Component Files for App Host

Table of Contents

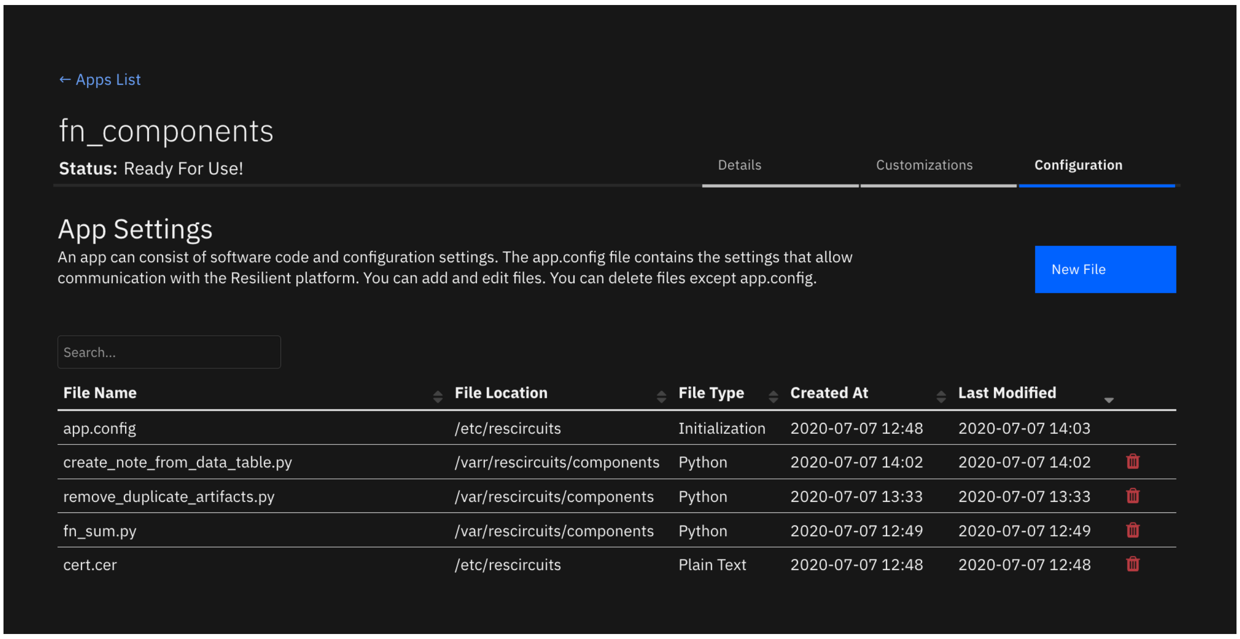
Revision History

Version	Date	Notes
1.0.0	07/2020	Initial Release

About This Package

This package is used to convert existing, single-file Python integrations to use the App Host framework. Today, the componentsdir parameter in the app.config file references a directory where these files reside.

To use these files in an App Host environment, install this app using the Apps tab within Administrative Settings and, through the Configuration tab of the app, add each single-file integration.



fn_components

Status: Ready For Use!

App Settings

An app can consist of software code and configuration settings. The app.config file contains the settings that allow communication with the Resilient platform. You can add and edit files. You can delete files except app.config.

File Name	File Location	File Type	Created At	Last Modified
app.config	/etc/rescircuits	Initialization	2020-07-07 12:48	2020-07-07 14:03
create_note_from_data_table.py	/varr/rescircuits/components	Python	2020-07-07 14:02	2020-07-07 14:02
remove_duplicate_artifacts.py	/var/rescircuits/components	Python	2020-07-07 13:33	2020-07-07 13:33
fn_sum.py	/var/rescircuits/components	Python	2020-07-07 12:49	2020-07-07 12:49
cert.cer	/etc/rescircuits	Plain Text	2020-07-07 12:48	2020-07-07 12:48

Container Environment

The container runs resilient-circuits similar to an Integration Server and continues to use the componentsdir app.config parameter. The following additional Python packages have been added to the container:

- ldap3
- jinja2
- json2html
- pytz
- requests
- resilient-lib
- six
- tldextract

If you require additional Python packages, refer to the section below on how to modify the container build enviornment.

Requirements

This App Host package assumes that the message destination, functions, and rules for each single-file integration are already defined in your Resilient platform. If you require moving your integrations between Resilient platform, consider converting your single-file integrations to fully packaged Apps using the resilient-sdk codegen tool and capability.

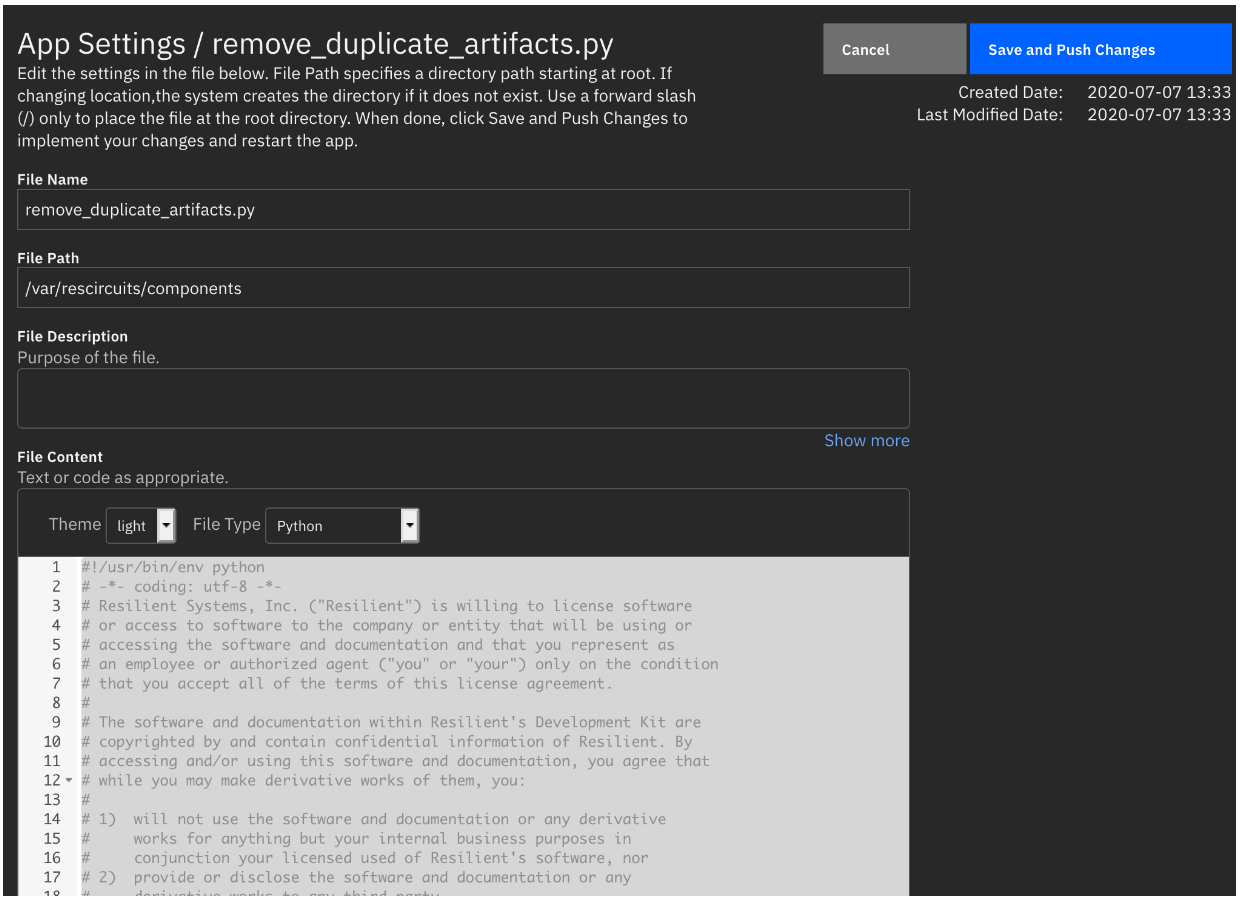
For each single-file integration:

- Each file must be Python 3 compatible.
- Have no additional Python packages required other than those specified in the container environment.
- Message destinations, functions, and rules used must already exist on your Resilient platform.

Installation and Configuration

With the app-fn_components-x.x.x.zip file downloaded from the AppExchange, navigate to the Apps tab within the Administrative Settings and install the package.

Navigate to the Configuration tab and click the New File button to specify the file name, file path and file contents of your single-file integration. Use /var/rescircuits/components for the file path and specify the file type as: Python.
Finally, cut and paste the content of the single-file Python code into the File Content window. Repeat these steps for each single-file Python integration.



App Settings / remove_duplicate_artifacts.py

Edit the settings in the file below. File Path specifies a directory path starting at root. If changing location,the system creates the directory if it does not exist. Use a forward slash (/) only to place the file at the root directory. When done, click Save and Push Changes to implement your changes and restart the app.

Created Date: 2020-07-07 13:33
Last Modified Date: 2020-07-07 13:33

File Name: remove_duplicate_artifacts.py

File Path: /var/rescircuits/components

File Description
Purpose of the file.

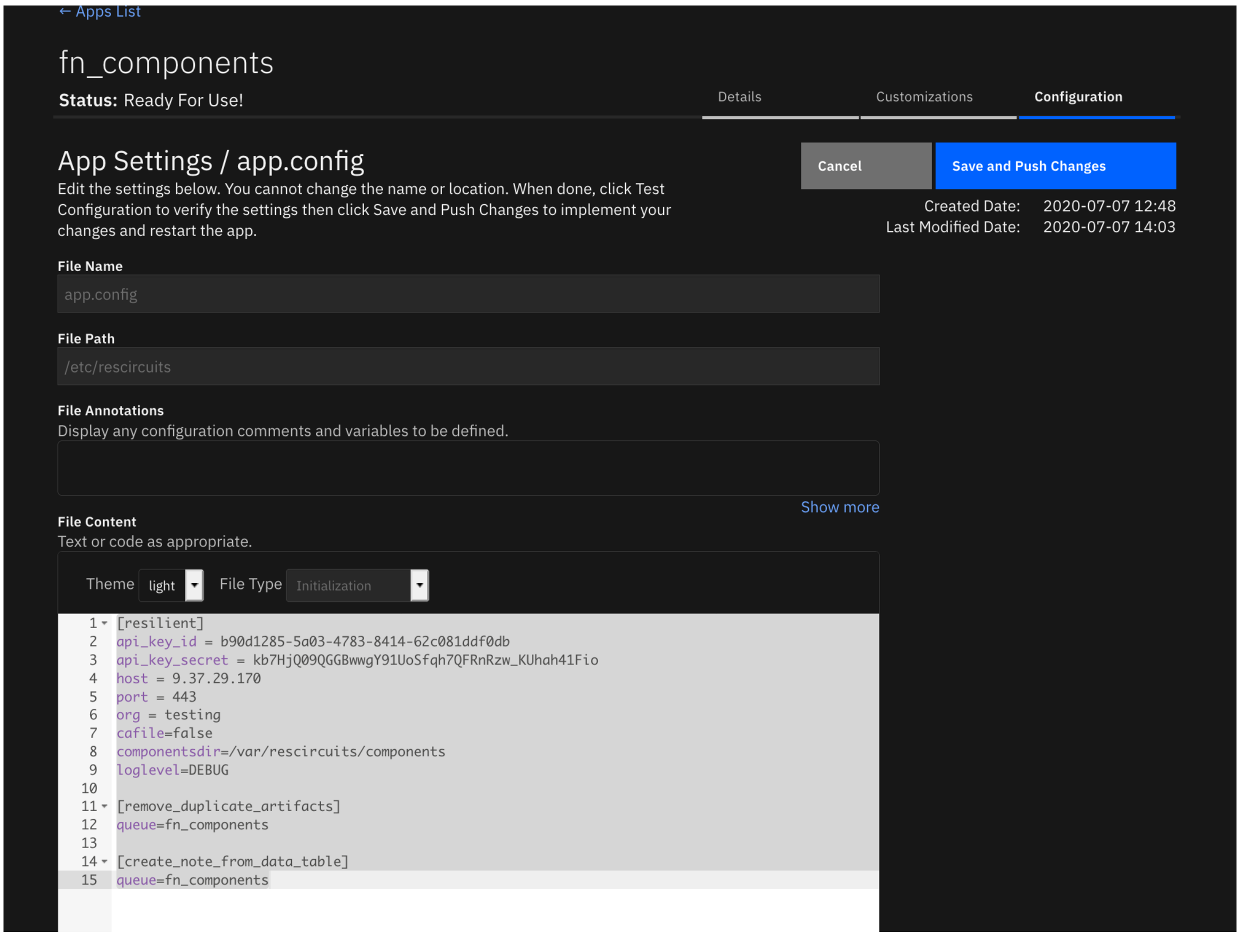
File Content
Text or code as appropriate.

Theme: light File Type: Python

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# Resilient Systems, Inc. ("Resilient") is willing to license software
# or access to software to the company or entity that will be using or
# accessing the software and documentation and that you represent as
# an employee or authorized agent ("you" or "your") only on the condition
# that you accept all of the terms of this license agreement.
#
# The software and documentation within Resilient's Development Kit are
# copyrighted by and contain confidential information of Resilient. By
# accessing and/or using this software and documentation, you agree that
# while you may make derivative works of them, you:
#
# 1)  will not use the software and documentation or any derivative
#     works for anything but your internal business purposes in
#     conjunction your licensed used of Resilient's software, nor
# 2)  provide or disclose the software and documentation or any
```

Within the app.config file, add the [resilient] parameter: componentsdir=/var/rescircuits/components

Each single-file integration may have additional sections and parameters to include in this file similar the settings you have already specified on your Integration Server's app.config file.



fn_components

Status: Ready For Use!

App Settings / app.config

Edit the settings below. You cannot change the name or location. When done, click Test Configuration to verify the settings then click Save and Push Changes to implement your changes and restart the app.

Created Date: 2020-07-07 12:48
Last Modified Date: 2020-07-07 14:03

File Name
app.config

File Path
/etc/rescircuits

File Annotations
Display any configuration comments and variables to be defined.

File Content
Text or code as appropriate.

Theme light File Type Initialization

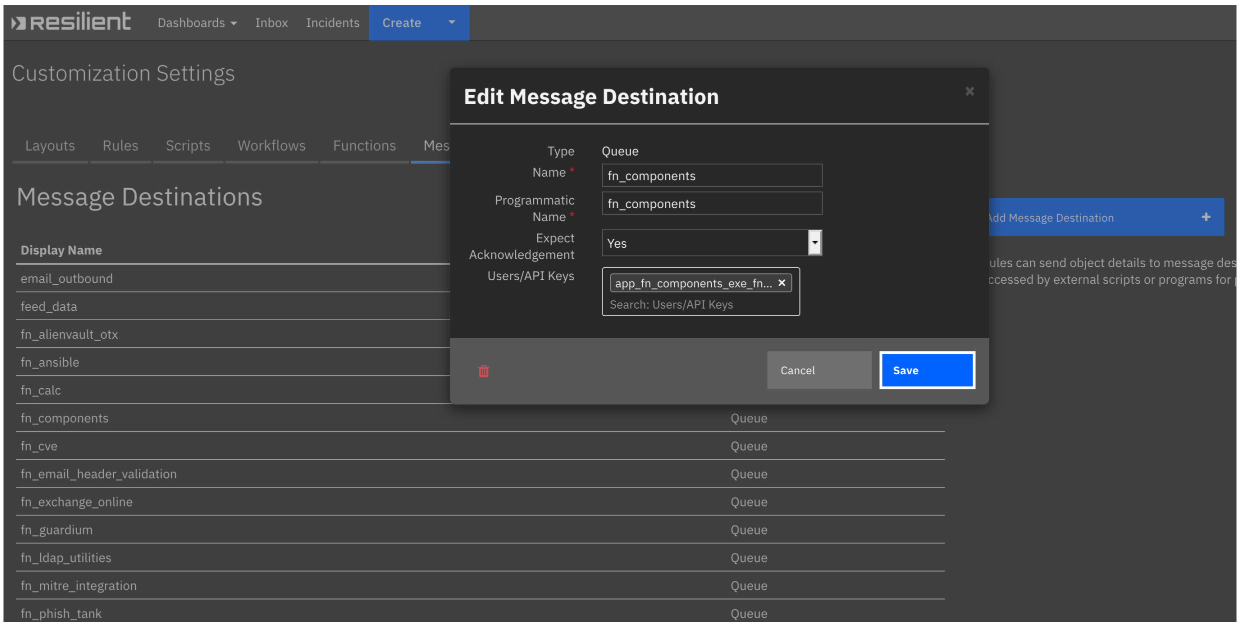
```
[resilient]
api_key_id = b90d1285-5a03-4783-8414-62c081ddf0db
api_key_secret = kb7HjQ09QGGBwwgY91UoSfqh7QFRnRzw_KUhah41Fio
host = 9.37.29.170
port = 443
org = testing
cafile=false
componentsdir=/var/rescircuits/components
loglevel=DEBUG

[remove_duplicate_artifacts]
queue=fn_components

[create_note_from_data_table]
queue=fn_components
```

Once all the single-file integrations and configurations added, return to the Details tab and click on the Deploy button.

Note: Once deployed, your single-file integrations are enabled for rule execution. It is best to remove these files from your Integration Server and restart resilient-circuits as both will be active otherwise.

Message Destination Setup

The next step is to add the API Key created for this container, app_fn_components_exe_fn_components, to each of the message destinations used by your single-file integrations. For convenience, all your single-file integrations can share the same message destination.



Message Destinations

Edit Message Destination

Type: Queue
Name *: fn_components
Programmatic Name *: fn_components
Expect Acknowledgement: Yes
Users/API Keys: app_fn_components_exe_fn... ×
Search: Users/API Keys

...ules can send object details to message des...
...ccessed by external scripts or programs for...

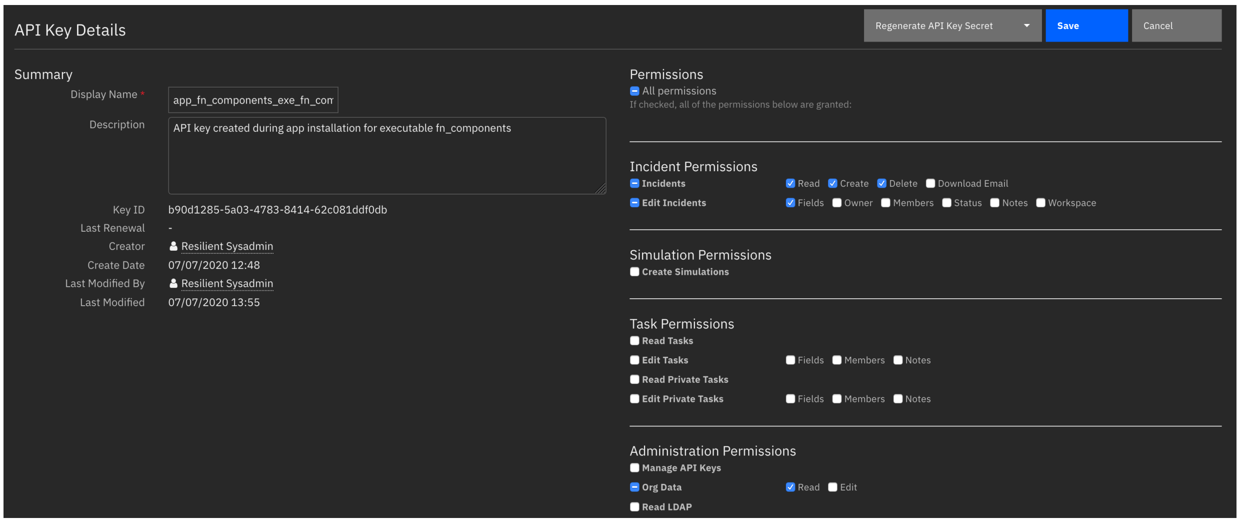
Display Name	
email_outbound	
feed_data	
fn_alienvault_otx	
fn_ansible	
fn_calc	
fn_components	Queue
fn_cve	Queue
fn_email_header_validation	Queue
fn_exchange_online	Queue
fn_guardium	Queue
fn_ldap_utilities	Queue
fn_mitre_integration	Queue
fn_phish_tank	Queue

API Key Permission Setup

Since it's not known in advance which API key permissions are required for your single-file integrations, it is necessary to review each integration for the specific API calls performed for their operation. The base permissions for this API key are:

- read and edit incident data.
- create, edit and delete incident elements, such as artifacts, attachments, notes, milestones and tasks.

If your single-file integrations require more or less permissions, edit the key's permissions set as necessary. Insufficient permissions will cause your integration to fail with an error message of forbidden.



API Key Details

Summary

Display Name *: app_fn_components_exe_fn_com
Description: API key created during app installation for executable fn_components
Key ID: b90d1285-5a03-4783-8414-62c081ddf0db
Last Renewal: -
Creator: Resilient Sysadmin
Create Date: 07/07/2020 12:48
Last Modified By: Resilient Sysadmin
Last Modified: 07/07/2020 13:55

Permissions
All permissions
If checked, all of the permissions below are granted:

Incident Permissions
Incidents: [✓ Read] [✓ Create] [✓ Delete] Download Email
Edit Incidents: [✓ Fields] Owner Members Status Notes Workspace

Simulation Permissions
Create Simulations

Task Permissions
Read Tasks
Edit Tasks: Fields Members Notes
Read Private Tasks
Edit Private Tasks: Fields Members Notes

Administration Permissions
Manage API Keys
Org Data: [✓ Read] Edit
Read LDAP

Adding Additional Python Files after Deployment

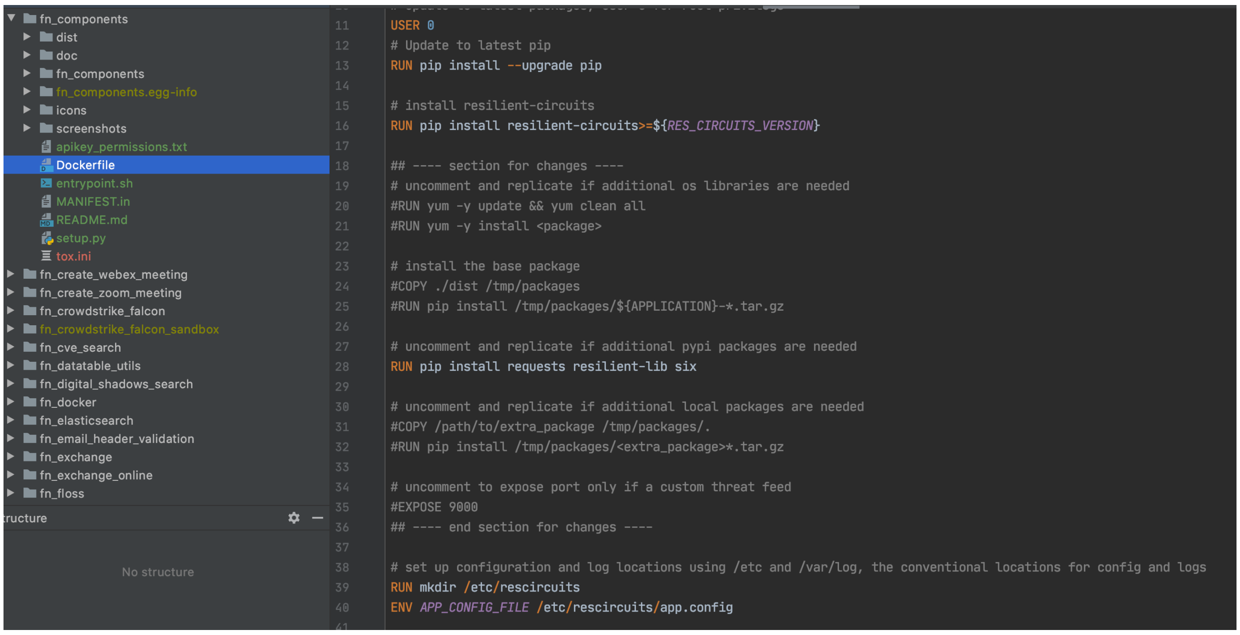
After the initial deployment of your fn_components app, repeat the steps in the Installation and Configuration section when adding additional single-file Python integrations. Changes to the app.config file will trigger the container to restart. Alternatively, click the Restart button in the Details tab.

Adding Additional Python Packages

In order to enable the container to include additional Python packages, it is necessary to rebuild the container. This is possible by unzipping the app-fn_components-x.x.x.zip file and then uncompressing the `fn_components-x.x.x.tar.gz` archive. Edit the enclosed `Dockerfile` to include additional Python packages. See the existing RUN command as an example:

`RUN pip install requests resilient-lib six`

Build the container using either `docker build` or `podman build` in your development environment. You will need to push the new container to your own registry and reference that repository in your App Host. Information on the use of private repositories can reviewed on the IBM Knowledge Center.



```
USER 0
# Update to latest pip
RUN pip install --upgrade pip

# install resilient-circuits
RUN pip install resilient-circuits>=${RES_CIRCUITS_VERSION}

## ---- section for changes ----
# uncomment and replicate if additional os libraries are needed
#RUN yum -y update && yum clean all
#RUN yum -y install <package>

# install the base package
#COPY ./dist /tmp/packages
#RUN pip install /tmp/packages/${APPLICATION}-*.tar.gz

# uncomment and replicate if additional pypi packages are needed
RUN pip install requests resilient-lib six

# uncomment and replicate if additional local packages are needed
#COPY /path/to/extra_package /tmp/packages/.
#RUN pip install /tmp/packages/<extra_package>*.tar.gz

# uncomment to expose port only if a custom threat feed
#EXPOSE 9000
## ---- end section for changes ----

# set up configuration and log locations using /etc and /var/log, the conventional locations for config and logs
RUN mkdir /etc/rescircuits
ENV APP_CONFIG_FILE /etc/rescircuits/app.config
```

App Host Troubleshooting

If your integration isn't running, there are a few ways to determine the cause and take corrective actions. Below are a few common issues and the steps to correct.

App Restart

Make sure to restart the App anytime you make additions and changes to component files.

Message destination

Check the logs to make sure your message destination is listened to. If not your log will have an entry similar to this:

```
2020-07-13 20:07:15,226 ERROR [actions_component] STOMP listener: Error:
b'java.lang.SecurityException: User a@example.com is not authorized to read from
queue://actions.201.fn_ansible
```

File names

Ensure your python files end in .py. Otherwise the list of loaded component files will bypass your integration. This log statement shows the files loaded when a container starts:

```
2020-07-27 18:35:46,000 INFO [app] Components auto-load directory:
/var/rescircuits/components
2020-07-27 18:35:46,007 INFO [component_loader] Loading 'create_note_from_data_table'
from /var/rescircuits/components/create_note_from_data_table.py
2020-07-27 18:35:46,008 INFO [component_loader] Loading 'utilities_json2html' from
/var/rescircuits/components/utilities_json2html.py
2020-07-27 18:35:46,008 INFO [component_loader] Loading 'utilities_expand_url' from
/var/rescircuits/components/utilities_expand_url.py
2020-07-27 18:35:46,008 INFO [component_loader] Loading 'utilities_call_rest_api' from
/var/rescircuits/components/utilities_call_rest_api.py
2020-07-27 18:35:46,009 INFO [component_loader] Loading 'utilities_attachment_to_base64'
from /var/rescircuits/components/utilities_attachment_to_base64.py
```

Import statements

Import statements which are unsupported will cause the container to become unusable and display a stack trace similar to below. Refer to section Adding Additional Python Packages on how to build containers for your additional packages.

```
2020-07-27 19:15:10,757 ERROR [component_loader] Failed to load component
'task_utils_create'
2020-07-27 19:15:10,757 DEBUG [component_loader] Name does not exist in modules
2020-07-27 19:15:10,802 ERROR [component_loader] No module named
'fn_cisco_umbrella_inv'
Traceback (most recent call last):
  File "/opt/app-root/lib/python3.6/site-packages/resilient_circuits/component_loader.py", line
40, in safe_but_noisy_import
    return __import__(name, globals(), locals(), [""])
  File "/var/rescircuits/components/task_utils_create.py", line 10, in <module>
    from fn_cisco_umbrella_inv.util.resilient_inv import ResilientInv
ModuleNotFoundError: No module named 'fn_cisco_umbrella_inv'
```