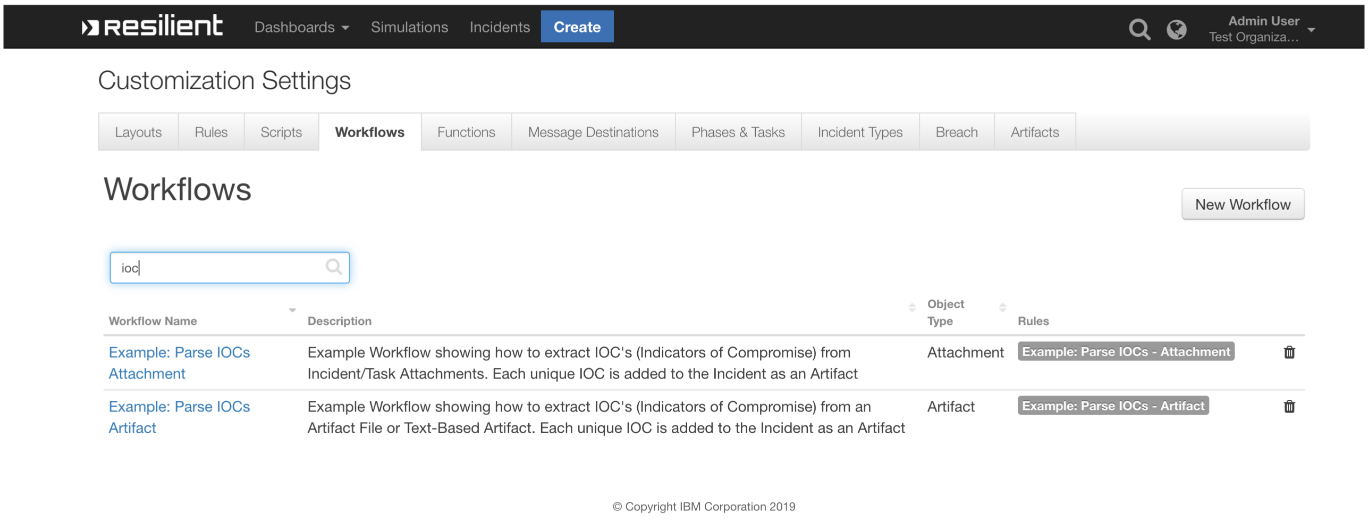
IOC Parser Functions for IBM Resilient

Overview

Resilient Function to extract Indicators of Compromise from Attachments + Artifact Files



Workflows

Workflow Name	Description	Object Type	Rules
Example: Parse IOCs Attachment	Example Workflow showing how to extract IOC's (Indicators of Compromise) from Incident/Task Attachments. Each unique IOC is added to the Incident as an Artifact	Attachment	Example: Parse IOCs - Attachment
Example: Parse IOCs Artifact	Example Workflow showing how to extract IOC's (Indicators of Compromise) from an Artifact File or Text-Based Artifact. Each unique IOC is added to the Incident as an Artifact	Artifact	Example: Parse IOCs - Artifact

© Copyright IBM Corporation 2019

Uses the IOCParser Python Library to extract IOCs from Resilient Attachments and Artifacts. All unique IOCs that are found are added to the Resilient Incident as an Artifact

Release Notes

v1.0.0

- Initial Release

v2.0.0

- Message Destination renamed from `iocpdest` to `fn_ioc_parser`
- Function renamed from `ioc_parser` to `function_ioc_parser`

- Removed Function Inputs: `incidentId`, `inputType` and `artifactId`
- Python FunctionComponent file renamed from `ioc_parser.py` to `function-ioc-parser.py`
- Added *Example* Rules and Workflows
- New Function Result:

  ```
  results = {
    "iocs": [{
      'count': 1,
      'type': 'IP',
      'value': '127.0.0.0'
    }, {
      'count': 1,
      'type': 'uri',
      'value': 'https://www.example.com'
    }],
    "attachment_file_name": u'test_file_name.pdf'
  }
  ```

Requirements

- IBM Resilient >= `v31.0.4254`
- An Integration Server running `resilient_circuits>=30.0.0`
  - To setup an Integration Server see: ibm.biz/res-int-server-guide

---

Installation

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

  ```
  $ unzip fn_ioc_parser-x.x.x.zip
  ```

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

  ```
  $ cd fn_ioc_parser-x.x.x
  ```

- **Install** the package:

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

- Import the fn_ioc_parser **customizations** into the Resilient Appliance:

```
$ resilient-circuits customize -y -l fn-ioc-parser
```

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

```
$ resilient-circuits selftest -l fn-ioc-parser
```

- **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-ioc-parser
```

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_ioc_parser	2.0.0	Resilient Labs	http://ibm.biz/resilientcommunity

User Guide: fn_ioc_parser_v2.0.0

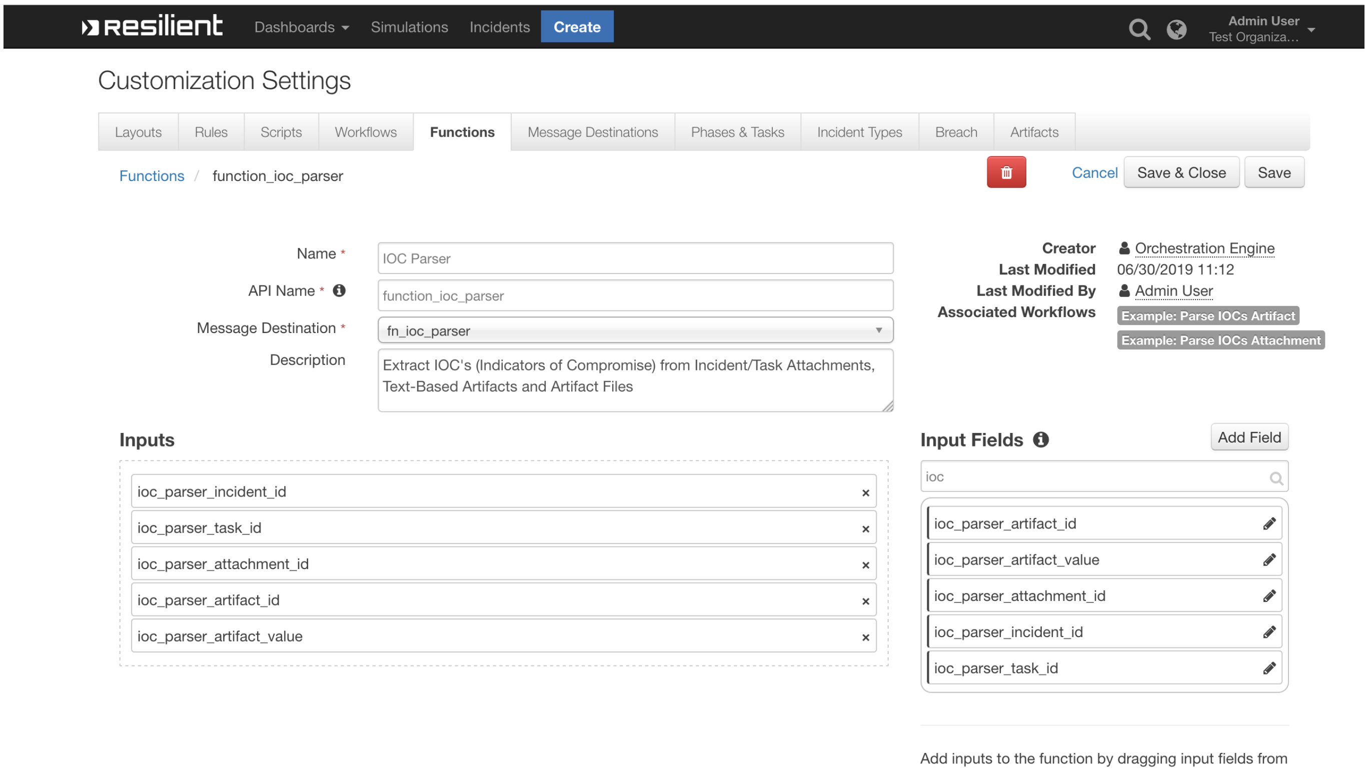
Table of Contents

Key Features

- Extract unique Indicators Of Compromise from PDF, docx, xls and more text based files
- Count duplicate IOCs and increment its `count`
- Add each IOC as an Artifact and update its Description with the IOC's `count`

Function - IOC Parser

Extract IOC's (Indicators of Compromise) from Incident/Task Attachments, Text-Based Artifacts and Artifact Files



Customization Settings

Functions / function_ioc_parser

Name * IOC Parser
API Name * function_ioc_parser
Message Destination * fn_ioc_parser
Description Extract IOC's (Indicators of Compromise) from Incident/Task Attachments, Text-Based Artifacts and Artifact Files

Creator Orchestration Engine
Last Modified 06/30/2019 11:12
Last Modified By Admin User
Associated Workflows Example: Parse IOCs Artifact, Example: Parse IOCs Attachment

Inputs
- ioc_parser_incident_id
- ioc_parser_task_id
- ioc_parser_attachment_id
- ioc_parser_artifact_id
- ioc_parser_artifact_value

Input Fields
ioc
- ioc_parser_artifact_id
- ioc_parser_artifact_value
- ioc_parser_attachment_id
- ioc_parser_incident_id
- ioc_parser_task_id

Add inputs to the function by dragging input fields from

▶ Inputs:

Name	Type	Required	Example	Tooltip
`ioc_parser_incident_id`	`number`	Yes	`-`	The ID of the Incident
`ioc_parser_task_id`	`number`	No	`100001`	The ID of the Task
`ioc_parser_attachment_id`	`number`	No	`123`	The ID of the Attachment

Name	Type	Required	Example	Tooltip
`ioc_parser_artifact_id`	`number`	No	`123`	The ID of the Artifact
`ioc_parser_artifact_value`	`text`	No	`-`	The Artifact's value

▶ Outputs:

```
results = {
  'iocs': [{
    'count': 1,
    'type': 'IP',
    'value': '127.0.0.0'
  }, {
    'count': 1,
    'type': 'uri',
    'value': 'https://www.example.com'
  }, {
    'count': 1,
    'type': 'uri',
    'value': 'example.com'
  }, {
    'count': 1,
    'type': 'md5',
    'value': '22sd233b26debdfb8c7cfbd3a55abbd'
  }, {
    'count': 1,
    'type': 'CVE',
    'value': 'CVE-4242-4242'
  }, {
    'count': 5,
    'type': 'email',
    'value': 'info@example.com'
  }],
  'attachment_file_name': u'test_indicators_of_compromise.pdf'
}
```

▶ Example Pre-Process Script:

```
# Define Pre-Process Inputs
inputs.ioc_parser_incident_id = incident.id
inputs.ioc_parser_artifact_id = artifact.id
inputs.ioc_parser_artifact_value = artifact.value
```

▶ Example Post-Process Script:

```
import re
```

```
def get_artifact_type(artifact_value, artifact_type):
  """Use some regex expressions to try and identify
  from the Artifact's value, what Artifact type it is.
  Return original artifact_type if we cannot figure it out"""

  dns_name_regex = re.compile(r'^(([a-zA-Z]{1})|([a-zA-Z]{1}[a-zA-Z]{1})|
([a-zA-Z]{1}[0-9]{1})|([0-9]{1}[a-zA-Z]{1})|([a-zA-Z0-9][a-zA-Z0-9-_]
{1,61}[a-zA-Z0-9]))\.([a-zA-Z]{2,6}|[a-zA-Z0-9-]{2,30}\.[a-zA-Z]{2,3})$')

  if re.match(dns_name_regex, artifact_value):
    return "DNS Name"

  return artifact_type

# Map ioc.type to Resilient Artifact Type
ioc_type_to_artifact_type_map = {
    'uri': 'URI Path',
    'IP': 'IP Address',
    'md5': 'Malware MD5 Hash',
    'sha1': 'Malware SHA-1 Hash',
    'sha256': 'Malware SHA-256 Hash',
    'CVE': 'Threat CVE ID',
    'email': 'Email Sender',
    'filename': 'File Name',
    'file': 'File Name'
}

# Get the IOCs
iocs = results.iocs

if iocs:
    # Loop IOCs and add each on as an Artifact
    for ioc in iocs:

      # If attachment_file_name is not defined, use the ioc.value as in
the Artifact's Description
      if results.attachment_file_name:
        artifact_description = u"This IOC occurred {0} time(s) in the
artifact: {1}".format( unicode(ioc.count),
unicode(results.attachment_file_name) )

      else:
        artifact_description = u"This IOC occurred {0} time(s) in the
artifact: {1}".format( unicode(ioc.count), unicode(ioc.value) )

      artifact_value = ioc.value
      artifact_type = ioc_type_to_artifact_type_map.get(ioc.type,
"String")

      # If the artifact_type is 'URI Path', call get_artifact_type to try
intentify the type using regex
      if artifact_type == "URI Path":
        artifact_type = get_artifact_type(artifact_value, artifact_type)
      incident.addArtifact(artifact_type, artifact_value,
artifact_description)
```

Rules

Rule Name	Object	Workflow Triggered
Example: Parse IOCs - Artifact	artifact	`example_parse_iocs_artifact`
Example: Parse IOCs - Attachment	attachment	`example_parse_iocs_attachment`