Microsoft Azure Sentinel – ServiceNow app installation guide

Table of Contents

[Introduction 2](#_Toc74303540)

[Prerequisites 2](#_Toc74303541)

[Azure: Create the Service Principal 2](#_Toc74303542)

[Delegate permissions to the Service Principal 5](#_Toc74303543)

[ServiceNow: create a user for Azure Sentinel 6](#_Toc74303544)

[Installation 8](#_Toc74303545)

[Import the application in ServiceNow 8](#_Toc74303546)

[Configure the OAuth credentials 10](#_Toc74303547)

[Configure the application 12](#_Toc74303548)

[Classes 17](#_Toc74303549)

[Alerts 17](#_Toc74303550)

[getIncidentAlerts 17](#_Toc74303551)

[alertsToHtmlTable 17](#_Toc74303552)

[AppUtils 17](#_Toc74303553)

[CustomMapping 17](#_Toc74303554)

[Entities 17](#_Toc74303555)

[getIncidentEntities 17](#_Toc74303556)

[entitiesToHtmlTable 17](#_Toc74303557)

[getEntitiesByType 18](#_Toc74303558)

[SentinelIncidents 18](#_Toc74303559)

[getSentinelIncidents 18](#_Toc74303560)

[createIncidents 18](#_Toc74303561)

[updateSentinelIncident 18](#_Toc74303562)

[updateChangedIncidents 18](#_Toc74303563)

[createUrlForObject 19](#_Toc74303564)

[getIncidentComments 19](#_Toc74303565)

[addIncidentComments 19](#_Toc74303566)

# Introduction

This integration is a ServiceNow application providing bi-directional synchronization of incidents, between both platforms. It has the following capabilities:

* Incident creation (Azure Sentinel to ServiceNow only)
* Incident alerts synchronization
* Incident entities synchronization
* Incident comments synchronization
* Incident status synchronization
* Incident severity synchronization
* Incident owner assignment synchronization
* Incident custom properties support (requires custom code)

# Prerequisites

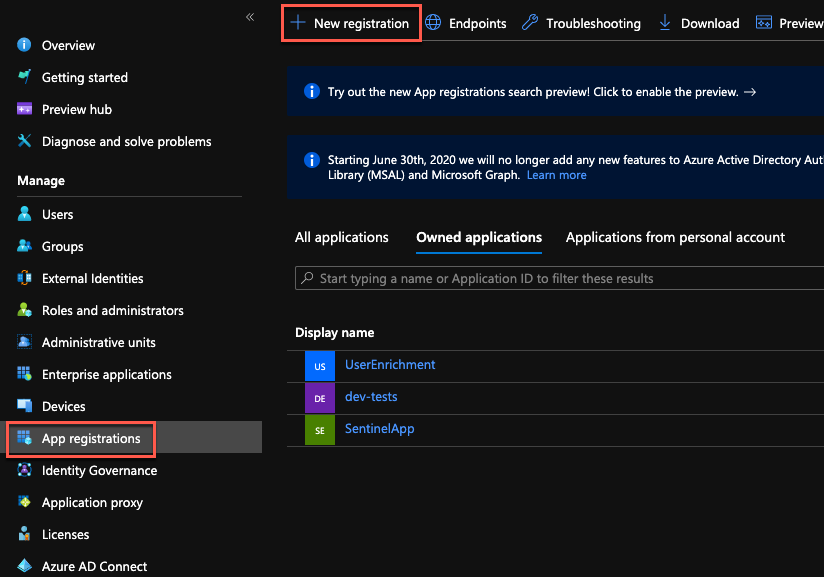
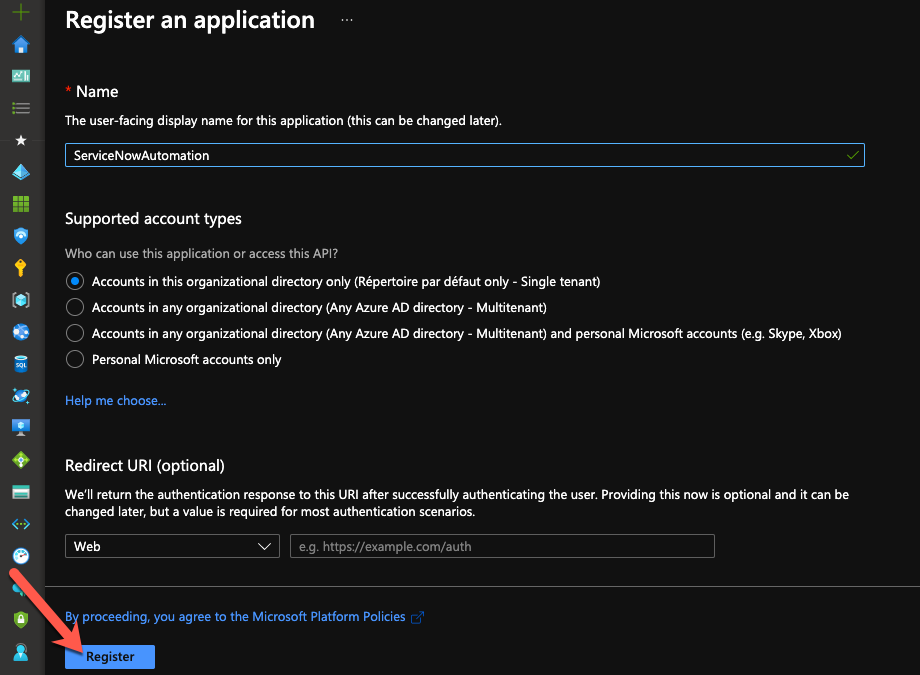
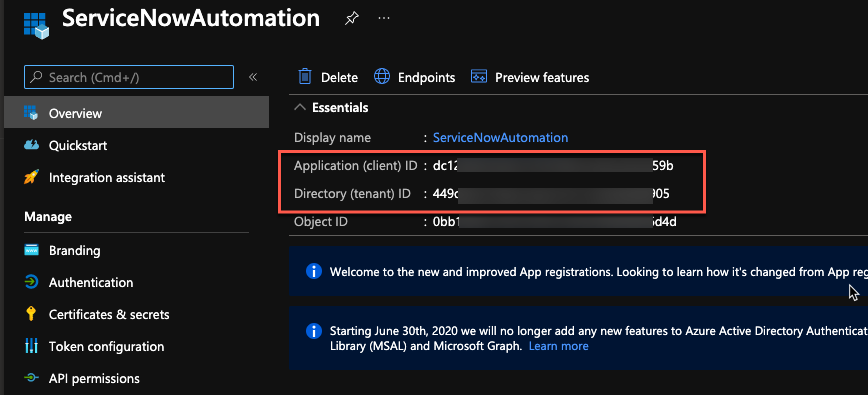
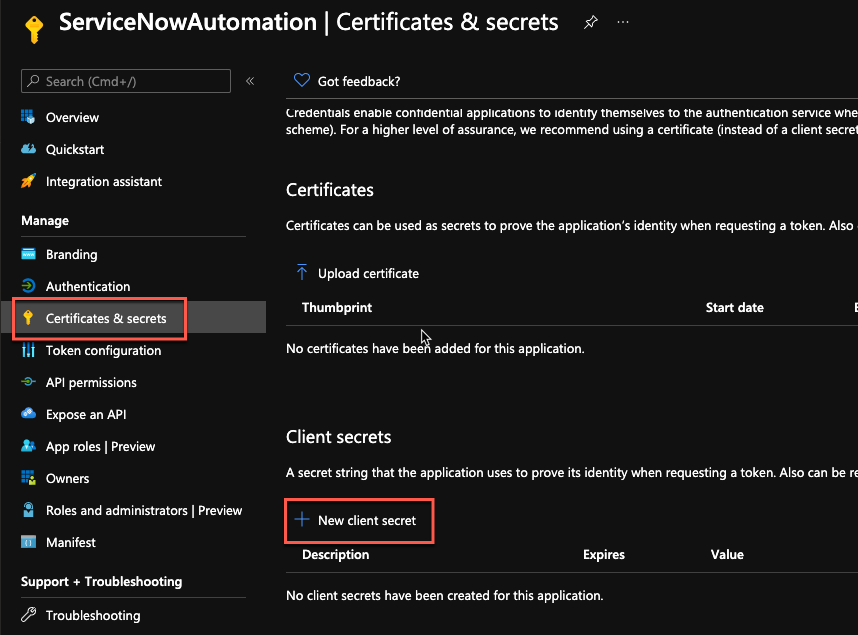
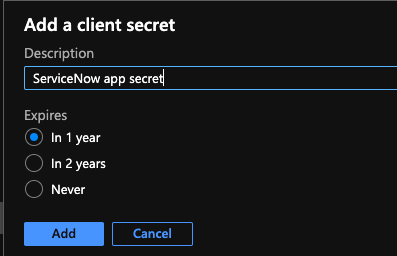
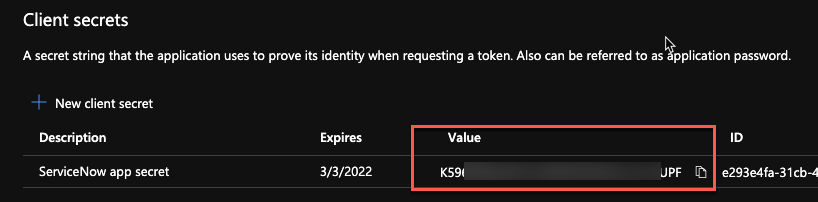
This ServiceNow application fully rely on the Azure Sentinel **Management API** to provide bi-directional sync between both platforms.

thumbnail image 1 of blog post titled 
 
 
  
 
 
 
    
  
   
    
      
       Azure Sentinel API 101
       
      
     
   
  
 
   
 
 
 
 
 

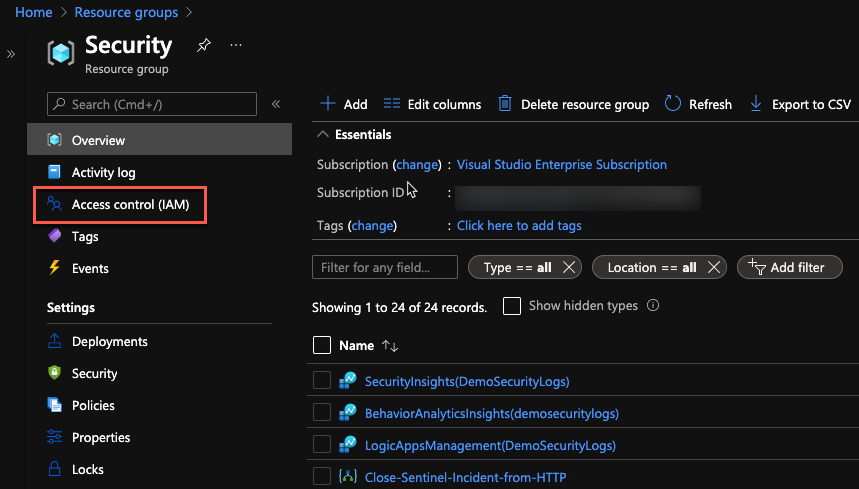
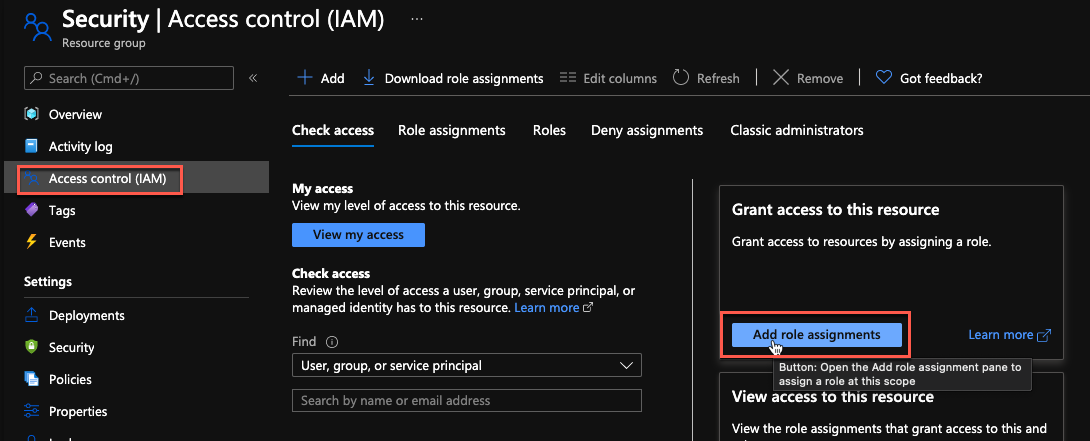
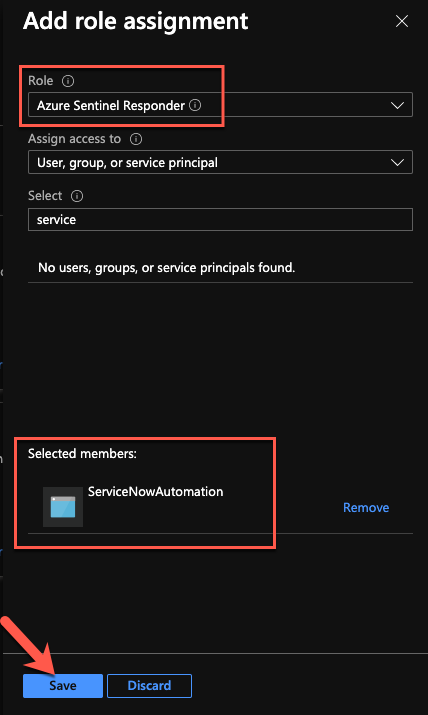

Azure Sentinel APIs reference

To provide access to our application, we have to create a Service Principal in Azure Active Directory, and assign to it the required permissions.

## Azure: Create the Service Principal

1. Go to the Azure portal, in Azure AD service, App Registrations:  
   <https://portal.azure.com/#blade/Microsoft_AAD_IAM/ActiveDirectoryMenuBlade/RegisteredApps>
2. Click on “New registration”.  
   
3. Provide a name for the app and click “Register”.  
   
4. Take note of the Application (client) ID and Directory (tenant) ID. We’ll need them during the ServiceNow configuration.
5. Go to “Certificates & secrets” and click on “New client secret”.  
   
6. Provide a name for the secret and a validity period.  
   Important: when the secret will expire, you’ll have to create a new one and update the ServiceNow configuration.  
   
7. Note the secret and keep it in a safe location for later use.  
   

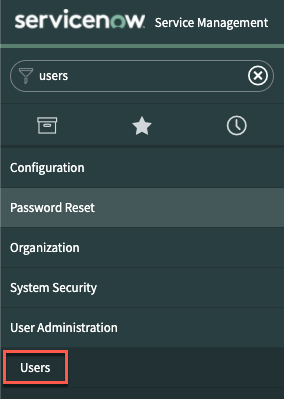
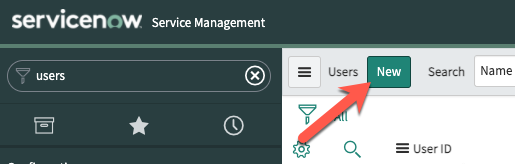
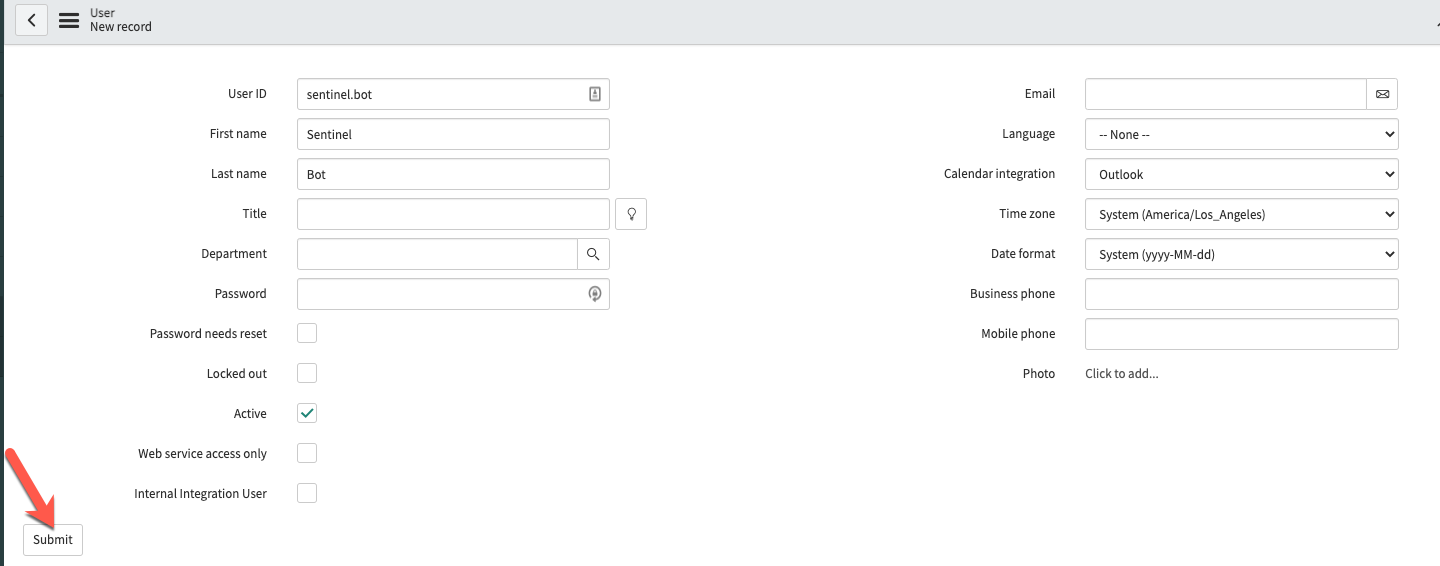
## Delegate permissions to the Service Principal

1. In the Azure portal, go to the Resource Group containing your Azure Sentinel workspace and click on “Access control (IAM)”.  
   
2. Click on “Add role assignments”.  
   
3. In the new blade, select the “Azure Sentinel Responder” role, then select the Service Principal we created before, and click on the “Save” button.  
   

We are now done with the Azure configuration part.

## ServiceNow: create a user for Azure Sentinel

To identify the incidents created from Azure Sentinel incidents, we will create a user. This user will be used as the “caller\_id” property, when creating new records.

1. In ServiceNow, under “User Administration”, click on “Users”.  
   
2. Click on the ”New” button.  
   
3. Provide the required details and click on “Submit”.  
   

# Installation

## Import the application in ServiceNow

1. Search for “update set” and select the “Retrieved Update Sets” module. Then, click on the “**Import Update Set from XML**” link.

Graphical user interface, text, application

Description automatically generated

1. Click on the “Choose File” button and select the application XML file.  
   Then, click on the “Upload” button.

Graphical user interface

Description automatically generated

1. Once uploaded, you will see the new imported update set. Click on it to open it.

Graphical user interface, application

Description automatically generated

1. Click on the “Preview Update Set” button.

Graphical user interface

Description automatically generated

Shape, rectangle

Description automatically generated

1. Click on the “Commit Update Set” button.

Graphical user interface, application

Description automatically generated

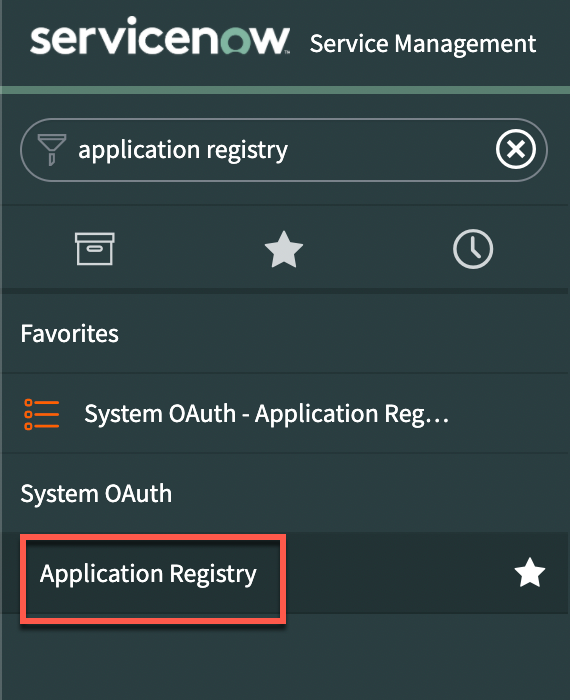
Rectangle

Description automatically generated

The application is now imported and is available in “Studio”.

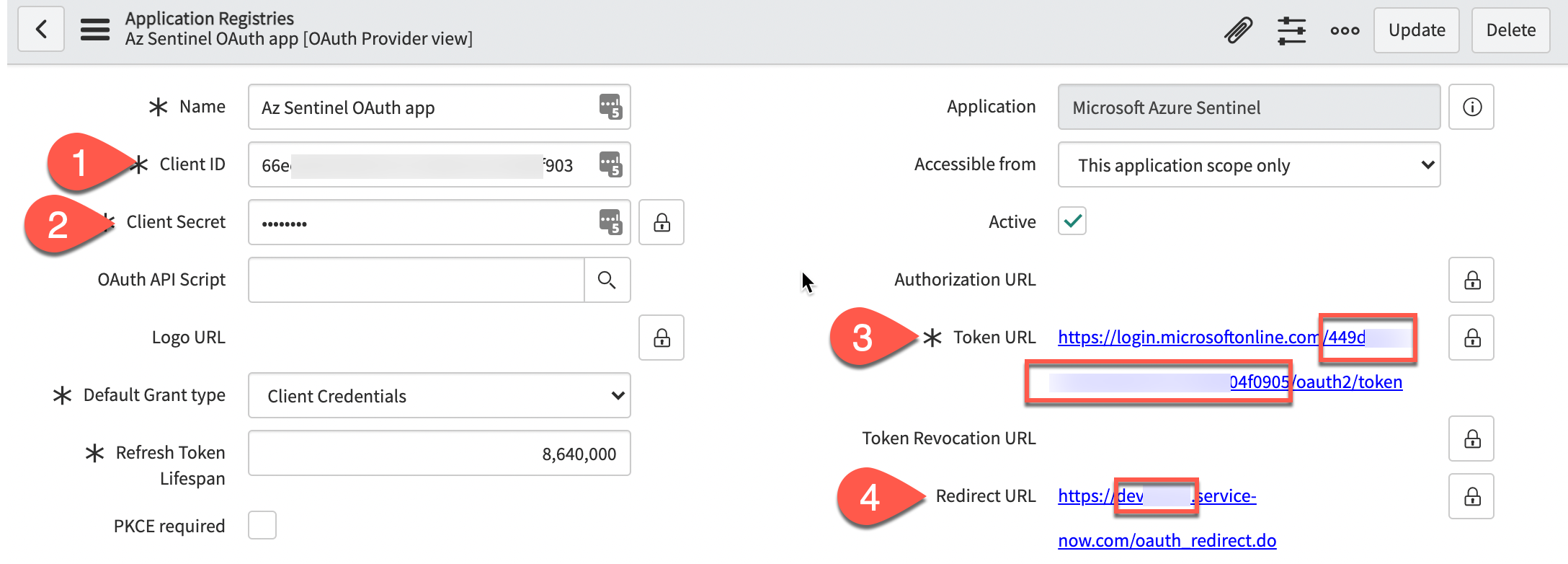
## Configure the OAuth credentials

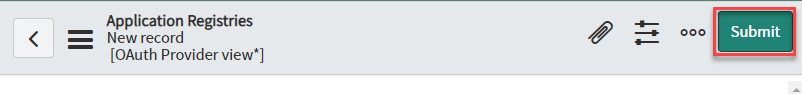
To be able use the Azure Sentinel Management API from ServiceNow, we must configure the credentials we created previously in Azure AD. This is done using an “Application Registry”.

1. Search for “Application Registry” and click on the link.  
   
2. Create a new set of credentials. We’ll use “Az Sentinel OAuth app” but you can use any name you want.  
   Graphical user interface, text, application

   Description automatically generated.
3. Select “Connect to a third party OAuth Provider”.  
   Graphical user interface, text, application

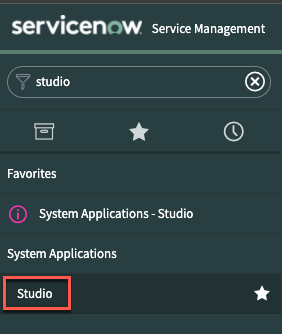
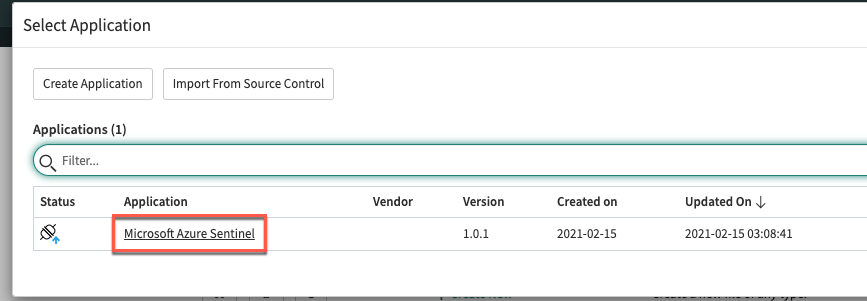
   Description automatically generated
4. On the credentials configuration page, we must provide the information we collected during the Service Principal creation:
   * Name: Az Sentinel OAuth app (can be different. This is the default name used by the workspace configuration)
   * Client ID (1): Azure AD application/client ID
   * Client secret (2): Azure AD client secret
   * Default Grant type: Client Credentials
   * Token URL (3): add your Azure AD tenant ID in the URL:  
     [https://login.microsoftonline.com/{AAD\_tenant\_id}/oauth2/token](https://login.microsoftonline.com/%7bAAD_tenant_id%7d/oauth2/token)
   * Token Revocation URL (4): add your ServiceNow instance name in the URL:  
     https://{instance\_name}.service-now.com/oauth\_redirect.do



1. Click on the “Submit” button to save your changes.  
   

## Configure the application

Now that we have imported the application, we must configure the details to connect to the Azure Sentinel Management API.

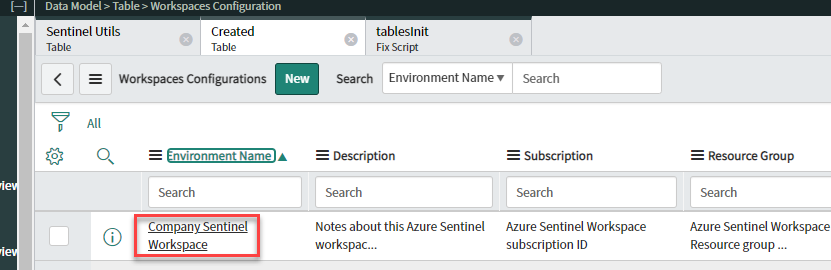
1. Search for “Studio” and open it. Then, select the newly imported application.  
     
   
2. Scroll to the “FixScript” section and run the “tablesInit” script.  
   This script will populate the tables  
   Graphical user interface, text, application

   Description automatically generated  
     
   Graphical user interface, application

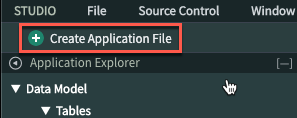
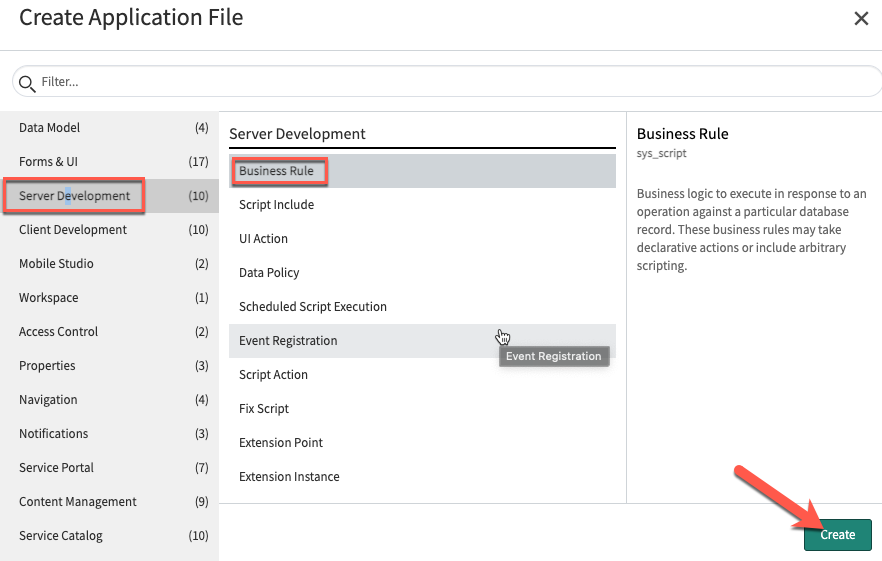
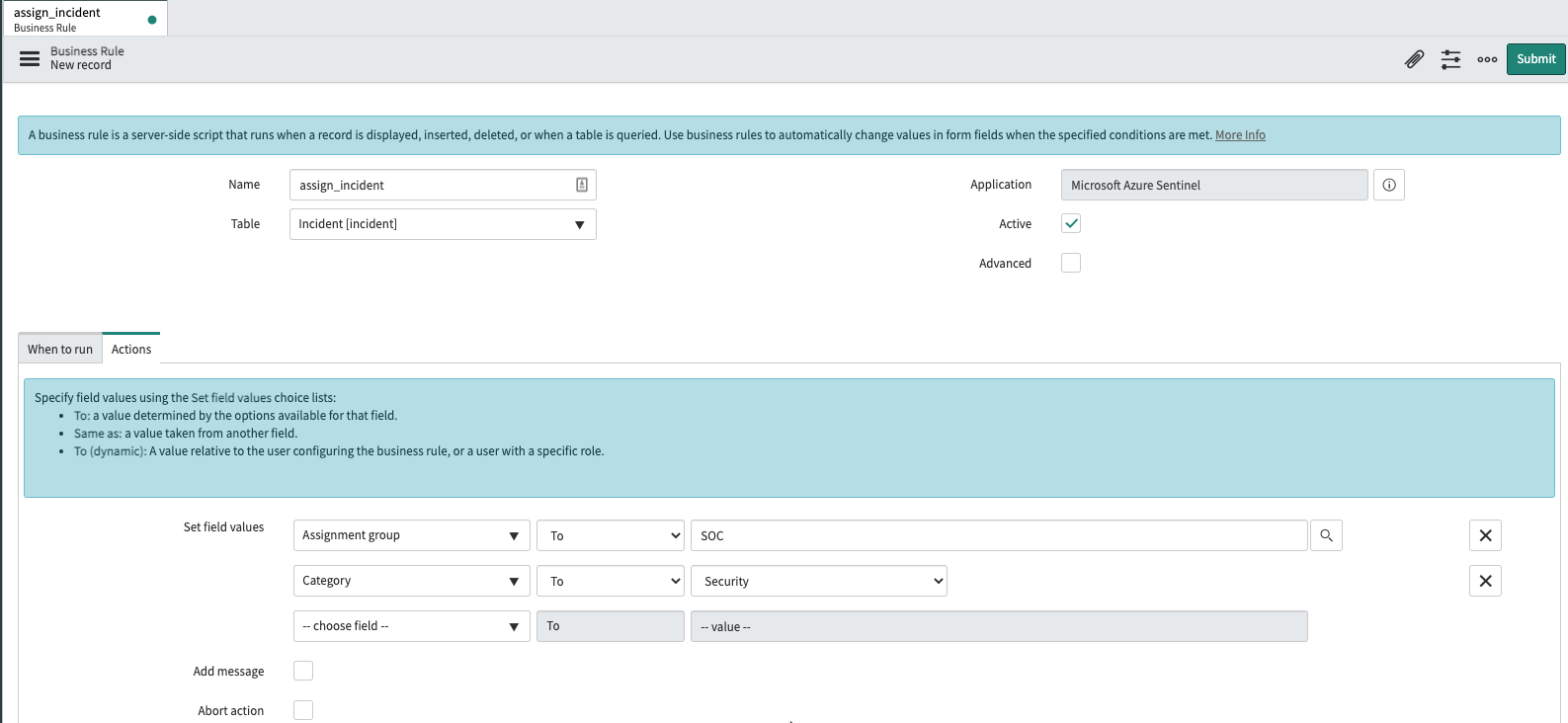
   Description automatically generated  
     
   Graphical user interface, text, application

   Description automatically generated
3. Verify the “Sentinel Severity to ServiceNow” table mapping.  
   This table is used to map the Sentinel severity to the ServiceNow value, when creating or updating Azure Sentinel incidents.  
   Open the table => Show list, and review the values.
4. Verify the “Sentinel State to ServiceNow” table mapping.  
   This table is used to map the Sentinel state to the ServiceNow value, when creating or updating Azure Sentinel incidents.  
   Open the table => Show list, and review the values.
5. Verify the “ServiceNow Severity to Sentinel” table mapping.  
   This table is used to map the ServiceNow severity to the Sentinel value, when updating ServiceNow incidents.  
   Open the table => Show list, and review the values.
6. Verify the “ServiceNow State to Sentinel” table mapping.  
   This table is used to map the ServiceNow severity to the Sentinel value, when updating ServiceNow incidents.  
   Open the table => Show list, and review the values.
7. Configure the workspace(s) details. Open the “Workspaces Configuration” table and click on **Show List**.  
   Graphical user interface, text

   Description automatically generated
8. Open to edit the current row.



1. Provide the required values (available in Azure Sentinel) and click on the **Update** button.  
   Graphical user interface, text, application

   Description automatically generated
2. If needed, create a new Business Rule to assign specific properties, like the “Assignment Group” or the “Category”.  
   This can be achieved by clicking on “Create Application File”, selecting “Server Development”, “Business Rule”.  
     
     
     
   

# Classes

## Alerts

This class contains functions to manipulate incidents alerts.

### getIncidentAlerts

Function used to retrieve an incident’s alerts.

Parameters:

* Environment: Sentinel environment for which we want to retrieve the alerts
* IncidentId: incident for which we want to retrieve the alerts
* Format: used to specify the format to return (json or html)
* LastSync: used to filter the incident alerts, based on the last time we queried the Sentinel API

### alertsToHtmlTable

Function formatting a list of alerts to an html table.

Parameters:

* Alerts: list of alerts objects

## AppUtils

## CustomMapping

## Entities

This class contains functions to manipulate an incident entities, like users, Ips, hosts and more.

### getIncidentEntities

Function used to retrieve an incident’s entities.

Parameters:

* Environment: Sentinel environment for which we want to retrieve the alerts
* IncidentId: incident for which we want to retrieve the alerts
* Format: used to specify the format to return (json or html)

### entitiesToHtmlTable

Function formatting a list of entities to an html table.

Parameters:

* Alerts: list of alerts objects

### getEntitiesByType

Function returning a list of entities of a specific type (ex: users, Ips, …).

Parameters:

* Entities: list of entities to filter
* Type: type of entities to return

## SentinelIncidents

This class contains the functions to retrieve Sentinel incidents, create or update incidents in ServiceNow and Sentinel, plus supporting functions for those operations.

### getSentinelIncidents

Function retrieving Azure Sentinel incidents, based on a time and custom filter (by default, a specific tag), stored in the environment configuration table.

Parameters:

* Environment: Azure Sentinel environment from which we want to retrieve the incidents
* Id: Azure Sentinel incident ID. Used when you want to get a specific incident details
* Operation: used to specify if we want to retrieve the new (created) or updated incidents, when calling the API

### createIncidents

Function creating ServiceNow incidents, based on the Sentinel incidents.

Parameters:

* Environment: Azure Sentinel environment from which the incidents has been retrieved
* Incidents: Azure Sentinel incidents to create in ServiceNow

### updateSentinelIncident

Function updating an Azure Sentinel incident, based on the ServiceNow incident changes to sync.

Parameters:

* Environment: Azure Sentinel environment where the incident to update is located
* incidentId: Id of the Azure Sentinel incident to update
* properties: changes from the ServiceNow incident to sync to Azure Sentinel

### updateChangedIncidents

Function updating a ServiceNow incident, based on the Azure Sentinel changes, like the status, severity or comments.

Parameters:

* Environment: Azure Sentinel environment where the incident is located
* modifiedIncidents: list of incidents to update in ServiceNow
* modifiedLastSync: timestamp from the updated incidents last sync. Used when retrieving latest comments or alerts

### createUrlForObject

Function used to generate the direct link to the ServiceNow incident, before being added to Azure Sentinel comments.

Parameters:

### getIncidentComments

Function used to retrieve comments from an Azure Sentinel incident, based on the last sync.

Parameters:

### addIncidentComments

Function used to add new comments to an Azure Sentinel incident, once a new work note is added to a ServiceNow incident.

Parameters:

## 