URLhaus Integration for Resilient

Release Notes

v1.0.2

- Added App Host support

v1.0.1

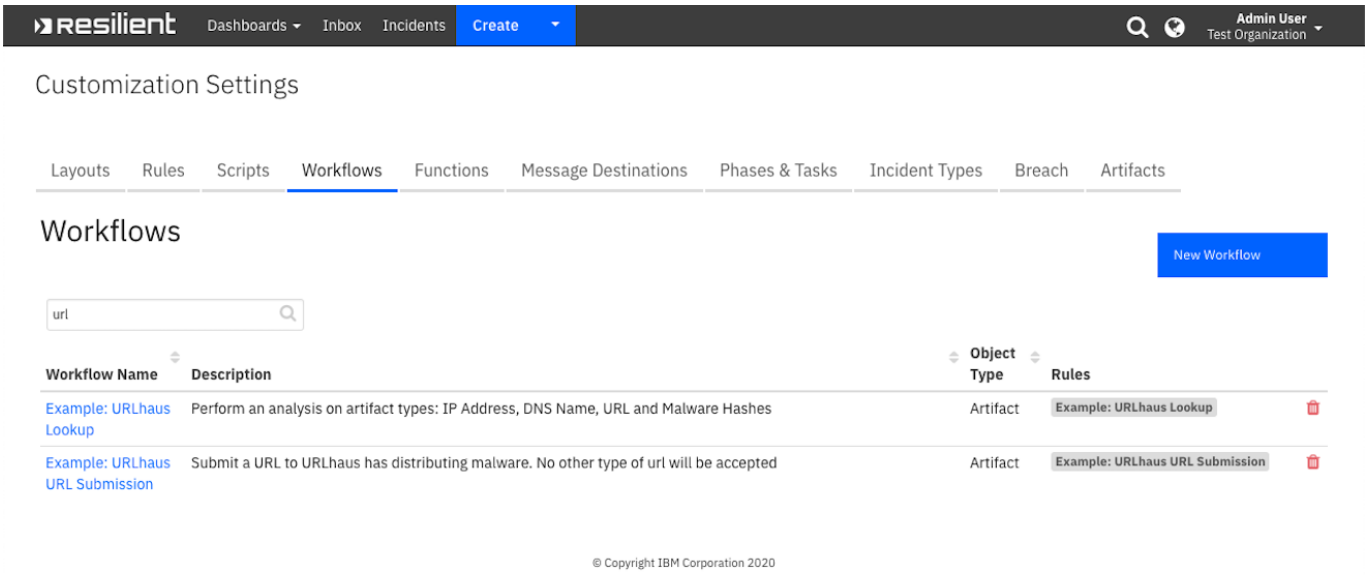
- Minor bug fix

v1.0.0

- Initial Release

Overview

Look up artifacts in URLhaus and submit malicious URLs



Look up supported artifacts in URLhaus to get more enrichment information and submit a malicious URL to the security community

---

Requirements

- Resilient platform >= `v35.0.0`
- An Integration Server running `resilient_circuits>=33.0.0`
  - To set up an Integration Server see: ibm.biz/res-int-server-guide
  - If using API Keys, minimum required permissions are:
    - Org Data: Read, Edit
    - Function: Read

Installation (App Host)

With App Host, all the run-time components are pre-built. Perform the following steps to install and configure:

- Download the `app-fn_urlhaus-x.x.x.zip`.
- In Resilient navigate to **Adiminstrator Settings > Apps**
- Click the Install button and select the downloaded `app-fn_urlhaus-x.x.x.zip`. This will install the associated customizations.
- Once installed, navigate to the app's Configuration tab and edit the app.config file updating the `[resilient]` section as necessary and updating the `[fn_urlhaus]` section as necessary.

Installation (Integration Server)

- Download the `app-fn_urlhaus-x.x.x.zip`.
- Copy the `.zip` to your Integration Server and SSH into it.

- **Unzip** the package:

```
$ unzip app-fn_urlhaus-x.x.x.zip
```

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

```
$ cd app-fn_urlhaus-x.x.x
```

- **Install** the package:

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

- Import the **configurations** into your app.config file:

```
$ resilient-circuits config -u -l fn-urlhaus
```

- Import the fn_urlhaus **customizations** into the Resilient platform:

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

- Open the config file, scroll to the bottom and edit your fn_urlhaus configurations:

```
$ nano ~/.resilient/app.config
```

Config	Required	Example	Description
url	Yes	`https://urlhaus-api.abuse.ch/v1`	The URL for URLhaus Lookup
submit_url	Yes	`https://urlhaus.abuse.ch/api/`	The URL for URLhaus Submissions
submit_api_key	Yes	`xxxxxxxxxx`	The API Key that you get when you authorize URLhaus in your twitter account

- **Save** and **Close** the app.config file.
- [Optional]: Run selftest to test the Integration you configured:

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

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

```
$ resilient-circuits run
```

Uninstall (Integration Server)

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

```
$ pip uninstall fn-urlhaus
```

- Open the config file, scroll to the [fn_urlhaus] section and remove the section or prefix `#` to comment out the section.
- **Save** and **Close** the app.config file.

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

Name	Version	Author	Support URL
fn_urlhaus	1.0.2	Resilient Labs	https://ibm.biz/resilientcommunity

User Guide: fn_urlhaus_v1.0.2

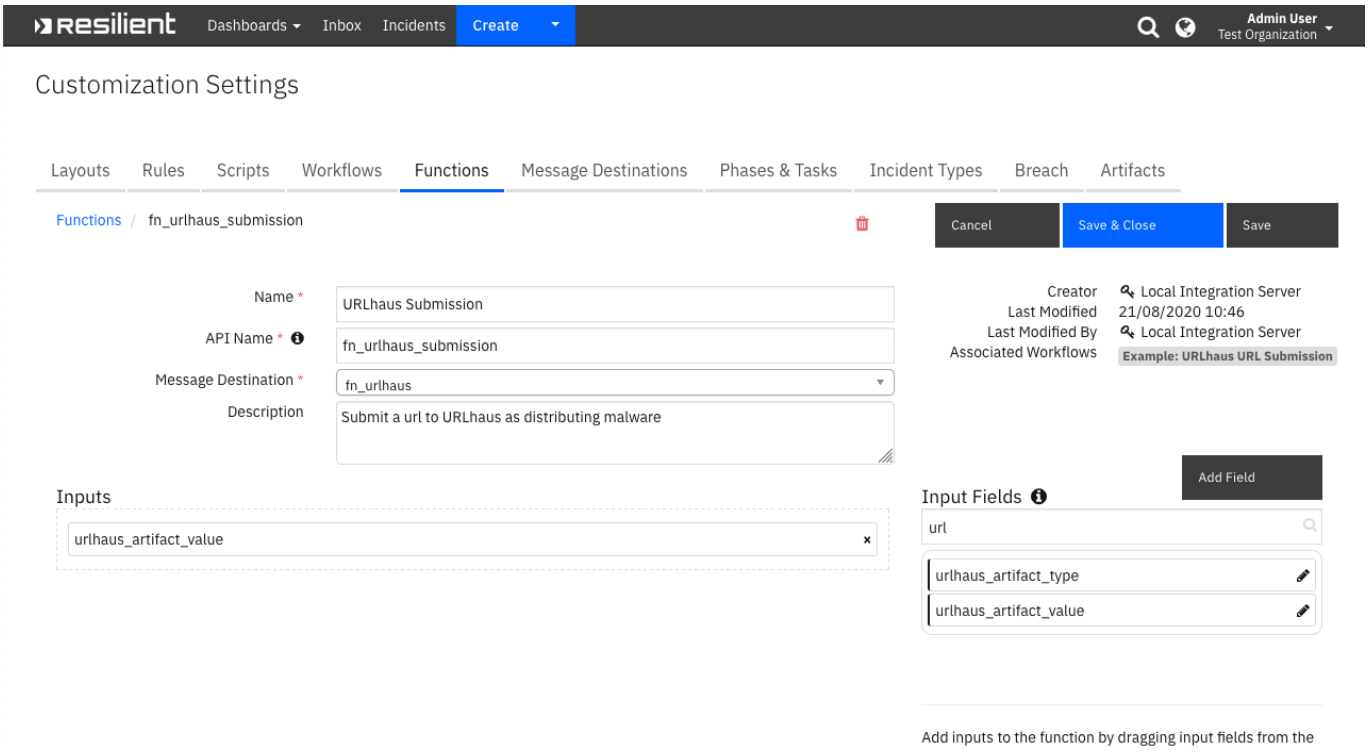
Table of Contents

Key Features

- Incorporates APIs available from URLhaus (https://urlhaus.abuse.ch/) to enrich data on:
  - urls
  - domains
  - IP Addresses
  - MD5 and SHA-256 Hash
  - tags (ex. Troldesh)
- Includes the capability to submit a url as distributing malware to a public searchable database.

Function - URLhaus Submission

Submit a url to URLhaus as distributing malware



▶ Inputs:

Name	Type	Required	Example	Tooltip
`urlhaus_artifact_value`	`text`	Yes	-	The value of the artifact

▶ Outputs:

```
results = {
    'version': '1.0',
    'success': True,
    'reason': None,
    'content': 'already_known: http://example.com\n',
    'raw': '"already_known: http://example.com\\n"',
    'inputs': {
        'urlhaus_artifact_value': 'http://example.com'
    },
    'metrics': {
        'version': '1.0',
        'package': 'fn-urlhaus',
        'package_version': '1.0.2',
        'host': 'example',
        'execution_time_ms': 174,
        'timestamp': '2020-08-24 17:04:19'
    }
}
```

▶ Workflows

▶ Example Pre-Process Script:

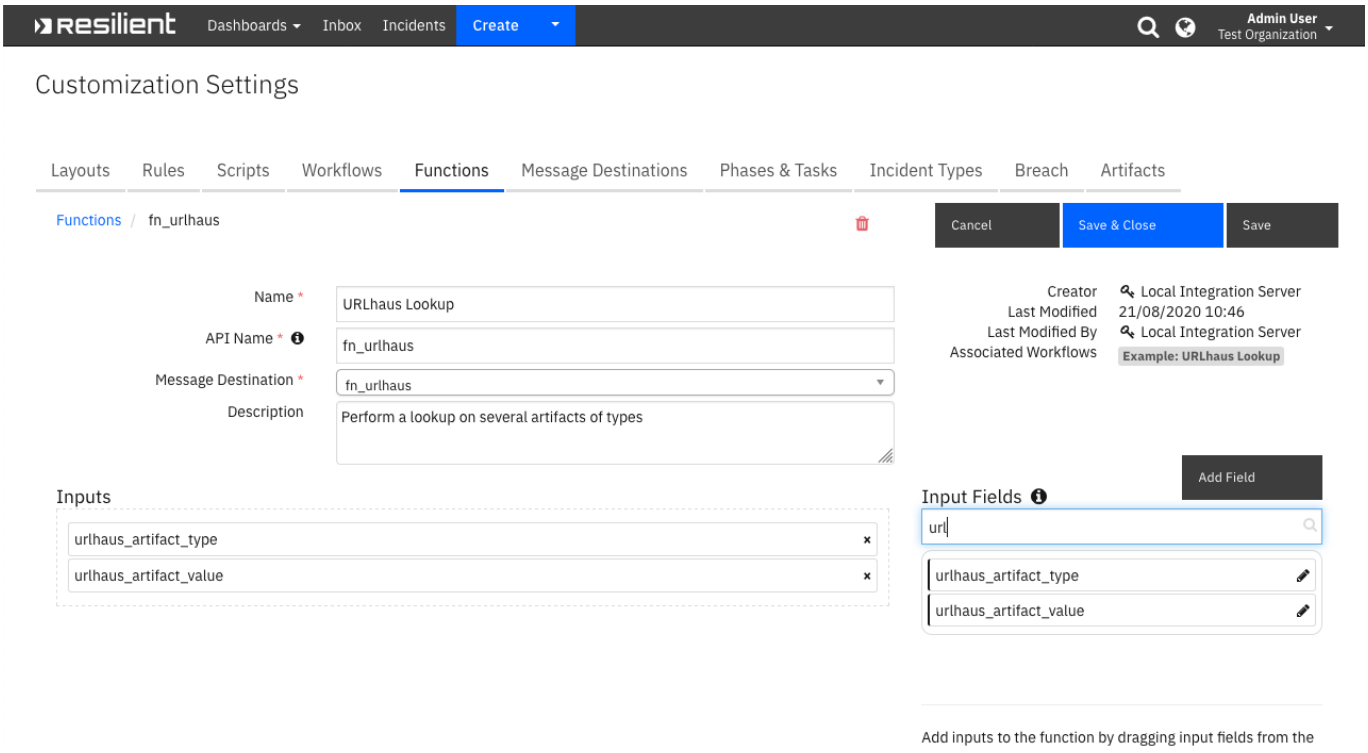
```
inputs.urlhaus_artifact_value = artifact.value
```

▶ Example Post-Process Script:

```
incident.addNote(u"Artifact {} submitted to
URLhaus\n{}".format(artifact.value, results.content))
```

Function - URLhaus Lookup

Perform a lookup on several artifacts of types



▶ Inputs:

Name	Type	Required	Example	Tooltip
urlhaus_artifact_type	text	Yes	–	The artifact's type
urlhaus_artifact_value	text	Yes	–	The value of the artifact

▶ Outputs:

```
results = {
    'version': '1.0',
    'success': True,
    'reason': None,
    'content': {
        'query_status': 'ok',
        'md5_hash': '51dcd062feb030e94da803b3fa0c0f8d',
        'sha256_hash':
'a13897aff5bbdee2bf78782be00ac516731e334463b3846c57df74c6167e97c8',
        'file_type': 'doc',
        'file_size': '186404',
        'signature': 'Heodo',
        'firstseen': '2020-08-21 13:04:06',
        'lastseen': None,
        'url_count': '30',
        'urlhaus_download': 'https://urlhaus-
api.abuse.ch/v1/download/a13897aff5bbdee2bf78782be00ac516731e334463b3846c5
7df74c6167e97c8/',
        'virustotal': None,
        'imphash': None,
        'ssdeep':
'3072:V4PrXcuQuvpzm4bkiaMQgAlSDyxS50XjwlxJ:iDRv1m4bnQgISDyxAYjwlxJ',
        'tlsh':
```

```
'6F040CDE30D9FC3EE74EA03A9C4AAE6E7212DF901EC8F1A910B4377D34B5390556A112',
        'urls': [{
            'url_id': '437782',
            'url': 'http://drshekharbiswas.com/cgi-
bin/report/4p38q98727977179317930u06alhqz405xm7q6yerm2a/',
            'url_status': 'offline',
            'urlhaus_reference': 'https://urlhaus.abuse.ch/url/437782/',
            'filename': 'INV_PO_08212020EX.doc',
            'firstseen': '2020-08-21',
            'lastseen': None
        }, {
            'url_id': '438159',
            'url': 'http://www.866qk.cn/f8a/invoice/',
            'url_status': 'online',
            'urlhaus_reference': 'https://urlhaus.abuse.ch/url/438159/',
            'filename': 'OIIN_ZFD_080120_YJN_082120.doc',
            'firstseen': '2020-08-21',
            'lastseen': None
        }, {
            'url_id': '436638',
            'url':
'https://rollofkati.com/temp/INC/lenbxnn059968010058223ah6ti7jimemv10i/',
            'url_status': 'offline',
            'urlhaus_reference': 'https://urlhaus.abuse.ch/url/436638/',
            'filename': 'DOC_PO_08212020EX.doc',
            'firstseen': '2020-08-21',
            'lastseen': None
        }]
    },
    'raw': '{"query_status": "ok", "md5_hash":
"51dcd062feb030e94da803b3fa0c0f8d", "sha256_hash":
"a13897aff5bbdee2bf78782be00ac516731e334463b3846c57df74c6167e97c8", ...}',
    'inputs': {
        'urlhaus_artifact_type': 'Malware MD5 Hash',
        'urlhaus_artifact_value': '51dcd062feb030e94da803b3fa0c0f8d'
    },
    'metrics': {
        'version': '1.0',
        'package': 'fn-urlhaus',
        'package_version': '1.0.2',
        'host': 'example',
        'execution_time_ms': 234,
        'timestamp': '2020-08-24 17:01:14'
    }
}
```


▶ Workflows

▶ Example Pre-Process Script:

```
inputs.urlhaus_artifact_type = artifact.type
inputs.urlhaus_artifact_value = artifact.value
```

▶ Example Post-Process Script:

```
def format_link(item):
if item and ("https://" in item or "http://" in item):
  return "<a target='blank' href='{0}'>{0}</a>".format(item)
else:
  return item

def expand_list(list_value, separator="<br>"):
if not isinstance(list_value, list):
  return format_link(list_value)
else:
  try:
    items = []
    for item in list_value:
      if isinstance(item, dict):
        items.append("<div style='padding:10px'>{}
</div>".format(walk_dict(item)))
      else:
        items.append(format_link(item))
    return separator.join(items)
  except:
      pass

def walk_dict(sub_dict):
notes = []
for key, value in sub_dict.items():
  if key not in ['display_content']:
    if isinstance(value, dict):
      notes.append(u"<b>{}</b>: <div style='padding:10px'>{}
</div>".format(key, walk_dict(value)))
    else:
      notes.append(u"<b>{}</b>: {}".format(key, expand_list(value)))
return u"<br>".join(notes)

note = u"<b>URLhaus look up for artifact:</b> {}<br>
<br>".format(artifact.value)

if results["success"]:
note = note + walk_dict(results["content"])
else:
note = note + u"This Artifact has no accessible registry information"

incident.addNote(helper.createRichText(note))
```

Rules

Rule Name	Object	Workflow Triggered
Example: URLhaus URL Submission	artifact	`example_urlhaus_url_submission`
Example: URLhaus Lookup	artifact	`example_urlhaus_lookup`