

IBM Resilient



Incident Response Platform Integrations

BigFix Function V1.0.0

Release Date: Aug 2018

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

This guide describes the BigFix Integration Function.

Overview

BigFix is an endpoint management tool that allows users to keep systems or endpoints in an environment under its control, updated, compatible and free of security issues. It allows for the identification and remediation of a vulnerable endpoint from a central console.

The BigFix integration with the Resilient platform allows querying of a BigFix environment using the REST APIs, where the returned results can be used to remediate issues or hits, such as a malicious path or filename, a service or process name, or a registry key.

The four functions supplied in this Resilient package support the following use cases.

- Beginning with an Indicator of Compromise (IOC) such as a malicious path or filename, service or process name, registry key, or IP address, the BigFix integration allows you to search a BigFix environment for all affected endpoints with a hit, and then update a data table with this information where it can be displayed on the Resilient platform.
- Allows you to query BigFix for all available BigFix properties of an endpoint with a hit, and then attach an XML file with these properties to the Resilient incident.
- Allows you to execute BigFix remediation procedures from the Resilient platform against an endpoint with a hit. These procedures include killing a process, stopping a service, deleting a registry key (Microsoft Windows only) or a file.
- Allows you to query and update the status of a BigFix remediation action from the Resilient platform on an endpoint with a hit.

The remainder of this document describes the included Functions, how to configure example custom workflows, and any additional customization options.

Installation

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

- BigFix version must be 9.5 patch 2, or later.
- Resilient platform must be version 30 or later.
- A designated BigFix Console Operator account, with the Create Custom Content permission enabled. This account must be configured to access all those endpoints that you wish to have accessible to the Resilient platform.
- You must have a Resilient account to use for the integrations. This can be any account that has the permission to view and modify administrator and customization settings, and read and update incidents. You must know the account username and password.
- You have access to the command line of the Resilient appliance, which hosts the Resilient platform; or to a separate integration server where you will deploy and run the functions code. If you are using a separate integration server, you must install Python version 2.7.10 or later, or version 3.6 or later, and “pip”. (The Resilient appliance is preconfigured with a suitable version of Python.)

Install the Python components

The functions package contains Python components that are 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.

Complete the following steps to install the Python components:

1. Ensure that the environment is up-to-date, as follows:

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

2. Run the following command to install the package:

```
sudo pip install --upgrade fn_bigfix-1.0.0.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.

Complete the following steps to configure and run the integration:

1. Using `sudo`, switch to the `integration` user, as follows:

```
sudo su - integration
```

2. Use one of the following commands to create or update the `resilient-circuits` configuration file. Use `-c` for new environments or `-u` for existing environments.

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

or

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

3. Edit the `resilient-circuits` configuration file, as follows:

- a. In the `[resilient]` section, ensure that you provide all the information required to connect to the Resilient platform.
- b. In the `[fn_bigfix]` section, edit the settings as follows:

```
bigfix_url. URL of your BigFix server; for example: https://bigfix-  
url.com  
  
bigfix_port. Port number of your BigFix server.  
  
bigfix_user. Username of the BigFix Console Operator account used for  
this integration.  
  
bigfix_pass. Password for the BigFix Console Operator account.  
  
bigfix_polling_interval. Time in seconds that the integration waits  
between polling BigFix to get query results or the final status of the  
remediation actions. Default is 30  
  
bigfix_polling_timeout. Time in seconds that the integration waits before  
timing out while polling BigFix to get query results or the final status  
of the remediation actions. Default is 600  
  
hunt_results_limit. Limits the number of results sent to the Resilient  
platform. Default is 200.
```

Deploy customizations to the Resilient platform

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

1. Use the following command to deploy these customizations to the Resilient platform:

```
resilient-circuits customize
```

2. Respond to the prompts to deploy functions, message destinations, workflows and rules.

Run the integration framework

To test the integration package before running it in a production environment, you must run the integration manually, using the following command:

```
resilient-circuits run  
...
```

```
2018-08-01 16:49:02,931 INFO [app] Configuration file:
2018-08-01 16:49:02,932 INFO [app] Resilient server: <host>
2018-08-01 16:49:02,933 INFO [app] Resilient user: <acct>
2018-08-01 16:49:02,933 INFO [app] Resilient org: <org>
2018-08-01 16:49:02,934 INFO [app] Logging Level: INFO
...
2018-08-01 16:49:03,431 INFO [component_loader] Loading 4 components
2018-08-01 16:49:03,432 INFO [component_loader]
'fn_bigfix.components.fn_bigfix_assets.FunctionComponent' loading
2018-08-01 16:49:03,434 INFO [component_loader]
'fn_bigfix.components.fn_bigfix_artifact.FunctionComponent' loading
2018-08-01 16:49:03,435 INFO [component_loader]
'fn_bigfix.components.fn_bigfix_action_status.FunctionComponent' loading
2018-08-01 16:49:03,437 INFO [component_loader]
'fn_bigfix.components.fn_bigfix_remediation.FunctionComponent' loading
...
2018-08-01 16:49:03,451 INFO [actions_component]
'fn_bigfix.components.fn_bigfix_assets.FunctionComponent' function
'fn_bigfix_assets' registered to 'bigfix_asset'
2018-08-01 16:49:03,452 INFO [actions_component]
'fn_bigfix.components.fn_bigfix_artifact.FunctionComponent' function
'fn_bigfix_artifact' registered to 'bigfix_artifact'
2018-08-01 16:49:03,453 INFO [actions_component]
'fn_bigfix.components.fn_bigfix_action_status.FunctionComponent' function
'fn_bigfix_action_status' registered to 'bigfix_remediation'
2018-08-01 16:49:03,453 INFO [app] App Started
2018-08-01 16:49:03,455 INFO [actions_component]
'fn_bigfix.components.fn_bigfix_remediation.FunctionComponent' function
'fn_bigfix_remediation' registered to 'bigfix_remediation'
2018-08-01 16:49:03,456 INFO [app] Components loaded
...
2018-08-01 16:49:03,794 INFO [actions_component] Subscribe to message
destination 'bigfix_remediation'
2018-08-01 16:49:03,795 INFO [actions_component] Subscribe to message
destination 'bigfix_asset'
2018-08-01 16:49:03,796 INFO [actions_component] Subscribe to message
destination 'bigfix_artifact'
2018-08-01 16:49:03,797 INFO [stomp_component] Subscribe to message
destination actions.202.bigfix_remediation
2018-08-01 16:49:03,798 INFO [stomp_component] Subscribe to message
destination actions.202.bigfix_asset
2018-08-01 16:49:03,799 INFO [stomp_component] Subscribe to message
destination actions.202.bigfix_artifact
```

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.

Configure Resilient Circuits for restart

For normal operation, Resilient Circuits must run continuously. The recommended way to do this is to configure it to automatically run at start up. On a Red Hat appliance, you can do this using a systemd unit file such as the one below. You might need to change the paths to your working directory and `app.config`.

1. The unit file must be named `resilient_circuits.service` To create the file, enter the following command:

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

2. Add the following contents to the file and change as necessary:

```
[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
```

3. Ensure that the service unit file is correctly permissioned, as follows:

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

4. Use the `systemctl` command to manually start, stop, restart and return status on the service:

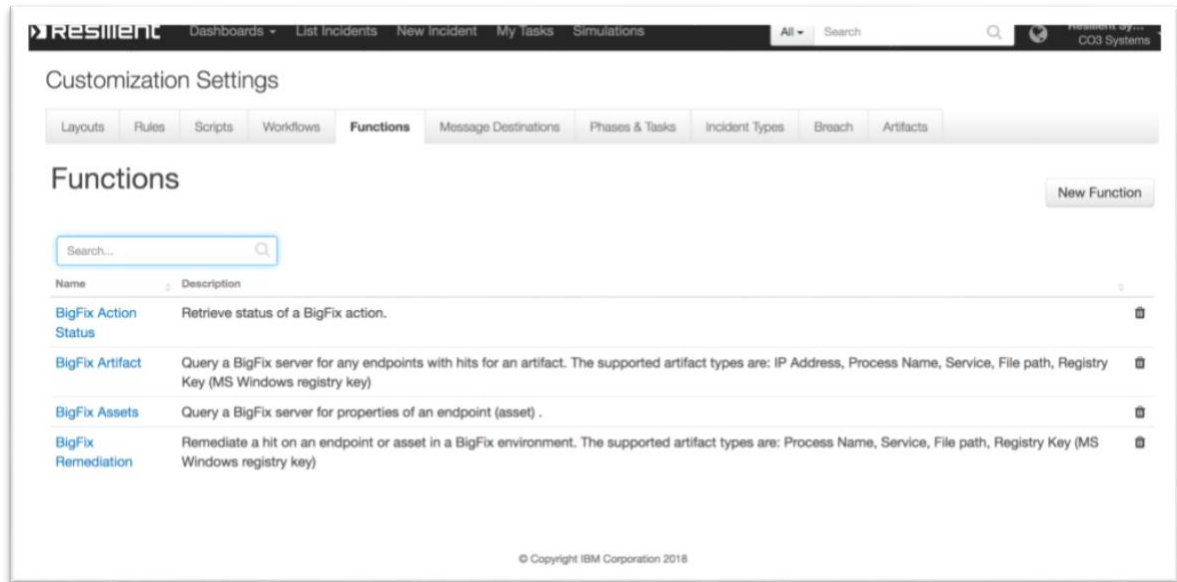
```
sudo systemctl resilient_circuits [start|stop|restart|status]
```

You can view log files for systemd and the resilient-circuits service using the `journalctl` command, as follows:

```
sudo journalctl -u resilient_circuits --since "2 hours ago"
```

Function Descriptions

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



Customizations

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

BigFix Artifact

This function performs a query that retrieves a list of endpoints with hits from a BigFix environment.

Customization Settings

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

[Functions](#) / `fn_bigfix_artifact`

Name *
API Name * ⓘ
Message Destination *
Description

bigfix_artifact ▾

Query a BigFix server for any endpoints with hits for an artifact.
The supported artifact types are:

Inputs

bigfix_artifact_id ×

bigfix_artifact_value ×

bigfix_artifact_type ×

bigfix_incident_id ×

bigfix_incident_plan_status ×

bigfix_artifact_properties_name ×

bigfix_artifact_properties_value ×

This function takes the following parameters:

- `bigfix_artifact_id` - Resilient artifact ID
- `bigfix_artifact_value` - Resilient artifact value
- `bigfix_artifact_type` - Resilient artifact type
- `bigfix_incident_id` - Resilient incident ID
- `bigfix_incident_plan_status` - Resilient incident status
- `bigfix_artifact_properties_name` - Resilient artifact properties name; optional, used for registry key value name (MS Windows)
- `bigfix_artifact_properties_value` - Resilient artifact properties name; optional, used for registry key value data (MS Windows)

The example workflow (object type = Artifact) that calls this function is “Example: BigFix Query for Artifact”.

The parameter assignments are done in the Pre-Process Script tab.

Workflows / Example: BigFix Query for Artifact

Name * Example: BigFix Query for Artifact

API Name * bigfix_query_for_artifact

Description Query a BigFix server for any endpoints in the BigFix environment with hits for an artifact.

Object Type * Artifact

Input Pre-Process Script Output Post-Process Script

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

```

1 inputs.bigfix_artifact_id = artifact.id
2 inputs.bigfix_artifact_value = artifact.value
3 inputs.bigfix_artifact_type = artifact.type
4 if artifact.properties is not None:
5     inputs.bigfix_artifact_properties_name = artifact.properties[0]["name"]
6     inputs.bigfix_artifact_properties_value = artifact.properties[0]["value"]
7 inputs.bigfix_incident_id = incident.id
8 inputs.bigfix_incident_plan_status = incident.plan_status

```

A Menu Item rule called “Example: BigFix Query for Artifact” is also included. This rule calls the workflow above. A user can invoke the workflow by right-clicking on this rule from the Actions drop-down menu of a suspect artifact.

RESILIENT Dashboards - List Incidents New Incident My Tasks Simulations All Search Resilient Sysadmin CO3 Systems

BigFix test

Actions

Summary

ID 2095

Phase Respond

Severity Low

Date Created 09/21/2018

Date Occur... —

Date Discov... 09/21/2018

Data Compr... Unknown

Incident Type —

People

Created By Resilient Sysadmin

Owner Resilient Sysadmin

Members There are no members.

Related Incidents

No related incidents.

Attachments

There are no attachments.

Newsfeed

Resilient Sysadmin added a row to the

Description

No description.

Tasks Details Breach Notes Members News Feed Attachments Stats Timeline **Artifacts**

Artifacts

Add Artifact Table Graph

Search...

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

Show 25 entries

Type	Value	Created	Relate?	Actions
File Path	C:\temp\testfile.txt	09/21/2018	As specified in artifact type settings	Example: BigFix Query for Artifact
File Path	/tmp/testfile.txt	09/21/2018	As specified in artifact type settings	
Registry Key	HKEY_LOCAL_MACHINE\SOFTWARE\TEST\TEST\com.tst.browsercore	09/21/2018	As specified in artifact type settings	

BigFix Query Results

Search... Print Export

Query Execution Date	Artifact Type	Artifact Value	BigFix Computer ID	BigFix Computer Name	Remediation Status	BigFix Action ID	Remediation Date
There is no data for this table							

If any endpoints are detected in the BigFix environment with the suspected artifact, entries are added to the data table “BigFix Query Results”.

BigFix Query Results							
				Search...	Q	Print	Export
Query Execution Date	Artifact Type	Artifact Value	BigFix Computer ID	BigFix Computer Name	Remediation Status	BigFix Action ID	Remediation Date
09-21-2018 12:14:31	File Path	/tmp/testfile.txt	12315195	bigfix.test	None	—	—

BigFix Remediation

This function creates a BigFix action to remediate a hit found on an endpoint in the BigFix environment.

Customization Settings

LayoutsRulesScriptsWorkflows**Functions**Message DestinationsPhases & TasksIncident Types

Functions / fn_bigfix_remediation

Name *

BigFix Remediation

API Name * ⓘ

fn_bigfix_remediation

Message Destination *

bigfix_remediation

Description

Remediate a hit on an endpoint or asset in a BigFix environment.
The supported artifact types are:

Inputs

bigfix_asset_id

bigfix_artifact_value

bigfix_artifact_type

bigfix_incident_id

This function takes the following parameters:

- bigfix_asset_id – Bigfix endpoint or asset ID
- bigfix_artifact_value - Resilient artifact value
- bigfix_artifact_type - Resilient artifact type
- bigfix_incident_id - Resilient incident ID

- The example workflow (object type = Data Table) that calls this function is “Example: BigFix Remediate”.

Workflows / Example: BigFix Remediate

Name * Example: BigFix Remediate

API Name * ⓘ bigfix_remediate

Description Remediate or fix a hit in a BigFix environment and return status of the remediating action.

Object Type * Data Table

Data table * BigFix Query Results

Input Pre-Process Script Output Post-Process Script

Input Parameter	Value
bigfix_asset_id * ⓘ	
bigfix_artifact_value * ⓘ	
bigfix_artifact_type * ⓘ	
bigfix_incident_id * ⓘ	

The parameter assignments are done in the Pre-Process Script tab.

Workflows / Example: BigFix Remediate

Name * Example: BigFix Remediate

API Name * ⓘ bigfix_remediate

Description Remediate or fix a hit in a BigFix environment and return status of the remediating action.

Object Type * Data Table

Data table * BigFix Query Results

Input Pre-Process Script Output Post-Process Script

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

```

1 inputs.bigfix_asset_id = row.res.bigfix_computer_id
2 inputs.bigfix_artifact_value = row.res.artifact_value
3 inputs.bigfix_artifact_type = row.res.artifact_type
4 inputs.bigfix_incident_id = incident.id

```

A Menu Item rule called “Example: BigFix Remediate” is also included. This rule calls the workflow. A user can invoke the workflow by right-clicking on this rule from the Actions drop-down or a data table entry for an endpoint with a hit.

Artifacts

Search...

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

Show 25 entries

Type	Value	Created	Relate?	Actions
Registry Key	HKEY_LOCAL_MACHINE\SOFTWARE\TEST\TEST\com.tst.browsercore	09/21/2018	As specified in artifact type settings	<div></div> <div></div>
File Path	/tmp/testfile.txt	09/21/2018	As specified in artifact type settings	<div></div> <div></div>
File Path	C:\temp\testfile.txt	09/21/2018	As specified in artifact type settings	<div></div> <div></div>

BigFix Query Results

Search...

Print Export

Artifact Type	Artifact Value	BigFix Computer ID	BigFix Computer Name	Remediation Status	BigFix Action ID	Remediation Date	
File Path	/tmp/testfile.txt	12315195	bigfix.test	None	—	—	...

Example: BigFix Remediate

Example: BigFix Retrieve Resource Details

Displaying 1 - 1 of 1

If a remediating BigFix action is successfully created, the entry in the data table “BigFix Query Results” which the workflow was invoked against, is updated with the status, remediation date and action ID.

BigFix Query Results

Search...

Print Export

Query Execution Date	Artifact Type	Artifact Value	BigFix Computer ID	BigFix Computer Name	Remediation Status	BigFix Action ID	Remediation Date
09-21-2018 12:14:31	File Path	/tmp/testfile.txt	12315195	bigfix.test	BigFix action created successfully.	268	09-21-2018 2:17:42

Displaying 1 - 1 of 1

BigFix Action Status

Customization Settings

LayoutsRulesScriptsWorkflows**Functions**Message DestinationsPhases & TasksIncident Types

Functions / fn_bigfix_action_status

Name *

BigFix Action Status

API Name * ⓘ

fn_bigfix_action_status

Message Destination *

bigfix_remediation

Description

Retrieve status of a BigFix action.

Inputs

bigfix_action_id

This function takes the following parameter:

- bigfix_action_id – Bigfix action ID
- The example workflow (object type = Data Table) that calls this function is “Example: BigFix Update Action status”.

Workflows / Example: BigFix Update Action status

Name *

Example: BigFix Update Action status

API Name * ⓘ

bigfix_update_action_status

Description

Update status of a BigFix action which mediates a hit.

Object Type *

Data Table

Data table *

BigFix Query Results

InputPre-Process ScriptOutputPost-Process Script

Input Parameter

Value

bigfix_action_id * ⓘ

The parameter assignment is done in the Pre-Process Script tab.

Customization Settings

LayoutsRulesScripts**Workflows**FunctionsMessage DestinationsPhases & TasksIncident Types

Workflows / Example: BigFix Update Action status

Name *Example: BigFix Update Action status

API Name * ⓘbigfix_update_action_status

DescriptionUpdate status of a BigFix action which mediates a hit.

Object Type *Data Table

Data table *BigFix Query Results

InputPre-Process ScriptOutputPost-Process Script

language: PythonTheme: lightMode: DefaultTab Size: 2- Font+ Font

1inputs.bigfix_action_id = row.res_bigfix_action_id

A Menu Item rule called “Example: BigFix Update Action status” is also included. This rule calls the workflow. A user can invoke the workflow by right-clicking on this rule from the Actions drop-down of a data table entry for an endpoint with a hit and where an action ID has been set.

Artifacts

Edit

Add Artifact

Table

Graph

Search...

Artifact Type: All

Date Created: All

Has Attachment: All

Show 25 entries

Type	Value	Created	Relate?	Actions
File Path	C:\temp\testfile.txt	09/21/2018	As specified in artifact type settings	
File Path	/tmp/testfile.txt	09/21/2018	As specified in artifact type settings	
Registry Key	HKEY_LOCAL_MACHINE\SOFTWARE\TEST\TEST\com.tst.browsercore	09/21/2018	As specified in artifact type settings	

BigFix Query Results

Search...

Print

Export

Artifact Type	Artifact Value	BigFix Computer ID	BigFix Computer Name	Remediation Status	BigFix Action ID	Remediation Date	
File Path	/tmp/testfile.txt	12315195	bigfix.test	BigFix action created successfully	269	09-21-2018 13:34:53	

Example: BigFix Remediate

Example: BigFix Retrieve Resource Details

Example: BigFix Update Action status

Displaying 1 - 1 of 1

If a remediating BigFix action was executed successfully, the entry in the data table “BigFix Query Results” which the workflow was invoked against, is updated with the new status.

BigFix Query Results

Search...

Print

Export

Query Execution Date	Artifact Type	Artifact Value	BigFix Computer ID	BigFix Computer Name	Remediation Status	BigFix Action ID	Remediation Date
09-21-2018 13:34:40	File Path	/tmp/testfile.txt	12315195	bigfix.test	The action executed successfully.	269	09-21-2018 13:34:53

Displaying 1 - 1 of 1

This function is also included in the “Example: BigFix Remediate” workflow and it is invoked automatically as part of that workflow. This would be the more common method of invocation.

Workflows / Example: BigFix Remediate

Name *

Example: BigFix Remediate

API Name * ⓘ

bigfix_remediate

Description

Remediate or fix a hit in a BigFix environment and return status of the remediating action.

Object Type *

Data Table

Data table *

BigFix Query Results

Hand icon

Plus icon

Lightning bolt icon

Circle icon

Circle with dot icon

Circle with cross icon

Circle with plus icon

Circle with minus icon

Person icon

Group icon

Document icon

Document with checkmark icon

f(x) icon

Start your workflow here

Start node

BigFix Remediation

BigFix Action Status

In cases where the “Example: BigFix Remediate” workflow does not receive the status within the specified time, this workflow can be invoked manually at a later time.

Page 15

BigFix Assets

This function performs a query to fetch BigFix properties of an endpoint with a hit from a BigFix environment.

The screenshot shows the 'Customization Settings' interface for a function named 'BigFix Assets'. The interface has a top navigation bar with tabs: Layouts, Rules, Scripts, Workflows, **Functions**, Message Destinations, Phases & Tasks, and Incident Type. Below the tabs, the breadcrumb 'Functions / fn_bigfix_assets' is visible. The main form contains the following fields:

- Name ***: A text input field containing 'BigFix Assets'.
- API Name ***: A text input field containing 'fn_bigfix_assets'.
- Message Destination ***: A dropdown menu with 'bigfix_asset' selected.
- Description**: A text area containing 'Query a BigFix server for properties of an endpoint (asset) .'. There is a small icon in the bottom right corner of the text area.

Below the main form is an 'Inputs' section, which is a dashed box containing three input fields, each with a close button (x) on the right:

- bigfix_asset_id
- bigfix_asset_name
- bigfix_incident_id

This function takes the following parameter:

- bigfix_asset_id – Bigfix endpoint or asset ID
- bigfix_asset_name - Bigfix endpoint or asset name
- bigfix_incident_id - Resilient incident ID
- The example workflow (object type = Data Table) that calls this function is “Example: BigFix Retrieve Resource Details”.

Customization Settings

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

Workflows / Example: BigFix Retrieve Resource Details

Name *
Example: BigFix Retrieve Resource Details
API Name * ⓘ
bigfix_retrieve_resource_details
Description
Retrieve properties of an endpoint in a BigFix environment.
Object Type *
Data Table
Data table *
BigFix Query Results

Input
Pre-Process Script
Output
Post-Process Script

Input Parameter	Value
bigfix_asset_id * ⓘ	
bigfix_asset_name * ⓘ	
bigfix_incident_id * ⓘ	

The parameter assignments are done in the Pre-Process Script tab.

Customization Settings

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

Workflows / Example: BigFix Retrieve Resource Details

Name *
Example: BigFix Retrieve Resource Details
API Name * ⓘ
bigfix_retrieve_resource_details
Description
Retrieve properties of an endpoint in a BigFix environment.
Object Type *
Data Table
Data table *
BigFix Query Results

Input
Pre-Process Script
Output
Post-Process Script

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

```

1 inputs.bigfix_asset_name = row.res_bigfix_computer_name
2 inputs.bigfix_asset_id = row.res_bigfix_computer_id
3 inputs.bigfix_incident_id = incident.id

```

A Menu Item rule called “Example: BigFix Retrieve Resource Details” is also included. This rule calls the workflow. A user can invoke the workflow by right-clicking on this rule from the Actions drop-down of a data table entry for an endpoint with a hit.

Artifacts

Edit

Add Artifact

Table

Graph

Search...

Artifact Type: All

Date Created: All

Has Attachment: All

Show 25 entries

Type	Value	Created	Relate?	Actions
File Path	C:\temp\testfile.txt	09/21/2018	As specified in artifact type settings	
File Path	/tmp/testfile.txt	09/21/2018	As specified in artifact type settings	
Registry Key	HKEY_LOCAL_MACHINE\SOFTWARE\TEST\TEST\com.tst.browsercore	09/21/2018	As specified in artifact type settings	

BigFix Query Results

Search...

Print

Export

Artifact Type	Artifact Value	BigFix Computer ID	BigFix Computer Name	Remediation Status	BigFix Action ID	Remediation Date	
File Path	/tmp/testfile.txt	12315195	bigfix.test	The action ex	269	09-21-2018 1	...

Displaying 1 - 1 of 1

Example: BigFix Remediate

Example: BigFix Retrieve Resource Details

Example: BigFix Update Action status

An attachment is added to the incident containing BigFix properties of the targeted endpoint.

Attachments

Drag file here

Upload File

Maximum file size: 25 MB

Search...

Show Task Attachments

Uploaded By: All

Date Created: All

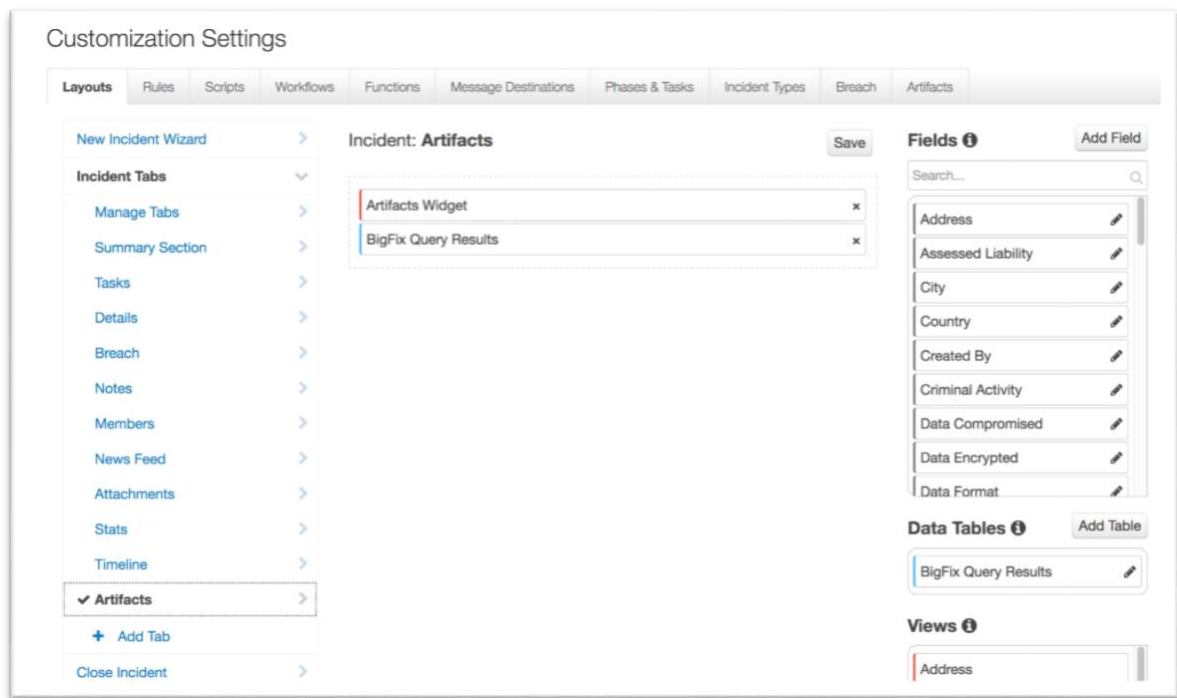
Type	Name	Uploaded By	Date Added	Size	Actions
	bigfix-properties-bigfix.test-20180921.xml	Resilient Sysadmin	09/21/2018	37 KB	

Showing 1 to 1 of 1 entries

Resilient Platform Configuration

To display query results, users need to manually add the “BigFix Query Results” data table to the Artifacts tab.

1. Navigate to the Customization Settings and select the Layouts tab.
2. Select Artifacts.
3. Drag the “BigFix Query Results” data table to your Artifacts tab.
4. Click Save.



Next are the details about how each function is used in the example workflow and rule after the function package customizations are deployed to the Resilient instance, you can view the functions in the Functions tab in the Resilient platform, as shown in the following screenshots.

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.