

IBM Resilient SOAR Platform

QRadar Advisor Guide

V2.0.2



Date: September 2020

Licensed Materials – Property of IBM

© Copyright IBM Corp. 2010, 2020.  All Rights Reserved.

US Government Users Restricted Rights: Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

IBM Resilient SOAR Platform   
QRadar Integration Guide

|  |  |  |
| --- | --- | --- |
| Version | Publication | Notes |
| 2.0.2 | September 2020 | Support added for App Host, support added for proxies, updated deprecated API endpoints, bug fixes |
| 2.0.1 | February 2019 | Bug fixes for Python 3 |
| 2.0 | April 2019 | Supports the V2.0 release |
| 1.0.1 | March 2019 | For Watson Search fixed version compatibility with search that returns no data, and fix typo in post-process script |
| 1.0 | August 2018 | Initial publication. |

Table of Contents

[Overview 4](#_Toc49951293)

[Check Prerequisites 4](#_Toc49951294)

[Configure QRadar Advisor 5](#_Toc49951295)

[Cyber Adversary Framework Mapping Application Configuration 6](#_Toc49951296)

[Install the Integration 7](#_Toc49951297)

[Function descriptions 8](#_Toc49951298)

[Example of Watson Search with Local Context 10](#_Toc49951299)

[Example of Watson Search 16](#_Toc49951300)

[Example of QRadar Advisor Offense Analysis 18](#_Toc49951301)

[Example of mapping QRadar 25](#_Toc49951302)

Overview

Resilient Functions simplify development of integrations by wrapping each activity into an individual workflow component. These components can be easily installed, then used and combined in Resilient workflows. The Resilient platform sends data to the function component that performs an activity then returns the results to the workflow. The results can be acted upon by scripts, rules, and workflow decision points to dynamically orchestrate the security incident response activities.

This guide describes the QRadar Advisor Integration Function.

Backed by IBM Watson, QRadar Advisor applies artificial intelligence to automatically investigate indicators of compromise (IOC), utilizes cognitive reasoning to provide critical insights, and ultimately accelerates the response cycle. It can augment a security analyst to gain a head start in assessing incidents and reduce the risk of missing threats.

QRadar Advisor Integration Function enables Resilient users to gather Cyber Threat Intelligence(CTI) data from IBM Watson and QRadar. This information is critical for effective identification of potential IOC and quick response to incidents. In addition, this integration receives MITRE ATT&CK information from QRadar Advisor. As a result, an example workflow of this integration depends on the MITRE ATTACK function integration.

QRadar Advisor integration includes four functions:

* Perform a Watson Search on an indicator and retrieve suspicious observables related to it.
* Perform a Watson Search with Local Context on an indicator and retrieve a cyber threat intelligence (CTI) report on it in Structured Threat Information eXpression ([*STIX2*](https://stixproject.github.io/)) format.
* Perform an analysis on a QRadar offense, and retrieve CTI data from QRadar Advisor and IBM Watson in STIX format.
* Map a given QRadar rule to MITRE ATT&CK tactics.

The package also includes workflow examples to demonstrate the usage of the above functions.

The remainder of this document describes the functions and how to configure them in custom workflows or using the configuration file.

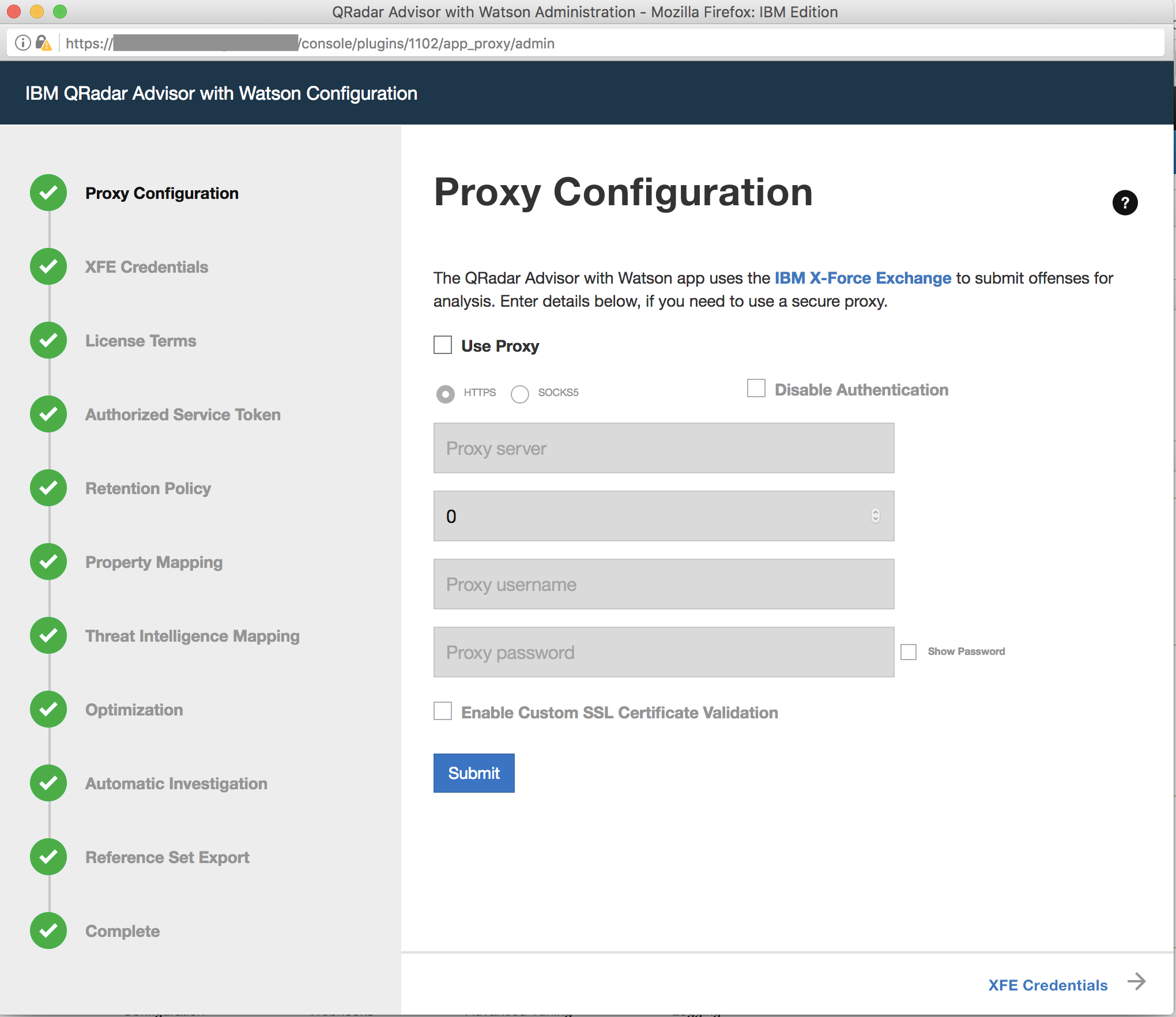
Check Prerequisites

Before installing, verify that your environment meets the following prerequisites:

* Resilient platform is version 31 or later.
* You have a Resilient account or API key to use for the integrations. This can be any account or API key that has the permission to view and modify administrator and customization settings, and read and update incidents. You need to know the account username and password or API key and secret.
* If you are not installing on a Resilient platform configured with an App Host, you have access to a Resilient integration server. An *integration server* is the system that you use to deploy integration packages to the Resilient platform. See the [Resilient Integration Server Guide (PDF)](https://github.com/ibmresilient/resilient-reference/blob/master/developer_guides/Integration%20Server%20Guide.pdf) for more information.

# Configure QRadar Advisor

You need to have QRadar Advisor installed to a QRadar server, and fully configured, as shown in the following configuration page.



To access the QRadar Advisor REST API, you need to know its app\_id, which you can access by clicking the QRadar Advisor’s Configuration icon. For example, in the URL address shown in the configuration page screenshot, the app\_id is 1102 for this QRadar Advisor instance.

You also need an access token to use the REST API. You can obtain access tokens from the Authorized Service Token section of the Admin page.

Cyber Adversary Framework Mapping Application Configuration

QRadar Advisor 2.0 comes with Cyber Adversary Framework Mapping Application (CAFM). This needs to be properly configured as well. From the Admin page of QRadar, select the Configuration page for CAFM. Click the configuration button on the top right corner, and then enter an authorization token. This token can be the same as the authorization token used for QRadar Advisor above.A screenshot of a social media post

Description automatically generated

Write down the app id for CAFM. It is shown in the URL address of this page. For the example above, the app id for CAFM is 1051. This app id is needed in the app.config file.

Install the Integration

If installing the integration on an integration server, follow the procedures in the [Resilient Integration Server Guide (PDF)](https://github.com/ibmresilient/resilient-reference/blob/master/developer_guides/Integration%20Server%20Guide.pdf). To complete the installation, you need to perform the following:

1. Edit the [fn\_qradar\_advisor] section of the resilient-circuits configuration file, as follows:

qradar\_host=host of your QRadar server with QRadar Advisor installed

qradar\_advisor\_token=qradar token (res-keyring protected recommended)

qradar\_advisor\_app\_id=qradar app id for qradar advisor

verify\_cert=[true|false] whether to validate the QRadar server cert

﻿qradar\_cafm\_token=qradar token (res-keyring protected recommended)

qradar\_cafm\_app\_id=qradar app id for CAFM

#optional settings

full\_search\_timeout=timeout for full search in seconds (1200 default)

full\_search\_period=period for full search in seconds (5 default)

offense\_analysis\_timeout=timeout for analysis in seconds (1200 default)

offense\_analysis\_period=period for analysis in seconds (5 default)

# Settings for access to Qradar Advisor via a proxy

#http\_proxy=http://proxy:80

#https\_proxy=https://proxy:80

1. (Recommended) Use res-keyring to store the qradar advisor token:
   1. Instead of storing your token in plaintext, use this instead in your app.config for the token

qradar\_advisor\_token=^qradar\_advisor\_token

qradar\_cafm\_token=^qradar\_cafm\_token

* 1. Run the following command from a terminal in the same folder of your app.config

res-keyring

* 1. Follow the prompt to enter your token.

Function descriptions

This package contains the following functions, example workflows and rules that invoke those functions.

|  |  |  |
| --- | --- | --- |
| **Function** | **Example Workflow** | **Rule** |
| Watson Search | Example of Watson Search | Watson Search |
| Watson Search with Local Context | Example of Watson Search with Local Context | Watson Search with Local Context |
| QRadar Advisor Offense Analysis | Example of QRadar Advisor Offense Analysis | QRadar Advisor Offense Analysis |
| QRadar Advisor Map Rule | Example of mapping QRadar Rule | Map QRadar rule |

In addition, the package contains two custom data tables called “QRadar Advisor analysis results” and “Watson Search with Local Context results”. They are used by the example workflows to show the observables that are extracted from the QRadar Advisor STIX response. Two demo scripts and two associated rules are also included. Each rule is a menu item added to its own data table. The user can click on a rule to create an artifact based on the selected row.

|  |  |  |
| --- | --- | --- |
| **Data Table** | **Rule** | **Script** |
| QRadar Advisor analysis results | Create Artifact (QRadar Advisor Analysis) | Create Artifact for QRadar Advisor Analysis Observable |
| Watson Search with Local Context results | Create Artifact (Watson Search with Local Context) | Create Artifact for Watson Search with Local Context |

Once the function package deploys the functions, you can view them in the Resilient platform Functions tab, as shown below.

A screenshot of a social media post

Description automatically generatedThe package also includes example workflows and rules that show how the functions can be used. You can copy and modify these workflows and rules for your own needs.

A screenshot of a cell phone

Description automatically generated

A screenshot of a cell phone

Description automatically generated

The example workflows shown above demonstrate how to use the functions included in the integration package, as explained below.

Example of Watson Search with Local Context

This example workflow invokes the function “Watson Search with Local Context”. The function calls the QRadar Advisor REST API to perform a Watson Search with Local Context on an indicator.

To use this example workflow and rule included in the package for this function, the user needs to create an incident and add an artifact. For this function to work, the artifact type must correspond to one indicator type. QRadar Advisor supports searches on the following indicators:

* IP addresses
* Hashes
* Domains
* URLs
* Persons

QRadar Advisor supports three return stages:

* Stage1: feature hunt
* Stage2: cognitive investigation added on top of the result of stage 1
* Stage3: wider feature hunt added on top of the result of stage 2

The user can specify the desired return stage in the pre-process script of the example workflow.

A screenshot of a social media post

Description automatically generated

The search REST API of QRadar Advisor returns CTI information in Structured Threat Information Expression (STIX 2.0) format. It is normally a STIX bundle with STIX objects. The function processes the STIX data and performs the following:

* Generates a HTML representation of the STIX data
* Extracts observables from the STIX objects
* Generates a summary from the STIX data

The return data from this function includes the raw STIX data in json dictionary format.

In the post-process script, the HTML representation is used to create a note. The observables are used to populate the custom data table, “Watson Search with Local Context results”, and the summary is used to create a task. Note that the raw STIX data is accessible from the post-process script as results.stix, and can be parsed to create custom code.

A screenshot of a social media post

Description automatically generated

In the following example, a User Account artifact was added to an incident with value “jsmith”. The user can then select Watson Search with Local Context from the artifact menu to search QRadar Advisor for the observable.

A screenshot of a social media post

Description automatically generated

Please note that both Watson Search and Watson Search with Local Context perform queries for information about an indicator. Therefore, only those artifacts that can be mapped into indicators are supported. The types of artifacts that can be searched include:

* DNS Name
* Malware SHA-256 Hash
* Malware SHA-1 Hash
* Malware MD5 Hash
* IP Address
* URL
* User Account

The menu item Watson Search with Local Context is the only one shown for the artifact type listed above.

Note that a full search like this could take up to 15 minutes. Once it is completed, the note created for this indicator can be viewed from the Notes tab of this incident.

A screenshot of a cell phone

Description automatically generated

Please note that the icons shown in the above note use external URL referencing to the official site for STIX2 icons (https://raw.githubusercontent.com/freetaxii/stix2-graphics/master/icons/png\_standard). Therefore, those icons are shown only if the Resilient platform can access the above website.

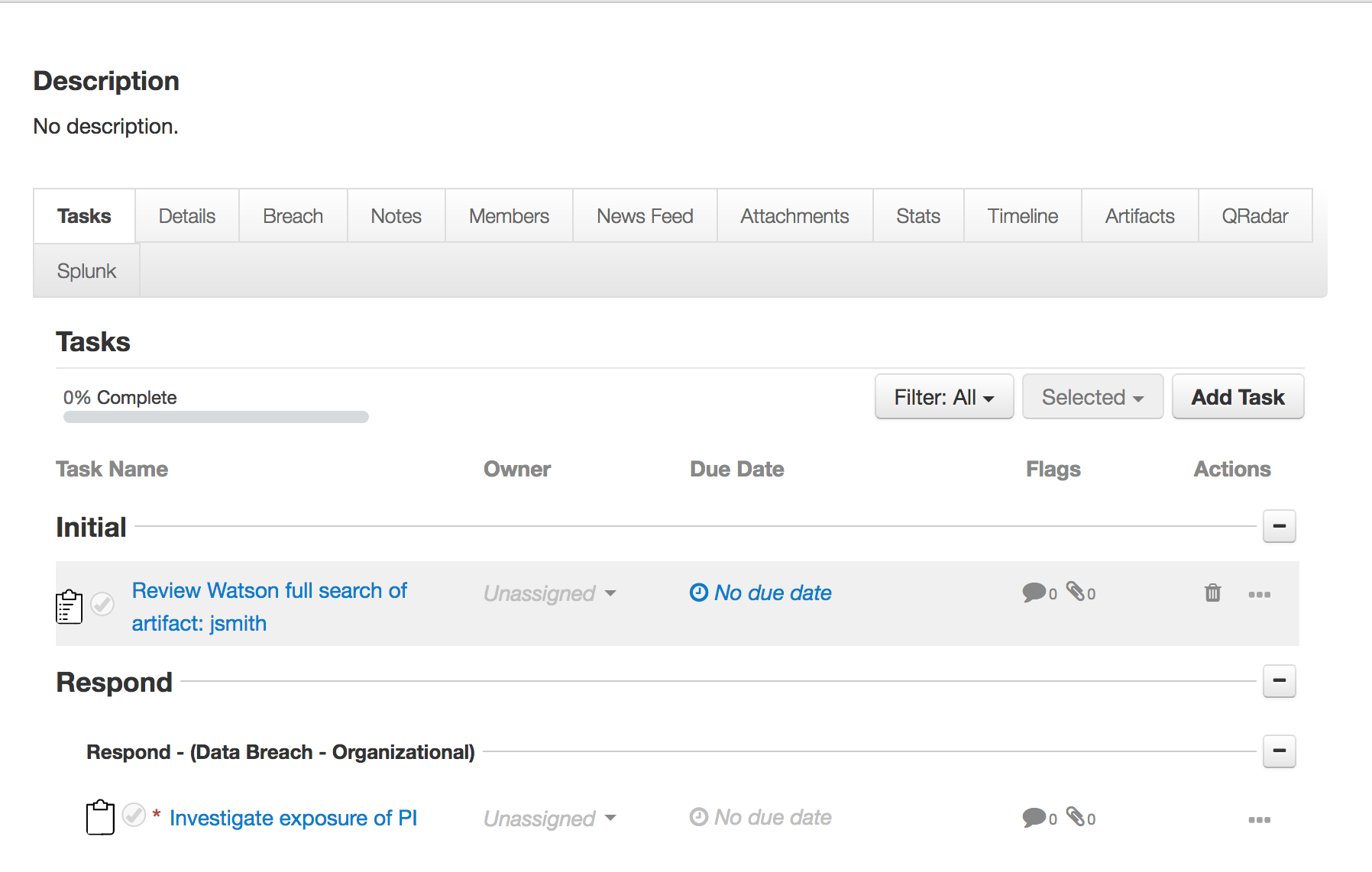
Also note that some indicators have a link icon at the end. These indicators are basically placeholders for the other (real) indicators with the same value. Think of them as symbolic links in a folder tree.

The data table can be viewed if the user adds the “Watson Search with Local Context results” data table into one tab of an incident. Note that this package includes a rule, “Create Artifact (Watson Search with Local Context)”, which is added to the “Watson Search with Local Context results” data table. This enables the user to create an artifact based on a selected row from this data table as shown below.

A screenshot of a social media post

Description automatically generated

The newly created task can be viewed from the Tasks tab.



Since Watson Search with Local Context could potentially take a long time to complete depending on the performance of QRadar Advisor, additional configuration settings are available in the app.config file.

|  |  |
| --- | --- |
| **Setting** | **Explanation** |
| full\_search\_timeout | Timeout in seconds. It is the time the function waits for the result returned from QRadar Advisor. It is optional, and defaulted to 1200 seconds if absent. |
| full\_search\_period | In seconds. It specifies how often the function checks the search status. It is optional, and defaulted to 5 seconds if absent. |

Example of Watson Search

This example workflow invokes the function “Watson Search”. The function calls the QRadar Advisor REST API to perform a quick search on an indicator.

To use this example workflow, the user creates an incident and then adds an artifact with the desired artifact type as shown in the Watson Search with Local Context function.

The QRadar Advisor REST API for Watson Search returns data in JSON format. The JSON dictionary contains two lists, one for suspicious\_observables, and the other for other\_observables. In the post-process script of this example workflow, the suspicious\_observables are mapped to default artifact types, using a dictionary defined there. The user can easily map observables to custom artifacts by modifying the dictionary mapping.

Note the other\_observables are not used in this example workflow. If user wants to make use of them, they can be accessed in the post-process script as results.other\_observables.

In the following example, a Watson Search on the artifact, “domain.com”, is initiated when selecting Watson Search from the artifact menu

A screenshot of a cell phone

Description automatically generated

For this example, the Watson Search of “domain.com” does not return any suspicious observables. As a result, no new artifacts are added. A note was added to the incident to summarize this.

A screenshot of a social media post

Description automatically generated

Example of QRadar Advisor Offense Analysis

This example workflow invokes two functions.

|  |  |  |
| --- | --- | --- |
| **Function** | **Explanation** | **Outputs** |
| QRadar Advisor Offense Analysis | Call QRadar Advisor API to perform the following:   * get the insights of a QRadar Advisor offense. * perform analysis of the offense. | * QRadar Advisor Observable data table * Incident note * Task |

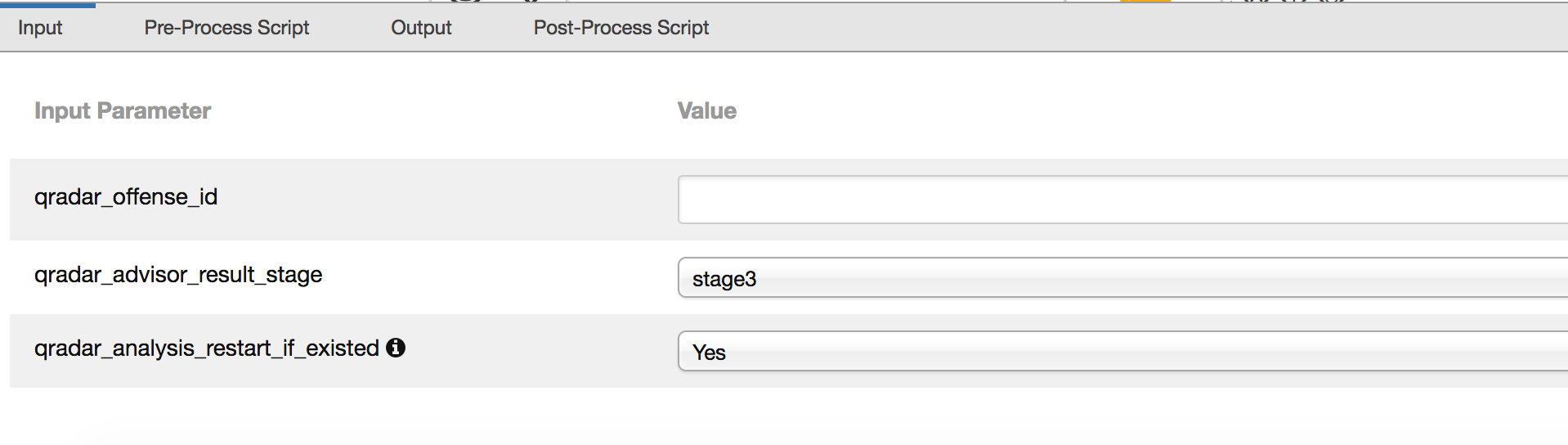
A screenshot of a cell phone

Description automatically generated

The QRadar Advisor return of insights is in JSON format, and the result of an analysis is in STIX format.

Similar to the Watson Search with Local Context, the QRadar Advisor Offense Analysis generates a HTML representation of the STIX data. It also extracts observables from the STIX objects.

Just like Watson Search with Local Context, the user can also specify the return stage from the pre-process script of the example workflow.



One more setting is qradar\_analysis\_restart\_if\_existed. If this flag is set to Yes, the function restarts a new analysis even if a previous result exists for this offense.

In the post-process script of the “QRadar Advisor Offense Analysis”, the HTML representation is used to create a note. The observables are used to populate the “QRadar Advisor analysis results” data table. The insights are used to create a task. The MITRE ATT&CK tactic information is written to a custom field, “mitre\_tactic\_name”, which is displayed in this example as “MITRE ATT&CK Tactic name.

A screenshot of a social media post

Description automatically generated

The raw STIX data from QRadar Advisor is accessible from the post-process script as results.stix, if the user wants to create custom code to parse the STIX data.

To use the example workflow, a Resilient incident must have a valid QRadar offense ID stored in a custom field, qradar\_id. In the following example, the incident is linked to QRadar offense 2696.

The offense analysis begins upon selection of the rule “QRadar Advisor Offense Analysis” from the Action menu of the incident.

A screenshot of a social media post

Description automatically generated

A normal analysis can take up to 20 minutes. Once completed, the HTML representation is shown in the Notes tab.

A screenshot of a cell phone

Description automatically generated

Observables are added to the data table, “QRadar Advisor analysis results”. A menu rule is included for this data table. Users can use it to create a new artifact based on the selected row.

The MITRE ATT&CK tactic name(s) are assigned to the custom field “MITRE ATT&CK Tactic name”.

A screenshot of a social media post

Description automatically generated

A task is created based on the insights returned from QRadar Advisor. The insights are kept in the instruction of the task.

A screenshot of a social media post

Description automatically generated

Since an analysis could potentially take a long time to complete depending on the performance of QRadar Advisor, additional configuration settings are available in the app.config file.

|  |  |
| --- | --- |
| **Setting** | **Explanation** |
| offense\_analysis\_timeout | Timeout in seconds. It is the time the function waits for the result returned from QRadar Advisor. It is optional, and defaults to 1200 seconds if absent. |
| offense\_analysis\_period | In seconds. It specifies how often the function checks the analysis status. It is optional, and defaults to 5 seconds if absent. |

Example of mapping QRadar

This example workflow invokes the following function.

|  |  |  |
| --- | --- | --- |
| **Function** | **Explanation** | **Outputs** |
| QRadar Advisor Map Rule | Call the QRadar CAMF API to map a given QRadar rule to a MITRE tactic. | MITRE ATT&CK Tactic name |

A screenshot of a social media post

Description automatically generated

To use this workflow example, a user needs to enter a QRadar rule into the custom field qradar\_rule first, then select “Map QRadar rule”.

A screenshot of a social media post

Description automatically generated

In the above example, there is an example QRadar rule named “RF Risklist Source Log”. This rule detects if a local host contacts an external IP that is in a reference set called “RF Risklist”. If so, it stores the local IP into a reference set. QRadar Advisor maps this rule to a MITRE ATT&CK tactic called “Initial Access” and the returned name is entered into the custom field MITRE ATT&CK Tactic name.

A screenshot of a social media post

Description automatically generated