IBM Resilient

Incident Response Platform Integrations

Exchange Function V1.0.0

Release Date: August 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 Exchange Function.

Overview

This Resilient Function package provides seven functions that work with Exchange:

1) Exchange Create Meeting – Creates a meeting in Exchange and send out invites
2) Exchange Delete Emails – Deletes queried emails from a user's mailbox
3) Exchange Find Emails – Queries emails from a user's mailbox
4) Exchange Get Mailbox Info – Gets mailbox info for a sender
5) Exchange Move Folder Contents and Delete Folder – Moves the contents of one folder to another folder and deletes the original
6) Exchange Move Emails – Moves queried emails from one folder to another folder
7) Exchange Send Email – Sends email to a list of recipients

The package also includes corresponding menu item rules and workflows that create Notes and Artifacts from the function results.

Installation

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

- Resilient platform is version 31 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_exchange-<version>.<zip>
```

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_exchange] section, edit the settings as follows:
```
verify_cert=[True|False]
server=example.com
username=domain\username
email=admin@example.com - this is the default account to send emails and
create meetings if one was not specified. Specifying an account that is
not this one will require impersonation access.
password=password
default_folder_path=Some folder path after root i.e. Top of Information
Store/Inbox
```

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 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
```

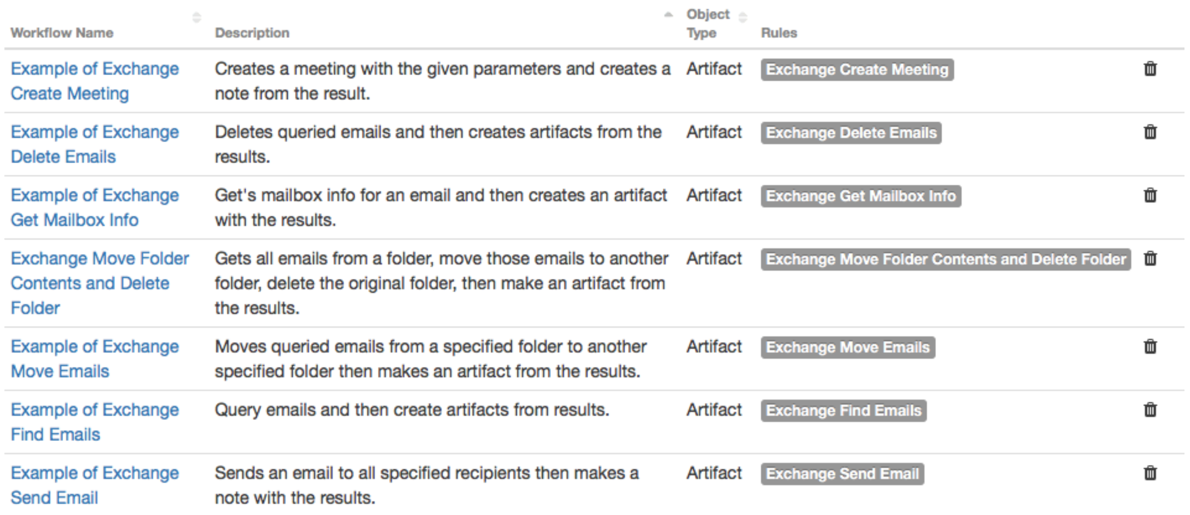
4. Use the systemctl command to manually start, stop, restart and return status on the service:

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



Workflow Name	Description	Object Type	Rules
Example of Exchange Create Meeting	Creates a meeting with the given parameters and creates a note from the result.	Artifact	Exchange Create Meeting
Example of Exchange Delete Emails	Deletes queried emails and then creates artifacts from the results.	Artifact	Exchange Delete Emails
Example of Exchange Get Mailbox Info	Get's mailbox info for an email and then creates an artifact with the results.	Artifact	Exchange Get Mailbox Info
Exchange Move Folder Contents and Delete Folder	Gets all emails from a folder, move those emails to another folder, delete the original folder, then make an artifact from the results.	Artifact	Exchange Move Folder Contents and Delete Folder
Example of Exchange Move Emails	Moves queried emails from a specified folder to another specified folder then makes an artifact from the results.	Artifact	Exchange Move Emails
Example of Exchange Find Emails	Query emails and then create artifacts from results.	Artifact	Exchange Find Emails
Example of Exchange Send Email	Sends an email to all specified recipients then makes a note with the results.	Artifact	Exchange Send Email

Functions and Components

The package includes example workflows and rules that show how you can use the functions, as shown in the following table. Resilient users can view the rules in the Rules tab and the workflows in the Workflows tab, and modify them as needed. The object type for the workflows is Artifact. The provided sample workflows create notes and artifacts from the function results.

Function	Rule	Workflow
exchange_create_meeting	Exchange Create Meeting	Example of Exchange Create Meeting
exchange_delete_emails	Exchange Delete Emails	Example of Exchange Delete Emails
exchange_find_emails	Exchange Find Emails	Example of Exchange Find Emails
exchange_get_mailbox_info	Exchange Get Mailbox Info	Example of Exchange Get Mailbox Info
exchange_move_emails	Exchange Move Emails	Example of Exchange Move Emails
exchange_move_folder_contents_and_delete_folder	Exchange Move Folder Contents and Delete Folder	Example of Exchange Move Folder Contents and Delete Folder

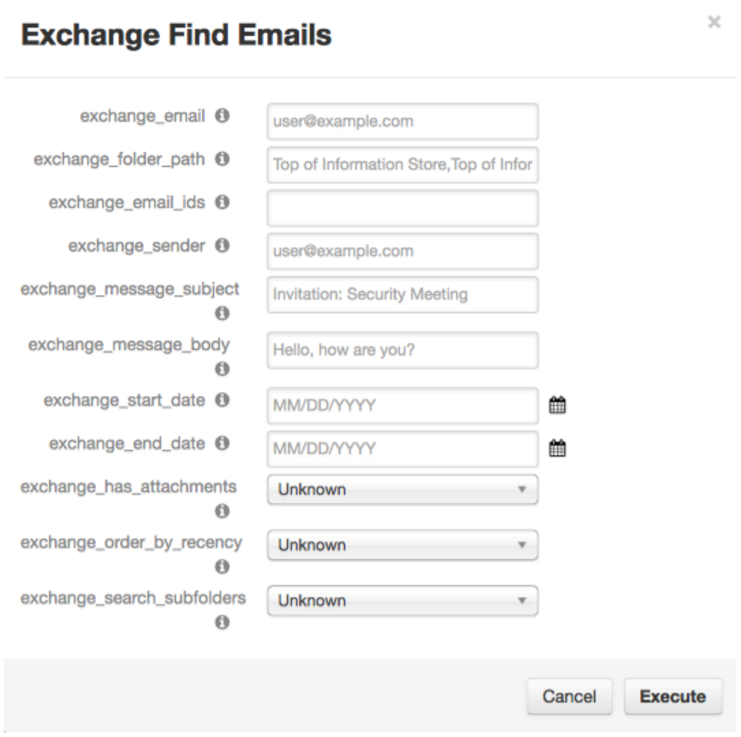
Function	Rule	Workflow
exchange_send_email	Exchange Send Email	Example of Exchange Send Email

Inputs

Each function has a set of inputs, which you can view by clicking the function name in the Functions tab of the Resilient platform.

The Resilient functions use input parameters starting with `exchange_`, examples include `exchange_email`, `exchange_subject` and `exchange_body`

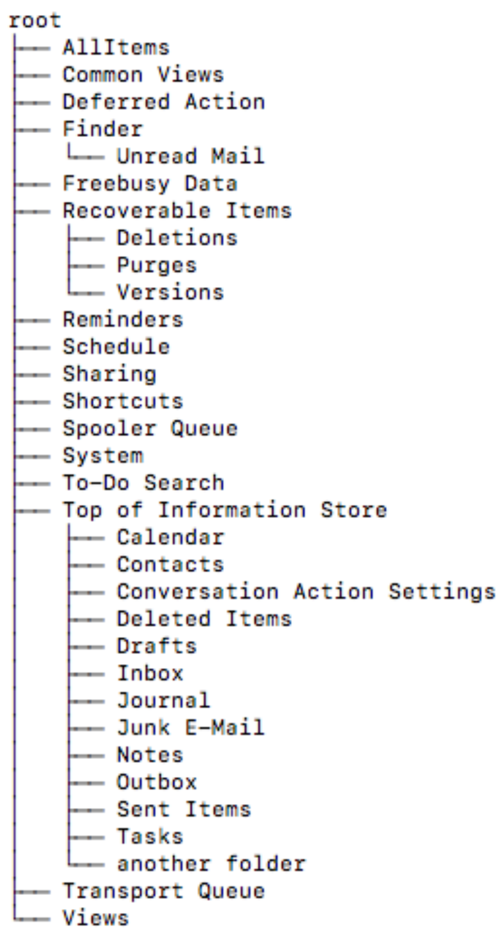
The function inputs can also be set by the user when clicking a menu item. For example:


Exchange Find Emails

exchange_email	user@example.com
exchange_folder_path	Top of Information Store,Top of Infor
exchange_email_ids	
exchange_sender	user@example.com
exchange_message_subject	Invitation: Security Meeting
exchange_message_body	Hello, how are you?
exchange_start_date	MM/DD/YYYY
exchange_end_date	MM/DD/YYYY
exchange_has_attachments	Unknown
exchange_order_by_recency	Unknown
exchange_search_subfolders	Unknown

The example preprocessing script uses the inputs from the popup menu when executing a function, if no fields are provided, then it uses the value from the workflow input/artifact.

The `exchange_folder_path` or `exchange_destination_folder_path` fields may be difficult to configure and are dependent on the Exchange environment. Upon entering an invalid folder path, a tree structure of the folder hierarchy will be printed. Here is an example:



```
root
├── AllItems
├── Common Views
├── Deferred Action
├── Finder
│   └── Unread Mail
├── Freebusy Data
├── Recoverable Items
│   ├── Deletions
│   ├── Purges
│   └── Versions
├── Reminders
├── Schedule
├── Sharing
├── Shortcuts
├── Spooler Queue
├── System
├── To-Do Search
├── Top of Information Store
│   ├── Calendar
│   ├── Contacts
│   ├── Conversation Action Settings
│   ├── Deleted Items
│   ├── Drafts
│   ├── Inbox
│   ├── Journal
│   ├── Junk E-Mail
│   ├── Notes
│   ├── Outbox
│   ├── Sent Items
│   ├── Tasks
│   └── another folder
├── Transport Queue
└── Views
```

Example folder paths given this folder structure could be any path following the root path:

```
-    Top of Information Store/Inbox
-    Top of Information Store/Deleted Items
-    Finder/Unread Mail
-    Finder
```

Additionally, if the `exchange_search_subfolders` path is set to `true`, every folder in its branch will be included in the query. For example if the specified folder is `Recoverable Items`, then the searched folders would be:

```
-   Recoverable Items
-   Recoverable Items/Deletions
-   Recoverable Items/Purges
-   Recoverable Items/Versions
```

To search folder paths, the specified account in config file must have access to the searched folders.

Folders that contain `/` or `,` must be wrapped in quotes.

```
-   Example/”One/With/Quotes”/Folder
-   Example/”One, with, commas”/Folder
-   Example/”One/with, both”/Folder
```

Multiple folder paths can be specified by separating them with commas and following the above rules.

For more information on specific function inputs, check the tooltips.

The Resilient functions use input parameters starting with Resilient Platform Configuration

The configuration file must specify credentials that have access to mailboxes that are being queried otherwise the functions can only query the account that is specified.

The package only currently supports on-premise Exchange servers. Functionality for Ofifce365 is not guaranteed.

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.