IBM Resilient

Incident Response Platform Integrations

ElasticSearch Query Function V1.0.0

Release Date: July 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 ElasticSearch Function.

Overview

The ElasticSearch integration allows users of the Resilient Platform to connect to and query an ElasticSearch Database.

Users can specify the location of a remote ElasticSearch instance and query this instance for data which is then returned to Resilient for display or use by other functions.

Queries provided to the function must be properly formed to work.

Please review ElasticSearch documentation for examples on how to form your query.

https://www.elastic.co/guide/en/elasticsearch/reference/6.3/search-request-body.html

A number of example queries are available when setting up the function in a workflow.

Installation

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

- Resilient platform is version 30 or later.
- You 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 need to 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 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. To install the package, you must first unzip it then install the package as follows:

```
sudo pip install --upgrade fn_<fn_name>-<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.

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_elasticsearch] section, for a non-encrypted connection edit the settings as follows:

```
[fn_elasticsearch]
```

```
es_datastore_url = <URL of your Elasticsearch store>
es_datastore_scheme = http
```

c. In the [fn_elasticsearch] section, **for an encrypted connection** over **SSL/TLS** edit the settings as follows:

```
[fn_elasticsearch]
es_datastore_url = <URL of your Elasticsearch store>
es_datastore_scheme = https
es_auth_username = <YOUR_USERNAME>
es_auth_password = <YOUR_PASSWORD>
es_cafile = <CA_File If used>
```

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 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.

Configure Resilient Circuits for restart

For normal operation, Resilient Circuits must run continuously. The recommend way to do this is to configure it 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.

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 contents to the file similar to the following 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 function(s), 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.

Fn_ElasticSearch_Query

The ElasticSearch query function allows users of the Resilient Platform to connect to and query an ElasticSearch Database using ElasticSearch's DSL query language.

Users can specify the location of a remote ElasticSearch instance and query this instance for data which is then returned to Resilient for display or use by other functions.

The function takes 3 inputs :

Input Name	Type	Example
es_index	An string index to search for data. Defaults to searching all indices if none provided	Test_data
es_Doc_type	A type of document that will be searched for data. Defaults to all doc_types	Test_type
es_query	A query that will be submitted to the elastic datastore. Requires a valid JSON payload	{"query": {match_all": {}}}

Queries provided to the function must be properly formed to work. The query submitted by the Resilient user is sent as-is to ElasticSearch.

A number of example queries are available when setting up the function in a workflow.

Two options are available for connection:

HTTP connection to localhost or remote

HTTPS connection with username:password authentication

If you wish to connect to a resource with a self signed cert can provide a cafile as one of the config options.

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.