

# fn-teams Functions for IBM Resilient

---

- [Release Notes](#)
  - [Overview](#)
  - [Requirements](#)
  - [Installation](#)
  - [Uninstall](#)
  - [Troubleshooting](#)
  - [Support](#)
  - [Futures](#)
- 

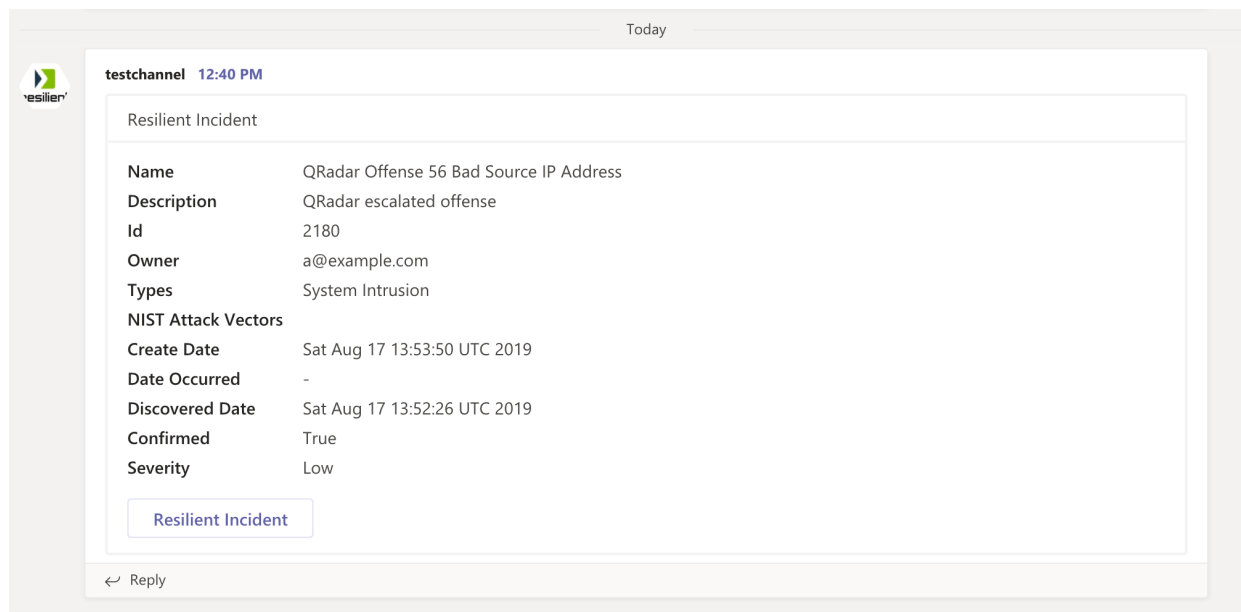
## Release Notes

v1.0.0

- Initial Release
- 

## Overview

### Resilient Circuits Components for fn\_teams



The screenshot shows a Microsoft Teams chat window. At the top, it says 'testchannel 12:40 PM'. Below this, a message from 'Resilient' is displayed. The message content is a table of incident details:

Resilient Incident	
Name	QRadar Offense 56 Bad Source IP Address
Description	QRadar escalated offense
Id	2180
Owner	a@example.com
Types	System Intrusion
NIST Attack Vectors	
Create Date	Sat Aug 17 13:53:50 UTC 2019
Date Occurred	-
Discovered Date	Sat Aug 17 13:52:26 UTC 2019
Confirmed	True
Severity	Low

Below the table is a button labeled 'Resilient Incident'. At the bottom of the chat window, there is a 'Reply' button.

This python package contains the integration code necessary to connect to Microsoft Soft teams to escalate incident data to an existing Teams channel.

Included are example workflows and rules for pushing incident and task information to a Teams channel. See the README documentation in the doc/ folder for further information on how to use this integration.

---

## Requirements

- Resilient platform >= **v31.0.4254**

- An Integration Server running `resilient_circuits>=30.0.0` 'resilient\_lib'
    - To set up an Integration Server see: [ibm.biz/res-int-server-guide](https://ibm.biz/res-int-server-guide).
- 

## Installation

- Download the `fn_teams.zip`.
- Copy the `.zip` to your Integration Server and SSH into it.
- **Unzip** the package:

```
$ unzip fn_teams-x.x.x.zip
```

- **Change Directory** into the unzipped directory:

```
$ cd fn_teams-x.x.x
```

- **Install** the package:

```
$ pip install fn_teams-x.x.x.tar.gz
```

- Import the `fn_teams` **customizations** into the Resilient platform:

```
$ resilient-circuits customize -y -l fn-teams
```

- Configure the `fn_teams` **app.config** settings. See [Configure](#) for setting changes

```
$ resilient-circuits config -l fn-teams
```

- [Optional]: Run selftest to test the Integration you configured:

```
$ resilient-circuits selftest -l fn-teams
```

- **Run** resilient-circuits or restart the Service on Windows/Linux:

```
$ resilient-circuits run
```

---

## Uninstall

- SSH into your Integration Server.
- **Uninstall** the package:

```
$ pip uninstall fn-teams
```

---

## 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.

---

## Configure fn\_teams

After running `resilient-circuits config -l fn-teams`, your app.config file will contain the following section

```
[fn_teams]
# add multiple parameters for the channels to access and their
webhook.
# The channel name is used in the function input: teams_channel
#<channel_name>=<teams channel webhook>
# use this channel reference to use the self-test capability of
resilient-circuits
#selftest=<teams channel webhook>
```

Any number of channels can be configured, each with it's own inbound webhook. If you're unfamiliar with Teams' inbound webhooks, refer to the setup documentation such as this [medium article](#)

Copy the webhook URL and add it to your `[fn_teams]` section using a label which refers to the channel. This label is then used in your Resilient workflow, configuring the `Teams Post Message` function's `teams_channel` input parameter.

## Support

Name	Version	Author	Support URL
fn_teams	1.0.0	Resilient Labs	Resilient Labs

## Futures

Microsoft is planning an full API for Teams communications. This API is presently in beta. When this API is released, this integration will be modified for more complete, bi-directional message posting.