IBM Resilient

resilient

# Incident Response Platform Integrations

## Splunk Function V1.0.1

Release Date: October 2018

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 Splunk Function.

## Overview

The Splunk function, fn_splunk_integration, provides an automated way of managing bidirectional actions between Resilient artifact items and Splunk items in threat intelligence collections.

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

- Search function to query a Splunk intelligence collection for threat items.
- Update function to change the status of a Splunk ES notable event.
- Add function to create a new threat intelligence item in a given Splunk collection.
- Delete function to disable a threat intelligence item from a given Splunk collection.

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

The remainder of this document describes each included function, how to configure them in custom workflows, and any additional customization options.

# Installation

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

- Resilient platform is version 30 or later.
- 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) for more information.
- Splunk version 6.6 or later.
- Splunk ES 4.7.2 or later (only required for the function to update a Splunk ES notable event).

## Install the Python components

The functions package contains Python components that will be called by the Resilient platform to execute the functions during your workflows. These components run in the 'resilient-circuits' integration framework.

The package also includes Resilient customizations that will be imported into the platform later.

Ensure that the environment is up to date:

```
sudo pip install --upgrade pip
sudo pip install --upgrade setuptools
sudo pip install --upgrade resilient-circuits
```

To install the package, you must first unzip it then install the package as follows:

```
sudo pip install fn_splunk_integration-<version>.tar.gz
```

## Configure the Python components

The 'resilient-circuits' components run as an unprivileged user, typically named `integration`. If you do not already have an `integration` user configured on your appliance, create it now.

Perform the following to configure and run the integration:

1. Using 'sudo', become the integration user.

   ```
   sudo su - integration
   ```

2. From the account used for Integrations, use one of the following commands to configure the Splunk settings. Use –c to create new environments or –u to update existing environments:

   ```
   resilient-circuits config -c
   ```

   OR

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

3. Edit the .resilient/app.config configuration.

   a. In the [resilient] section, ensure that you provide all the information needed to connect to the Resilient platform.

   b. In the [fn_splunk_integration] section, edit the settings as required.

   ```
   host=<splunk url>
   port=<8089 or the customized port>
   username=<splunk access user>
   splunkpassword=<splunk access password, key-ring protection recommended>
   verify_cert=[true|false]
   ```

Use "false" for self-signed certificates.

## Deploy customizations into the Resilient platform

The package contains function definitions that you can use in workflows, and also includes example workflows and rules that show how to use these functions.

Install these customizations to the Resilient platform with the following command:

```
resilient-circuits customize
```

You will be prompted to import the functions and associated message destinations, workflows, and so on.  Note that users can arrange custom fields and data tables in the Layout to suit their particular needs.

## Run the integration framework

Run the integration manually with the following command:

```
resilient-circuits run
```

The resilient-circuits command starts, loads its components, and continues to run until interrupted. If it stops immediately with an error message, check your configuration values and retry.

## Configuration of resilient-circuits for restartability

For normal operation, resilient-circuits must run continuously.  The recommended way to do this is to configure the service to automatically run at startup. On a Red Hat appliance, this is done using a systemd unit file such as the one below. You may need to change the paths to your working directory and app.config.

The unit file should be named ‘resilient_circuits.service’:

```
sudo vi /etc/systemd/system/resilient_circuits.service
```

The contents:

```
[Unit]
Description=Resilient-Circuits Service
After=resilient.service
Requires=resilient.service

[Service]
Type=simple
User=integration
WorkingDirectory=/home/integration
ExecStart=/usr/local/bin/resilient-circuits run
Restart=always
TimeoutSec=10
Environment=APP_CONFIG_FILE=/home/integration/.resilient/app.config
Environment=APP_LOCK_FILE=/home/integration/.resilient/resilient_circuits.
lock

[Install]
WantedBy=multi-user.target
```

Ensure that the service unit file is correctly permissioned:

```
sudo chmod 664 /etc/systemd/system/resilient_circuits.service
```

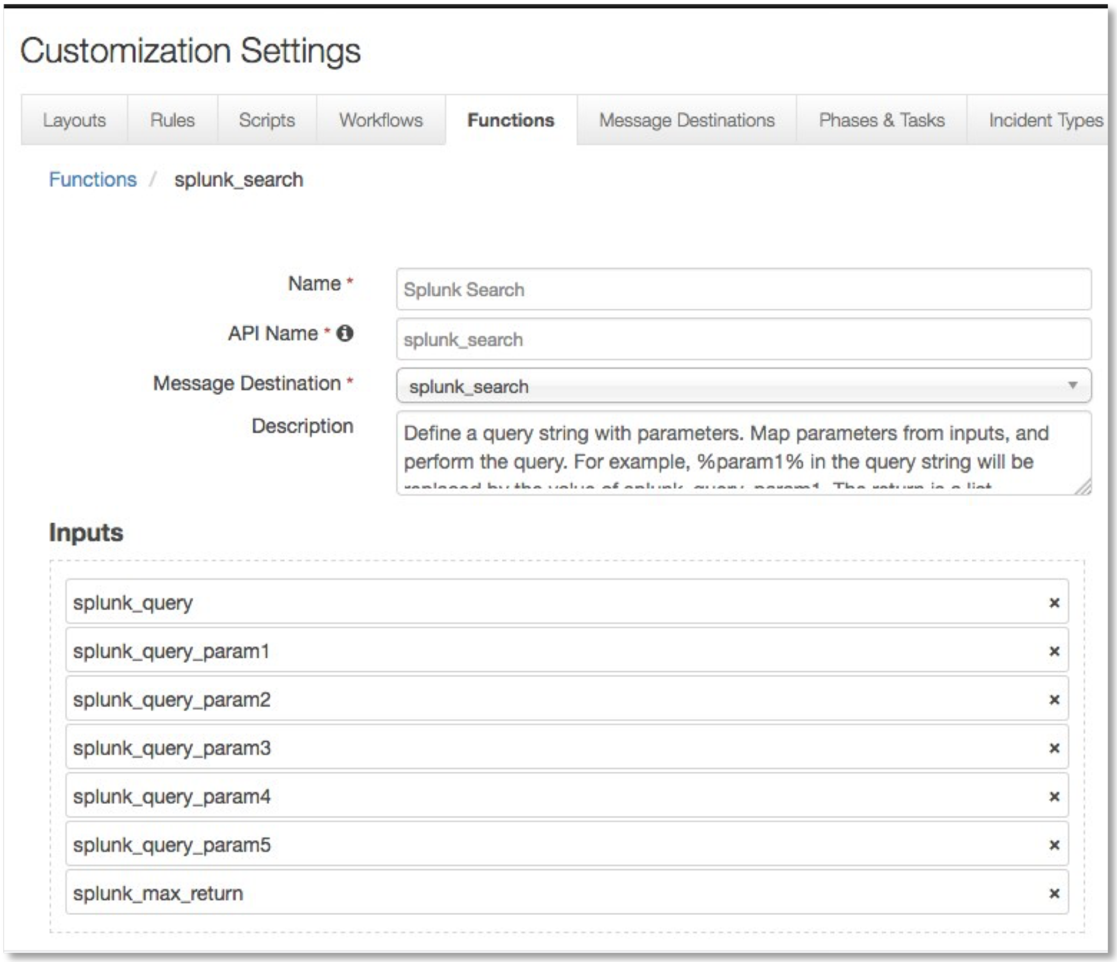
# Function Descriptions

In the Customization Settings section of the Resilient platform, you can verify that the following Splunk specific functions, workflows, data table, and rules are available by clicking their respective tabs.

Here are the details about how each function is used in the example workflows and rules.

## Splunk Search

This function performs a query to fetch data from the Splunk server.



Name * Splunk Search

API Name * splunk_search

Message Destination * splunk_search

Description Define a query string with parameters. Map parameters from inputs, and perform the query. For example, %param1% in the query string will be replaced by the value of splunk_query_param1. The return is a list

**Inputs**

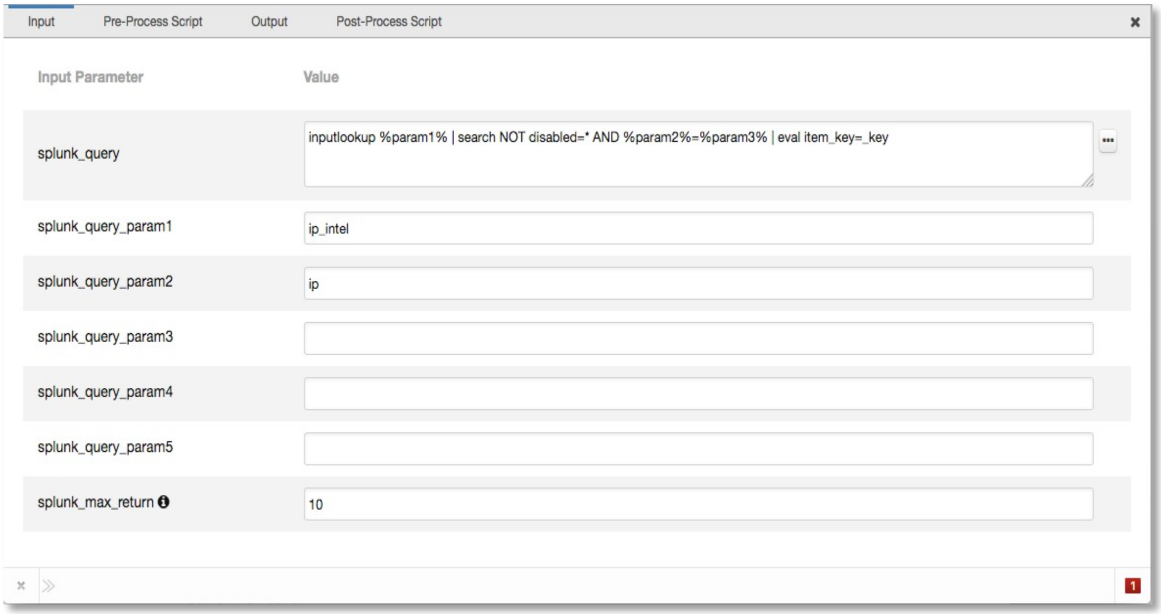
- splunk_query
- splunk_query_param1
- splunk_query_param2
- splunk_query_param3
- splunk_query_param4
- splunk_query_param5
- splunk_max_return

*Figure 1: Splunk Search*

As shown above, this function takes the following parameters:

- splunk_query: Query to perform. It contains demo template queries that you can select from within the workflow. The demo queries contain parameters which are replaced by the splunk_query_param[n] below. For example, one demo query is: SELECT %param1% FROM events WHERE username=%param2% LAST %param3% MINUTES. Users can then set values for splunk_query_param1, splunk_query_param2, and splunk_query_param3 in the workflow.
- splunk_query_param[n]: parameters used in the query.
- splunk_max_return: specifies how many events to return from Splunk.

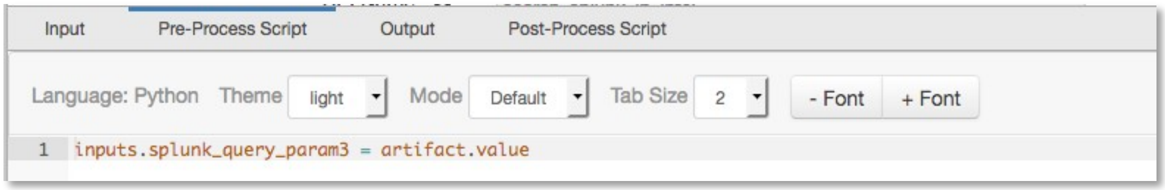
The example workflow (object type = Artifact) that calls this function is “Example of searching Splunk ES ip_intel”. The Input tab of this function is shown in Figure 2. It shows the mapping of the parameters; for example, %param1% in the query is mapped to ip_intel.



| Input Parameter | Value |
|---|---|
| splunk_query | inputlookup %param1% \| search NOT disabled=* AND %param2%=%param3% \| eval item_key=_key |
| splunk_query_param1 | ip_intel |
| splunk_query_param2 | ip |
| splunk_query_param3 | |
| splunk_query_param4 | |
| splunk_query_param5 | |
| splunk_max_return | 10 |

*Figure 2: Example of searching Splunk ES ip_intel*

In the Pre-Process Script, the %param3% is set to the value of the artifact associated with this workflow as shown in Figure 3.

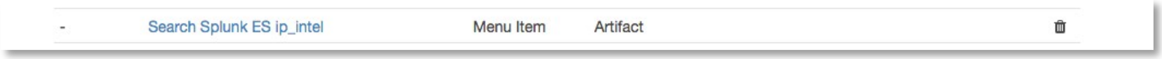


Language: Python Theme light Mode Default Tab Size 2

```
inputs.splunk_query_param3 = artifact.value
```

*Figure 3: Pre-Process Script*

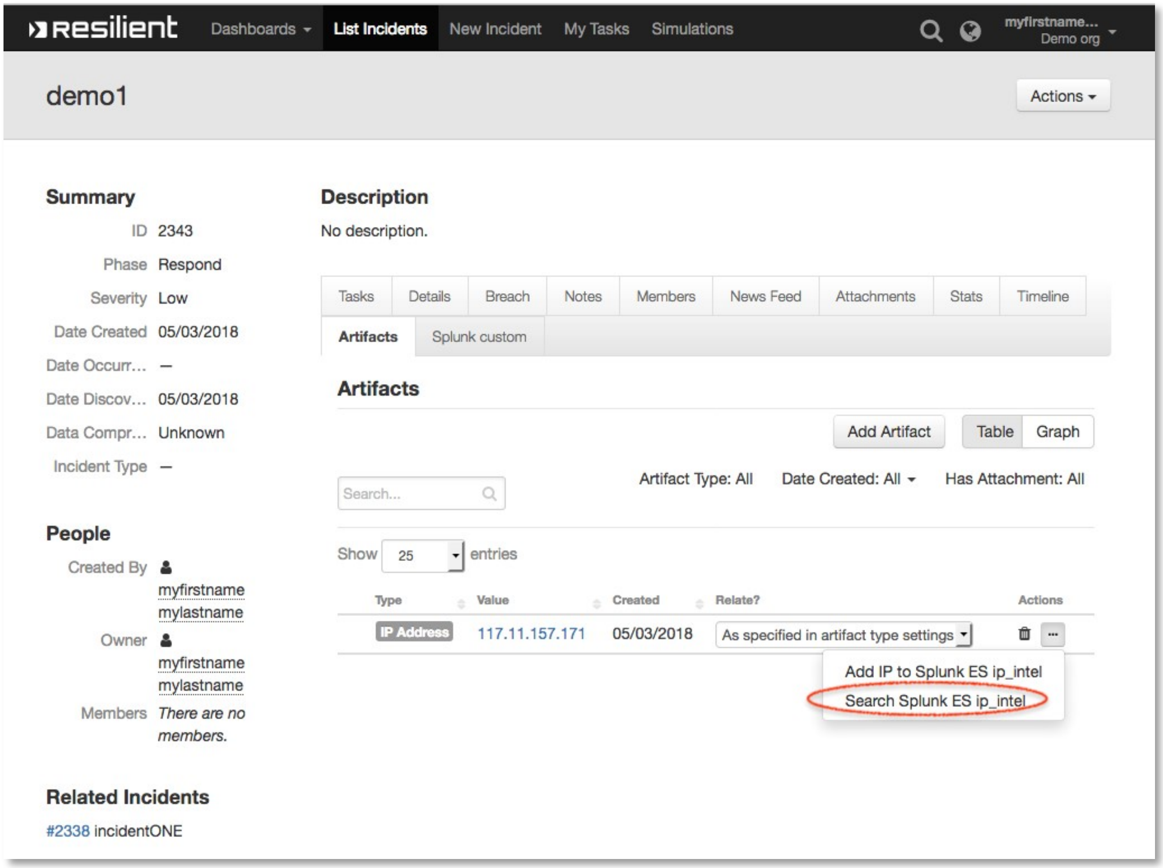
A Menu Item rule called "Search Splunk ES ip_intel" is also included. This rule calls the provided workflow.



| | | | |
|---|---|---|---|
| - | Search Splunk ES ip_intel | Menu Item | Artifact |

*Figure 4: Rule*

With these components in place, if an IP Address artifact is added, users can click the Actions button, and the above rule appears as shown in Figure 5. By clicking the menu item, this search function is activated. The search result from Splunk is used to update the custom data table called "splunk_ip_intel" shown in Figure 6. The definition of this data table is also included in the package.



demo1

**Summary**

| | |
|---|---|
| ID | 2343 |
| Phase | Respond |
| Severity | Low |
| Date Created | 05/03/2018 |
| Date Occurr... | — |
| Date Discov... | 05/03/2018 |
| Data Compr... | Unknown |
| Incident Type | — |

**People**

Created By myfirstname mylastname

Owner myfirstname mylastname

Members *There are no members.*

**Related Incidents**

#2338 incidentONE

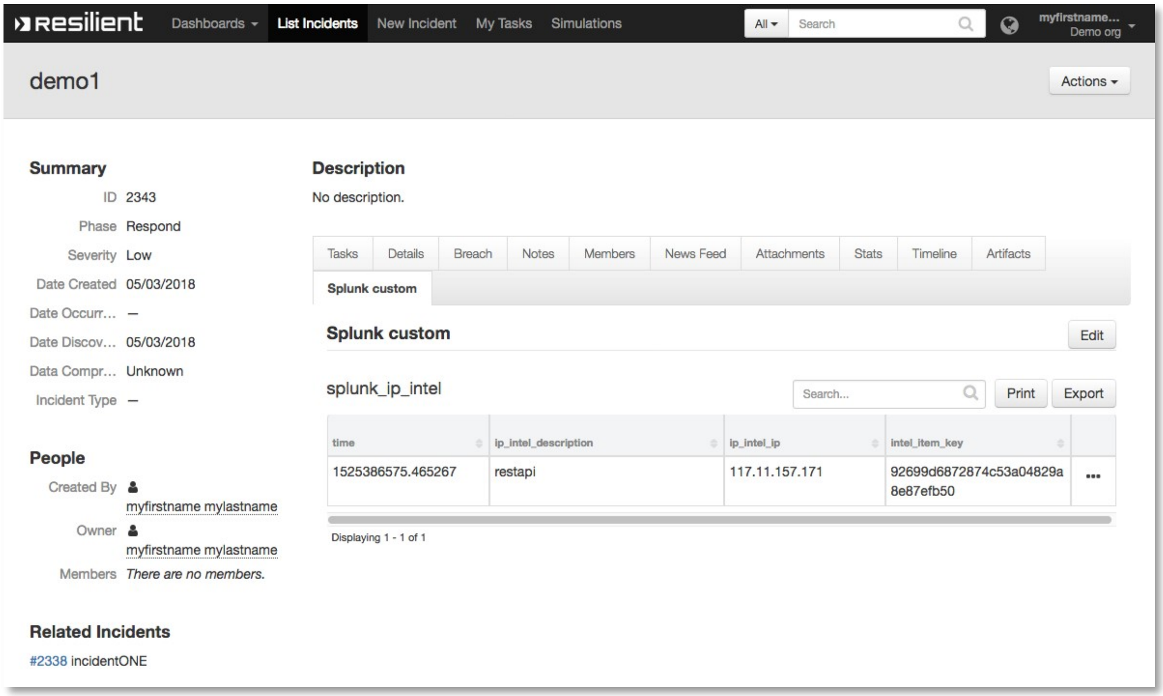
**Description**

No description.

**Artifacts**

| Type | Value | Created | Relate? | Actions |
|---|---|---|---|---|
| IP Address | 117.11.157.171 | 05/03/2018 | As specified in artifact type settings | |

*Figure 5: Menu item*



demo1

**Summary**

| | |
|---|---|
| ID | 2343 |
| Phase | Respond |
| Severity | Low |
| Date Created | 05/03/2018 |
| Date Occurr... | — |
| Date Discov... | 05/03/2018 |
| Data Compr... | Unknown |
| Incident Type | — |

**People**

Created By myfirstname mylastname

Owner myfirstname mylastname

Members *There are no members.*

**Related Incidents**

#2338 incidentONE

**Description**

No description.

**Splunk custom**

splunk_ip_intel

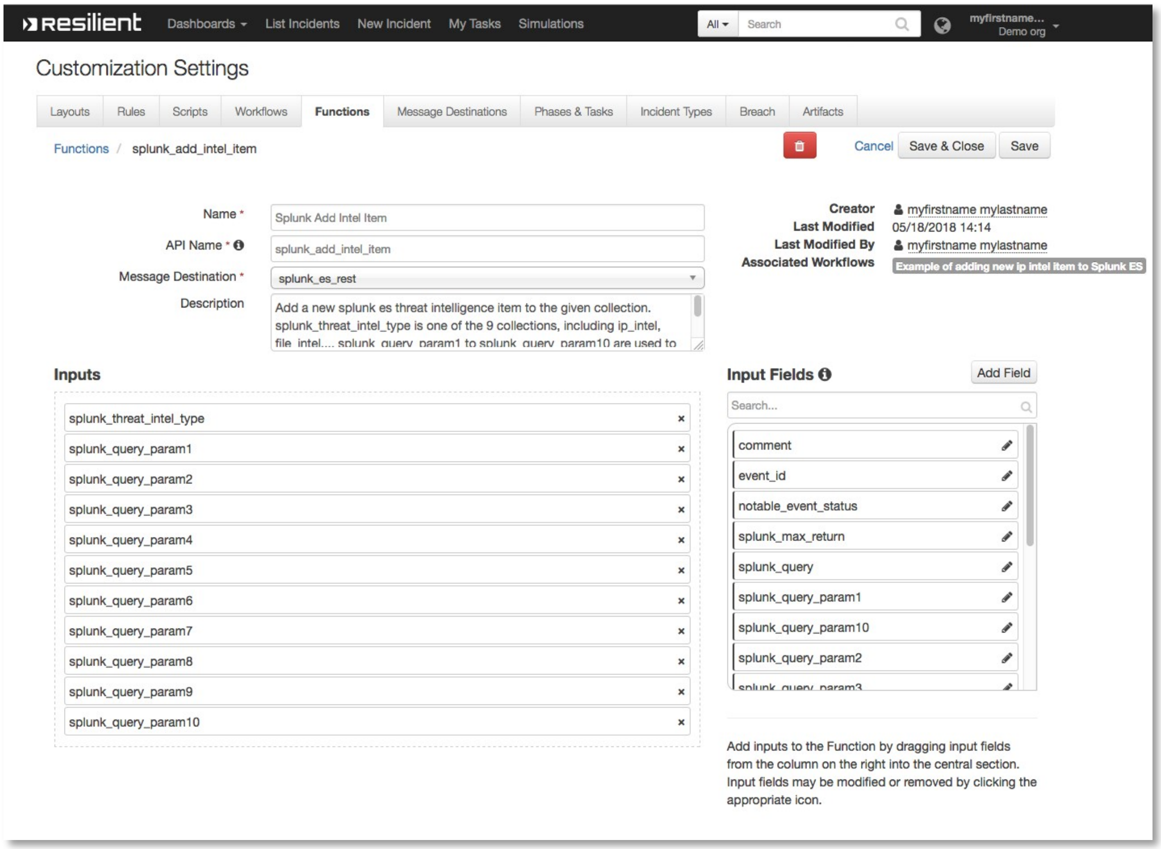
| time | ip_intel_description | ip_intel_ip | intel_item_key |
|---|---|---|---|
| 1525386575.465267 | restapi | 117.11.157.171 | 92699d6872874c53a04829a8e87efb50 |

Displaying 1 - 1 of 1

*Figure 6: Data table*

## Splunk Add Intelligence Item

This function adds a new threat intelligence item to a given collection.



Customization Settings

Functions / splunk_add_intel_item

Name * Splunk Add Intel Item

API Name * splunk_add_intel_item

Message Destination * splunk_es_rest

Description: Add a new splunk es threat intelligence item to the given collection. splunk_threat_intel_type is one of the 9 collections, including ip_intel, file_intel.... splunk_query_param1 to splunk_query_param10 are used to

Creator: myfirstname mylastname
Last Modified: 05/18/2018 14:14
Last Modified By: myfirstname mylastname
Associated Workflows: Example of adding new ip intel item to Splunk ES

**Inputs**

- splunk_threat_intel_type
- splunk_query_param1
- splunk_query_param2
- splunk_query_param3
- splunk_query_param4
- splunk_query_param5
- splunk_query_param6
- splunk_query_param7
- splunk_query_param8
- splunk_query_param9
- splunk_query_param10

**Input Fields**

- comment
- event_id
- notable_event_status
- splunk_max_return
- splunk_query
- splunk_query_param1
- splunk_query_param10
- splunk_query_param2
- splunk_query_param3

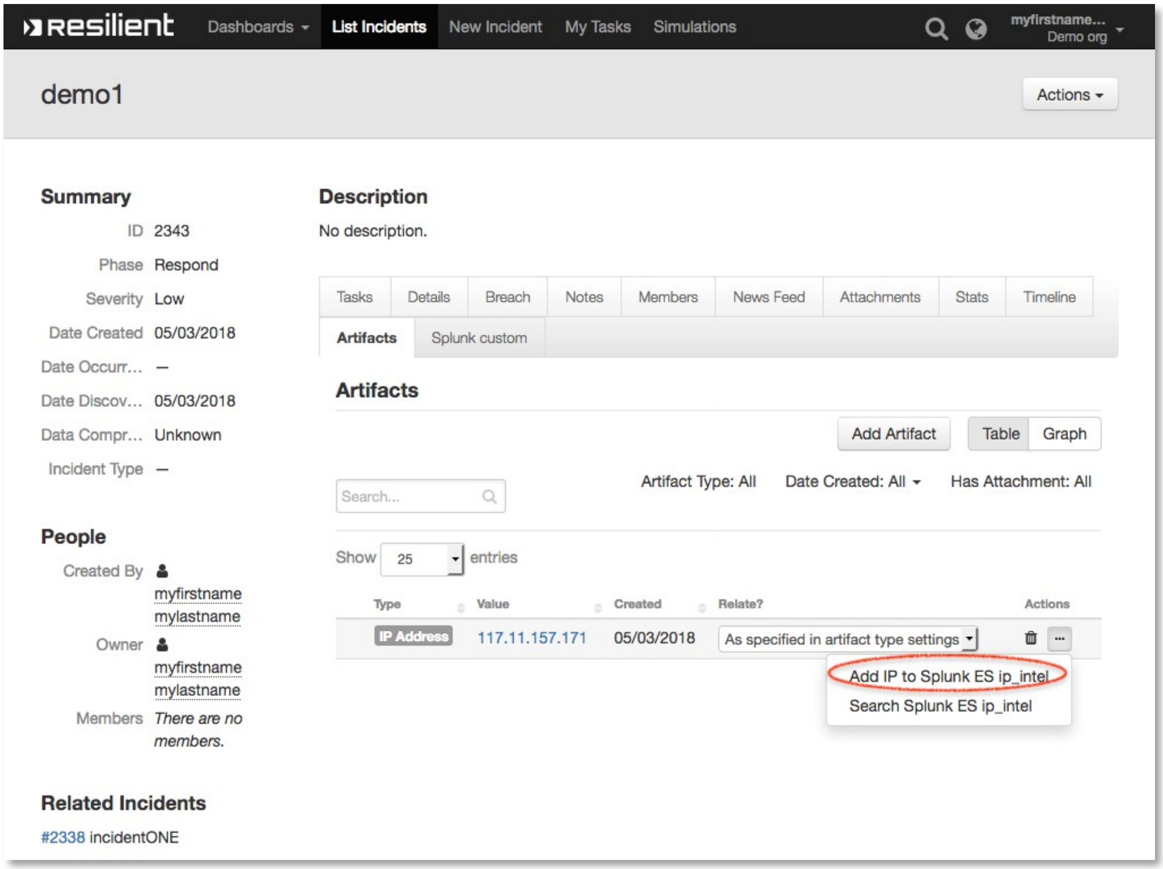
Add inputs to the Function by dragging input fields from the column on the right into the central section. Input fields may be modified or removed by clicking the appropriate icon.

*Figure 7: Splunk Add Intelligence Item*

Here, splunk_threat_intel_type is the name of the Splunk threat intelligence collection, and splunk_query_param1 to splunk_query_param10 are inputs used to create a python dictionary that adds a new threat intelligence item to a given collection.

In the Input tab of the example workflow for artifact, splunk_threat_intel_type is set to ip_intel, and splunk_query_param1 to ip. In the Pre-Process Script, splunk_query_param2 is the value of the associated artifact. This creates a python dictionary: {“ip”: “the_associated_artifact_value”}, and a new item is added to the ip_intel collection.

An example rule, “Add IP to Splunk ES ip_intel”, calls this example workflow. As a result, a user can click on this menu item to add an IP Address artifact to the ip_intel collection of Splunk ES.



demo1

**Summary**

ID 2343
Phase Respond
Severity Low
Date Created 05/03/2018
Date Occurr... —
Date Discov... 05/03/2018
Data Compr... Unknown
Incident Type —

**People**

Created By myfirstname mylastname
Owner myfirstname mylastname
Members *There are no members.*

**Related Incidents**

#2338 incidentONE

**Description**

No description.

**Artifacts**

Artifact Type: All   Date Created: All   Has Attachment: All

Show 25 entries

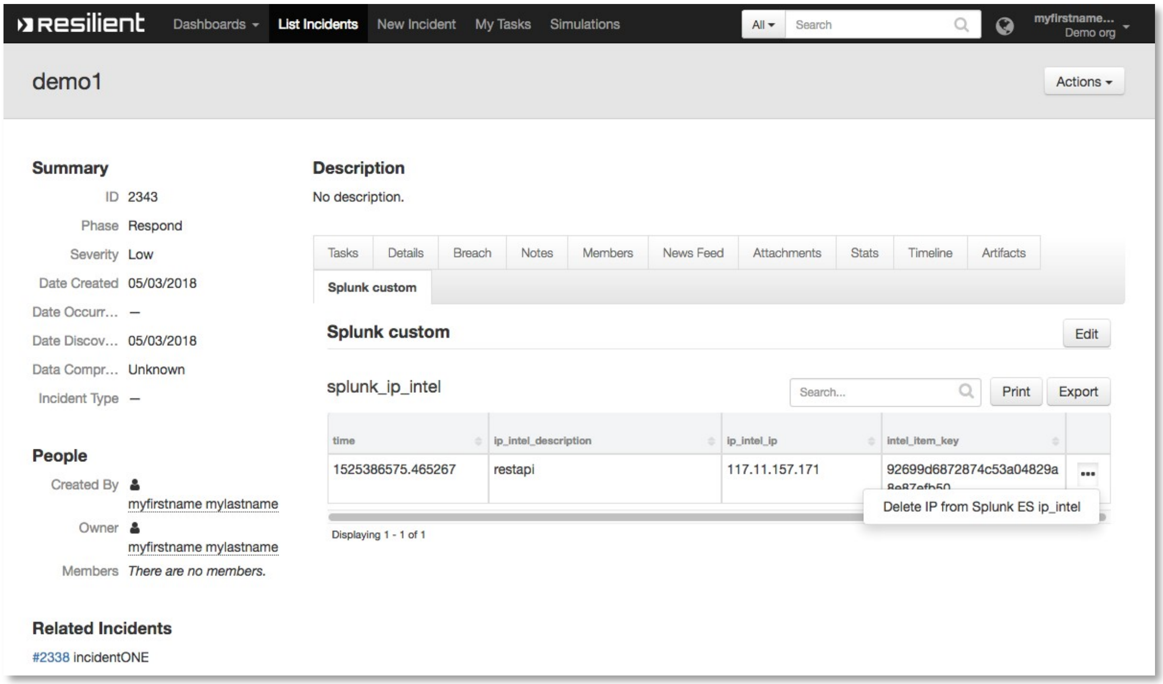
| Type | Value | Created | Relate? | Actions |
|---|---|---|---|---|
| IP Address | 117.11.157.171 | 05/03/2018 | As specified in artifact type settings | |

*Figure 8: Rule and Menu Item*

## Splunk Delete Intelligence Item

This function is used to disable a threat intelligence item from a given collection. A workflow, “Example of deleting a Splunk ES ip_intel item”, calls this function, and is activated by a rule called “Delete IP from Splunk ES ip_intel”.

The rule is a menu item to a row in the included data table. As shown in Figure 9, a row contains the intel_item_key corresponding to this intelligence item. This menu item calls the function to delete the item associated with that item_key.



demo1

**Summary**

| | |
|---|---|
| ID | 2343 |
| Phase | Respond |
| Severity | Low |
| Date Created | 05/03/2018 |
| Date Occurr... | — |
| Date Discov... | 05/03/2018 |
| Data Compr... | Unknown |
| Incident Type | — |

**People**

| | |
|---|---|
| Created By | myfirstname mylastname |
| Owner | myfirstname mylastname |
| Members | *There are no members.* |

**Related Incidents**

#2338 incidentONE

**Description**

No description.

**Splunk custom**

splunk_ip_intel

| time | ip_intel_description | ip_intel_ip | intel_item_key |
|---|---|---|---|
| 1525386575.465267 | restapi | 117.11.157.171 | 92699d6872874c53a04829a 8e87efb50 |

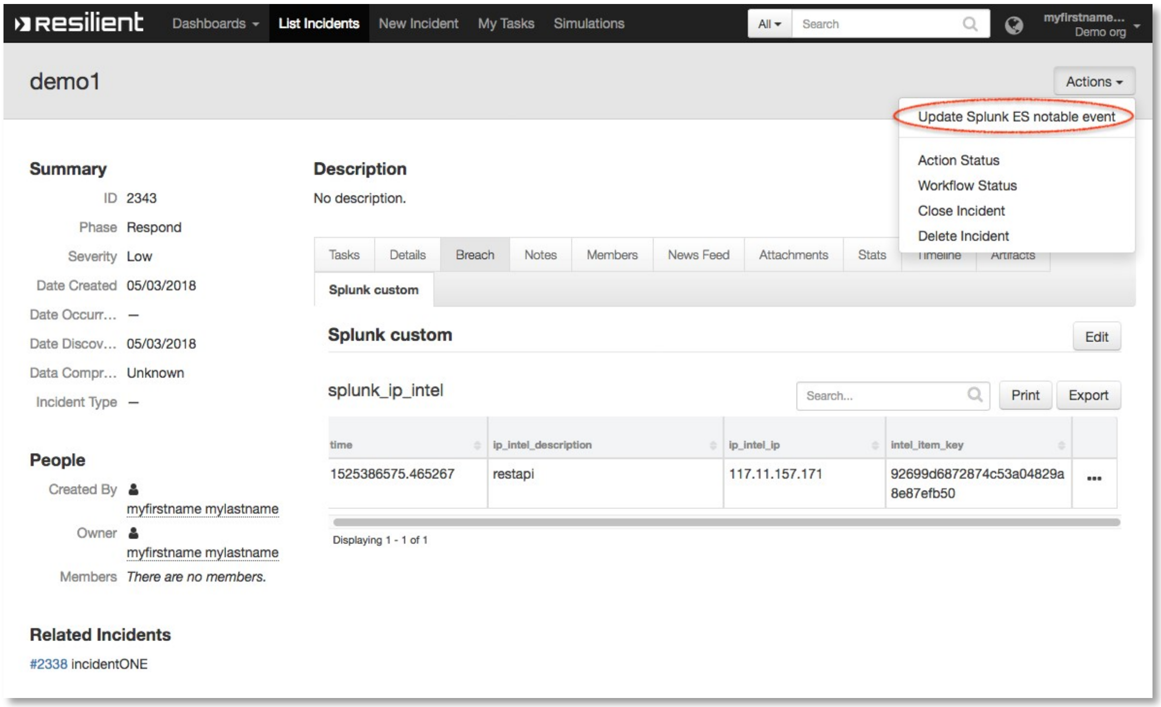
Displaying 1 - 1 of 1

*Figure 9: Data table row with data including intel_item_key*

## Splunk ES Notable Event

This function updates the status and comment of a given notable event, using the event_id stored in an incident. It can be used together with the “Resilient Integration for Splunk ES” addon.

An incident escalated from the “Resilient Integration for Splunk and Splunk ES” addon contains a custom property called splunk_notable_event_id. In the workflow, the status of the incident is mapped to the status of notable event. Also, a comment is given in the Input tab. As a result, this menu item updates the notable event identified by this event id accordingly.



demo1

**Summary**

| | |
|---|---|
| ID | 2343 |
| Phase | Respond |
| Severity | Low |
| Date Created | 05/03/2018 |
| Date Occurr... | — |
| Date Discov... | 05/03/2018 |
| Data Compr... | Unknown |
| Incident Type | — |

**People**

| | |
|---|---|
| Created By | myfirstname mylastname |
| Owner | myfirstname mylastname |
| Members | *There are no members.* |

**Related Incidents**

#2338 incidentONE

**Description**

No description.

**Splunk custom**

splunk_ip_intel

| time | ip_intel_description | ip_intel_ip | intel_item_key |
|---|---|---|---|
| 1525386575.465267 | restapi | 117.11.157.171 | 92699d6872874c53a04829a 8e87efb50 |

Displaying 1 - 1 of 1

*Figure 10: Update Splunk ES Notable Event*

# 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 additional support, contact support@resilientsystems.com.

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