

IBM Resilient SOAR Platform

QRadar Functions Guide

V2.0.7

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.

Resilient SOAR Platform QRadar Functions Guide

| Version | Publication | Notes |
|----------------|--------------------|-----------------------------------------------|
| 2.0.7 | June 2020 | Correct typos. |
| 2.0.6 | May 2020 | Add option to return all results from Search. |
| 2.0.4 | April 2020 | Additional configuration notes. |
| 2.0 | March 2019 | Supports the 2.0 release. |
| 1.0 | July 2018 | Initial publication. |

Table of Contents

- Overview 5
 - What’s New in V2.05
- Installation 6
- Function Descriptions 8
 - QRadar Search8
 - QRadar Add Reference Set Item10
 - QRadar Delete Reference Set Item14
 - QRadar Find Reference Set Item14
 - QRadar Find Reference Sets15
- Troubleshooting 16
- Support..... 17

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 Function.

The QRadar integration with the Resilient platform package provides the following:

- Search function to perform a QRadar Ariel query
- Search function to query an item in a QRadar reference set
- Search function to find all the reference sets that contain an item
- Add function to insert a new item in a QRadar reference set
- Delete function to remove an item from a QRadar reference set

With the above functions, this package includes example workflows that demonstrate how to call the functions, rules that start the example workflows, and custom data tables updated by the example workflows.

What's New in V2.0

The following components were added to the function package:

- Function: "QRadar Find Reference Sets"
- Workflow: "Example of finding all QRadar reference sets for artifact"
- Rule: "Find All QRadar Reference Sets"
- Data table: "QRadar Reference Sets"

Installation

Before you install the IBM Resilient QRadar integration package, make sure that your environment meets the following prerequisites:

- QRadar version 7.2.8 or later.
- Your Resilient platform version is 30 or later. If supporting the Resilient for MSSPs multi-organization feature, Resilient platform V33 or later is required.
- A Resilient integration server running Resilient Circuits V30 or later. To setup an integration server, see <https://ibm.biz/res-int-server-guide>.
- A dedicated Resilient account to use as the API user. This can be any account that has the permission to create incidents, and view and modify administrator and customization settings. You need to know the account username and password.

NOTE: Should you later change the dedicated Resilient account to another user, the new user must also have the permission to edit incidents, in addition to the permission to create incidents and view and modify administrator and customization settings. The edit permission is necessary so that the integration can continue to modify or synchronize the incidents escalated by the original user account.

If supporting the Resilient for MSSP feature, the Resilient account must have permission to access the configuration, global dashboard and all child organizations.

Perform the following procedure to install the IBM Resilient QRadar package.

1. Download the IBM Resilient QRadar .zip file from the [IBM Security App Exchange](#).
2. Copy the zip file to your Integration Server and SSH into it.
3. Unzip the package:

```
unzip fn_qradar_integration-x.x.x.zip
```

4. Change directory into the unzipped directory:

```
cd fn_qradar_integration-x.x.x
```

5. Install the package:

```
pip install fn_qradar_integration-x.x.x.tar.gz
```

6. Import the configurations into your file:

```
resilient-circuits config -u
```

7. Import the fn_qradar_integration customizations into your Resilient platform:

```
resilient-circuits customize -y -l fn-qradar-integration
```

8. Open the config file, scroll to the bottom and edit your [fn_qradar_integration] configurations:

```
host=<qradar url>
username=<qradar access user>
qradarpassword=<qradar access password, key-ring protection recommended>
verify_cert=[true|false]
qradartoken=xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxx
#search_timeout=
```

Use 'false' for self-signed certificates.

You have two choices for authentication with QRadar – username and a password, or a token. QRadar token is an authentication token created in QRadar’s Authorized Services manager.

If the username and password and the token are given, the username and password authentication takes precedence.

9. Save and close the app.config file.

10. Optionally, run selftest to test the integration you configured:

```
resilient-circuits selftest -l fn_qradar_integration
```

11. Run Resilient Circuits or restart the service on Linux or Windows.

```
resilient-circuits run
```

Function Descriptions

Once the function package deploys the functions, workflows, rules, and message destination, you can view them in the Resilient platform in their respective tabs. destination tab,

QRadar Search

This sample function performs an Ariel query to fetch data from the QRadar server.

The screenshot shows the configuration page for the 'QRadar Search' function. At the top, there are navigation tabs: Layouts, Rules, Scripts, Workflows, Functions (selected), Message Destinations, Phases & Tasks, Incident Types, Breach, and Artifacts. Below the tabs, the breadcrumb 'Functions / qradar_search' is visible. On the right, there are buttons for 'Cancel', 'Save & Close', and 'Save'. The main configuration area includes fields for 'Name' (QRadar Search), 'API Name' (qradar_search), 'Message Destination' (fn_qradar_integration), and 'Description' (Search QRadar for events). To the right of these fields, there is a section for 'Creator' (Resilient Sysadmin), 'Last Modified' (05/08/2020 16:41), 'Last Modified By' (Resilient Sysadmin), and 'Associated Workflows' (Example of searching QRadar events using offense id). Below the configuration fields, there are two sections: 'Inputs' and 'Input Fields'. The 'Inputs' section lists various parameters like qradar_query, qradar_query_param1 through qradar_query_param5, qradar_query_range_start, qradar_query_range_end, and qradar_query_all_results. The 'Input Fields' section shows a search bar and a list of the same parameters with edit icons.

This function takes the following parameters:

- **qradar_query**: Query to perform. It contains demo template queries you can select from within the calling workflow. The demo queries contain parameters that are replaced by the `qradar_query_param[n]` below. For example, one template query is: *SELECT %param1% FROM events WHERE INOFFENSE(%param2%) LAST %param3% MINUTES*. Users can set values for `qradar_query_param1`, `qradar_query_param2`, and `qradar_query_param3` in the workflow.
- **qradar_query_param[n]**: Optional. Parameters used in the query.
- **qradar_query_range_start**: Optional. An integer specifying QRadar return start range.
- **qradar_query_range_end**: Optional. An integer specifying QRadar return end range. The workflow (object type = Incident), "Example of searching QRadar events using offense id", sets the function's input fields and runs the function. The Input tab maps the function's input fields.
- **qradar_query_all_results**: Optional. If selected, `qradar_query_range_end` and `qradar_query_range_start` will not be used and instead all the results will be returned from the search.

| Input Parameter | Value |
|----------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| qradar_query ⓘ | SELECT %param1% FROM events WHERE INOFFENSE(%param2%) LAST %param3% MINUTES |
| qradar_query_param1 | DATEFORMAT(starttime, 'YYYY-MM-dd HH:mm') as StartTime, CATEGORYNAME(category), LOGSOURCENAME(logsourceid), PROTOCOLNAME(protocolid), RULENAME(creeventlist) |
| qradar_query_param2 | |
| qradar_query_param3 | 43320 |
| qradar_query_param4 | |
| qradar_query_param5 | |
| qradar_query_range_start | 1 |
| qradar_query_range_end | 5 |
| qradar_query_all_results ⓘ | No |

In the Pre-Process Script, the input field, “qradar_query_param2” is set to the incident’s custom field, “qradar_id” and, if the rule is to be run manually, “qradar_query_all_results” is set to the value from the activity field, otherwise the value set under Input in the workflow will be used.

| Input | Pre-Process Script | Output | Post-Process Script |
|-------|--------------------|--------|---------------------|
|-------|--------------------|--------|---------------------|

Language: Python Theme light ▼ Mode Default ▼ Tab Size 2 ▼ - Font + Font

```

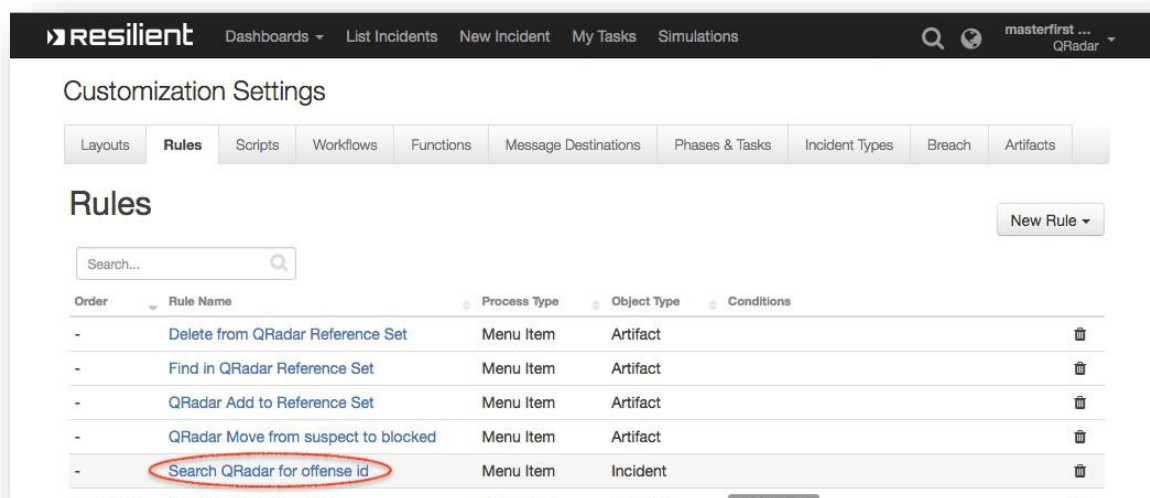
1 inputs.qradar_query_param2 = incident.properties.qradar_id
2 if rule.properties.qradar_query_all_results:
3     inputs.qradar_query_all_results = rule.properties.qradar_query_all_results

```

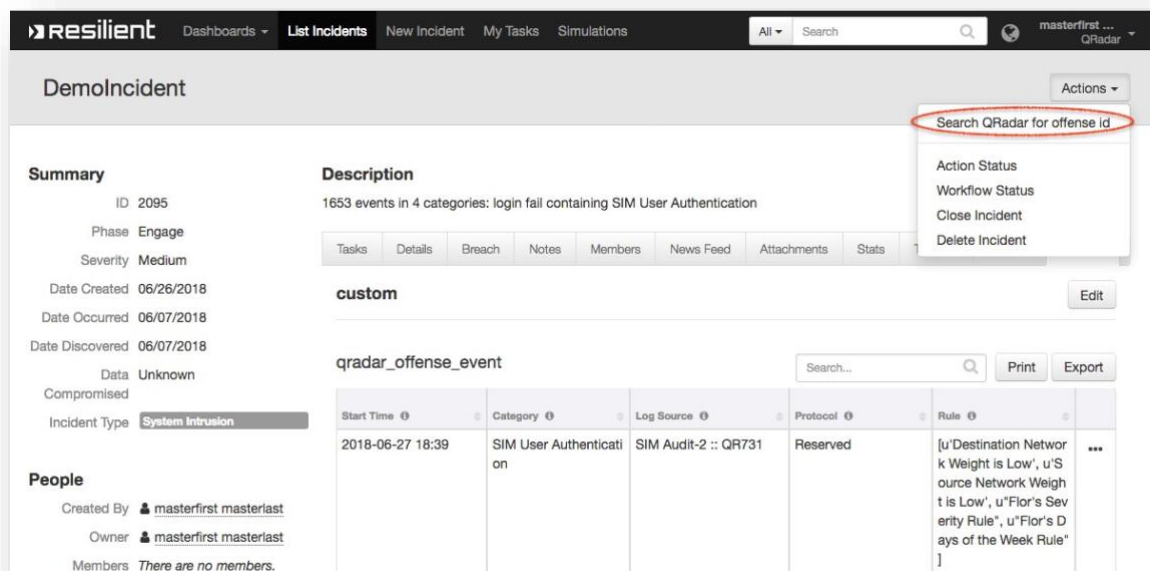
NOTES:

- You should create the custom field, qradar_id, and add it to your layout for accessibility, along with the custom data tables.
- If a reduced number of QRadar events are returned when the search is run in Resilient, allocate more time to the Resilient input parameter: qradar_query. This will allow Resilient to return the same number of QRadar events.

The example rule, “Search QRadar for offense id”, is a menu item rule for an Incident. The user can select this menu item to initiate the workflow.



This workflow can work together with the Resilient QRadar App. An incident escalated from a QRadar offense stores the offense ID. This workflow can make use of that offense ID to perform an Ariel query and update a custom data table, which is also included in the package.



QRadar Add Reference Set Item

This function adds a new item to an existing QRadar reference set. It uses two input parameters: `qradar_reference_set_name` is the name of an existing reference set in QRadar, and `qradar_reference_set_item_value` is the value to be added. The input is populated by the workflow, “Example of adding an item to QRadar reference set”.

resilient

Dashboards

List Incidents

New Incident

My Tasks

Simulations

AllSearch

masterfirst ...
QRadar

Customization Settings

Layouts

Rules

Scripts

Workflows

Functions

Message Destinations

Phases & Tasks

Incident Types

Breach

Artifacts

Functions / qradar_add_reference_set_item

Name *

QRadar Add Reference Set Item

API Name *

qradar_add_reference_set_item

Message Destination *

fn_qradar_integration

Description

Add an item to a given QRadar reference set

Creator

masterfirst masterlast

Last Modified

06/27/2018 18:26

Last Modified By

masterfirst masterlast

Associated Workflows

Example of adding an item to QRadar

Example of moving QRadar item from

Cancel

Save & Close

Save

qradar_reference_set_name

qradar_reference_set_item_value

Search...

qradar_query_param1

qradar_query_param2

qradar_query_param3

qradar_query_param4

qradar_query_param5

qradar_query_range_end

qradar_query_range_start

qradar_reference_set_item_value

qradar_reference_set_name

The workflow, “Example of adding an item to QRadar reference set”, sets the function’s input fields: “qradar_reference_set_name” is mapped to “Sample Blocked IPs”, and qradar_reference_set_item_value is mapped to the artifact value, and then runs the function.

The workflow is initiated by the rule, “QRadar Add to Reference Set”.

The screenshot shows the 'Customization Settings' page in the Resilient interface. The 'Workflows' tab is selected, and the workflow 'Example of adding an item to QRadar reference set' is being edited. The workflow details include a name, API name, description, and object type. A visual workflow diagram shows a start node leading to a function node 'QRadar Add Reference Set Item'. Below the diagram, the input parameters are defined: 'qradar_reference_set_name' is set to 'Sample Blocked IPs', and 'qradar_reference_set_item_value' is an empty field.

resilient Dashboards ▾ List Incidents New Incident My Tasks Simulations All ▾ Search masterfirst ... QRadar ▾

Customization Settings

Layouts Rules Scripts **Workflows** Functions Message Destinations Phases & Tasks Incident Types Breach Artifacts

Workflows / Example of adding an item to QRadar reference set Cancel Save & Close Save

Name * Example of adding an item to QRadar reference set

API Name * qradar_add_reference_set_item

Description Add an IP address artifact to the QRadar reference set, "Sample Blocked IPs".

Object Type * Artifact

Creator masterfirst masterlast
Last Modified 06/27/2018 18:26
Last Modified By masterfirst masterlast
Associated Rules QRadar Add to Reference Set

Start your workflow here

QRadar Add Reference Set Item

| Input Parameter | Value |
|-----------------------------------|--------------------|
| qradar_reference_set_name * | Sample Blocked IPs |
| qradar_reference_set_item_value * | |

The example rule, “QRadar Add to Reference Set”, is a menu item rule for an artifact.

The screenshot shows the 'Customization Settings' page in the Resilient interface. The 'Rules' tab is selected, and the specific rule being configured is 'QRadar Add to Reference Set'. The 'Display Name' is 'QRadar Add to Reference Set' and the 'Object Type' is 'Artifact'. Below this, there are sections for 'Activities' including 'Ordered', 'Workflows', and 'Destinations'. The 'Ordered' section states that activities are invoked in the specified order. The 'Workflows' section indicates that workflow activities start after ordered activities complete, with an example of adding an item to the QRadar reference set. The 'Destinations' section shows that transaction data is posted to message destinations after all activities complete. At the bottom, there is a copyright notice for IBM Corporation 2018.

The user can select this action in the menu to initiate the workflow.

The screenshot displays the 'DemolIncident' details page in the Resilient interface. The page is divided into several sections: 'Summary' (ID: 2095, Phase: Engage, Severity: Medium, Date Created: 06/26/2018, Date Occurred: 06/07/2018, Date Discovered: 06/07/2018, Data: Unknown, Compromised, Incident Type: System Intrusion), 'Description' (1653 events in 4 categories: login fail containing SIM User Authentication), 'People' (Created By: masterfirst masterlast, Owner: masterfirst masterlast, Members: There are no members), and 'Related Incidents' (No related incidents). The 'Artifacts' section is active, showing a table of artifacts. The first artifact is 'IP Address: Source' with a value of '9.108.150.53' and a creation date of '06/26/2018'. A context menu is open over the artifact, listing actions: 'Delete from QRadar Reference Set', 'Find in QRadar Reference Set', 'QRadar Add to Reference Set' (which is circled in red), and 'QRadar Move from suspect to blocked'.

QRadar Delete Reference Set Item

This function deletes an item from an existing QRadar reference set. It has two input fields: “qradar_reference_set_name” and “qradar_reference_set_item_value”.

The function is called by the workflow, “Example of deleting QRadar reference set item”.

The workflow, “Example of deleting QRadar reference set item”, sets the function’s input fields:

“qradar_reference_set_name” is mapped to “Sample Suspect IPs”, and “qradar_reference_set_item_value” is mapped to the artifact value, and then runs the function. The workflow is initiated by the rule, “Delete from QRadar Reference Set”.

The rule, “Delete from QRadar Reference Set” is a menu item rule for artifacts. The user can select this menu item to initiate the workflow.

QRadar Find Reference Set Item

This function looks for an item in an existing QRadar reference set. It has two input fields:

“qradar_reference_set_name” and “qradar_reference_set_item_value”. The function is called by the workflow, “Example of finding an item from a QRadar reference set”.

The workflow, “Example of finding an item from a QRadar reference set”, sets the function’s input fields:

“qradar_reference_set_name” is mapped to “Sample Blocked IPs”, and “qradar_reference_set_item_value” is mapped to the artifact value. After running the function, the workflow adds a note to the corresponding incident.

The rule, “Find in QRadar Reference Set,” is a menu item rule for artifacts. The user can select this action from the menu to initiate the workflow.

QRadar Find Reference Sets

This function looks for all QRadar reference sets that contain a given item. It has one input field, “qradar_reference_set_item_value.” The function is called by the workflow, “Example of finding all QRadar reference sets for artifact.”

The workflow, “Example of finding all QRadar reference sets for artifact”, sets the input field, “qradar_reference_set_item_value” to the artifact value. After running the function, the workflow populates the “QRadar Reference Sets” data table with the returned reference sets.

The rule, “Find All QRadar Reference Sets,” is a menu item rule for the artifact. The user can click this menu item for a selected artifact, and the example workflow commences.

The screenshot displays the QRadar interface. At the top, there is a 'Show' dropdown set to '25'. Below this is a table with columns: Type, Value, Created, Relate?, and Actions. The table contains two rows: one for 'DNS Name' with value 'mysuspiciousdomain.com' and another for 'IP Address' with value '178.137.87.242'. A context menu is open over the 'IP Address' row, showing options: 'Delete from QRadar Reference Set', 'Find All QRadar Reference Sets' (highlighted with a blue circle), 'Find in QRadar Reference Set', 'QRadar Add to Reference Set', and 'QRadar Move from suspect to blocked'. Below the main table is a section titled 'QRadar Reference Sets' with a search bar. It contains a table with columns: Reference Set, Item Value, Source, and an empty column. The table lists two entries: 'Sample Blocked IPs' with item value '178.137.87.242' and source 'admin', and 'DNS Servers' with item value '178.137.87.242' and source 'reference data api'. At the bottom, it says 'Displaying 1 - 2 of 2'.

| Type | Value | Created | Relate? | Actions |
|------------|------------------------|------------------|----------------------------------------|---------|
| DNS Name | mysuspiciousdomain.com | 03/25/2019 11:31 | As specified in the artifact type sett | ... |
| IP Address | 178.137.87.242 | 03/25/2019 11:27 | As specified | ... |

QRadar Reference Sets

| Reference Set | Item Value | Source | |
|--------------------|----------------|--------------------|-----|
| Sample Blocked IPs | 178.137.87.242 | admin | ... |
| DNS Servers | 178.137.87.242 | reference data api | ... |

Displaying 1 - 2 of 2

Troubleshooting

There are several ways to verify the successful operation of a function.

- Resilient Action Status

When viewing an incident, use the Actions menu to view Action Status. By default, pending and errors are displayed. Modify the filter for actions to also show Completed actions. Clicking on an action displays additional information on the progress made or what error occurred.

- Resilient Scripting Log

A separate log file is available to review scripting errors. This is useful when issues occur in the pre-processing or post-processing scripts. The default location for this log file is: `/var/log/resilient-scripting/resilient-scripting.log`.

- Resilient Logs

By default, Resilient logs are retained at `/usr/share/co3/logs`. The `client.log` may contain additional information regarding the execution of functions.

- Resilient-Circuits

The log is controlled in the `.resilient/app.config` file under the section `[resilient]` and the property `logdir`. The default file name is `app.log`. Each function will create progress information. Failures will show up as errors and may contain python trace statements.

Support

For support, visit <https://ibm.com/mysupport>.

Including relevant information from the log files will help us resolve your issue.