ServiceNow Change Management Synchronization PowerPack

Version 3.1.2

Table of Contents

Chapter 1

Introduction to the ServiceNow Change Management Synchronization PowerPack

Overview

This chapter describes the *ServiceNow Change Management* Synchronization PowerPack, which is the ScienceLogic integration with the ServiceNow Change Management Module. You can use this Synchronization PowerPack to place SL1 devices into and out of maintenance mode.

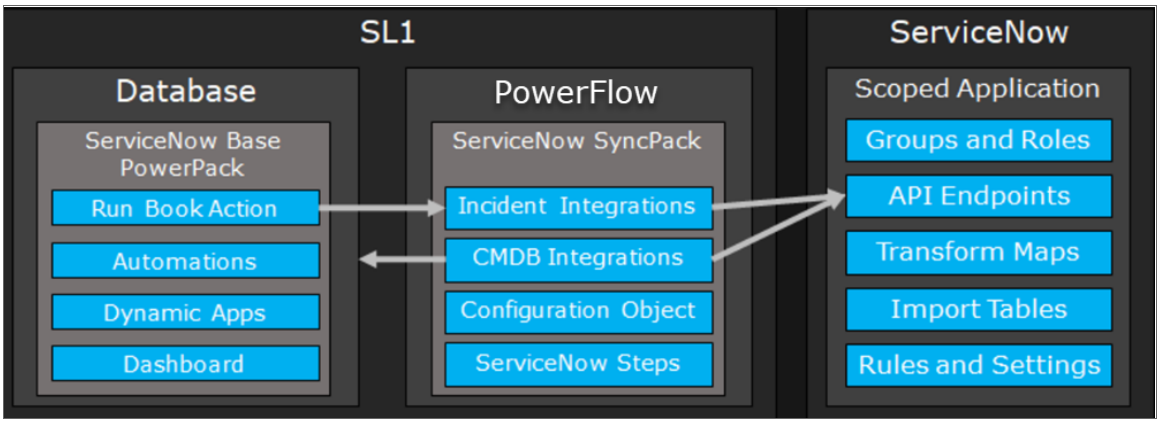
NOTE: After the 2.1.0 platform release, the *Integration Service* was rebranded as *SL1 PowerFlow*, and the *Automation Builder* was rebranded as *SL1 PowerFlow builder*.

NOTE: The label "SyncPack" is used in place of "Synchronization PowerPack" in the PowerFlow user interface.

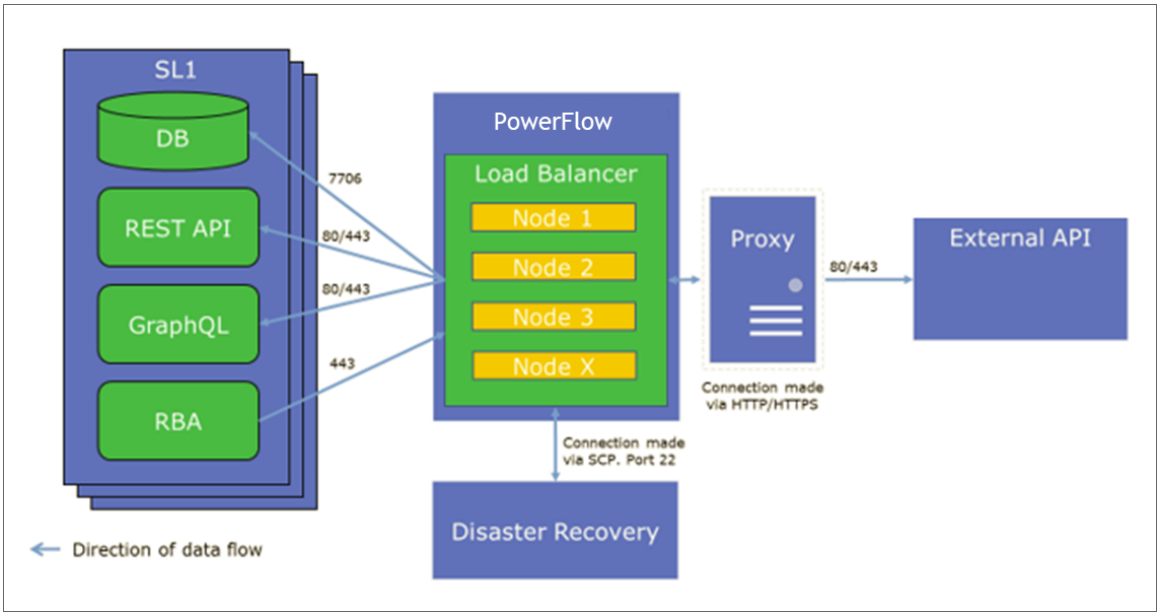
This chapter covers the following topics:

Architecture Overview for ServiceNow Synchronization PowerPacks

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:



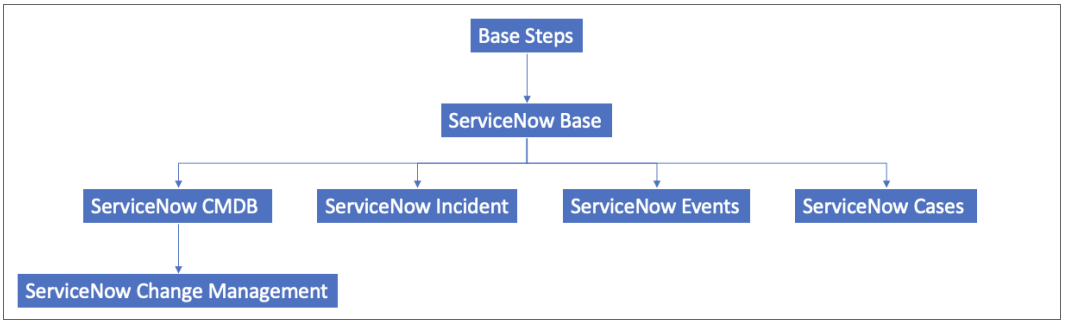
SL1 and ServiceNow Terminology

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 Synchronization PowerPack 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 Synchronization PowerPacks

The following graphic describes which Synchronization PowerPacks depend on other Synchronization PowerPacks:



TIP: For more information about the Standard *Base Steps* Synchronization PowerPack, see the ***SL1 PowerFlow Platform*** manual.

Prerequisites for ServiceNow Synchronization PowerPacks

This section describes the prerequisites for the ServiceNow Synchronization PowerPacks. For more information about the specific software versions required by a ServiceNow Synchronization PowerPack, see the release notes for that Synchronization PowerPack.

To install any of the ScienceLogic ServiceNow Synchronization PowerPacks, you must have administrator access to both SL1 and ServiceNow. Specifically, you will need:

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

Additional information related to uploading, installing, and upgrading the ServiceNow Synchronization PowerPacks:

- If you want to upload and install multiple ServiceNow Synchronization PowerPacks at the same time, you should upload *all* of the Synchronization PowerPacks first, and then install them to address any dependencies between the Synchronization PowerPacks.
- If you are upgrading from version 1.8.4 of PowerFlow (formerly called the "Integration Service"), you can first move to version 2.5.0 of the *ServiceNow* Synchronization PowerPack, then upgrade to version 2.0 of the PowerFlow platform, and finally move to version 3.0.0 or later of the relevant ServiceNow Synchronization PowerPacks.
- If you made customizations to version 2.5.0 or earlier of the *ServiceNow* Synchronization PowerPack, you must make a copy of that Synchronization PowerPack and save it as a new Synchronization PowerPack to keep your customizations. For more information, see the ***SL1 PowerFlow for Developers*** manual.
- If you are starting out with version 2.0.0 of the PowerFlow platform, you should skip version 2.5.0 of the *ServiceNow* Synchronization PowerPack and instead install version 3.0.0 of the relevant ServiceNow Synchronization PowerPacks.
- There is no backwards compatibility after you upgrade a ServiceNow Synchronization PowerPack from version 2.5.0 to 3.0.0.

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

Source IP	PowerFlow Destination	PowerFlow Source Port	Destination Port	Requirement
PowerFlow	SL1 API	Any	TCP 443	SL1 API Access
PowerFlow	ServiceNow API	Any	TCP 443	ServiceNow API Access

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

Contents of the Synchronization PowerPack

This section lists the contents of the *ServiceNow Change Management* Synchronization PowerPack.

PowerFlow Applications

The following PowerFlow applications are included with the *ServiceNow Change Management* Synchronization PowerPack:

- ***Create or Update Maintenance Schedule from ServiceNow Trigger***. Handles scheduling and canceling maintenance schedules in SL1 using a trigger from a ServiceNow Management, Instrumentation, and Discovery (MID) Server. This application requires SL1 version 8.14.0 or later. For more information, see *Creating or Updating Device Maintenance from a ServiceNow Trigger*.
- ***Sync Change Management Requirements***. Sends configuration data from the PowerFlow to ServiceNow to use with change management and other processes. For more information, see *Directly Enabling or Disabling Device Maintenance from ServiceNow*.
- ***Sync Maintenance Schedules from ServiceNow to SL1***. Performs maintenance of synced devices in ServiceNow and SL1. If you are scheduling this application, ScienceLogic recommends that you schedule it to run every hour or so, depending on your environment. For more information, see *Scheduling Device Maintenance*.
- ***Trigger Device Maintenance Updates via MID Server***. Lets you sync maintenance windows from ServiceNow to SL1 using a ServiceNow Management, Instrumentation, and Discovery (MID) Server. For more information, see *Directly Enabling or Disabling Device Maintenance from ServiceNow*.

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:

- ***Cancel Maintenance***. Cancels a scheduled maintenance in SL1.
- ***Create Maintenance***. Creates a scheduled maintenance in SL1.
- ***Modify Maintenance***. Updates a scheduled maintenance in SL1.
- ***Process Create Schedule Requests***. Schedules and posts maintenance tasks in ServiceNow.
- ***Remove Maintenance***. Removes a scheduled maintenance in SL1.
- ***Schedule Maintenance***. Creates a scheduled maintenance in SL1.

Chapter 2

Installing and Configuring the Change Management Synchronization PowerPack

Overview

This chapter describes how to install the *ServiceNow Change Management* Synchronization PowerPack.

This chapter covers the following topics:

Downloading the Synchronization PowerPack

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

WARNING: If you are *upgrading* to this version of the Synchronization PowerPack from a previous version, make a note of any settings you made on the **Configuration** pane of the various PowerFlow applications in this Synchronization PowerPack, as these settings are *not* retained when you upgrade.

To locate and download the Synchronization PowerPack:

1. Go to the ScienceLogic Support Site.
2. Click the **[Product Downloads]** tab and select *PowerPack*.
3. In the ***Search PowerPacks*** field, search for the Synchronization PowerPack and select it from the search results. The **Release Version** page appears.
4. On the **[PowerPack Versions]** tab, click the name of the Synchronization PowerPack version that you want to install. The **Release File Details** page appears.
5. Click the **[Download File]** button or click the name of the **.zip** file containing the **.whl** file for this Synchronization PowerPack to start downloading the file.

NOTE: Synchronization PowerPacks do not require a specific license. After you download a Synchronization PowerPack, you can import it to your PowerFlow system using the PowerFlow user interface.

NOTE: If you are installing or upgrading to the latest version of this Synchronization PowerPack in an offline deployment, see "Installing or Upgrading in an Offline Environment" in the Synchronization PowerPack release notes to ensure you install any external dependencies.

Importing the Synchronization PowerPack

To import a Synchronization PowerPack 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 Synchronization PowerPack you want to install.

TIP: You can also drag and drop a **.whl** file to the **SyncPacks** page.

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

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

Installing the Synchronization PowerPack

> **NOTE:** You must import and install the *ServiceNow Base* Synchronization PowerPack before uploading and installing any of the other ServiceNow Synchronization PowerPacks.

To install a Synchronization PowerPack in the PowerFlow user interface:

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

   > **TIP:** By default, the **SyncPacks** page displays only activated and installed PowerPacks. If you do not see the PowerPack that you want to install, click the Filter icon (⚌) on the **SyncPacks** page and select *Show All SyncPacks* to see a list of the uninstalled PowerPacks.

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

   > **TIP:** While the Synchronization PowerPack is installing, you cannot click any of the options that appear when you click the **[Actions]** button (⋮).

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 Synchronization PowerPack. 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 Synchronization PowerPack on your PowerFlow system, you can click the **[Actions]** button (⋮) for that Synchronization PowerPack and select *Change active version* to activate a different version other than the version that is currently running.

TIP: The most common error that occurs when installing a Synchronization PowerPack is that the Synchronization PowerPack dependencies are not installed. Review the "System Requirements" section of the release notes for that Synchronization PowerPack to ensure that you have installed all of the required applications for that Synchronization PowerPack. To view a list of additional PowerFlow and SL1 products that are required by the various Synchronization PowerPacks, see the SL1 PowerFlow Dependency Matrix.

Allowing Cross-Scoped Access in ServiceNow

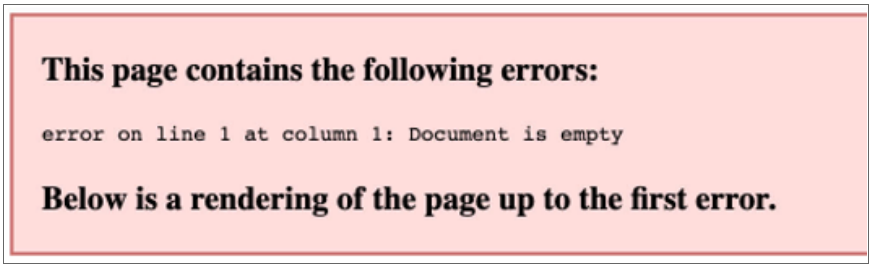
When using custom tables that are extended from the **cmdb_ci** table, you must configure cross-scope access for any custom tables created outside of the base ServiceNow deployment.

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.ScopeAccessNotGrantedException: 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:

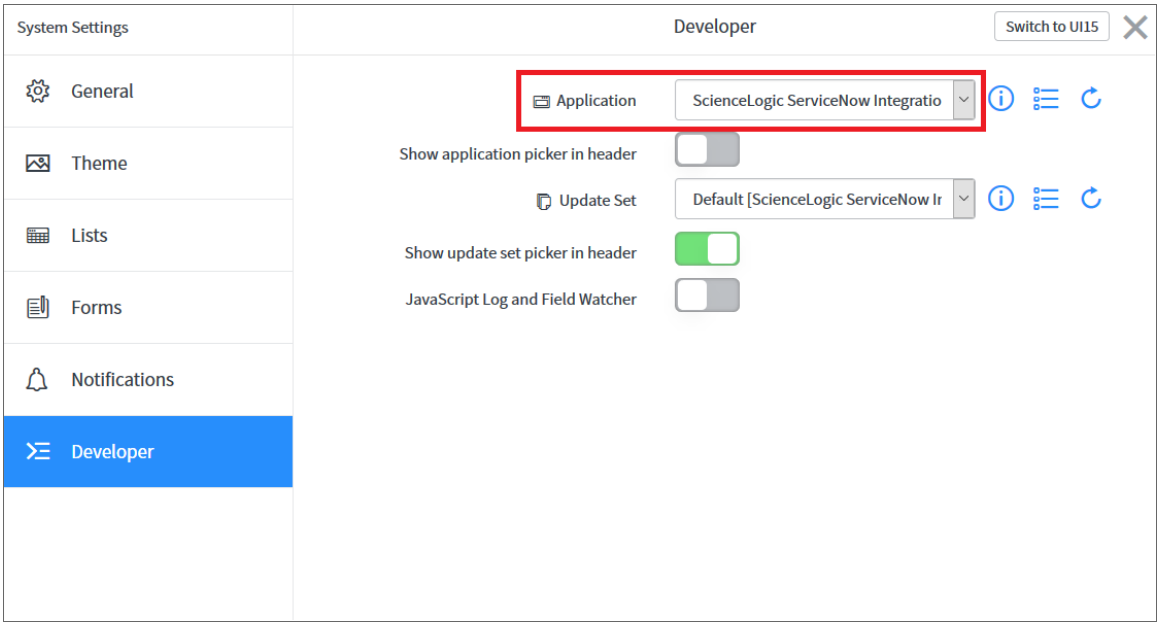


In this example, the table requires that you grant access to the ScienceLogic Scope to allow the API call to run correctly. In the above example, the target table is **u_test_hardware**.

NOTE: A ServiceNow account with System Administrator is required.

To grant access to the ScienceLogic Scope in ServiceNow:

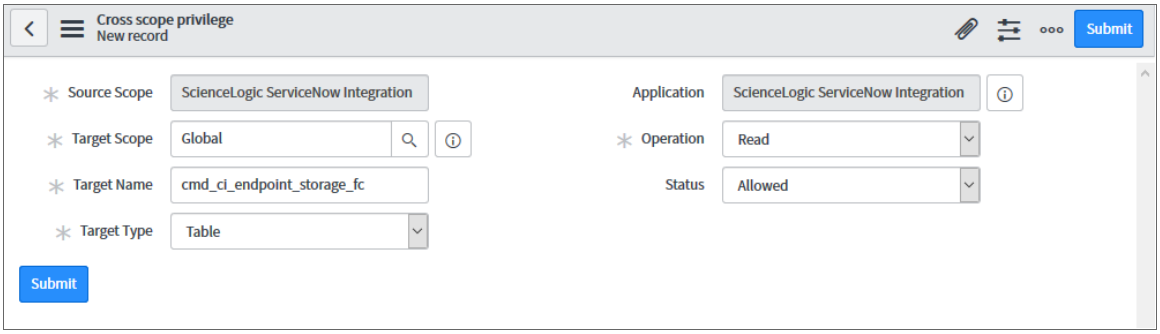
1. Log in to your ServiceNow instance.
2. Click the **Settings** icon ( ) and select the **Developer** tab. The **Developer System Settings** window appears:



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).

> **NOTE:** 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*.
    - ***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.

Chapter 3

Configuring Applications for the Change Management Synchronization PowerPack

Overview

This chapter describes how to configure PowerFlow application for the *ServiceNow Change Management* Synchronization PowerPack. You can use this Synchronization PowerPack to place SL1 devices into and out of maintenance mode.

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 the *ServiceNow CMDB* Synchronization PowerPack, 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 Synchronization PowerPacks.

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 Synchronization PowerPack for the purpose of distinguishing between domains is not supported, and will result in issues on upgrades.

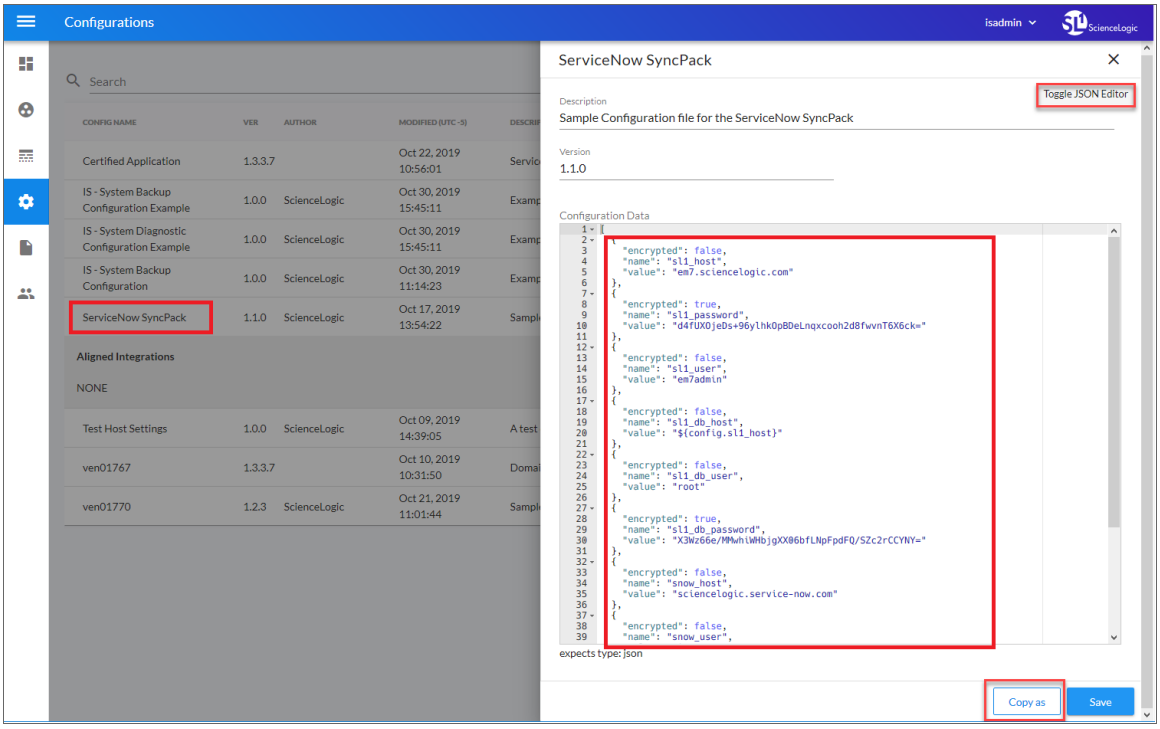
Creating a Configuration Object

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

TIP: The "ServiceNow SyncPack" configuration object contains all of the required variables. Simply 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. Click the **[Actions]** button ( ) for the "ServiceNow SyncPack" configuration object 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.
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"
}
```

TIP: Click **[Toggle JSON Editor]** to show the JSON code. Click the button again to see the fields. You can also click **[Add Value]** and add a new name-value pair in the **Configuration Data Values** section.

NOTE: If you are using SL1 with an External Database (SL1 Extended architecture or a cloud-based architecture), update the `"value"` of that block of code to be the host of your database. This field accepts IP addresses. For example: `"value": "db.sciencelogic.com"`. If you are *not* using the SL1 Extended architecture or a cloud-based architecture, you do not need to make any changes to the block of code other than pasting the code into the configuration object.

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.

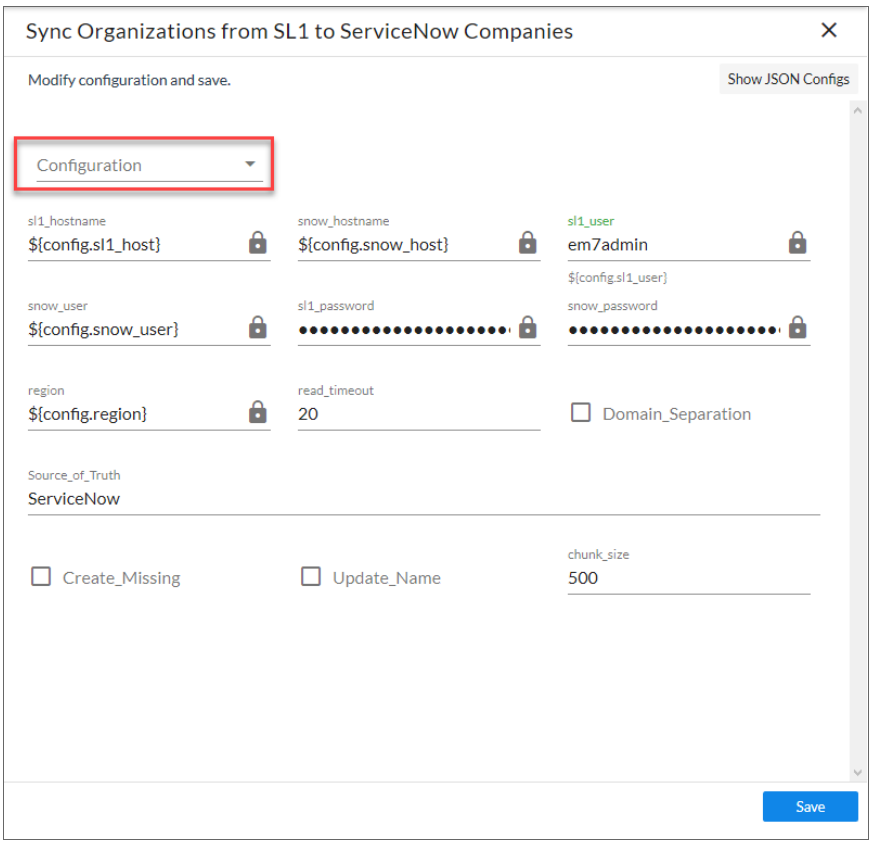
8. Click **[Save]**. You can now align this configuration object with one or more applications.

Aligning a Configuration Object

Before you can run the applications in this Synchronization PowerPack, 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. Wait for the "App & Config modifications saved" pop-up message to appear. The **Configuration** automatically closes after this message appears.
5. Repeat this process for every other application you want to use.

NOTE: The values for **eventDetails** and the other parameters that appear in the **Configuration** pane with a padlock icon ( ) are populated by the configuration object. Do not modify these values.

Syncing Device Maintenance from ServiceNow to SL1

You can use the following methods to put one or more devices into maintenance mode from ServiceNow to SL1:

- Use the "Sync Maintenance Schedules from ServiceNow to SL1" PowerFlow application if you want to perform *scheduled* maintenance on a set of devices in SL1. For more information, see ***Scheduling Device Maintenance***.
- Use the "Trigger Device Maintenance Updates via MID Server" application if you want to *immediately* enable or disable maintenance on a device. For more information, see ***Directly Enabling or Disabling Device Maintenance from ServiceNow***.
- Use the "Create or Update Maintenance Schedule from ServiceNow Trigger" application to schedule and cancel maintenance schedules in SL1 from a ServiceNow MID Server trigger. For more information, see ***Creating or Updating Device Maintenance from a ServiceNow Trigger***.

Scheduling Device Maintenance

You create a change request to perform *scheduled* maintenance through a maintenance window in ServiceNow.

WARNING: PowerFlow only syncs maintenance schedules that are aligned with devices that are already synced with ServiceNow. Before setting up maintenance schedule sync, you must first sync devices between SL1 and ServiceNow.

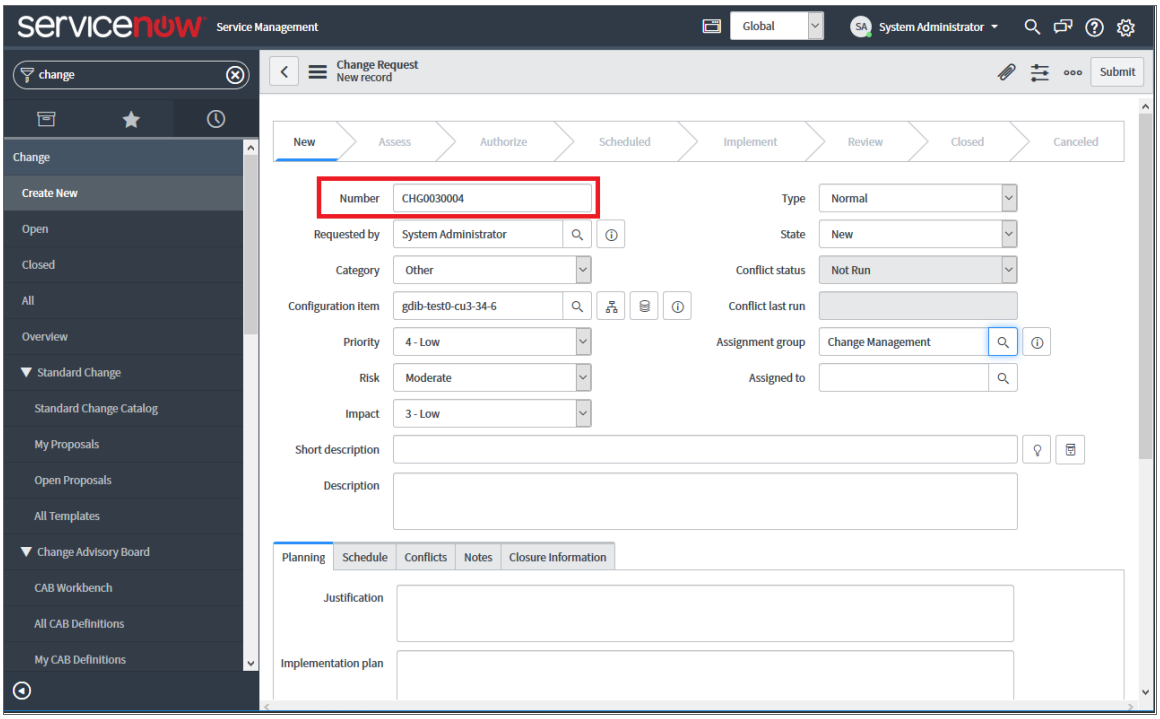
The CI Maintenance Sync process (non-scheduled) syncs maintenance windows from ServiceNow change requests (CHG)s to SL1 devices to place the synced devices into maintenance mode for the scheduled change window.

NOTE: If you update the scheduled times in a ServiceNow change request, you will need to cancel the change request, which also cancels the maintenance window in SL1. You will then need to create a new change request with the new time window and sync that change request to SL1.

NOTE: The SL1 Scheduler supports maintenance windows of at least one minute or more.

To set up maintenance sync:

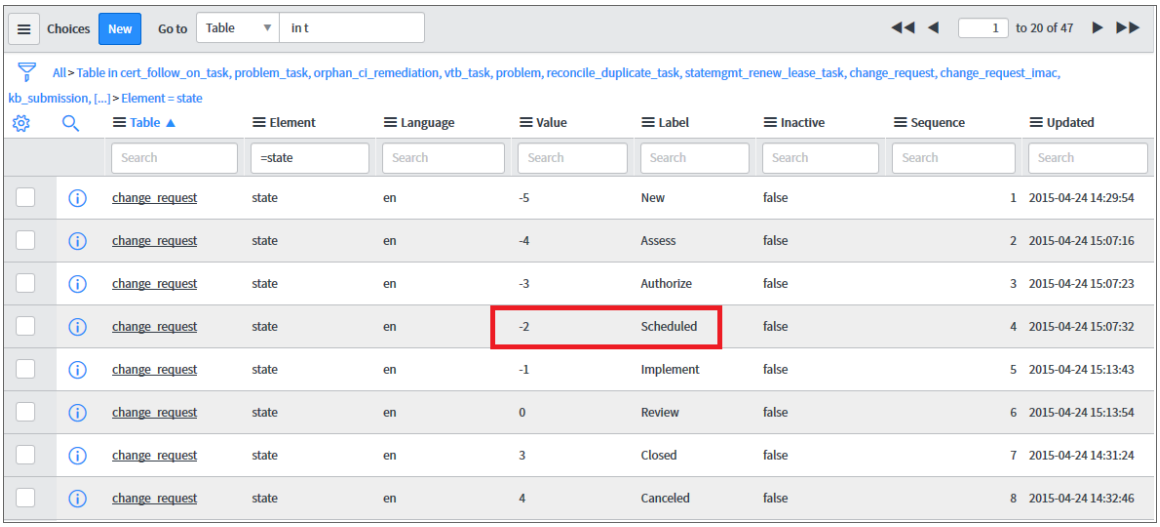
1. In ServiceNow, type "change" in the filter navigator and navigate to **Change > Create New**.
2. Click **[New]** to create a new change request of type "Normal". A new Change Request record appears:



3. Make a note of the change request number in the **Number** field. You will use this later to verify that the maintenance sync was created. In this example, the value is *CHG0030004*.
4. Update the following fields in the record:
   - ***Configuration Item***. Select the CI you want to configure for maintenance sync.
   - ***Assignment group***. Select the group for the CI.

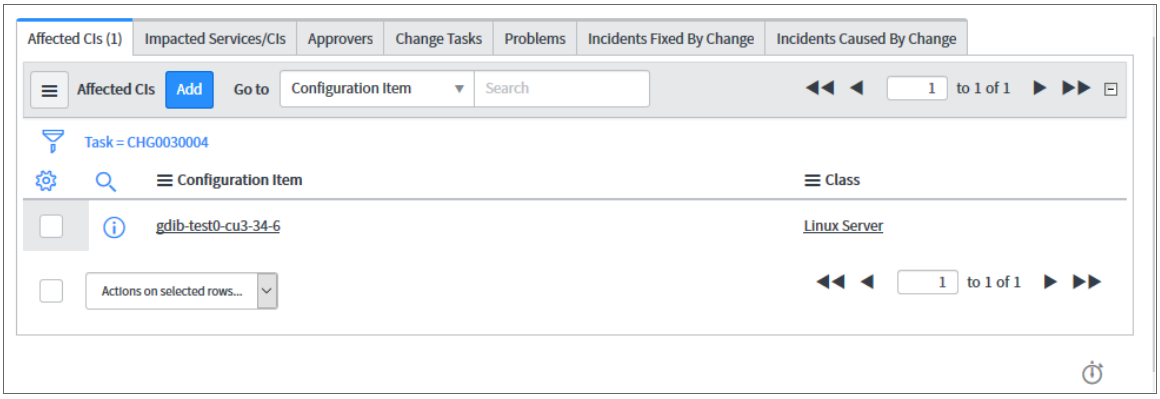
> **NOTE:** The aligned CI must have the ***SL1 Monitored*** field selected before the PowerFlow can use the maintenance schedule for that CI.

5. Click the **[Submit]** button. The change request is saved, and you are returned to the **Change Requests** page.
6. Select the change request you just created, and in the change request record, right-click the **State** label and select *Show Choice List*. The Choices list displays a list of the configurable choices and values:

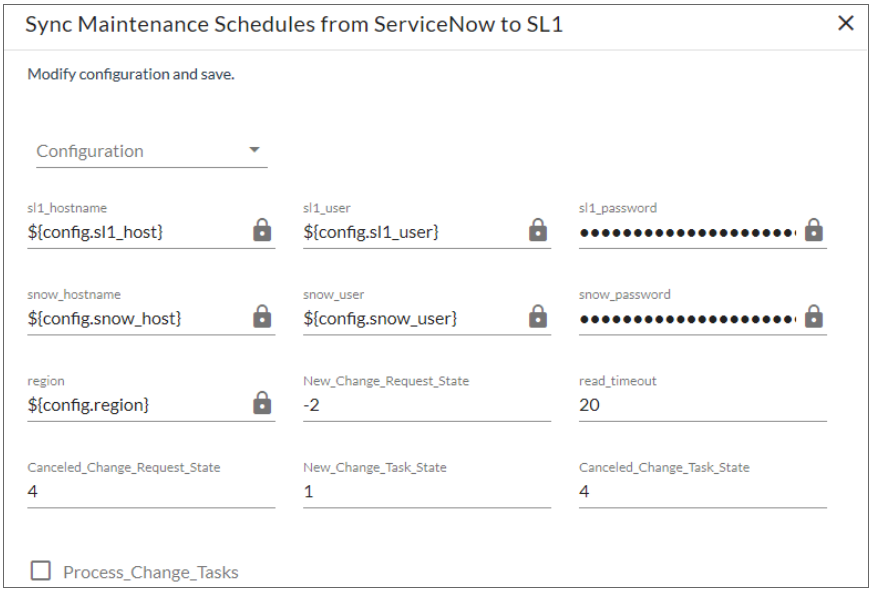


> **NOTE**: You need Administrator privileges to access this list.

7. Make a note of the values in the ***Value*** and ***Label*** fields. These values map to the ***New_Change_Request_State*** and ***Canceled_Change_Request_State*** fields in the "Sync Maintenance Schedules from ServiceNow to SL1" application.
8. Return to your new change request and scroll down in the change request to the **[Affected CIs]** tab, where you can click the **[Add]** button to add additional synced CIs to the maintenance sync:

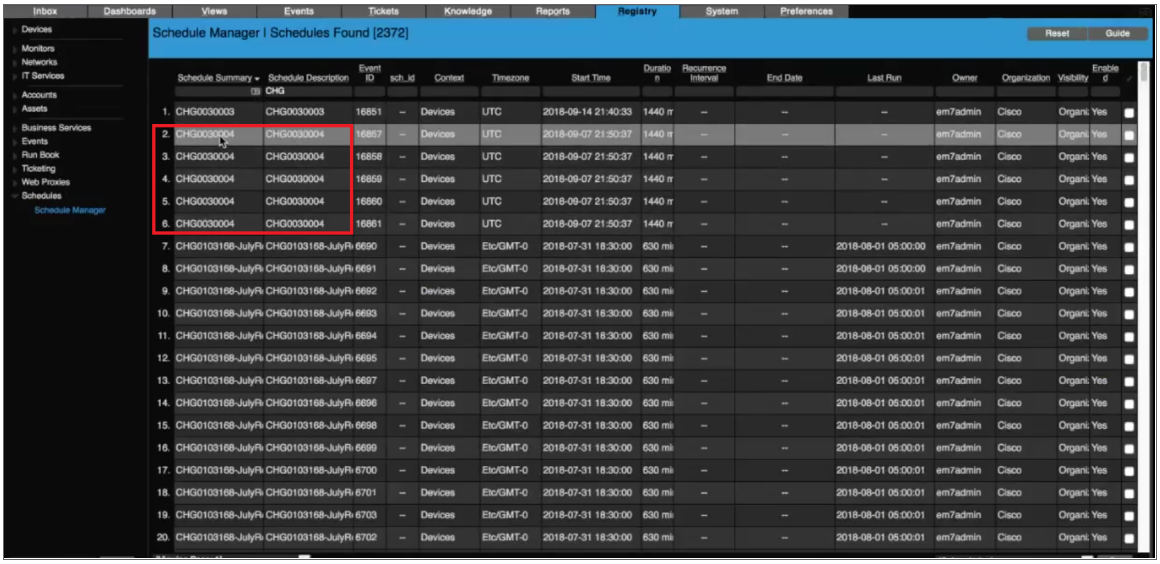


9. In the PowerFlow user interface, go to the **Applications** page and select the "Sync Maintenance Schedules from ServiceNow to SL1" application.
10. Click **[Configure]** ( ). The **Configuration** pane appears:



11. As needed, update 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.
    - ***New_Change_Request_State***. The State ID from ServiceNow of the scheduled change request that this application accesses to pull to schedule maintenance windows in SL1.The default is -2.
    - ***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.
    - ***Canceled_Change_Request_State***: The State ID for a canceled change request that this application accesses to pull to schedule maintenance windows in SL1. The default is 4.
    - ***New_Change_Task_State***: The State ID of the scheduled change task that this application accesses to pull to cancel maintenance windows in SL1. The default is 1.
    - ***Canceled_Change_Task_State***: The State ID for a canceled change task that this application accesses to pull to cancel maintenance windows in SL1. The default is 4.
    - ***Process_Change_Tasks***: Select this option to enable change task processing, which allows the request to be broken down into Change Tasks, which results in two extra pulls of the data. The default is unselected.

12. Verify that the value from the ***New_Change_Request_State*** field matches the value in the ***Value*** field from ServiceNow, and the value from the ***Canceled_Change_Request_State*** field matches the value from the ***Label*** field from ServiceNow. These values must match for the maintenance sync to work.
13. Click **[Save]** and wait for the "App & Config modifications saved" pop-up message to appear. The **Configuration** pane automatically closes after this message appears.
14. Click **[Run]** ( ) to run the application.
15. While the "Sync Maintenance from ServiceNow to SL1" application runs, you can monitor the status of the maintenance process by clicking the branch icon ( ) on the "Schedule Maintenance" step. Click the triggered application's run ID in the pop-up window, and then click the branch icon on the "Create SL Maintenance" or "Modify Maintenance" steps for more information.
16. After the "Sync Maintenance from ServiceNow to SL1" application completes, navigate to the **Schedule Manager** (Registry > Schedules > Schedule Manager) in SL1 to view the change requests.



17. Verify that the **Schedule Summary** field contains the same value from the ServiceNow **Number** field. In this example, the value in SL1 matches the value from ServiceNow: *CHG0030004*.
18. You can also verify that the schedule was created for a device by navigating to the **Device Manager** (Registry > Devices), clicking the wrench icon for the device, and clicking the **[Schedule]** tab.
19. If you want to cancel the scheduled time for the maintenance sync, open the change request in ServiceNow, click the **Additional actions** menu button ( ), and select *Cancel Change*. The next time the "Sync Maintenance from ServiceNow to SL1" application runs, the application cancels that maintenance sync.

TIP: As a best practice, schedule the "Sync Maintenance from ServiceNow to SL1" application to run every hour or so, depending on your environment.

Directly Enabling or Disabling Device Maintenance from ServiceNow

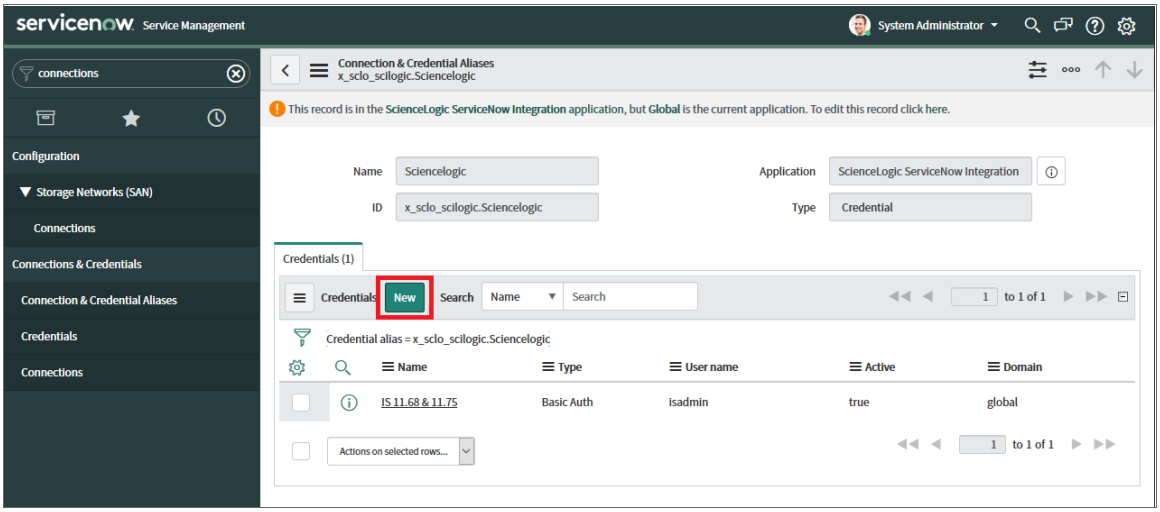
The "Trigger Device Maintenance Updates via MID Server" application receives a group of one or more synced devices from the ServiceNow Management, Instrumentation, and Discovery (MID) Server and checks for the *enable_maintenance* and *disable_maintenance* actions on those devices. If the application encounters devices with one of those actions, it will enable or disable the user maintenance status of those devices as needed.

You do *not* need to update any fields on the **Configuration** pane for the "Trigger Device Maintenance Updates via MID Server" application. You also do *not* need to run this application, as it is triggered by the MID Server, which is triggered first by a registered event in ServiceNow Event Management. For more information about registered events, including examples of other triggering events you can define in ServiceNow, see the *ServiceNow Registered Events* appendix.

Before you can set up maintenance sync with the MID Server, you need to create a credential for the MID Server. You should have access to the "Integration Services" section of the **Discovery Dependents** page in ServiceNow.

To create a credential to connect to PowerFlow:

1. In ServiceNow, go to **Connections & Credentials** > **Connection & Credential Aliases**.
2. From the **Connection & Credential Aliases** list, select **ScienceLogic**. This record is provided by the Integration Service Certified Application. The **Connection & Credential Aliases page** appears:



3. Click **[New]** to create a new credential. The **Credentials** page appears.
4. From the list of credentials, select **Basic Auth Credentials**. This is currently the only type of credential that is supported. The **Basic Auth Credentials** page appears.
5. Complete the fields related to the PowerFlow on the **Basic Auth Credentials** page. Make sure that the ***Credential alias*** field is set to *x_sclo_scilogic.Sciencelogic*.
5. Click **[Submit]**. The credential is added to the **Connection & Credential Aliases** page.
6. Select the new credential. The **Basic Auth Credentials** page for that credential appears.

7. Click **[New]** to create a Discovery IP Affinity record. A new **Credential Affinity** page appears.
8. Complete the following fields:
    - ***MID server***. The name of the Mid Server you want to use.
    - ***IP address***. Use the PowerFlow IP address that was listed in the ***IP*** field on the relevant record on the **Discovery Dependents** page in ServiceNow. To quickly find the relevant record on the **Discovery Dependents** page, right-click the **Type** column and select *Group By Type*, and then expand *Type: Integration Services*. Use the IP value from the record that matches the Region for the devices you want to use.
    - ***Credential ID***. This field should be completed for you.
9. Click **[Submit]**.

Creating or Updating Device Maintenance from a ServiceNow Trigger

The "Create or Update Maintenance Schedule from ServiceNow Trigger" application handles scheduling and canceling maintenance schedules in SL1 from a ServiceNow MID Server trigger. This application uses a similar workflow to the process used in *Directly Enabling or Disabling Device Maintenance from ServiceNow*, but this application works with the schedules instead.

This application receives a group of one or more synced devices from the ServiceNow Management, Instrumentation, and Discovery (MID) Server and checks for the *schedule* and *cancel* actions on those devices. If the application encounters devices with one of those actions, it will enable or disable the user maintenance status of those devices as needed.

As a result, in ServiceNow you can bring change requests out of a scheduled state and update them, and all of the updates to those change requests are synced back to SL1 in real time, even if those change requests were already scheduled.

You do *not* need to update any fields on the **Configuration** pane for the "Create or Update Maintenance Schedule from ServiceNow Trigger" application. You also do *not* need to run this application, as it is triggered by the MID Server, which is triggered first by a registered event in ServiceNow Event Management. For more information about registered events, including examples of other triggering events you can define in ServiceNow, see the *ServiceNow Registered Events* appendix.

Before you can set up maintenance sync with the MID Server, you need to create a credential for the MID Server. You should have access to the "Integration Services" section of the **Discovery Dependents** page in ServiceNow. For more information about creating the credential, see *Directly Enabling or Disabling Device Maintenance from ServiceNow*.

Chapter 4

Troubleshooting the Change Management Synchronization PowerPack

Overview

This chapter includes troubleshooting resources and procedures to use with the *ServiceNow Change Management* Synchronization PowerPack.

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.
2. Note the Docker container version, and verify that the Docker services are running.
3. If a certain service is failing, make a note the service name and version.
4. 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`.
5. 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 Synchronization PowerPack version, if applicable.
2. Make a note of whether the user is running an update set or a version of the Certified/Scoped application, if relevant.
3. Make a note of the ServiceNow application that is failing on 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. Use *guest/guest* for the login.
- **https://<*IP of PowerFlow*>/flower**. Provides access to Flower, a tool for monitoring and administrating Celery clusters.

NOTE: For version 2.0.0 and later of PowerFlow, 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 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.

Viewing Logs for a Specific Service

You can use the Docker command line to view the logs of any current running service in the PowerFlow cluster. To view the logs of any service, run the following command:

```
docker service logs -f iservices_<service_name>
```

Some common examples include the following:

```
docker service logs -f iservices_couchbase
docker service logs -f iservices_steprunner
docker service logs -f iservices_contentapi
```

> **NOTE:** *Application* logs are stored on the central database as well as on all of the Docker hosts in a clustered environment. These logs are stored at **/var/log/iservices** for both single-node or clustered environments. However, the logs on each Docker host only relate to the services running on that host. For this reason, using the Docker service logs is the best way to get logs from all hosts at once.

Clearing RabbitMQ Volume

RabbitMQ is a service that distributes tasks to be executed by PowerFlow workers. This section covers how to handle potential issues with RabbitMQ.

The following error message might appear if you try to run a PowerFlow application via the API:

```
Internal error occurred: Traceback (most recent call last):\n File \"./content_
api.py\", line 199, in kickoff_application\n task_status = ... line 623, in _on_
close\n (class_id, method_id), ConnectionError)\nInternalError: Connection.open:
(541) INTERNAL_ERROR - access to vhost '/' refused for user 'guest': vhost '/' is
down
```

First, verify that your services are up. If there is an issue with your RabbitMQ volume, you can clear the volume with the following commands:

```
docker service rm iservices_rabbitmq
docker volume rm iservices_rabbitdb
```

If you get a message stating that the volume is in use, run the following command:

```
docker rm <id of container using volume>
```

Re-deploy PowerFlow by running the following command:

```
docker stack deploy -c /opt/iservices/scripts/docker-compose.yml iservices
```

> **NOTE:** Restarting Docker does not clear the queue, because the queue is persistent. However, clearing the queue with the commands above might result in data loss due to the tasks being removed from the queue.

Viewing the Process Status of All Services

Run the following command:

```
docker ps
```

Deploying Services from a Defined Docker Compose File

Run the following command:

```
docker stack deploy -c <compose-file> iservices
```

Dynamically Scaling for More Workers

Run the following command:

```
docker service scale iservices_steprunner=10
```

Completely Removing Services from Running

Run the following command:

```
docker stack rm iservices
```

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 with "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, POST the following to the API endpoint:

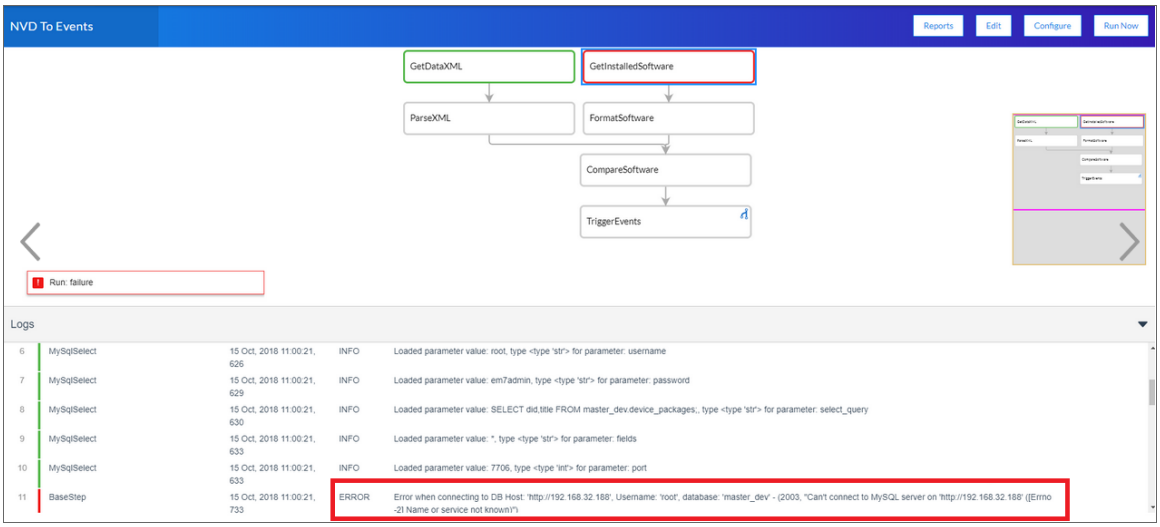
```
https://<PowerFlow>/api/v1/applications/run
```

Request body:

```
{
  "name": "<application_name>",
  "params": {
    "loglevel": 10
  }
}
```

After running the application in Debug Mode, go back to the PowerFlow user interface and review the step logs to see detailed debug output for each step in the application. When run in Debug Mode, the step log output shows additional debug statements such as "Saved data for next step", which displays the data being sent from one step to the next.

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:em7admin -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}}'
```

Appendix

A

ServiceNow API Endpoints

Overview

This appendix describes the customized ServiceNow API Endpoints that were created for the ServiceNow Synchronization PowerPacks. These scripted endpoints reduce the amount of REST calls that PowerFlow makes to ServiceNow.

Please note that for pagination, the following Query parameters are not required: `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 appendix includes the following topics:

Business Services

HTTP Method

GET

Pagination

Enabled

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/business_service
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/business_service
```

This operation pulls all the fields from just the Business Service (**cmdb_ci_service**) 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*** and ***SL1 ID*** field on the ServiceNow side. 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/v1/
sciencelogic/business_service
```

Example (Response)


```
{
    "results": [
        {
            "operational_status": "1",
            "sys_updated_on": "2019-02-06 19:32:34",
            "discovery_source": "Other Automated",
            "first_discovered": "2019-02-06 19:31:19",
            "sys_updated_by": "admin",
            "sys_created_on": "2019-02-06 19:31:19",
            "sys_domain": "global",
            "used_for": "Production",
            "sys_created_by": "is4user1",
            "sys_domain_path": "/",
            "install_status": "1",
            "name": "One Service to rule them",
            "subcategory": "Service",
            "busines_criticality": "1 - most critical",
            "last_discovered": "2019-02-06 19:31:19",
            "sys_class_name": "cmdb_ci_service",
            "sys_id": "52da95dcdb6323009f7dd7a0cf961918",
            "sys_class_path": "/!!/#C",
            "comments": "Postman",
            "sys_mod_count": "1",
            "x_sclo_scilogic_id": "1570",
            "model_id": "e8aaeb3f3763100044e0bfc8bcbe5d20",
            "cost_cc": "USD",
            "x_sclo_scilogic_monitored": "true",
            "category": "Business Service",
            "service_classification": "Technical Service",
            "x_sclo_scilogic_region": "ScienceLogic"
        }
    ],
    "sysparm_offset": 0,
    "sysparm_limit": 100,
    "return_count": 1,
    "total_count": 1
}
```

Change Requests

HTTP Method

GET

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/change_requests?record_type=change_
request&state=1&region=ScienceLogic
```

Default Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/change_requests?record_type=change_
request&state=1&region=ScienceLogic
```

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 operation returns all records with their configuration items in formatted JSON strings that include planned start and end time.

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\' is not defined and is required.

Fixed Internal Query

State:

```
'task.sys_class_name=' + recordType + 'task.state=' + state + '^ci_item.x_sclo_
scilogic_monitored=true^ci_item.x_sclo_scilogic_region=' + region
```

Non-State:

```
'task.active=true^task.sys_class_name=' + recordType + 'ci_item.x_sclo_scilogic_
monitored=true^ci_item.x_sclo_scilogic_region=' + region
```

Example

```
https://<your Instance>.service-now.com/api/x_sclo_scilogic/v1/sciencelogic/change_
requests?record_type=change_request&state=-5&region=ScienceLogic
```

Example (Response)


```
{
    "results": [
        {
            "sys_id": "48ebaba0db962f00dc44f00fbf961961",
            "number": "CHG0030001",
            "state_value": "-5",
            "state": "New",
            "short_description": "Test Change",
            "planned_start_date": "2019-01-01 06:00:01",
            "planned_end_date": "2019-01-01 18:00:01",
            "device": [
                {
                    "sys_id": "d83dac0adb4dab00dc44f00fbf961919",
                    "name": "Postman Test Server 11",
                    "id": "11",
                    "region": "ScienceLogic"
                }
            ]
        }
    ]
}
```

Classification version 1

NOTE: This API Endpoint has been deprecated. The last version of the "ScienceLogic SL1: CMDB & Incident Automation" application" (also called the Certified or Scoped application), that used this endpoint was version 1.0.18.

HTTP Method

GET

Pagination

Enabled

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/classification
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/classification
```

To support the identification and reconciliation framework, SL1 requires a large amount of information to know how to correctly fill out the JSON formatted string defined by the Identification Engine documentation. This operation uses the **getTableExtension()** function to find all the tables extended from the **cmdb_ci** table and then goes through each table one by one. 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.

Headers	
Key	***Value***
Content-Type	application/json
Accept	application/json

Parameters	
Key	***Value***
action	attributes

Attributes require x_sclo_scilogic.Admin be added to **sys_dictionary.*** (read) ACL to allow the API to access field names on each class table.

Fixed Internal Query

Example

```
https://<your Instance>.service-now.com/api/x_sclo_scilogic/sciencelogic/
classification
```

Example (Response)


```
{
  "results": [
    {
      "class_label": "Storage Area Network",
      "class_table": "cmdb_ci_san",
      "criterion_attributes": [
        ""
      ],
      "independent": "false",
      "containment_rule": [

      ],
      "hosting_rule": [

      ],
      "reference_rule": [

      ],
      "attributes": [
        "asset",
        "asset_tag",
        "assigned",
        "assigned_to",
        "assignment_group",
        "attributes",
        "can_print",
        "category",
        "change_control",
        "checked_in",
        "checked_out",
        "comments",
        "company",
        "correlation_id",
        "cost",
        "cost_cc",
        "cost_center",
        "delivery_date",
        "department",
        "discovery_source",
        "dns_domain",
        "due",
        "due_in",
        "fault_count",
```

```
        "first_discovered",
        "fqdn",
        "gl_account",
        "install_date",
        "install_status",
        "invoice_number",
        "ip_address",
        "justification",
        "last_discovered",
        "lease_id",
        "location",
        "mac_address",
        "maintenance_schedule",
        "managed_by",
        "manufacturer",
        "model_id",
        "model_number",
        "monitor",
        "name",
        "operational_status",
        "order_date",
        "owned_by",
        "po_number",
        "purchase_date",
        "san_id",
        "schedule",
        "serial_number",
        "short_description",
        "skip_sync",
        "start_date",
        "subcategory",
        "supported_by",
        "support_group",
        "sys_class_name",
        "sys_class_path",
        "sys_created_by",
        "sys_created_on",
        "sys_domain",
        "sys_domain_path",
        "sys_id",
        "sys_mod_count",
        "sys_updated_by",
        "sys_updated_on",
        "unverified",
        "vendor",
        "warranty_expiration",
        "x_sclo_scilogic_id",
        "x_sclo_scilogic_monitored",
        "x_sclo_scilogic_region",
        "x_sclo_scilogic_url"
      ]
    }
  ]
}
```

Classification version 2

HTTP Method

GET

Pagination

Enabled

Resource Path

```
/api/x_sclo_scilogic/v2/sciencelogic/classification
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/classification
```

To support the identification and reconciliation framework, SL1 requires a large amount of information to know how to correctly fill out the JSON formatted string defined by the Identification Engine documentation. This operation uses the **getTableExtension()** function to find all the tables extended from the **cmdb_ci** table and then goes through each table one by one. 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.

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

Attributes require x_sclo_scilogic.Admin be added to **sys_dictionary.*** (read) ACL to allow the API to access field names on each class table.

Fixed Internal Query

Example

```
https://<your Instance>.service-now.com/api/x_sclo_scilogic/v2/sciencelogic/
classification
```

Example (Response)


```
{
  "results": [
    {
      "class_label": "Storage Area Network",
      "class_table": "cmdb_ci_san",
      "criterion_attributes": [
        ""
      ],
      "independent": "false",
      "containment_rule": [

      ],
      "hosting_rule": [

      ],
      "reference_rule": [

      ],
      "attributes": [
        "asset",
        "asset_tag",
        "assigned",
        "assigned_to",
        "assignment_group",
        "attributes",
        "can_print",
        "category",
        "change_control",
        "checked_in",
        "checked_out",
        "comments",
        "company",
        "correlation_id",
        "cost",
        "cost_cc",
        "cost_center",
        "delivery_date",
        "department",
        "discovery_source",
        "dns_domain",
        "due",
        "due_in",
        "fault_count",
```

```
        "first_discovered",
        "fqdn",
        "gl_account",
        "install_date",
        "install_status",
        "invoice_number",
        "ip_address",
        "justification",
        "last_discovered",
        "lease_id",
        "location",
        "mac_address",
        "maintenance_schedule",
        "managed_by",
        "manufacturer",
        "model_id",
        "model_number",
        "monitor",
        "name",
        "operational_status",
        "order_date",
        "owned_by",
        "po_number",
        "purchase_date",
        "san_id",
        "schedule",
        "serial_number",
        "short_description",
        "skip_sync",
        "start_date",
        "subcategory",
        "supported_by",
        "support_group",
        "sys_class_name",
        "sys_class_path",
        "sys_created_by",
        "sys_created_on",
        "sys_domain",
        "sys_domain_path",
        "sys_id",
        "sys_mod_count",
        "sys_updated_by",
        "sys_updated_on",
        "unverified",
        "vendor",
        "warranty_expiration",
        "x_sclo_scilogic_id",
        "x_sclo_scilogic_monitored",
        "x_sclo_scilogic_region",
        "x_sclo_scilogic_url"
      ]
    }
  ]
}
```

CMDB Group

HTTP Method

POST

Pagination

Enabled

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/cmdb_group
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/cmdb_group
```

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***
Content-Type	application/json
Accept	application/json

Example (Request URL

```
https://<your Instance>.service-now.com/api/x_sclo_scilogic/v1/sciencelogic/cmdb_
group
```

Example (Body)

```
[
  {
    "items": [
      {
        "name": "test",
        "description": "",
        "group": "",
        "manualCIList": "d83dac0adb4dab00dc44f00fbf961919,2e6b7046db8dab00dc44f00fbf
        961929,7fb39667dba12380dc44f00fbf961901,77b39667dba12380dc44f00fbf961917,7bb
        39667dba12380dc44f00fbf96191c",
        "region": "Cisco",
        "id": "1"
```

```
        }
      ]
    }
  ]
```

Example (Response)

```
{
  "result": [
    {
      "idList": [

      ],
      "partialCIListDueToACLFlag": false,
      "nextBatchStart": 0,
      "result": true
    },
    {
      "idList": [

      ],
      "partialCIListDueToACLFlag": false,
      "nextBatchStart": 0,
      "result": true
    }
  ]
}
```

Companies

HTTP Method

GET

Pagination

Enabled

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/companies
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/companies
```

This operation supports Domain Separation enabled or not enabled. This operation pulls all the fields for from the company table that are not NULL values. 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*** and ***region*** values. The region mus be supplied by the requester, and it will only return region-specific companies.

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.

Fixed Internal Query

Domain:

```
'x_sclo_scilogic_region=' + region + '^x_sclo_scilogic_monitored=true^sys_
domain!=global'
```

Non-Domain:

```
'x'_sclo_scilogic_monitored=true^x_sclo_scilogic_idISNOTEMPTY^x_sclo_scilogic_
region'' + region
```

Example

```
https://<your instance>.service-now.com/api/x_sclo_scilogic/v1/sciencelogic/
companies?region=ScienceLogic&sysparm_offset=0&sysparm_limit=100
```

Example (Response)


```
{
  "results": [
    {
      "country": "USA",
      "notes": "What's on your digital horizon?",
      "city": "San Jose",
      "sys_updated_on": "2018-11-30 16:03:45",
      "sys_class_name": "core_company",
      "sys_id": "1ac84f95dbce2700dc44f00fbf9619c8",
      "sys_updated_by": "is4user1",
      "market_cap": "0",
      "street": "170 West Tasman Dr.",
      "sys_created_on": "2018-11-27 16:32:33",
      "state": "CA",
      "sys_created_by": "admin",
      "zip": "95134",
      "profits": "0",
      "revenue_per_year": "0",
      "sys_mod_count": "4",
      "x_sclo_scilogic_id": "1",
      "x_sclo_scilogic_monitored": "true",
      "phone": "18005532447",
      "name": "Cisco Systems, Inc.",
      "x_sclo_scilogic_region": "Cisco"
    }
  ],
  "sysparm_offset": 0,
  "sysparm_limit": 1,
  "return_count": 1,
```

```
  "total_count": 1
}
```

Device Identification Engine

HTTP Method

POST

Pagination

Enabled

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/IdentificationEngine
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/IdentificationEngine
```

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.

Headers	
Key	***Value***
Content-Type	application/json
Accept	application/json

Parameters	
Key	***Value***
test	true

Example (Request URL)

```
https://<your Instance>.service-now.com/api/x_sclo_scilogic/v1/sciencelogic/
IdentificationEngine?test=true
```

Example (Body)

```
[
  {
    "items": [
      {
        "className": "cmdb_ci_linux_server",
```

```
        "values": {
          "name": "Postman Test Server 1",
          "serial_number": "9876EFGH",
          "mac_address": "BF:D4:D6:6E:56:F1",
          "ip_address": "10.10.10.4",
          "ram": "16000",
          "x_sclo_scilogic_region": "ScienceLogic",
          "x_sclo_scilogic_id": "1"
        }
      }
    ]
  },
  {
    "items": [
      {
        "className": "cmdb_ci_linux_server",
        "values": {
          "name": "Postman Test Server 2",
          "serial_number": "HGFE6789",
          "mac_address": "87:54:3C:8C:2A:A3",
          "ip_address": "10.10.10.5",
          "ram": "16000",
          "x_sclo_scilogic_region": "ScienceLogic",
          "x_sclo_scilogic_id": "2"
        }
      }
    ]
  }
]
```

Example Business Service (Body)

```
[
{
"items": [
{
"className": "cmdb_ci_service",
"values": {
"name": "Integration Service",
"busines_criticality": "1 - most critical",
"used_for": "Production",
"operational_status": "1",
"service_classification": "Technical Service",
"comments": "Postman",
"x_sclo_scilogic_region": "ScienceLogic",
"x_sclo_scilogic_id": "1570"
}
},
{
"className": "cmdb_ci_linux_server",
"values": {
"name": "Postman Test Server",
"serial_number": "7MDvqrSNyd",
"manufacturer": "ScienceLogic, Inc.",
"model_id": "",
"mac_address": "EE:D6:0B:79:32:C7",
```

```
"ip_address": "10.10.10.224",
"ram": "16000",
"x_sclo_scilogic_region": "ScienceLogic",
"x_sclo_scilogic_id": "10"
}
}
],
"relations": [
{
"type": "Depends on::Used by",
"parent": 0,
"child": 1
}
]
}
]
```

Example (Response)

```
{
  "result": [
    {
      "items": [
        {
          "className": "cmdb_ci_linux_server",
          "operation": "NO_CHANGE",
          "sysId": "7fb39667dba12380dc44f00fbf961936",
          "identifierEntrySysId": "fb27f69cc3000200d8d4bea192d3ae67",
          "identificationAttempts": [
            {
              "identifierName": "Hardware Rule",
              "attemptResult": "SKIPPED",
              "attributes": [
                "serial_number",
                "serial_number_type"
              ],
              "searchOnTable": "cmdb_serial_number"
            },
            {
              "identifierName": "Hardware Rule",
              "attemptResult": "MATCHED",
              "attributes": [
                "serial_number"
              ],
              "searchOnTable": "cmdb_ci_hardware"
            }
          ]
        }
      ],
      "relations": [

      ]
    }
  ]
}
```

Discovery Dependents

HTTP Method

GET

Pagination

Enabled

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/discovery_dependent
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/discovery_dependent
```

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. This operation pulls all the required information for both SL1 processes: Discovery Session and Create Virtual Device. 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 Service Catalog 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
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": "2019-08-28 18:03:50",
            "type": "credential",
            "type_label": "Credentials",
            "sys_id": "0491aae51b273f0045c8db1dcd4bcbc2",
            "hostname": "example.com",
            "sys_updated_by": "is4user1",
            "sys_created_on": "2019-08-28 18:03:50",
            "name": "AppDynamics Example",
            "id": "93",
            "category": "soapCredentials",
            "region": "del_test",
            "sys_created_by": "is4user1"
        }
    ],
    "sysparm_offset": 0,
    "sysparm_limit": 1,
    "return_count": 1,
    "total_count": 150
}
```

File Systems

HTTP Method

GET

Pagination

Enabled

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/file_systems
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/file_systems
```

This operation pulls all the fields from the File System 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*** and ***SL1 ID*** field on the ServiceNow side. This operation requires the region to be supplied by the requester, it returns only 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

HTTP Status	
Code	***Value***

200	OK
400	Query Parameter \'region\' is not defined and is required.

Fixed Internal Query

```
'x_sclo_scilogic_monitored=true^x_sclo_scilogic_idISNOTEMPTY^x_sclo_scilogic_
region=' + region
```

Example

```
https://<your Instance>.service-now.com/api/x_sclo_scilogic/v1/sciencelogic/file_
systems?region=ScienceLogic&sysparm_offset=0&sysparm_limit=100
```

Example (Response)


```
{
  "results": [
    {
      "operational_status": "1",
      "sys_updated_on": "2018-11-12 21:59:52",
      "media_type": "fixed",
      "sys_created_by": "admin",
      "sys_domain_path": "/",
      "sys_class_name": "cmdb_ci_file_system",
      "computer": "d83dac0adb4dab00dc44f00fbf961919",
      "x_sclo_scilogic_monitored": "true",
      "x_sclo_scilogic_region": "ScienceLogic",
      "sys_updated_by": "admin",
      "sys_created_on": "2018-11-12 21:59:06",
      "sys_domain": "global",
      "install_status": "1",
      "name": "/root",
      "subcategory": "File Share",
      "sys_id": "afd30ba0dbf5a380dc44f00fbf961951",
      "file_system": "ntfs",
      "sys_class_path": "/!!/!K/!!",
      "mount_point": "/root",
      "sys_mod_count": "3",
      "x_sclo_scilogic_id": "31",
      "label": "/root",
      "cost_cc": "USD",
      "category": "Resource"
    }
  ],
  "sysparm_offset": 0,
  "sysparm_limit": 100,
  "return_count": 1,
  "total_count": 1
}
```

Import Set

HTTP Method

POST

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/import_set
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/import_set
```

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 operations 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
    }
    ]
  }
]
```

Incidents

HTTP Method

GET

Pagination

Enabled

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/incidents
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/incidents
```

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

Parameters	
Key	***Value***
user_id (required)	is4user1
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": "0141807bdbb16300dc44f00fbf9619fc",
      "number": "INC0010135",
      "state": "2",
      "state_label": "In Progress",
      "events": [
        {
          "event_id": "16908",
          "device": {
            "sys_id": {

            }
          }
        },
        {
          "event_id": "16874",
          "device": {
            "sys_id": {

            }
          }
        },
        {
          "event_id": "16865",
          "device": {
            "sys_id": {

            }
          }
        }
      ]
    }
  ],
  "sysparm_offset": 0,
  "sysparm_limit": 1,
  "return_count": 1,
```

```
  "total_count": 44
}
```

Installed Software

HTTP Method

GET

Pagination

Enabled

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/installed_software
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/installed_software
```

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
}
```

Manufacturer

HTTP Method

POST

Pagination

Enabled

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/manufacture
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/manufactures
```

This operation does not populate any data into ServiceNow. Instead, this operation takes an array of manufacturer names and attempts to line them up with manufacturers already in ServiceNow. Then the operation returns the sys_id of manufacturers it was able find based on matching name. If the Normalization Data Services Client is active on the target instance, this operation uses those tables to find a matching company record; otherwise the operation will match on whether name and manufacturer is true on the core_company table.

Headers	
Key	***Value***
Content-Type	application/json
Accept	application/json

Parameters	
Key	***Value***
region (required)	ScienceLogic

Example (Request URL)

```
https://<your Instance>.service-now.com/api/x_sclo_scilogic/v1/sciencelogic/
manufacture
```

Example (Body)

```
{
"manufactures": [
```

```
"Cisco Systems Inc",
"Cisco Systems, Incorporated",
"CiscoSystems",
"American Power Conversion Inc.",
"APC Corp",
"Apc",
"IBM",
"IBM CORP",
"International Business Machines",
"Juniper Systems",
"Juniper Networks,Inc",
"Juniper Solutions"

]
}
```

Example (Response)

```
{
"result": {
"Cisco Systems Inc": "",
"Cisco Systems, Incorporated": "",
"CiscoSystems": "",
"American Power Conversion Inc.": "",
"APC Corp": "",
"Apc": "",
"IBM": "",
"IBM CORP": "",
"International Business Machines": "",
"Juniper Systems": "",
"Juniper Networks,Inc": "",
"Juniper Solutions": ""
}
}
```

Model

HTTP Method

POST

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/model
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/models
```

This operation does not populate any data into ServiceNow. Instead, this operation takes an array of model names and attempts to line them up with models already in ServiceNow and returns the sys_id of models it was able to find based on matching name.

Headers	
Key	***Value***
Content-Type	application/json
Accept	application/json

Parameters	
Key	***Value***
region (required)	ScienceLogic

Example (Request URL)

```
https://<your Instance>.service-now.com/api/x_sclo_scilogic/v1/sciencelogic/models
```

Example (Body)

```
{
"models": [
"4331 ISR",
"7206VXR",
"7609S",
"AS5300",
"ASR5000",
"Catalyst 3560G-24TS",
"Catalyst 4948",
"Catalyst 6509-CatOS",
```

```
"BIG-IP Viprion B4300",
"F5 BIG-IP DNS",
"BIG-IP Wide IP Container",
"BIG-IP Data Center Container"
]
}
```

Example (Response)

```
{
"result": {
"4331 ISR": "",
"7206VXR": "",
"7609S": "",
"AS5300": "",
"ASR5000": "",
"Catalyst 3560G-24TS": "",
"Catalyst 4948": "",
"Catalyst 6509-CatOS": "",
"BIG-IP Viprion B4300": "",
"F5 BIG-IP DNS": "",
"BIG-IP Wide IP Container": "",
"BIG-IP Data Center Container": ""
}
}
```

Network Adapters

HTTP Method

GET

Pagination

Enabled

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/network_adapters
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/network_adapters
```

This operation pulls all the fields from the network adapter 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*** and ***SL1 ID*** field on the ServiceNow side. This operation requires the region to be supplied by the requester, and it only returns 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

HTTP Status	
Code	***Value***

200	OK
400	Query Parameter \'region\' is not defined and is required.

Fixed Internal Query

```
'x_sclo_scilogic_monitored=true^x_sclo_scilogic_idISNOTEMPTY^x_sclo_scilogic_
region=' + region
```

Example (Request URL)

```
https://<your Instance>.service-now.com/api/x_sclo_scilogic/v1/sciencelogic/network_
adapters?region=ScienceLogic&sysparm_offset=0&sysparm_limit=100
```

Example (Response)


```
{
  "results": [
    {
      "operational_status": "1",
      "sys_updated_on": "2018-11-12 21:29:23",
      "sys_updated_by": "admin",
      "sys_created_on": "2018-11-12 21:27:48",
      "sys_domain": "global",
      "sys_created_by": "admin",
      "cmdb_ci": "d83dac0adb4dab00dc44f00fbf961919",
      "sys_domain_path": "/",
      "install_status": "1",
      "name": "eth0",
      "subcategory": "Network",
      "sys_class_name": "cmdb_ci_network_adapter",
      "sys_id": "33ac36acdbb5a380dc44f00fbf961963",
      "netmask": "255.255.255.0",
      "sys_class_path": "/!!/!8",
      "mac_address": "BF:D4:D6:6E:56:F1",
      "sys_mod_count": "3",
      "x_sclo_scilogic_id": "20",
      "ip_address": "10.10.10.4",
      "cost_cc": "USD",
      "x_sclo_scilogic_monitored": "true",
      "category": "Hardware",
      "x_sclo_scilogic_region": "ScienceLogic"
    }
  ],
  "sysparm_offset": 0,
  "sysparm_limit": 1,
  "return_count": 1,
  "total_count": 5
}
```

Service Requests

HTTP Method

GET

Pagination

Enabled

Resource Path

```
/api/x_sclo_scilogic/v1/sciencelogic/service_request
```

Default Resource Path

```
/api/x_sclo_scilogic/sciencelogic/service_request
```

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": "RITM0010018",
      "sysid": "00365de2db1a2340dc44f00fbf961941",
      "state": "2",
      "request_type": "Discover Device",
      "region": "Cisco",
      "log_all": "false",
      "ip_hostname_list": "167.132.14.15",
      "credentials": [
        {
          "Category": "Linux",
          "ID": "1"
        }
      ],
      "discover_non_snmp": "false",
      "model_devices": "true",
      "dhcp": "false",
      "device_model_cache_ttl_h": "2",
      "collection_server": "1",
      "organization": "1",
      "add_devices_to_device_groups": [
        "test"
      ],
      "device_template": "1",
```

```
      "initial_scan_level": "System Default (Recommended)",
      "scan_throttle": "System Default (Recommended)",
      "scan_ports": "21,22,23,25,80",
      "port_scan_all": "System Default (Recommended)",
      "port_scan_timeout": "System Default (Recommended)",
      "interface_inventory_timeout": "600000",
      "maximum_allowed_interfaces": "10000",
      "bypass_interface_inventory": "false"
    },
    {
      "number": "RITM0010016",
      "sysid": "194447e8db162f00dc44f00fbf96195b",
      "state": "2",
      "request_type": "Discover Device",
      "region": "Cisco",
      "log_all": "false",
      "ip_hostname_list": "192.168.1.1",
      "credentials": [
        {
          "Category": "Linux",
          "ID": "1"
        }
      ],
      "discover_non_snmp": "false",
      "model_devices": "false",
      "dhcp": "false",
      "device_model_cache_ttl_h": "2",
      "collection_server": "1",
      "organization": "1",
      "add_devices_to_device_groups": [

      ],
      "device_template": "1",
      "initial_scan_level": "System Default (Recommended)",
      "scan_throttle": "System Default (Recommended)",
      "scan_ports": "21,22,23,25,80",
      "port_scan_all": "System Default (Recommended)",
      "port_scan_timeout": "System Default (Recommended)",
      "interface_inventory_timeout": "600000",
      "maximum_allowed_interfaces": "10000",
      "bypass_interface_inventory": "false"
    },
    {
      "number": "RITM0010014",
      "sysid": "250dae2cdbd22f00dc44f00fbf961954",
      "state": "2",
      "request_type": "create_virtual_device",
      "region": "Cisco",
      "collection_server": "1",
      "virtual_device_class": "1"
    }
  ]
}
```

Appendix

B

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: You can access the update set from the **additional_materials.zip** file included in the main **.zip** file for the ServiceNow CMDB Synchronization PowerPack, which you can find on the **Synchronization PowerPack** page at the ScienceLogic Support Site.

This appendix includes 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.

Maintenance Mode Events

The following event is available through the "ScienceLogic ServiceNow Integration (Maintenance Mode)" update set in ServiceNow.

x_sclo_scilogic.device_maintenance

This event submits a list of devices to be put in to Maintenance via the Mid Server.

Trigger

Custom requirement supplied by ScienceLogic implementation or the Customer directly.

Command

```
gs.eventQueue('x_sclo_scilogic.device_maintenance',current, action, affected_ci);
```

Event Fields

Field	*Description*
`x_sclo_scilogic.device_maintenance`	Unique name of the event.
`current`	The table to which the event applies.
`action`	Parm1: An array that includes action to be performed (enable_maintenance or disable_maintenance) and the `sys_id` of the task. Task is not required, but the action is.
`affected_ci`	Parm2: An array of device sys_ids that need to be enable or disabled maintenance mode.

Example

Business rule (ScienceLogic ServiceNow Integration (Maintenance Mode Business rule)).

Maintenance Schedule Events

The following event is available through the "ScienceLogic ServiceNow Integration (Maintenance Schedule) Example" update set in ServiceNow.

x_sclo_scilogic.device_maintenance_skd

This event submits a list of devices and creates or cancels scheduled maintenance via the Mid Server.

Trigger

Custom requirement supplied by ScienceLogic implementation or the Customer directly.

Command

```
gs.eventQueue('x_sclo_scilogic.device_maintenance_skd',current, 'schedule',
current.getUniqueValue());
```

Event Fields

Field	*Description*
`x_sclo_scilogic.device_maintenance_skd`	Unique name of the event.
`current`	The table to which the event applies.
`'schedule'`	Parm1: Accepts two variables: *schedule* or *cancel*. - **[schedule]**. Creates a schedule in SL1. - ***cancel***: Deletes any currently scheduled maintenance.
`current.getUniqueValue())`	Parm2: An array of device `sys_id` values that need to be enabled or disabled for maintenance mode.

Example

The UI action / UI page is available via the "ScienceLogic ServiceNow Integration (Maintenance Schedule) Example" update set.

Copyrights and Trademarks

ScienceLogic, the ScienceLogic logo, and EM7 are trademarks of ScienceLogic, Inc. in the United States, other countries, or both.

Below is a list of trademarks and service marks that should be credited to ScienceLogic, Inc. The ® and ™ symbols reflect the trademark registration status in the U.S. Patent and Trademark Office and may not be appropriate for materials to be distributed outside the United States.

- ScienceLogic™
- EM7™ and em7™
- Simplify IT™
- Dynamic Application™
- Relational Infrastructure Management™

The absence of a product or service name, slogan or logo from this list does not constitute a waiver of ScienceLogic's trademark or other intellectual property rights concerning that name, slogan, or logo.

Please note that laws concerning use of trademarks or product names vary by country. Always consult a local attorney for additional guidance.

Other

If any provision of this agreement shall be unlawful, void, or for any reason unenforceable, then that provision shall be deemed severable from this agreement and shall not affect the validity and enforceability of any remaining provisions. This is the entire agreement between the parties relating to the matters contained herein.

In the U.S. and other jurisdictions, trademark owners have a duty to police the use of their marks. Therefore, if you become aware of any improper use of ScienceLogic Trademarks, including infringement or counterfeiting by third parties, report them to Science Logic's legal department immediately. Report as much detail as possible about the misuse, including the name of the party, contact information, and copies or photographs of the potential misuse to: legal@sciencelogic.com

800-SCI-LOGIC (1-800-724-5644)

International: +1-703-354-1010