custom requirement supplied by ScienceLogic implementation or the Customer directly.

Command

```
gs.eventQueue('x_sclo_scilogic.device_monitoring', region, ip_
list.toString(), region.getUniqueValue() + "," + region.x_sclo_scilogic_
region + "," + silo_template);
```

Event Fields

Field	Description
`x_sclo_scilogic.device_monitoring`	Unique name of the event.
`region`	The table to which the event applies.
`ip_list.toString()`	Parm1: The IP, or a comma-separated list of IP addresses, that is pulled from the `ip_address` field on the **cmdb_ci** table.
`getCompany.getUniqueValue(), silo_template`	Parm2: List of three requirements that the `sys_id` of the company associated with the Configuration Item and the catalog template selected through the user interface action.

Example

The UI action / UI page is available through the "ScienceLogic ServiceNow Integration (Catalog UI Action)" update set.

x_sclo_scilogic.remove_monitoring

This action takes the selected Configuration Item or Items and submits a request through the ServiceNow service catalog for each Configuration Item.

Trigger

Custom requirement supplied by ScienceLogic implementation or the Customer directly.

Command

```
gs.eventQueue('x_sclo_scilogic.remove_monitoring',current,
current.getUniqueValue(), current.company);
```

Event Fields

Field	Description
`x_sclo_scilogic.remove_monitoring`	Unique name of the event.
`current`	The table to which the event applies.
`current.getUniqueValue()`	Parm1: The `sys_id` of the Configuration Item that needs to be removed
`current.company);`	Parm2: The `sys_id` of the company that is associated with the Configuration Item.

Example

The UI action / UI page is available through the "ScienceLogic ServiceNow Integration (Catalog UI Action)" update set.

ScienceLogic