INTEGRATION PACK FOR SALESFORCE

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_Salesforce_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_Salesforce_x64_2.0**

User Guide

Version 2.0

Kelverion Integration Pack for Salesforce

Published: December 2023

Feedback

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

Contents

Introduction

The Integration Pack for Salesforce is an add-on for System Center Orchestrator that enables integration with the Salesforce Force.com platform.

System Requirements

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

Kelverion_Integration_Pack_for_Salesforce (32-bit)

- Microsoft System Center Orchestrator 2016, 2019
- Microsoft .NET Framework 4.7.2
- Salesforce.com subscription with REST API access

Kelverion_IP_for_Salesforce_x64 (64-bit)

- Microsoft System Center Orchestrator 2022
- Microsoft .NET Framework 4.7.2
- Salesforce.com subscription with REST API access

Important: The Integration Pack for Salesforce requires Salesforce REST API access, which is not available to all Salesforce editions. API access is available to the Enterprise, Unlimited, Developer and Performance editions. If you are using the Professional edition, API access must be purchased from Salesforce.

Important: The Integration Pack uses the Force.com REST API with OAuth 2.0 authentication. You must set up a Salesforce **connected app** as part of your Salesforce environment. For details, see **Connecting to Salesforce** in the Kelverion Integration Pack for Salesforce User Guide.

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

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:

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

Configuring the Integration Pack

A configuration establishes a reusable link between Orchestrator and the Salesforce application cloud service. You can create as many configurations as you require specifying links to multiple sites. You can also create multiple configurations on the same site to allow for differences in security permissions for different user accounts.

To setup a Salesforce configuration in Orchestrator:

1. In Orchestrator Runbook Designer, click the **Options** menu, and select *KA Salesforce*. The **KA Salesforce** 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 Spectrum server or a descriptive name to distinguish the type of configuration.

4. Click the ellipsis button (…) next to the **Type** box and select *Salesforce Configuration.*

5. In the **Server URL** box, enter the URL to your Salesforce site. For example: https://202103190104170395.my.salesforce.com.

    Note: The lightning URL displayed in your web browser may not work. Instead, use the site URL as configured under Setup > User Interface > Sites and Domains > Domains.

6. In the **Consumer Key** box, enter the consumer key for the connected app used to integrate the integration pack with your Salesforce environment. For details, please see Connecting to Salesforce.

7. In the **Consumer Secret** box, enter the consumer secret for the connected app used to integrate the integration pack with your Salesforce environment. For details, please see Connecting to Salesforce.

8. In the **Session Timeout (Minutes)** box, enter the Salesforce Session Timeout value, **in minutes**, for your Salesforce environment. You can find the Session Timeout in Salesforce at **Setup > Administer > Security Controls > Session Settings**.

    If the value configured in your Session Settings is in hours, you must convert this into corresponding minutes and then enter it in Session Timeout (Minutes) box. **This value should be the same or lower than the Session Timeout value, in minutes, configured in Session Settings on your organization Salesforce site**. By default, this value is set in the Integration Pack to 15 minutes, which is the lowest available setting.
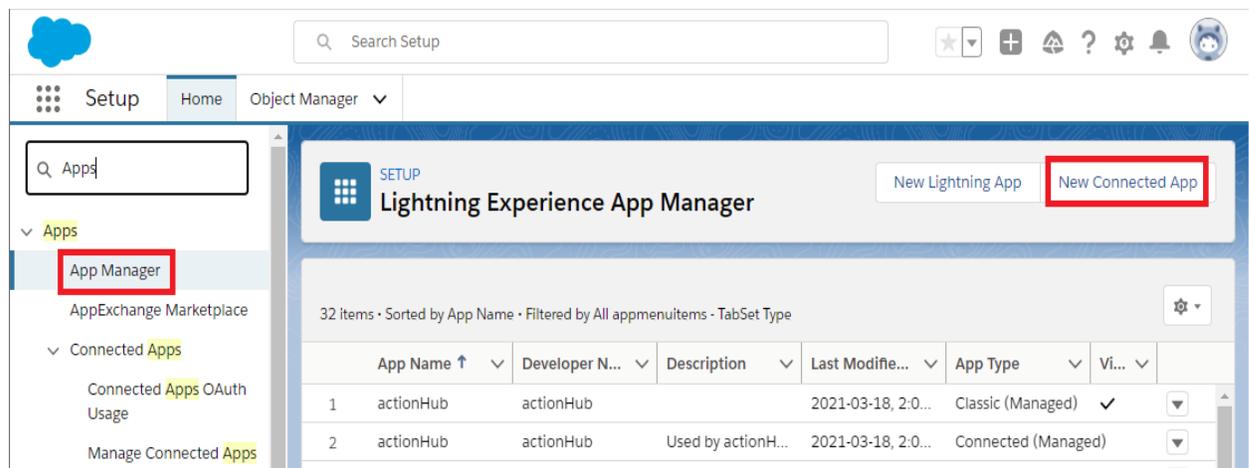
9. Click **OK** to close the configuration dialog box, and then click **Finish**.

Connecting to Salesforce

Salesforce Force.COM REST API uses the OAuth protocol to allow users to securely access their Salesforce environment. For the Integration Pack to gain access to your Salesforce environment, in your Salesforce organization, you must configure a connected app that defines OAuth settings. For more information, see [Create a Connected App](#) in the online documentation for Salesforce.

Use the following steps to create a new connected app. The user performing this should be configured with the **System Administrator** profile.

1. Click on **Setup** on upper right side under the gear menu, and then enter "Apps" in the Quick Find box.

2. Select **App Manager** under Apps.

3. In the **App Manager** click **New Connected App** to create a new connected app.

4. Enter the **Name** of your application, for example **Kelverion Integration**.

5. Enter **Contact Email** information, as well as any other information appropriate for your application.

6. Select **Enable OAuth Settings**.

7. The **Callback URL** is required; however, this is not needed for integrating with the integration pack. You can enter an arbitrary HTTPS URL, for example https://Kelverion_Integration.

8. In the **Selected OAuth Scopes** add **Manage user data via APIs (api)**.

9. Make sure to **Enable Client Credentials Flow**.

10. Click **Save** and then **Continue**.

11. After the connected app is created, click **View** in the App Manager and then click **Manage Consumer Details**.

12. Copy the **Consumer Key** and the **Consumer Secret** and enter them in the respective boxes in IP configuration.

13. Type **Manage Connected** in the **Setup** Quick Find box and click on **Manage Connected Apps** under **Apps > Connected Apps**.

14. Find the newly created Connected App in the list and click **Edit**.

15. Under **Client Credentials Flow** specify a **Run As** user for the Connected App. The Integration Pack will connect to Salesforce in the context of this user. We recommend using a dedicated Salesforce user for the integration, for example **Kelverion Integration Admin**, and assigning appropriate permissions for the integration scope.

Salesforce API Limits

The Kelverion Integration Pack for Salesforce uses the Salesforce Force.com REST API to communicate with your environment. Salesforce imposes several limits on its REST API and the IP is also subject to these limitations and restrictions. For more information, see API Request Limits and Allocations and API calls reports and limits FAQ in the online documentation for Salesforce.

Daily API Calls (Requests) Limit

The main API limit affecting the Integration Pack is the **Daily API Calls (or Requests) limit**. This value controls how many requests IP activities can send to Salesforce application, per day.

The maximum number of daily API calls depends on your Salesforce organization and edition. Once the maximum number of daily API calls has been reached, IP activities will fail with REQUEST_LIMIT_EXCEEDED exception.

The Get Limits Activity

To help control the requests volume of your Salesforce integration, the IP provides the **Get Limits** activity, which can be used to check the status of your daily limits.

Before running the integration pack in a production environment, it is recommended to measure/estimate your typical daily API request volume. If your integration needs tend to reach close to the daily limits for your organization, the Get Limits activity can be used to throttle your Salesforce runbooks such that the API call limit is not reached during periods of high volume.

Concurrent API Calls (Requests) Limit

Another important limit to keep in mind is the Concurrent API Requests limit. This value specifies the maximum number of concurrent calls that the integration pack can make, with a duration of 20 seconds or longer.

To avoid reaching this limit, it is recommended to design Salesforce activities in your workflow sequentially.

Stored Salesforce Information

The Integration Pack uses Salesforce object metadata to configure activities with object and field information. To improve performance and usability, the integration pack stores object and field information locally on the Orchestrator server.

All information stored by the integration pack can be found at the following location:

> **[Program Data Directory]\Kelverion Automation\Integration Pack for Salesforce\Schema_v1.0\[Salesforce Server]**

Folders at this location:

- **Schema** folder contains object and field information.
- **Data** folder contains partial records used by the integration pack to provide friendly names for Reference fields.

Stored data in both folders grows gradually as you access Salesforce objects in the integration pack. It initially starts by storing the Schema/*SObjects.json* file, which contains summary information about all Salesforce objects available in your organization. Afterwards, for each object accessed by the integration pack, detailed information is stored in a corresponding JSON file.

For example, contact information is stored at **Schema/*Contact.json*.** and browser information for the Account ID field for a Contact is stored at **Data/Account/ReferenceFrom/Contact.AccountId.json.**

The information stored by the integration pack includes data such as object and field names, labels and browser list values and IDs. If a new custom object is added in your organization, the information in the **SObjects.json** must be refreshed. The best way to accomplish this is to delete the **SObjects.json** file, and then reload one of the integration pack activities. This will result in the integration pack saving a new up to date **SObjects.json** metadata file.

If the definition of an existing object changes in your organization, for example if a field is added or removed, the stored object schema information must be refreshed. The best way to accomplish this is to identify the corresponding JSON file in the **Schema** folder, delete it, and then reload an integration pack activity which uses the object. This will result in the integration pack saving a new up-to-date object schema file.

Reference Fields

One of the key features of stored object information is that it provides browser information for reference fields. Reference fields are object fields which refer to other objects.

Typically, the Salesforce API used by the integration pack requires internal **Record IDs** to be specified when referencing other records. For example, when creating a new Contact record and specifying an Account for the contact, you would have to provide the **Account ID** value, instead of the **Account Name**. This can be inconvenient and tedious, especially when specifying multiple reference fields.

To make things easier, the integration pack stores browser information for reference fields - a list of records where friendly values are stored along with record IDs. This way, when presenting available choices for reference fields, the integration pack displays friendly values which then internally map to corresponding record IDs at runtime, when invoking the Salesforce API.

When specifying the value for a reference field, unless selecting from the available browser list, always specify the referenced record ID.

Although useful in cases where referenced data remains static, working with stored reference information can become difficult when the referenced object contains many records or when reference object data is continuously changing. To mitigate these problems, the integration pack only store reference field browser list items when *the number of records in the referenced object table is less than 250.* This behavior can be overridden by configuring the **ReferencePicklist** property in the schema file corresponding to the target object.

For example, if the integration pack does not initially provide browser values for the Account ID on the Contact object, you can override this default behavior by making the change *"ReferencePicklist": true* in the in the **Schema/Contact.json** file and restarting the Runbook Designer.

Tip: Setting the **ReferencePicklist** value in the *SObjects.json* file has no effect. You must set the **ReferencePicklist** value in the JSON file corresponding to the target object.

In cases when data in the target tables has changed and the integration pack presents an outdated browser list, you must delete the corresponding data file:

   Data/[Target table]/ReferenceFrom/[Source table].[Source field].json

The integration pack will rebuild the file with up-to-date information next time you access an object which is referring the target object.

Finally, for cases when referenced object tables contain many records, or when data in referenced tables is constantly changing, or for any other cases when browser list information is not available, we recommend using **Get Record** activities to retrieve and filter records from the referenced object table, to obtain the needed **Record ID**.

Salesforce Activities

This integration pack adds the KA Salesforce category to the **Activities** pane in the Client. This category contains the following activities:

- Create Record
- Delete Record
- Get Limits
- Get Record
- Monitor Record
- Update Record

Common Configuration Instructions for All Activities

The following configuration instructions apply to all 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 Client.

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/or **Filters** tab, and a **Run Behavior** tab. Some activities may have additional tabs.

General Tab

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

Properties/Filters Tab

These tabs contain properties that are specific to the activity.

All activities in this integration pack have the **Configuration Name** property at the top of the **Properties** tab. This property is used to specify the connection to a Salesforce server.

To configure the Configuration Name property:

1. Click the ellipsis **(…)** button next to the **Name** field, and then select the applicable connection name. Connections displayed in the list have been previously configured as described in [Configuring the Integration Pack](#).

Filter Behavior

The Monitor and Get activities use filters to determine the values that will invoke a runbook or retrieve activities. When matching against property values, you select one of the available methods of comparison. The Monitor activity will only trigger if all filters match and the Get activity will only return Alarms that match all filters.

- **Equals**: the field of the record exactly matches the text or number specified in the filter.

- **Does not equal**: the field of the record does not exactly match the text or number specified in the filter.

- **Is less than**: the field of the record is less than the number specified in the filter.

- **Is less than or equal to**: the field of the record is less than or equal to the number specified in the filter.

- **Is greater than**: the field of the record is greater than the number specified in the filter.

- **Is greater than or equal to**: the field of the record is greater than or equal to the number specified in the filter.

- **Contains**: the field of the record contains the text specified in the filter.

- **Does not contain**: the field of the record does not contain the text specified in the filter.

- **Starts with**: the field of the record starts with the text specified in the filter.

- **Ends with**: the field of the record ends with the text specified in the filter.

- **Matches**: the field of the record matches the specified pattern expression.

- **Does not match**: the field of the record does not match the specified pattern expression.

When specifying pattern expressions for **Matches** and **Does not match** filters, use the following syntax and rules:

- The % and _ wildcards are supported.

- The % wildcard matches zero or more characters.

- The _ wildcard matches exactly one character.

- Matching is case-insensitive.

- Special characters % or _ can be escaped using backslash.

- Only use backslash to escape a special character.

For example, the filter expression "appl%" matches Appleton, Apple, and Appl, but not Bappl.

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.

Multi-Value Published Data Behavior

The Get 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 Collection Member activity, the data output from that activity might be a list of computers that belong to the specified collection.

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](https://technet.microsoft.com/en-us/library/hh489610.aspx#Event) (https://technet.microsoft.com/en-us/library/hh489610.aspx#Event Notifications).

Published Data

Published data is the foundation of a working runbook. It is the data produced because 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 activities.

To use published data:

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

3. Click the **Activity** drop-down box and select the activity from which you want to obtain the data.

4. To view additional data elements common to all runbooks, select **Show Common Published Data**.

5. 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.

Create Record

The **Create Record** activity can be used in a runbook to insert a new record of the specified object type. The activity will only let you create *creatable* object types,

Required Properties

You must configure the **Object Type** property. This activity may provide additional required properties depending on the **Object Type** that you select.

Object Type	Specifies the object for the new record. Note that the selected object type includes the internal name in square brackets, for example: *Account [Account]*

Optional Properties

Depending on the **Object Type** that was selected, this activity may provide one or more optional properties that you can configure as required.

Published Data

This activity publishes the following activity-specific data items.

Record ID	Identifies the newly inserted record.

Delete Record

The **Delete Record** activity can be used in a runbook to delete a record of the specified object type. The activity will only let you delete object types that support deleting.

Required Properties
You must configure the following properties.

Object Type	Specifies the object for the record to be deleted. Note that the selected object type includes the internal name in square brackets. Ex: *Account [Account]*
Record ID	Identifies the record to be deleted.

Published Data
This activity does not publish any activity-specific data items.

Get Limits

The **Get Limits** activity can be used in a runbook to retrieve Salesforce API limits for your Salesforce environment. The activity returns both the maximum allowed, and the current (remaining) values for the day.

You can use this activity to monitor the status of your daily API calls (requests). If your integration needs tend to reach close to the daily limits for your organization, the Get Limits activity can be used to throttle your Salesforce runbooks such that the API call limit is not reached during periods of high volume. For details, see Salesforce API Limits.

Tip: Running this activity affects the Daily API Requests value, which means that running this activity repeatedly will deplete the value of your daily API calls (requests). It is recommended that you use this activity sparingly, to minimize impact on your daily allowable API calls.

Published Data
This activity publishes the following activity-specific data items.

Daily API Requests Max	The maximum number of Daily API requests (calls) allowed for your organization.
Daily API Requests Remaining	The remaining number of API requests (calls) for the day.
Daily Async Apex Executions Max	The maximum number of daily async Apex executions allowed for your organization.
Daily Async Apex Executions Remaining	The remaining number of async Apex executions for the day.
Daily Bulk API Batches Max	The maximum number of daily Bulk API batches allowed for your organization.
Daily Bulk API Batches Remaining	The remaining number of Bulk API batches for the day.
Data Storage (MB) Max	Maximum data storage for your organization.
Data Storage (MB) Remaining	Remaining data storage for your organization.

Get Record

The **Get Record** activity can be used in a runbook to retrieve records of the specified object type. The activity will only let you retrieve *retrievable* object type.

Required Properties
You must configure the following properties.

Object Type	Specifies the object for the records to be retrieved. Note that the selected object type includes the internal name in square brackets. For example, *Account [Account]*
Publish All Fields	Specifies whether the activity will publish all record fields, or just a subset of *summary* fields. Summary fields typically include record identifier fields, name fields, Create Date and Last Modified Date.
	Tip: Choosing not to publish all record fields can improve performance in cases where specific details for an object are not needed, for example, when only the Record ID for a specific record is required.
	Note that regardless of whether you chose to publish all record fields or not, the activity will provide all available filters for the specified object.

Optional Properties
This activity provides the following optional properties, that you can configure as required.

Descending	When **Order By** is specified, this property specifies whether the returned records will be ordered in ascending or descending order. By default, records are returned in descending order.
Order By	Specifies a field used for ordering the returned records. When not specified, there is no guarantee of the records order.
Record Limit	Specifies the maximum number of records to be returned by the activity.

Filters
This activity will provide appropriate filters, based on the **Object Type** that you selected. You can combine multiple filters to selectively control which objects records to retrieve.

Published Data

This activity publishes the following activity-specific data items. This activity will provide additional activity-specific data items, based on the **Object Type** that was selected and whether **Publish All Fields** is enabled.

Number of Records	The number of records retrieved by the activity.

Monitor Record

The **Monitor Record** activity can be used in a runbook to detect when new records are created and/or existing records are updated in your Salesforce environment.

Required Properties
You must configure the following properties.

Object Type	Specifies the object for the records to be retrieved. Note that the selected object type includes the internal name in square brackets. For example: *Account [Account]*
Publish All Fields	Specifies whether the activity will publish all record fields, or just a subset of "summary" fields. Summary fields typically include record identifier fields, name fields, Create Date and Last Modified Date.
	Choosing not to publish all record fields can improve performance in cases where specific details for an object are not needed, for example, when only the Record ID for a specific record is required.
	Note that regardless of whether you chose to publish all record fields or not, the activity will provide all available filters for the specified object.
Monitor New Records	Specifies whether the activity will detect newly created records.
Monitor Updated Records	Specifies whether the activity will detect updates to existing records.
Monitor Interval (seconds)	Specifies the number of seconds the activity waits between server requests. Minimum is 15 seconds.
	Tip: When configuring the monitor interval, please be aware of the API call limits for your organization. For details, see [Salesforce API Limits](#).

Monitor Record Optional Properties
You can configure the following properties, as required.

Descending	When *Order By* is specified, this property specifies whether the returned records will be ordered in ascending or descending order. When not specified, the activity returns records in ascending order.
Order By	Specifies a field used for ordering the returned records. When not specified, there is no guarantee of the records order.
Record Limit	Specifies the maximum number of records to be returned by the activity.

Filters
This activity will provide appropriate filters, based on the **Object Type** that you selected. You can combine multiple filters to selectively control which objects will trigger the monitor.

Published Data

This activity publishes the following activity-specific data items. This activity will provide additional activity-specific data items, based on the **Object Type** that was selected and whether **Publish All Fields** is enabled.

Number of Records	The number of records retrieved by the activity.

Update Record

The **Update Record** activity can be used in a runbook to update an existing Salesforce record of the specified object type. The activity will only let you update *updateable* object types.

Required Properties
You must configure the following properties.

Object Type	Specifies the object for the record to be updated. Note that the selected object type includes the internal name in square brackets. For example: *Account [Account]*
Record ID	Identifies the record to be updated.

Optional Properties
This object will provide optional fields based on the **Object Type** that was selected. You can use these fields to update the Salesforce record as required.

Tip: To reset a field value to null, specify **$null** as the activity property value.

Published Data
This activity does not publish any activity-specific data items.