# **IBM** Resilient



# **Incident Response Platform Integrations**

Email Header Validation Function V1.0.0

Release Date: September 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 Email Header Validation Function.

#### **Overview**

The Email Header Validation Function allows users of the Resilient Platform to analyze DKIM (DomainKeys Identified Mail) and ARC (Authenticated Received Chain) headers in a specified RFC822 formatted email.

#### 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. Run the following command to install the package:

```
sudo pip install --upgrade fn email header validation-<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
```

#### **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 <u>continuously</u>. 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
```

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
```

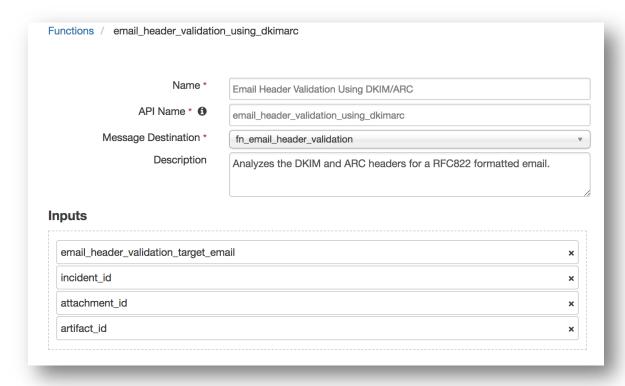
4. Use the systematl command to manually start, stop, restart and return status on the service:

```
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:

# **Function Descriptions**

Once the function package deploys the function(s), you can view them in the Resilient platform Functions tab, as shown below.



# **Resilient Platform Configuration**

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

# **Support**

For additional support, contact <a href="mailto:support@resilientsystems.com">support@resilientsystems.com</a>.

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