INTEGRATION PACK FOR SOLARWINDS

For Microsoft System Center Orchestrator

For System Center 2016 and 2019, you must use the 32-bit version of the integration pack, which has the name **Kelverion_Integration_Pack_for_SolarWinds_2.0**

For System Center 2022 and later, you must use the 64-bit version of the integration pack, which has the name **Kelverion_IP_SolarWinds_x64_2.0**

User Guide

Version 2.0

Kelverion Integration Pack for SolarWinds

Published: February 2023

Feedback

Send suggestions and comments about this document support@kelverion.com

Contents

Kelverion Integration Pack for SolarWinds

The Integration Pack for SolarWinds is an add-on for System Center Orchestrator that enables integration with the SolarWinds Orion Platform.

System Requirements

The Kelverion Integration Pack for SolarWinds requires the following software to be installed and configured prior to implementing the integration. For more information about installing and configuring System Center Orchestrator and SolarWinds refer to the respective product documentation.

Kelverion_Integration_Pack_for_SolarWinds (32-bit)

- Microsoft System Center Orchestrator 2016, 2019
- Microsoft .NET Framework 4.7.2
- SolarWinds Orion Platform
    - 2019.4 (NPM 2019.4)
    - 2020.2.4 (NPM 2020.2.4)
    - 2022.4 (NPM 2022.4)
    - 2023.1 (NPM 2023.1)
- SolarWinds Information Service (SWIS)

Kelverion_IP_SolarWinds_x64 (64-bit)

- Microsoft System Center Orchestrator 2022
- Microsoft .NET Framework 4.7.2
- SolarWinds Orion Platform
    - 2019.4 (NPM 2019.4)
    - 2020.2.4 (NPM 2020.2.4)
    - 2022.4 (NPM 2022.4)
    - 2023.1 (NPM 2023.1)
- SolarWinds Information Service (SWIS)

Registering and Deploying the Integration Pack

After you download the integration pack file, you must register it with the Orchestrator management server and then deploy it to Runbook Servers and Runbook Designers. For more information about how to install integration packs, see the How to Install an Integration Pack in the online documentation for System Center Orchestrator.

IMPORTANT: Ensure that you are deploying the correct version of the Integration Pack.

- For System Center 2016 and 2019, you must use the 32-bit version of the integration pack, which has the name **Kelverion_Integration_Pack_for_SolarWinds**

- For System Center 2022 and later, you must use the 64-bit version of the integration pack, which has the name **Kelverion_IP_SolarWinds _x64**

To register the integration pack:

1. On the management server, copy the **.OIP** file for the integration pack to a local hard drive or network share.
2. Confirm that the file is not set to **Read Only** to prevent unregistering the integration pack later.
3. Start the **Deployment Manager**.
4. In the navigation pane of the Deployment Manager, expand **Orchestrator Management Server**, right-click **Integration Packs** to select **Register IP with the Orchestrator Management Server**. The **Integration Pack Registration Wizard** opens.
5. Click **Next**.
6. In the **Select Integration Packs or Hotfixes** dialog box, click **Add**.
7. Locate the **.OIP** file that you copied locally from step 1, click **Open** and then click **Next**.
8. In the **Completing the Integration Pack Wizard** dialog box, click **Finish**.
9. On the **End User Agreement** dialog box, read the Kelverion License Terms**,** and then click **Accept**.
10. The **Log Entries** pane displays a confirmation message when the integration pack is successfully registered.

To deploy the integration pack:

1. In the navigation pane of the **Deployment Manager**, right-click **Integration Packs**, click **Deploy IP to Runbook Server or Runbook Designer**.
2. Select the integration pack that you want to deploy, and then click **Next**.
3. Enter the name of the runbook server or computers with the Runbook Designer installed, on which you want to deploy the integration pack, click **Add**, and then click **Next**.
4. Continue to add additional runbook servers and computers running the Runbook Designer, on which you want to deploy the integration pack. Click **Next**.
5. In the **Installation Options** dialog box, configure the following settings.
6. To choose a time to deploy the integration pack, select the **Schedule installation** check box, and then select the time and date from the **Perform installation** list.
7. Click one of the following:
    a. **Stop all running runbooks before installing the integration pack** to stop all running runbooks before deploying the integration pack.
    b. **Install the Integration Packs without stopping the running Runbooks** to install the integration pack without stopping any running runbooks.
8. Click **Next**.
9. In the **Completing Integration Pack Deployment Wizard** dialog box, Click **Finish**.

10. When the integration pack is deployed, the Log Entries pane displays a confirmation message.
11. Restart each Runbook Server/Designer host system to ensure that the Kelverion Windows Automation Agent is running.

Licensing the Integration Pack

After you register and deploy the integration pack you must provide a valid Kelverion license before running any runbooks that contain activities from the integration pack.

To deploy the integration pack license file to System Center Orchestrator 2019 or earlier:

1. Copy the .KAL license file to %PROGRAMFILES(X86)%\Kelverion Automation\Licenses
2. Repeat for each Orchestrator Runbook Server and Runbook Designer host system.

To deploy the integration pack license file to System Center Orchestrator 2022 or later:

1. Copy the .KAL license file to %PROGRAMFILES%\Kelverion Automation\Licenses
2. Repeat for each Orchestrator Runbook Server and Runbook Designer host system.

Configuring the Integration Pack For SolarWinds

A configuration establishes a reusable link between Orchestrator and a SolarWinds instance. You can create as many connections as you require specifying links to multiple instances. You can also create multiple connections to the same instance to allow for differences in security permissions for different user accounts.

To set up a SolarWinds configuration:

1. In the Client, click the **Options** menu, and select *KA SolarWinds*. The **KA SolarWinds** dialog box appears.
2. On the **Configurations** tab, click **Add** to begin the configuration setup. The **Add Configuration** dialog box appears.
3. In the **Name** box, enter a name for the configuration. This could be the name of the SolarWinds instance or a descriptive name to distinguish the type of configuration.
4. Click the ellipsis button (…) next to the **Type** box and select *SolarWinds Configuration.*
5. In the **Server URL** box, enter the URL of your SolarWinds server, including the port used by the SolarWinds Information Service, in the form: **https://<SolarWinds Server>:<SWIS Port>**. The default SWIS port is 17778.

   Example: https://SolarWindsServer.YourDomain.com:17778

   Important Note: If you are using a proxy server, you must enter the URL in the form: **https://<SolarWinds Server IP>:<SWIS Port>**.

   Example: https://192.168.1.1:17778
6. In the **User Name** and **Password** boxes, enter the credentials of a user with sufficient privileges for your SolarWinds integration needs. The IP will use these credentials to connect to the SolarWinds server.

7. In the **Configuration File Path** box, enter the location of the Integration Pack configuration file. For more details, please see Additional Configuration.
8. In the **Request Timeout (seconds)** box, enter the number of seconds the IP should wait before timing out when sending API requests.
9. If you are using a proxy server then in the **Proxy Server URL** box, type the full URL of the proxy server, starting with http:// or https://. The URL should also contain the port used to access the proxy server. Example: http://PROXY-SRV:3128

   Important Note: The connection will only use the proxy provided if the **Server URL** entered in Step 5 is in the form of an IP address.
10. If you are using a proxy server then provide credentials using the **Proxy Use Name**, **Proxy Password**, and **Proxy Domain** boxes.
11. In the **Skip Certificate Validation** box, specify if you want the IP to perform server certificate validation or not. This applies only when connecting to the server over HTTPS. When set to True, the IP will not perform certificate validation. This is typically used in secure environments, when working with trusted servers and self-signed certificates. When set to False, the IP will validate the server certificate. The server must be configured with a valid certificate signed by a valid certificate authority and the server name in the **Server URL** must be listed on the certificate.
12. Add additional connections if applicable.
13. Click **OK** to close the configuration dialog box, and then click **Finish**.

Additional Configuration

In addition to the typical IP configuration options, this integration pack uses a configuration file, which enables further customization.

The configuration file **Kelverion.IntegrationPack.SolarWinds.Configuration.xml** is installed by default at the same location where integration pack files are installed.

On System Center Orchestrator 2019 or earlier:

<Program Files (x86)>\Common Files\Microsoft System Center 2012\
Orchestrator\Extensions\Support\Integration Toolkit\65F05F21-8820-4EA0-B24F-7304E5BA450C

On System Center Orchestrator 2022 or later:

<Program Files>\Common Files\Microsoft System Center 2012\
Orchestrator\Extensions\Support\Integration Toolkit\65F05F21-8820-4EA0-B24F-7304E5BA450C

Make sure to back up the original configuration file before you make any changes.

Note that the configuration file specified above is overwritten when the integration pack is updated. If you customize your configuration, make sure to save to save to a separate file and to re-apply your changes after updating the integration pack.

Configuration Structure

The main purpose of the configuration file is to provide information about the Orion fields and objects used by the integration pack activities. As an integration pack user, you can configure which fields are published by Get/Monitor activities, and you can customize which Orion objects are accessed to retrieve data from.

The configuration file is divided in two major sections under the root node. The first section contains information about the Orion objects used by the IP, the second section contains mapping information used by browser fields:

```xml
<Kelverion.IntegrationPack.SolarWinds>
    <Objects>
        <Object Name="AlertActive">
        <Object Name="AlertObject">
        <Object Name="AlertActiveObject">
        <Object Name="AlertSuppression">
        <Object Name="Node">
    </Objects>
    <Lists>
        <Map Name="ObjectSubType">
        <Map Name="PollerObjectType">
    </Lists>
    <Maps>
        <Map Name="AlertSeverity">
        <Map Name="NodeCategory">
        <Map Name="NodeStatus">
    </Maps>
</Kelverion.IntegrationPack.SolarWinds>
```

Objects are used to provide field and table information. For example:

- `AlertActive` is used by the *Get Alert* and *Monitor Alert* activities.
- `AlertObject` and `AlertActiveObject` are used by the *Get Trigger Object* activity.
- `AlertSuppression` is used by the *Get Suppressed* activity.
- `Node` is used by the *Node* activities.

Lists are used in browser fields to list property values. For example:

- `ObjectSubType` is used to provide a list of object subtypes in *Node* activities.
- `PollerObjectType` is used to provide a list of object types in *Poller* activities.

Maps are used in browser fields to map display values to internal SolarWinds values or IDs. For example:

- `AlertSeverity` is used by the *Severity* field the *Get Alert* and *Monitor Alert* activities.
- `NodeCategory` is used by the *Category* field the *Get Node* activity.
- `NodeStatus` is used by the *Status* field the *Node* activities.

Fields

Each section contains:

- A `<Fields>` element which lists the fields published by an activity.
- A `<Tables>` element containing information about one or more tables (objects) where data is retrieved from.

Each <Field> element contains the following attributes:

- `Name` – the field name, qualified with the table (or object) name, as defined in the Orion object.
- `Display Name` – the field name, as published by the integration pack.
- `Type` – the field type, as published by the integration pack.
- `List` – specifies a list in the Lists section, for providing browser field valid.
- `Map` – specifies a map in the Maps section, for providing browser field valid values and mapping to SolarWinds internal values.
- `CanCreate` –specifies if the field can be specified when the object is created.
- `CanUpdate` –specifies if the field can be specified when the object is updated.
- `Required` – specifies if the field is required when the object is created.
- `WriteOnly` – specifies if the field cannot be retrieved and can only be specified when the object is created or updated. By default, fields are retrievable unless specified.

For example:

```
<Field Name="AlertActive.AlertActiveID" DisplayName="Active Alert ID" Type="long" />
```

The `<Field>` element above describes the `Active Alert ID` published data field, which is published by the IP *Get Alert* and *Monitor Alert* activities as type `long`. This field maps internally to the `AlertActiveID` field in the `AlertActive` table.

You can customize the field display name for existing fields, this will change the published data name in the activity. ***Be aware that changing a published data name will require changing existing runbooks and activities which are already using that published data field.***

You can also specify additional fields, as long as the field name is correctly qualified with the table (object) name, and the field is available from one of the tables specified in the <Tables> element.

Tables

Each <Tables> element contains information about the tables (objects) where data is retrieved from. This can contain a single table, or multiple tables joined by SQL-like join clauses.

For example:

```
<Tables>
    Orion.AlertActive
    INNER JOIN Orion.AlertObjects
        ON AlertActive.AlertObjectID=AlertObjects.AlertObjectID
    INNER JOIN Orion.AlertConfigurations
        ON AlertObjects.AlertID=AlertConfigurations.AlertID
</Tables>
```

The above element specifies that alert data will be retrieved from the `AlertActive, AlertObjects and AlertConfigurations` tables, joined by their respective primary and foreign key IDs. You can specify additional tables in the join, if you require activities to publish additional data from those tables.

Additional Resources

Additional information about Alert objects, SWIS and the Orion SDK:

https://github.com/solarwinds/OrionSDK/wiki/Alerts

Additional information about SWQL – SolarWinds SQL-like query language:

https://thwack.solarwinds.com/docs/DOC-186990

SolarWinds Activities

This integration pack adds the **KA SolarWinds** category to the **Activities** pane in the Runbook Designer. This category contains the following activities:

- Add Poller
- Acknowledge Alert
- Create Node
- Clear Alert
- Delete Node
- Get Alert
- Get Alert History
- Get Engine
- Get Node
- Get Poller
- Get Suppressed
- Get Trigger Object
- Manage Node
- Monitor Alert
- Monitor Alert History
- Resume Alert
- Remove Poller
- Set Alert Note
- Set Custom Property
- Suppress Alert
- Un-Acknowledge Alert
- Un-Manage Node
- Update Node

Common Configuration Instructions for All Activities

The following configuration instructions apply to many of the activities in this integration pack. Links to this section are included in the configuration instructions for each activity.

Activity Properties

Each activity has a set of required or optional properties that define the configuration of that activity. This includes how it connects to other activities or how the activity performs its actions. You can view or modify activity properties in the Orchestrator Designer.

To configure the properties for an activity:

1. Double-click the activity. Alternatively, you can right-click the activity, and then click **Properties**.
2. To save your configuration entries, click **Finish**.

In the activity properties dialog box, several tabs along the left side provide access to general and specific settings for the activity. Although the number of available tabs for activity properties differs from activity to activity, all activities will have a **General** tab, a **Properties** tab, and a **Run Behavior** tab. Some activities may have a **Filters** tabs.

General Tab

This tab contains the **Name** and **Description** properties for the activity. By default, the **Name** of the object is the same as its object type, and the **Description** is blank. You can modify these properties to create more descriptive names or provide detailed descriptions of the activity.

Filters Tab

Monitor and Get activities provide filters which can be used to control data set retrieval in the activity. Field values of potential candidate records are compared to the values specified in filters to determine if they meet the filtering criteria. When matching against values, you select one of the available methods of comparison. An option is provided to either match or not match the filter using each method. For example, the "Does not" version of a method causes alerts that do not match the filter to trigger the runbook.

- **Equals**: the record field value exactly matches the text or number specified in the filter.
- **Does not equal**: the record field value does not exactly match the text or number specified in the filter.
- **Is less than**: the record field value is less than the number specified in the filter.
- **Is less than or equal to**: the record field value is less than or equal to the number specified in the filter.
- **Is greater than**: the record field value is greater than the number specified in the filter.
- **Is greater than or equal to**: the record field value is greater than or equal to the number specified in the filter.
- **Contains**: the record field value contains the exact text specified in the filter. Unlike the Equals behavior, there can be other text surrounding the matching text.
- **Does not contain**: the record field value does not contain the exact text specified in the filter. Unlike the Equals behavior, there can be other text surrounding the matching text.
- **Starts with**: the record field value starts with the exact text specified in the filter. Unlike the Equals behavior, there can be other text following the matching text.
- **Ends with**: the record field value ends with the exact text specified in the filter. Unlike the Equals behavior, there can be other text preceding the matching text.
- **Matches:** the record field value matches the specified pattern expression. You can use wildcard expression characters % (percent) and _ (underscore) to define pattern expressions.
- **Does not Match:** the record field value does not match the specified pattern expression. You can use wildcard expression characters % (percent) and _ (underscore) to define pattern expressions.

Null Filtering

This IP supports filtering by NULL values for **Equals** and **Does not equal** filters. You can type <NULL> in the filter value box to specify NULL comparison.

Run Behavior Tab

This tab contains the properties that determine how the activity handles multi-value published data and what notifications will be sent if the activity fails or runs for an excessive period of time.

Multi-Value Published Data Behavior

The Get Records and Monitor Records activities retrieve information from another activity or outside source, and can return one or more values in the published data. For example, when you use the Get Record activity, the data output from that activity might be a list of incident requests that belong to the specified support team.

By default, the data from the Get activity will be passed on as multiple individual outputs. This invokes the next activity as many times as there are items in the output. Alternatively, you can provide a single output for the activity by enabling the **Flatten** option. When you enable this option, you also choose a formatting option:

- **Separate with line breaks**: Each item is on a new line. This format is useful for creating human-readable text files for the output.
- **Separate with _:** Each item is separated by one or more characters of your choice.
- **Use CSV format**: All items are in CSV (comma-separated value) format. This format is useful for importing data into spreadsheets or other applications.

The activity will produce a new set of data every time it runs. The **Flatten** feature does not flatten data across multiple instances of the same activity.

Event Notifications

Some activities are expected to take a limited amount of time to complete. If they do not complete within that time they may be stalled or there may be another issue preventing them from completing. You can define the number of seconds to wait for completion of the action. After this period, a platform event will be sent, and the issue will be reported. You can also choose whether to generate a platform event if the activity returns a failure.

To be notified when the activity takes longer than a specified time to run or fails to run:

1. In the **Event Notifications** box, enter the **number of seconds** of run time before a notification is generated.
2. Select **Report if activity fails to run** to generate run failure notifications.

For more information about Orchestrator events, see the “Event Notifications " topics in the Runbook Properties in the online documentation for System Center Orchestrator.

Published Data

Returned data is the foundation of a working Runbook. It is the data produced as a result of the actions of an activity. This data is published to an internal data bus that is unique for each Runbook. Subsequent activities in the Runbook can subscribe to this data and use it in their configuration. Link conditions also use this information to add decision-making capabilities to Runbooks.

An activity can only subscribe to data from the activities that are linked before it in the Runbook. You can use published data to automatically populate the property values needed by activity.

To use published data:

1. Right-click the property value box, click **Subscribe**, and then click **Published Data**.

2. Click the **Activity** drop-down box and select the activity from which you want to obtain the data.
3. To view additional data elements common to all activities, select **Show Common Published Data**.
4. Click the published data element that you want to use, and then click **OK**.

For a list of the data elements published by each activity, see the Published Data tables in the activity topic. For information about the common published data items, see the Published Data in the online documentation for System Center Orchestrator.

Add Poller Activity

The **Add Poller** activity can be used in a runbook to add pollers to an existing SolarWinds object in your SolarWinds environment.

Required Properties

You must configure the following properties.

Object Type	Specifies the type of object for which the poller will be added. Allowed values are: • I • N • V
Object ID	Specifies the ID of the object for which the poller will be added.
Poller Type	Specifies the type of poller to be created. The activity provides a list of poller types based on the pollers that were found in your environment. You can find additional poller types in the SolarWinds Orion SDK documentation.

Optional Properties

You can use the following properties, as necessary, to control how this activity runs.

Enabled	Specifies if the created poller should be enabled or not. If not specified, the poller is enabled when created.

Published Data

This activity publishes the following activity-specific data.

Poller ID	Specifies the ID of the newly created poller object.
Poller Uri	Specifies the SWIS URI for the newly created poller object.

Acknowledge Alert Activity

The **Acknowledge Alert** activity can be used in a runbook to acknowledge an alert after it has triggered.

Required Properties

You must configure the following properties.

Alert Trigger Object ID	Alert Trigger Object ID for the alert that is to be acknowledged. You can specify multiple comma-separated IDs to acknowledge multiple alerts with one activity.
Append Note	Specifies a note to be appended to the alert when acknowledged. Note that when acknowledging the same alert instance multiple times, only the Append Note from the first acknowledgement is taken into consideration.

Published Data

This activity does not publish any activity-specific data.

Clear Alert Activity

The **Clear Alert** activity can be used in a runbook to clear an alert without running the normal reset actions. If the condition that triggered the alert persists, the alert will be triggered again the next time the condition is evaluated.

Required Properties

You must configure the following properties.

Alert Trigger Object ID	Alert Trigger Object ID for the alert that is to be cleared. You can specify multiple comma-separated IDs to clear multiple alerts with one activity.

Published Data

This activity does not publish any activity-specific data.

Create Node Activity

The **Create Node** activity can be used in a runbook to add new nodes in your SolarWinds environment.

Required Properties

You must configure the following properties.

Engine ID	Specifies the Engine ID for the node. You can retrieve the engine objects from your environment by using the *Get Engine* activity.
IP Address	Specifies the IP address for the node.
Subtype	Specifies the node subtype. Allowed values are: • Agent • ICMP • SNMP • WMI

Optional Properties

You can use the following properties, as necessary, to control how this activity runs.

Community	Specifies the SNMP community string for an SNMP node.
Custom Status	Specifies node custom status.
DNS	Specifies the node DNS.
Description	Node description.
Dynamic IP	Specifies if the node has dynamic IP address.
Is Server	Specifies if the node is a server.
Location	Specifies node location.
Node Name	Node name.
SNMP Version	Specifies the SNMP version for an SNMP node.
System Description	System description.
System Name	System name.
Vendor	Node vendor.

Published Data

This activity publishes the following activity-specific data.

Node ID	Specifies the ID of the newly created node object.
Node Uri	Specifies the SWIS URI for the newly created node object.

Delete Node Activity

The **Delete Node** activity can be used in a runbook to delete an existing SolarWinds node. Note that pollers associated with the specified node will also be deleted.

Required Properties

You must configure the following properties.

Node URI	Identifies the node that is to be deleted.

Published Data

This activity does not publish any activity-specific data.

Get Alert Activity

The **Get Alert** activity can be used in a runbook to retrieve and filter alerts that have triggered in your SolarWinds Orion environment.

Required Properties

You must configure the following properties.

Descending	Specifies that the returned record set should be ordered in descending order. If this property is not specified and *Order By* property is specified, records will be ordered in ascending order. This property is ignored if *Order By* property is not specified.
Order By	Specifies the field that the record set should be ordered by. If not specified, record ordering is arbitrary. By default, when this property is specified, records are ordered in ascending order. To order records in descending order, set *Descending* property to True.
Record Limit	Specifies the maximum number of records that the activity should return.

Filters

This activity provides filters, that you can combine, to control which alert records to retrieve.

Acknowledged	Filter by *Acknowledged*. **SolarWinds may not return expected results when filtering for False**. We recommend only using **True** when configuring filters for this field. For example: • Acknowledged Equals True • Acknowledged Does not equal True Alternatively, this IP supports filtering by NULL values for Equals and Does not equal filters. You can type <NULL> in the filter value box to specify NULL comparison. For example: • Acknowledged Equals <NULL> • Acknowledged Does not equal <NULL>
Acknowledged By	Filter by *Acknowledged By* value.
Acknowledged DateTime	Filter by *Acknowledged DateTime.*
Active Alert ID	Filter by *Active Alert ID.*
Alert Config ID	Filter by *Alert Config ID.*
Alert Note	Filter by *Alert Note.*
Alert Trigger Object ID	Filter by *Alert Trigger Object ID.*
Alert Ref ID	Filter by *Alert Ref ID.*
Description	Filter by *Description.*
Entity	Filter by *Entity.*

Entity Net Object ID	Filter by *Entity Net Object ID.*
Entity Type	Filter by *Entity Type.*
Last Executed Escalation Level	Filter by *Last Executed Escalation Level.*
Name	Filter by *Name.*
Object Type	Filter by *Object Type.*
Real Entity Type	Filter by *Real Entity Type.*
Related Node	Filter by *Related Node.*
Related Node ID	Filter by *Related Node ID.*
Severity	Filter by *Severity.*
Triggered Count	Filter by *Triggered Count.*
Triggered DateTime	Filter by *Triggered DateTime*.
Triggered Message	Filter by *Triggered Message.*

Published Data

This activity publishes the following activity-specific data.

Acknowledged	Indicates if this alert has been acknowledged or not. Note: SolarWinds may only return True or blank for this field. This is an external limitation.
Acknowledged By	The user that acknowledged the alert.
Acknowledged DateTime	The date and time when the alert was acknowledged.
Active Alert ID	Alert instance unique identifier.
Alert Config ID	Alert configuration unique identifier.
Alert Config Message	Alert configuration message.
Alert Note	Alert note.
Alert Trigger Object ID	Alert object unique identifier.
Alert Ref ID	Alert configuration unique identifier.
Description	Alert configuration description.
Entity	Entity associated with the alert.
Entity Net Object ID	Identifies the entity associated with the alert.
Entity Type	Type of entity associated with the alert.
Entity URI	Alert Entity URI.
Is Multi-Object Alert	Indicates if this alert is a multi-object alert. Multi-object alerts (or cumulative alerts) trigger only when a minimum number of objects meet

	the alert trigger condition.
Last Executed Escalation Level	Latest escalation level executed actions.
Name	Alert name as defined in alert configuration.
Object Type	Type of object triggering the alert.
Real Entity Type	Type of entity associated with the alert.
Record Count	Number of records returned by the activity.
Related Node	Related node associated with the alert.
Related Node ID	Unique identifier for the related node associated with the alert.
Severity	Alert severity.
Triggered Count	The number of times the alert has triggered.
Triggered DateTime	Date and time when the alert triggered.
Triggered Message	Alert instance message.

Get Alert History Activity

The **Get Alert History** activity can be used in a runbook to retrieve and filter alert history from your SolarWinds Orion environment.

Required Properties

You must configure the following properties.

Record Limit	Specifies the maximum number of records that the activity should return.

Optional Properties

You can use the following properties, as necessary, to control how this activity runs.

Descending	Specifies that the returned record set should be ordered in descending order. If this property is not specified and *Order By* property is specified, records will be ordered in ascending order. This property is ignored if *Order By* property is not specified.
Order By	Specifies the field that the record set should be ordered by. If not specified, record ordering is arbitrary. By default, when this property is specified, records are ordered in ascending order. To order records in descending order, set *Descending* property to True.

Filters

This activity provides filters, that you can combine, to control which alert history records to retrieve.

Active Alert ID	Filter by *Active Alert ID.*
Alert Config ID	Filter by *Alert Config ID.*
Active History ID	Filter by *Active History ID.*
Alert Note	Filter by *Alert Note.*
Alert Ref ID	Filter by *Alert Ref ID.*
Alert Trigger Object ID	Filter by *Alert Trigger Object ID.*
Description	Filter by *Description.*
Entity	Filter by *Entity.*
Entity Net Object ID	Filter by *Entity Net Object ID.*
Entity Type	Filter by *Entity Type.*
Entity URI	Filter by *Entity URI.*
Event Type	Filter by *Event Type.*
Message	Filter by Message.
Name	Filter by Name.
Object Type	Filter by *Object Type.*

Real Entity Type	Filter by *Real Entity Type.*
Related Node	Filter by *Related Node.*
Related Node ID	Filter by *Related Node ID.*
Severity	Filter by *Severity.*
Timestamp	Filter by *Timestamp*.
Triggered Count	Filter by *Triggered Count.*
Triggered DateTime	Filter by *Triggered DateTime*.

Published Data

This activity publishes the following activity-specific data.

Active Alert ID	Alert instance unique identifier. This value is null when the alert instance corresponding to the alert history record no longer exists.
Alert Config ID	Alert configuration unique identifier.
Alert History ID	Alert history unique identifier.
Alert Note	Alert note.
Alert Ref ID	Alert configuration unique identifier.
Alert Trigger Object ID	Alert object unique identifier.
Description	Alert configuration description.
Entity	Entity associated with the alert.
Entity Net Object ID	Identifies the entity associated with the alert.
Entity Type	Type of entity associated with the alert.
Entity URI	Alert Entity URI.
Is Multi-Object Alert	Indicates if the alert is a multi-object alert. Multi-object alerts (or cumulative alerts) trigger only when a minimum number of objects meet the alert trigger condition.
Message	Alert history message.
Name	Alert name as defined in alert configuration.
Object Type	Type of object that triggered the alert.
Real Entity Type	Type of entity associated with the alert.
Record Count	Number of records returned by the activity.
Related Node	Related node associated with the alert.
Related Node ID	Unique identifier for the related node associated with the alert.
Severity	Alert severity.
Timestamp	Date and time when the alert history record was created.
Triggered Count	The number of times the alert has triggered.

Triggered DateTime	Date and time when the alert triggered. This value is null when the alert instance corresponding to the alert history record no longer exists.

Get Engine Activity

The **Get Engine** activity can be used in a runbook to retrieve and filter engine records from your SolarWinds environment.

Optional Properties

You can use the following properties, as necessary, to control how this activity runs.

Descending	Specifies that the returned record set should be ordered in descending order. If this property is not specified and *Order By* property is specified, records will be ordered in ascending order. This property is ignored if *Order By* property is not specified.
Order By	Specifies the field that the record set should be ordered by. If not specified, record ordering is arbitrary. By default, when this property is specified, records are ordered in ascending order. To order records in descending order, set *Descending* property to True.
Record Limit	Specifies the maximum number of records that the activity should return.

Filters

This activity provides filters, that you can combine, to control which node records to retrieve.

Display Name	Filter by *Display Name*.
Engine ID	Filter by *Engine ID*.
Engine URI	Filter by *Engine URI*.
Is Free	Filter by *Is Free* value.
Server IP Address	Filter by *Server IP Address*.
Server Name	Filter by *Server Name*.
Server URI	Filter by *Server URI*.

Published Data

This activity publishes the following activity-specific data.

Display Name	Specifies the engine display name.
Engine ID	Uniquely identifies the engine.
Engine URI	Engine SWIS URI.
Is Free	Specifies if the engine is free or not.
Server IP Address	Engine server IP address.
Server Name	Engine server name.
Server URI	Engine server SWIS URI.

Get Node Activity

The **Get Node** activity can be used in a runbook to retrieve and filter nodes that have been discovered in your SolarWinds environment.

Optional Properties

You can use the following properties, as necessary, to control how this activity runs.

Descending	Specifies that the returned record set should be ordered in descending order. If this property is not specified and *Order By* property is specified, records will be ordered in ascending order. This property is ignored if *Order By* property is not specified.
Order By	Specifies the field that the record set should be ordered by. If not specified, record ordering is arbitrary. By default, when this property is specified, records are ordered in ascending order. To order records in descending order, set *Descending* property to True.
Record Limit	Specifies the maximum number of records that the activity should return.

Filters

This activity provides filters, that you can combine, to control which node records to retrieve.

Category	Filter by *Category.*
Comments	Filter by *Comments* value.
Custom Status	Filter by *Custom Status* value.
DNS	Filter by *DNS* value.
Description	Filter by *Description.*
Dynamic IP	Filter by *Dynamic IP*.
Engine ID	Filter by *Engine ID*.
Entity Type	Filter by *Entity Type.*
IP Address	Filter by *IP Address.*
IP Address Type	Filter by *IP Address Type.*
Is Orion Server	Filter by *Is Orion Server* value.
Is Server	Filter by *Is Server* value.
Location	Filter by *Location.*
Machine Type	Filter by *Machine Type.*
Node ID	Filter by *Node ID.*
Node Name	Filter by *Node Name.*
Node Uri	Filter by *Node Uri.*

Severity	Filter by *Severity.*
SNMP Version	Filter by *SNMP Version*.
Status	Filter by *Status.*
Status Description	Filter by *Status Description* value.
Subtype	Filter by *Subtype.*
System Description	Filter by *System Description* value.
System Name	Filter by *System Name.*
System Object ID	Filter by *System Object ID.*
System Up Time (sec)	Filter by *System Up Time (sec)* value.
Unmanaged	Filter by *Unmanaged* value.
Vendor	Filter by *Vendor.*

Published Data

This activity publishes the following activity-specific data.

Category	The Node category. Values include Other, Network and Server.
Comments	Node comments custom property.
Custom Status	Custom status
DNS	DNS name for this node.
Description	Node description.
Dynamic IP	Indicates if the node has a dynamic IP address or not.
Engine ID	ID of the engine for the node.
Entity Type	Node entity type.
IP Address	Node IP address.
IP Address Type	Node IP address type.
Is Orion Server	Indicates if the node is the Orion server.
Is Server	Indicates if the node is a server.
Location	Node location.
Machine Type	Node machine type.
Node ID	Uniquely identifies the node.
Node Name	Node name.
Node Uri	SWIS Uri for the node. Uniquely identifies the node.
Record Count	Number of records returned by the activity.
Severity	Node severity.
SNMP Version	SNMP version far an SNMP node.

Status	Node status.
Status Description	Description for the node status value.
Subtype	Node subtype.
System Description	Node system description.
System Name	Node system name.
System Object ID	Node system object ID.
System Up Time (sec)	Number of seconds since the node has been running.
Unmanaged	Indicates if this node is managed or not.
Vendor	Node vendor.

Get Poller Activity

The **Get Poller** activity can be used in a runbook to retrieve and filter poller records from your SolarWinds environment.

Optional Properties

You can use the following properties, as necessary, to control how this activity runs.

Descending	Specifies that the returned record set should be ordered in descending order. If this property is not specified and *Order By* property is specified, records will be ordered in ascending order. This property is ignored if *Order By* property is not specified.
Order By	Specifies the field that the record set should be ordered by. If not specified, record ordering is arbitrary. By default, when this property is specified, records are ordered in ascending order. To order records in descending order, set *Descending* property to True.
Record Limit	Specifies the maximum number of records that the activity should return.

Filters

This activity provides filters, that you can combine, to control which node records to retrieve.

Enabled	Filter by *Enabled* value.
Object	Filter by *Object* value
Object Type	Filter by *Object Type*.
Object ID	Filter by *Object ID*.
Poller ID	Filter by *Poller ID*.
Poller Type	Filter by *Poller Type*.
Poller URI	Filter by *Poller URI*.

Published Data

This activity publishes the following activity-specific data.

Enabled	Specifies if the poller is enabled or not.
Object	Object associated with the poller.
Object Type	Type of object associated with the poller.
Object ID	ID of the object associated with the poller.
Poller ID	Uniquely identifies the poller.
Poller Type	Specifies the type of poller.
Poller URI	Poller SWIS URI.

Get Trigger Object Activity

The **Get Trigger Object** activity can be used in a runbook to retrieve more information about the object that triggered the alert. In the case of multi-object alerts, the activity returns all the objects that contributed to the alert triggering.

Required Properties

You must configure the following properties.

Active Alert ID	Identifies the alert instance for which object details should be returned.

Optional Properties

You can use the following properties, as necessary, to control how this activity runs.

Descending	Specifies that the returned record set should be ordered in descending order. If this property is not specified and *Order By* property is specified, records will be ordered in ascending order. This property is ignored if *Order By* property is not specified.
Order By	Specifies the field that the record set should be ordered by. If not specified, record ordering is arbitrary. By default, when this property is specified, records are ordered in ascending order. To order records in descending order, set *Descending* property to True.
Record Limit	Specifies the maximum number of records that the activity should return.

Filters

This activity provides filters, which you can combine, to control which trigger records to retrieve.

Entity	Filter by *Entity.*
Entity Details URL	Filter by *Entity Details URL*.
Entity Type	Filter by *Entity Type.*
Entity URI	Filter by *Entity URI.*
Real Entity Type	Filter by *Real Entity Type.*
Real Entity URI	Filter by *Real Entity URI*.
Related Node	Filter by *Related Node*
Related Node Details URL	Filter by *Related Node Details URL*.
Related Node URI	Filter by *Related Node URI.*

Published Data

This activity publishes the following activity-specific data.

Entity	Entity associated with the alert.
Entity Details URL	Relative URL for the details view for the alert triggering object.
Entity Type	Type of entity associated with the alert.
Entity URI	URI for the object that triggered the alert.
Record Count	Number of records returned by the activity.
Real Entity Type	Type of entity associated with the alert.
Real Entity URI	URI for the object that triggered the alert.
Related Node	Related node associated with the alert.
Related Node Details URL	Relative URL for the details-view for the related node of the alert triggering object.
Related Node URI	URI for the related node of the object that triggered the alert.

Get Suppressed Activity

The **Get Suppressed** activity can be used in a runbook to retrieve and filter alert suppression records.

Each alert suppression record represents a scheduled occurrence of muting (or pausing) alerts for an entity. While alerts are suppressed (or muted) for a given entity, the entity will not trigger those alerts. The following tables list the properties, published data and filters for this activity.

Required Properties

This activity does not provide any required properties.

Optional Properties

You can use the following properties, as necessary, to control how this activity runs.

Descending	Specifies that the returned record set should be ordered in descending order. If this property is not specified and *Order By* property is specified, records will be ordered in ascending order. This property is ignored if *Order By* property is not specified.
Order By	Specifies the field that the record set should be ordered by. If not specified, record ordering is arbitrary. By default, when this property is specified, records are ordered in ascending order. To order records in descending order, set *Descending* property to True.
Record Limit	Specifies the maximum number of records that the activity should return.

Filters

This activity provides filters, that you can combine, to control which alert suppression records to retrieve.

Alert Suppression ID	Filter by *Alert Suppression ID.*
Entity URI	Filter by *Entity URI.*
Start Time	Filter by *Start Time.*
End Time	Filter by *End Time.*

Published Data

This activity publishes the following activity-specific data.

Alert Suppression ID	Uniquely identifies the alert suppression record.
Entity URI	URI for the entity for which alerts are being suppressed.
Start Time	Date and time when alert suppression comes into effect.
End Time	Date and time when alert suppression ends.
Record Count	Number of records returned by the activity.

Manage Node Activity

The **Manage Node** activity can be used in a runbook to re-enable node management for a SolarWinds node, after it has been placed into an un-managed state.

Required Properties

You must configure the following properties.

Node ID	Node ID for the node that is to be managed.

Published Data

This activity does not publish any activity-specific data.

Monitor Alert Activity

The **Monitor Alert** activity can be used in a runbook to detect when new alerts that have triggered in your SolarWinds Orion environment.

Required Properties

You must configure the following properties.

Monitor Interval (seconds)	Specifies the number of seconds the activity waits between server requests. Minimum is 15 seconds.

Optional Properties

You can use the following properties, as necessary, to control how this activity runs.

Descending	Specifies that the returned record set should be ordered in descending order. If this property is not specified and *Order By* property is specified, records will be ordered in ascending order. This property is ignored if *Order By* property is not specified.
Order By	Specifies the field that the record set should be ordered by. If not specified, record ordering is arbitrary. By default, when this property is specified, records are ordered in ascending order. To order records in descending order, set *Descending* property to True.
Record Limit	Specifies the maximum number of records returned by each monitor query.

Filters

This activity provides filters, that you can combine, to control which alerts will trigger the monitor.

Acknowledged	Filter by *Acknowledged*.
Acknowledged By	Filter by *Acknowledged By* value.
Acknowledged DateTime	Filter by *Acknowledged DateTime.*
Active Alert ID	Filter by *Active Alert ID.*
Alert Config ID	Filter by *Alert Config ID.*
Alert Note	Filter by *Alert Note.*
Alert Trigger Object ID	Filter by *Alert Trigger Object ID.*
Alert Ref ID	Filter by *Alert Ref ID.*
Description	Filter by *Description.*
Entity	Filter by *Entity.*
Entity Net Object ID	Filter by *Entity Net Object ID.*
Entity Type	Filter by *Entity Type.*
Last Executed Escalation	Filter by *Last Executed Escalation Level.*

Level	
Name	Filter by *Name.*
Object Type	Filter by *Object Type.*
Real Entity Type	Filter by *Real Entity Type.*
Related Node	Filter by *Related Node.*
Related Node ID	Filter by *Related Node ID.*
Severity	Filter by *Severity.*
Triggered Count	Filter by *Triggered Count.*
Triggered Message	Filter by *Triggered Message.*

Published Data

This activity publishes the following activity-specific data.

Acknowledged	Indicates if this alert has been acknowledged or not.
Acknowledged By	User that acknowledged the alert.
Acknowledged DateTime	Date and time when the alert was acknowledged.
Active Alert ID	Alert instance unique identifier.
Alert Config ID	Alert configuration unique identifier.
Alert Config Message	Alert configuration message.
Alert Note	Alert note.
Alert Trigger Object ID	Alert object unique identifier.
Alert Ref ID	Alert configuration unique identifier.
Description	Alert configuration description.
Entity	Entity associated with the alert.
Entity Net Object ID	Identifies the entity associated with the alert.
Entity Type	Type of entity associated with the alert.
Is Multi-Object Alert	Indicates if this alert is a multi-object alert. Multi-object alerts (or cumulative alerts) trigger only when a minimum number of objects meet the alert trigger condition.
Last Executed Escalation Level	Latest escalation level executed actions.
Name	Alert name as defined in alert configuration.
Object Type	Type of object triggering the alert.
Real Entity Type	Type of entity associated with the alert.
Record Count	Number of records returned by the activity.
Related Node	Related node associated with the alert.

Related Node ID	Unique identifier for the related node associated with the alert.
Severity	Alert severity.
Triggered Count	The number of times the alert has triggered.
Triggered DateTime	Date and time when the alert has triggered.
Triggered Message	Alert instance message.

Monitor Alert History Activity

The **Monitor Alert History** activity is be used in a runbook to detect when new alert history records have been generated in SolarWinds Orion environment.

Required Properties

You must configure the following properties.

Monitor Interval (seconds)	Specifies the number of seconds the activity waits between monitor queries. Minimum is 15 seconds.

Optional Properties

You can use the following properties, as necessary, to control how this activity runs.

Descending	Specifies that the returned record set should be ordered in descending order. If this property is not specified and *Order By* property is specified, records will be ordered in ascending order. This property is ignored if *Order By* property is not specified.
Order By	Specifies the field that the record set should be ordered by. If not specified, record ordering is arbitrary. By default, when this property is specified, records are ordered in ascending order. To order records in descending order, set *Descending* property to True.
Record Limit	Specifies the maximum number of records returned by each monitor query.

Filters

This activity provides filters, that you can combine, to control which alert history records will trigger the monitor.

Active Alert ID	Filter by *Active Alert ID.*
Alert Config ID	Filter by *Alert Config ID.*
Active History ID	Filter by *Active History ID.*
Alert Note	Filter by *Alert Note.*
Alert Ref ID	Filter by *Alert Ref ID.*
Alert Trigger Object ID	Filter by *Alert Trigger Object ID.*
Description	Filter by *Description.*
Entity	Filter by *Entity.*
Entity Net Object ID	Filter by *Entity Net Object ID.*
Entity Type	Filter by *Entity Type.*
Entity URI	Filter by *Entity URI.*
Event Type	Filter by *Event Type.*

Message	Filter by Message.
Name	Filter by Name.
Object Type	Filter by *Object Type.*
Real Entity Type	Filter by *Real Entity Type.*
Related Node	Filter by *Related Node.*
Related Node ID	Filter by *Related Node ID.*
Severity	Filter by *Severity.*
Triggered Count	Filter by *Triggered Count.*
Triggered DateTime	Filter by *Triggered DateTime*.

Published Data

This activity publishes the following activity-specific data.

Active Alert ID	Alert instance unique identifier. This value is null when the alert instance corresponding to the alert history record no longer exists.
Alert Config ID	Alert configuration unique identifier.
Alert History ID	Alert history unique identifier.
Alert Note	Alert note.
Alert Ref ID	Alert configuration unique identifier.
Alert Trigger Object ID	Alert object unique identifier.
Description	Alert configuration description.
Entity	Entity associated with the alert.
Entity Net Object ID	Identifies the entity associated with the alert.
Entity Type	Type of entity associated with the alert.
Entity URI	Alert Entity URI.
Is Multi-Object Alert	Indicates if the alert is a multi-object alert. Multi-object alerts (or cumulative alerts) trigger only when a minimum number of objects meet the alert trigger condition.
Message	Alert history message.
Name	Alert name as defined in alert configuration.
Object Type	Type of object that triggered the alert.
Real Entity Type	Type of entity associated with the alert.
Record Count	Number of records returned by the activity.
Related Node	Related node associated with the alert.
Related Node ID	Unique identifier for the related node associated with the alert.
Severity	Alert severity.

Timestamp	Date and time when the alert history record was created.
Triggered Count	The number of times the alert has triggered.
Triggered DateTime	Date and time when the alert triggered. This value is null when the alert instance corresponding to the alert history record no longer exists.

Remove Poller Activity

The **Remove Poller** activity can be used in a runbook to delete an existing SolarWinds poller.

Required Properties

You must configure the following properties.

Poller URI	Identifies the poller that is to be deleted.

Published Data

This activity does not publish any activity-specific data.

Resume Alert Activity

The **Resume Alert** activity can be used in a runbook to un-suppress previously suppressed (muted) alerts for a specified entity.

Required Properties

You must configure the following properties.

Entity URI	Specifies the entity for which alerts are to be resumed. You can obtain the Entity URIs for current alert suppressions by running the Get Suppressed activity.

Published Data

This activity does not publish any activity-specific data.

Set Alert Note Activity

The **Set Alert Note** activity can be used in a runbook to set an alert note, without acknowledging the alert.

Required Properties

You must configure the following properties.

Alert Trigger Object ID	Alert Trigger Object ID for the alert. You can specify multiple comma-separated IDs to set notes for multiple alerts with one activity.
Alert Note	Specifies the alert note.

Published Data

This activity does not publish any activity-specific data.

Set Custom Property Activity

The **Set Custom Property** activity can be used in a runbook to set custom property values for various SolarWinds entities.

Required Properties

You must configure the following properties.

Entity	Specifies the type of object for which the custom property is to be set. Note that this list is dependent on the custom properties supporting entities which are deployed in your environment.
<Entity> ID	Identifies the object for which the custom property is to be set. Depending on the selected entity, this may be an Alert ID, or a Node ID, etc.

Optional Properties

This activity will list the Custom Properties that are available to be set, based on the value specified in the *Entity* property.

Published Data

This activity does not publish any activity-specific data.

Suppress Alert Activity

The **Suppress Alert** activity can be used in a runbook to suppress (mute) alerts for a specified entity. While alerts are suppressed for a given entity, the entity will not trigger those alerts. You can use the Resume Alert activity to un-suppress previously suppressed alerts.

Required Properties

You must configure the following properties.

Entity URI	Specifies the entity for which alerts are to be suppressed. You can obtain the Entity URI for an alert instance by running the Get Trigger Object activity.

Optional Properties

You can use the following properties, as necessary, to control how this activity runs.

Start Time	Date and time when alert suppression comes into effect. If not specified, alert suppression comes into effect when the activity completes.
End Time	Date and time when alert suppression ends. If not specified, alert suppression remains into effect until specifically resumed.

Published Data

This activity does not publish any activity-specific data.

Un-Acknowledge Alert Activity

The **Un-Acknowledge Alert** activity can be used in a runbook to un-acknowledge a previously acknowledged alert.

Required Properties

You must configure the following properties.

Alert Trigger Object ID	Alert Trigger Object ID for the alert that is to be un-acknowledged. You can specify multiple comma-separated IDs to un-acknowledge multiple alerts with one activity.

Published Data

This activity does not publish any activity-specific data.

Un-Manage Node Activity

The **Un-Manage Node** activity can be used in a runbook to place a SolarWinds node in the un-managed state, for maintenance purposes. When a node is un-managed, it will not fire alerts and it will pause status data collection.

Required Properties

You must configure the following properties.

Node ID	Alert Trigger Object ID for the alert that is to be un-acknowledged. You can specify multiple comma-separated IDs to un-acknowledge multiple alerts with one activity.

Optional Properties

You can use the following properties, as necessary, to control how this activity runs.

Start Time	Date and time when the node should be placed in un-managed state. If not specified, Start Time will default to the date and time when the activity is running.
End Time	Date and time when the node should be placed back into managed state. If not specified, End Time will default to max DateTime value, effectively placing the node in un-managed state indefinitely.

Published Data

This activity does not publish any activity-specific data.

Update Node Activity

The **Update Node** activity can be used in a runbook to update an existing SolarWinds node.

Required Properties

You must configure the following properties.

Node URI	Identifies the node that is to be updated.

Optional Properties

You can use the following properties, as necessary, to control how this activity runs.

Community	Specifies the SNMP community string for an SNMP node.
Custom Status	Specifies node custom status.
DNS	Specifies the node DNS.
Description	Node description.
Dynamic IP	Specifies if the node has dynamic IP address.
Engine ID	Specifies the Engine ID for the node. You can retrieve the engine objects from your environment by using the *Get Engine* activity.
IP Address	Specifies the IP address for the node.
Is Server	Specifies if the node is a server.
Location	Specifies node location.
Node Name	Node name.
SNMP Version	Specifies the SNMP version for an SNMP node.
Subtype	Specifies the node subtype. Allowed values are: • Agent • ICMP • SNMP • WMI
System Description	System description.
System Name	System name.
Vendor	Node vendor.

Published Data

This activity does not publish any activity-specific data.