

# IBM Resilient



## Incident Response Platform Integrations

### Splunk Function V1.0.3

Release Date: August 2020

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 Integration functions.

## 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 35.2 or later.
- You have access to a Resilient integration server or an App Host environment.
- 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).

### App Host Installation

All the components for running this integration in a container already exist when using the App Host app.

To install,

- \* Navigate to Administrative Settings and then the Apps tab.
- \* Click the Install button and select the downloaded file: `app-fn_splunk_integration-x.x.x.zip`.
- \* Go to the Configuration tab and edit the `app.config` file, editing the URL and access credentials for Splunk, making any additional setting changes.

```
[fn_splunk_integration]
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]
```

### Integration Server Installation

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 app-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 -l fn-splunk-integration
```

OR

```
resilient-circuits config -u -l fn-splunk-integration
```

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 -l fn-splunk-integration
```

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, datatable, 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.

## Customization Settings

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

Functions / splunk\_search

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 an artifact in Splunk ES”. 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 the appropriate Splunk collection, such as `ip_intel`.

Input	Pre-Process Script	Output	Post-Process Script
Input Parameter	Value		
splunk_query	inputlookup %param1%   search NOT disabled=* AND %param2%=%param3%   eval item_key=_key		
splunk_query_param1			
splunk_query_param2			
splunk_query_param3			
splunk_query_param4			
splunk_query_param5			
splunk_max_return ⓘ	10		

*Figure 2: Example of searching an artifact in Splunk ES*

In the Pre-Process Script, the %param3% is set to the value of the artifact associated with this workflow and the collection is determined by a lookup table of artifact values and Splunk collection information.

```
lookup_map = {
  "DNS Name": ("ip_intel", "domain"),
  "Email Attachment": None,
  "Email Attachment Name": ("file_intel", "file_name"),
  "Email Body": None,
  "Email Recipient": None,
  "Email Sender": ("email_intel", "src_user"),
  "Email Sender Name": ("email_intel", "src_user"),
  "Email Subject": ("email_intel", "subject"),
  "File Name": ("file_intel", "file_name"),
  "File Path": None,
  "HTTP Request Header": None,
  "HTTP Response Header": None,
  "IP Address": ("ip_intel", "ip"),
  "Log File": None,
  "MAC Address": None,
  "Malware Family/Variant": None,
  "Malware MD5 Hash": ("file_intel", "file_hash"),
  "Malware Sample": None,
  "Malware Sample Fuzzy Hash": ("file_intel", "file_hash"),
  "Malware SHA-1 Hash": ("file_intel", "file_hash"),
  "Malware SHA-256 Hash": ("file_intel", "file_hash"),
  "Mutex": None,
  "Network CIDR Range": None,
  "Other File": None,
  "Password": None,
  "Port": None,
  "Process Name": ("process_intel", "process"),
  "Registry Key": ("registry_intel", "registry_value_name"),
  "RFC 822 Email Message File": None,
  "Service": ("service_intel", "service"),
  "String": None,
  "System Name": ("service_intel", "service"),
  "URI Path": None,
  "URL": ("http_intel", "url"),
  "URL Referer": ("http_intel", "http_referrer"),
  "User Account": None,
  "User Agent": ("http_intel", "http_user_agent")
}
```

```










if artifact.type in lookup_map and lookup_map[artifact.type]:
    threat_type, threat_field_name = lookup_map[artifact.type]
    inputs.splunk_threat_intel_type = threat_type
    inputs.splunk_query_param1 = threat_field_name
    inputs.splunk_query_param2 = artifact.value
else:
    helper.fail("Artifact type not supported: {}".format(artifact.type))

```

*Figure 3: Pre-Process Script*



A rule “Search Splunk ES for an artifact” is also included to call the provided workflow.

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 4. 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\_results” shown in Figure 5. The definition of this data table is also included in the package.

URL	<a href="https://this.is.com">https://this.is.com</a>	08/25/2020 14:15	As specified in the artifact type sett	  
IP Address	<a href="#">39.43.231.3</a>	08/25/2020 14:14	As specified in the artifact type sett	  
File Name	<a href="#">abc.dll</a>	08/25/2020 11:48	As specified in the artifact type sett	  
Email Sender	<a href="#">a@example.com</a>	08/25/2020 11:46	<div> Add artifact to Splunk ES  Search Splunk ES for an artifact </div>	

*Figure 4: Menu item*

Splunk Search Results Search... Print Export

Created Date	Source	Intel Collection	Intel Field	Intel Value	Intel Key	Status	
08/25/2020 12:13:10	restapi	file_intel	file_name	abc.dll	8fef544454b34b0ebb25c3a7e03d01c2	Active	
08/25/2020 14:14:11	restapi	ip_intel	ip	39.43.231.3	6c99df0fb0dc4524a8b7d118193f360b	Deleted	
08/25/2020 14:15:37	restapi	http_intel	url	https://this.is.com	d104847d69f9416eb97dbfe76cc52719	Deleted	
08/25/2020 14:17:31	restapi	http_intel	http_user_agent	user-agent:abc123	e49674856ed441289c279f8b75db6b8b	Active	

Displaying 1 - 4 of 4

*Figure 5: Data table*

## Splunk Add Intelligence Item

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

**Customization Settings**

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

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

**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

**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

**Cancel Save & Close Save**

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 6: 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. For an IP address artifact, a python dictionary: {"ip": "the\_associated\_artifact\_value"}, and a new item is added to the ip\_intel collection.

An example rule, "Add artifact to Splunk ES", calls this example workflow. As a result, a user can click on this menu item to add an artifact to the appropriate collection of Splunk ES, such as ip\_intel.

URL	<a href="https://this.is.com">https://this.is.com</a>	08/25/2020 14:15	As specified in the artifact type sett	
IP Address	<a href="#">39.43.231.3</a>	08/25/2020 14:14	As specified in the artifact type sett	
File Name	<a href="#">abc.ddl</a>	08/25/2020 11:48	As specified in the artifact type sett	
Email Sender	<a href="#">a@example.com</a>	08/25/2020 11:46	Add artifact to Splunk ES Search Splunk ES for an artifact	

Figure 7: 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 an artifact in Splunk ES", calls this function, and is activated by a rule called "Delete artifact from Splunk ES".



The rule is a menu item to a row in the included data table. As shown in Figure 8, 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.

Splunk Search Results

Created Date	Source	Intel Collection	Intel Field	Intel Value	Intel Key	Status	
08/25/2020 12:13:10	restapi	file_intel	file_name	abc.ddl	8fef544454b34b0ebb25c3a7e03d01c2	Active	⋮
08/25/2020 14:14:11	restapi	ip_intel	ip	39.43.231.3	6c99df0fb0dc4524a8b7d118193f360b		Delete artifact from Splunk ES
08/25/2020 14:15:37	restapi	http_intel	url	https://this.is.com	d104847d69f9416eb97dbfe76cc52719	Deleted	⋮
08/25/2020 14:17:31	restapi	http_intel	http_user_agent	user-agent:abc123	e49674856ed441289c279f8b75db6b8b	Active	⋮

Displaying 1 - 4 of 4

Figure 8: 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.

**Resilient** Dashboards ▾ List Incidents New Incident My Tasks Simulations All ▾ Search myfirstname... Demo org

demo1 Actions ▾

**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.

Tasks Details Breach Notes Members News Feed Attachments Stats Timeline Artifacts

**Splunk custom** Edit

splunk\_ip\_intel Search... Print Export

time	ip_intel_description	ip_intel_ip	intel_item_key
1525386575.465267	restapi	117.11.157.171	92699d6872874c53a04829a8e87efb50

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](mailto:support@resilientsystems.com).

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