IBM Security SOAR Functions for Cisco ASA

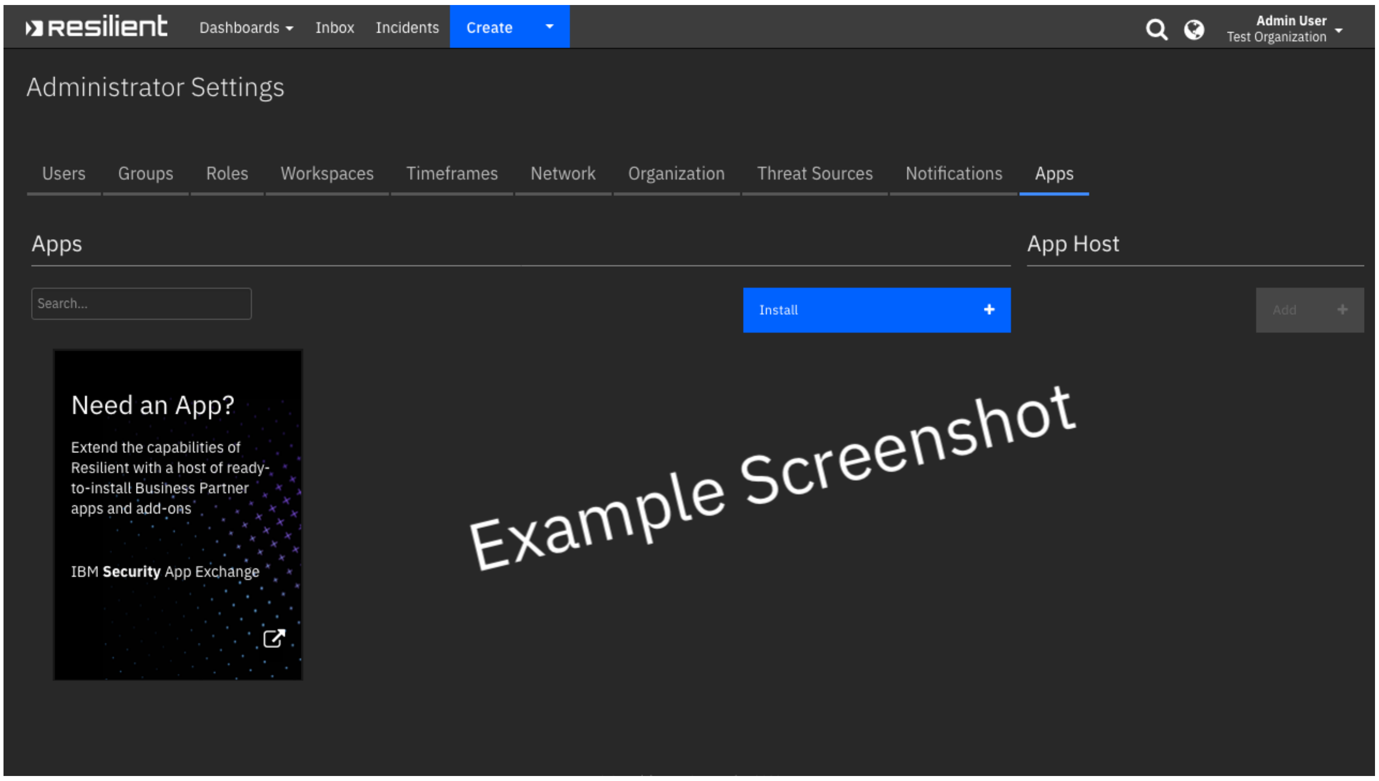
Table of Contents

Release Notes

Version	Date	Notes
1.0.0	04/4021	Initial Release

Overview

Resilient Circuits Components for 'fn_cisco_asa'



Resilient Circuits Components for 'fn_cisco_asa'

Key Features

- Key Feature 1
- Key Feature 2
- Key Feature 3

Requirements

This app supports the IBM Resilient SOAR Platform and the IBM Cloud Pak for Security.

Resilient platform

The Resilient platform supports two app deployment mechanisms, App Host and integration server.

If deploying to a Resilient platform with an App Host, the requirements are:

- Resilient platform >= `37.0.5832`.
- The app is in a container-based format (available from the AppExchange as a `zip` file).

If deploying to a Resilient platform with an integration server, the requirements are:

- Resilient platform >= `37.0.5832`.
- The app is in the older integration format (available from the AppExchange as a `zip` file which contains a `tar.gz` file).
- Integration server is running `None`.
- If using an API key account, make sure the account provides the following minimum permissions:

Name	Permissions
Org Data	Read
Function	Read

The following Resilient platform guides provide additional information:

- *App Host Deployment Guide*: provides installation, configuration, and troubleshooting information, including proxy server settings.
- *Integration Server Guide*: provides installation, configuration, and troubleshooting information, including proxy server settings.
- *System Administrator Guide*: provides the procedure to install, configure and deploy apps.

The above guides are available on the IBM Knowledge Center at ibm.biz/resilient-docs. On this web page, select your Resilient platform version. On the follow-on page, you can find the *App Host Deployment Guide* or *Integration Server Guide* by expanding **Resilient Apps** in the Table of Contents pane. The System Administrator Guide is available by expanding **System Administrator**.

Cloud Pak for Security

If you are deploying to IBM Cloud Pak for Security, the requirements are:

- IBM Cloud Pak for Security >= 1.4.
- Cloud Pak is configured with an App Host.
- The app is in a container-based format (available from the AppExchange as a `zip` file).

The following Cloud Pak guides provide additional information:

- *App Host Deployment Guide*: provides installation, configuration, and troubleshooting information, including proxy server settings. From the Table of Contents, select Case Management and Orchestration & Automation > **Orchestration and Automation Apps**.
- *System Administrator Guide*: provides information to install, configure, and deploy apps. From the IBM Cloud Pak for Security Knowledge Center table of contents, select Case Management and Orchestration & Automation > **System administrator**.

These guides are available on the IBM Knowledge Center at ibm.biz/cp4s-docs. From this web page, select your IBM Cloud Pak for Security version. From the version-specific Knowledge Center page, select Case Management and Orchestration & Automation.

Proxy Server

The app **does/does not** support a proxy server.

Installation

Install

- To install or uninstall an App or Integration on the *Resilient platform*, see the documentation at ibm.biz/resilient-docs.
- To install or uninstall an App on *IBM Cloud Pak for Security*, see the documentation at ibm.biz/cp4s-docs and follow the instructions above to navigate to Orchestration and Automation.

App Configuration

The following table provides the settings you need to configure the app. These settings are made in the app.config file. See the documentation discussed in the Requirements section for the procedure.

Config	Required	Example	Description
host	Yes	`<asa_ip>`	*Enter a description of the config here.*
username	Yes	`<asa_username>`	*Enter a description of the config here.*
password	Yes	`<asa_password>`	*Enter a description of the config here.*
network_object_lists	Yes	`BLACKLIST_IN, BLACKLIST_OUT`	*Enter a description of the config here.*

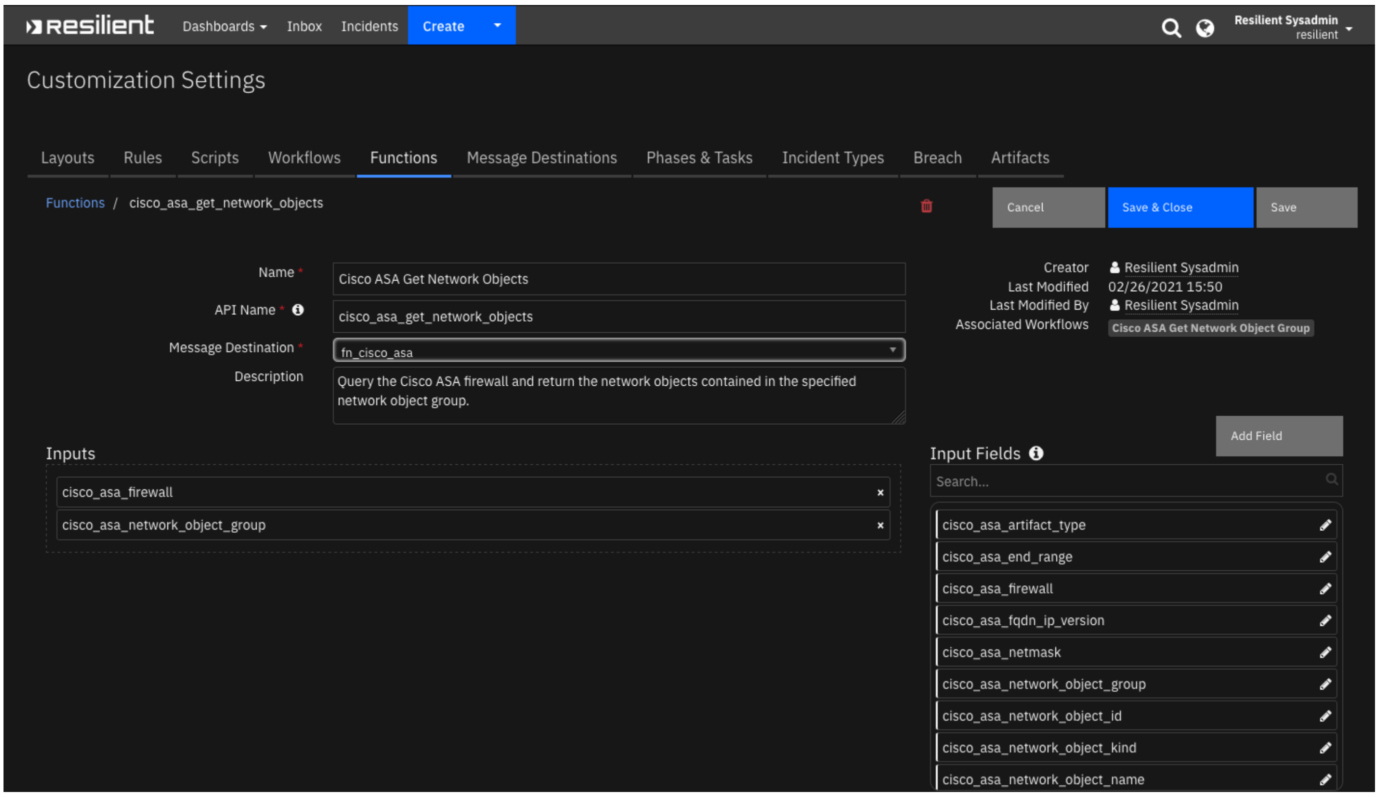
Custom Layouts

- Import the Data Tables and Custom Fields like the screenshot below:

  screenshot: custom_layouts

Function - Cisco ASA Get Network Objects

Query the Cisco ASA firewall and return the network objects contained in the specified network object group.



Customization Settings

Functions / cisco_asa_get_network_objects

Name *	Cisco ASA Get Network Objects
API Name *	cisco_asa_get_network_objects
Message Destination *	fn_cisco_asa
Description	Query the Cisco ASA firewall and return the network objects contained in the specified network object group.

Creator	Resilient Sysadmin
Last Modified	02/26/2021 15:50
Last Modified By	Resilient Sysadmin
Associated Workflows	Cisco ASA Get Network Object Group

Inputs

- cisco_asa_firewall
- cisco_asa_network_object_group

Input Fields

- cisco_asa_artifact_type
- cisco_asa_end_range
- cisco_asa_firewall
- cisco_asa_fqdn_ip_version
- cisco_asa_netmask
- cisco_asa_network_object_group
- cisco_asa_network_object_id
- cisco_asa_network_object_kind
- cisco_asa_network_object_name

▶ Inputs:

Name	Type	Required	Example	Tooltip
`cisco_asa_firewall`	`text`	Yes	`-`	-
`cisco_asa_network_object_group`	`text`	Yes	`-`	-

▶ Outputs:

```
results = {
    # TODO: Copy and paste an example of the Function Output within this code block.
    # To view the output of a Function, run resilient-circuits in DEBUG mode and invoke the Function.
    # The Function results will be printed in the logs: "resilient-circuits run --loglevel=DEBUG"
}
```

▶ Example Pre-Process Script:

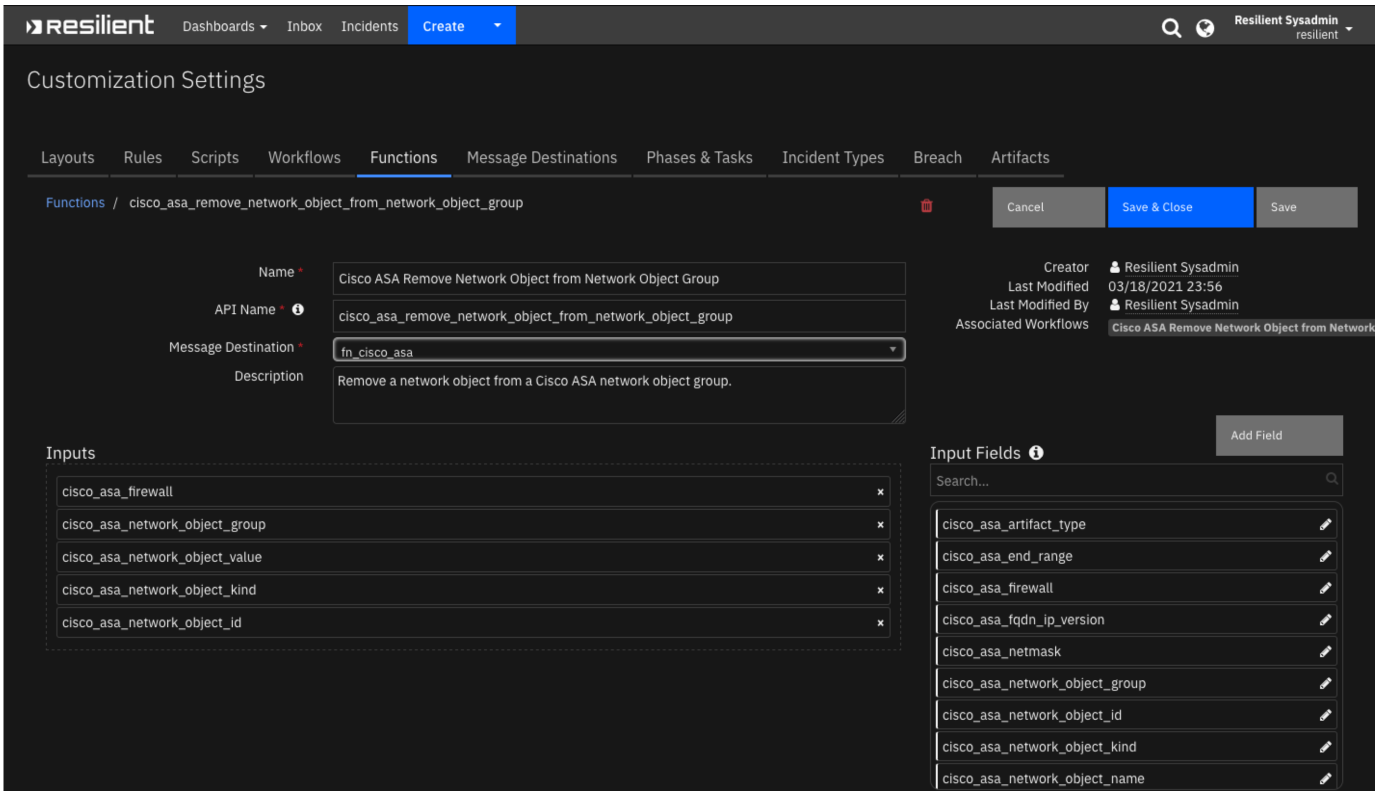
```python
override = rule.properties.cisco_asa_firewall_network_object_group_pair_overide
if override is "" or override is None:
  firewall_group_pair = rule.properties.cisco_asa_firewall_network_object_group_pair
else:
  firewall_group_pair = override

# Parse the firewall group pair, which is a string in "firewall:network_object_group" format
firewall_group_pair_list = firewall_group_pair.split(":")
inputs.cisco_asa_firewall = firewall_group_pair_list[0]
inputs.cisco_asa_network_object_group = firewall_group_pair_list[1]
```

▶ Example Post-Process Script:

```
from java.util import Date

content = results.get("content")
member_list = content.get("member_list")
firewall = results.inputs.get("cisco_asa_firewall")
network_object_group = results.inputs.get("cisco_asa_network_object_group")

# Add each email as a row in the query results data table
for network_object in member_list:
  network_object_row = incident.addRow("cisco_asa_network_object_dt")
  network_object_row.cisco_asa_query_date = Date()
  network_object_row.cisco_asa_firewall = firewall
  network_object_row.cisco_asa_network_object_group = network_object_group

  if network_object.get("kind")  == 'object#NetworkObj':
    network_object_row.cisco_asa_network_object_id = network_object.get("objectId")
    host = network_object.get("host")
    network_object_row.cisco_asa_network_object_kind = host.get("kind")
    network_object_row.cisco_asa_network_object_value = host.get("value")
  else:
    network_object_row.cisco_asa_network_object_kind = network_object.get("kind")
    network_object_row.cisco_asa_network_object_value = network_object.get("value")

  status_text = u"""<p style= "color:{color}">{status}</p>""".format(color="green", status="Active")
  network_object_row.cisco_asa_status = helper.createRichText(status_text)
```

Function - Cisco ASA Remove Network Object from Network Object Group

Remove a network object from a Cisco ASA network object group.



Customization Settings

Functions / cisco_asa_remove_network_object_from_network_object_group

Name *	Cisco ASA Remove Network Object from Network Object Group
API Name *	cisco_asa_remove_network_object_from_network_object_group
Message Destination *	fn_cisco_asa
Description	Remove a network object from a Cisco ASA network object group.

Creator	Resilient Sysadmin
Last Modified	03/18/2021 23:56
Last Modified By	Resilient Sysadmin
Associated Workflows	Cisco ASA Remove Network Object from Network

Inputs

- cisco_asa_firewall
- cisco_asa_network_object_group
- cisco_asa_network_object_value
- cisco_asa_network_object_kind
- cisco_asa_network_object_id

Input Fields

- cisco_asa_artifact_type
- cisco_asa_end_range
- cisco_asa_firewall
- cisco_asa_fqdn_ip_version
- cisco_asa_netmask
- cisco_asa_network_object_group
- cisco_asa_network_object_id
- cisco_asa_network_object_kind
- cisco_asa_network_object_name

▶ Inputs:

Name	Type	Required	Example	Tooltip
`cisco_asa_firewall`	`text`	Yes	`-`	-
`cisco_asa_network_object_group`	`text`	Yes	`-`	-
`cisco_asa_network_object_id`	`text`	No	`-`	-
`cisco_asa_network_object_kind`	`text`	Yes	`-`	-
`cisco_asa_network_object_value`	`text`	Yes	`-`	-

▶ Outputs:

```
results = {
    # TODO: Copy and paste an example of the Function Output within this code block.
    # To view the output of a Function, run resilient-circuits in DEBUG mode and invoke
the Function.
    # The Function results will be printed in the logs: "resilient-circuits run --
loglevel=DEBUG"
}
```

▶ Example Pre-Process Script:

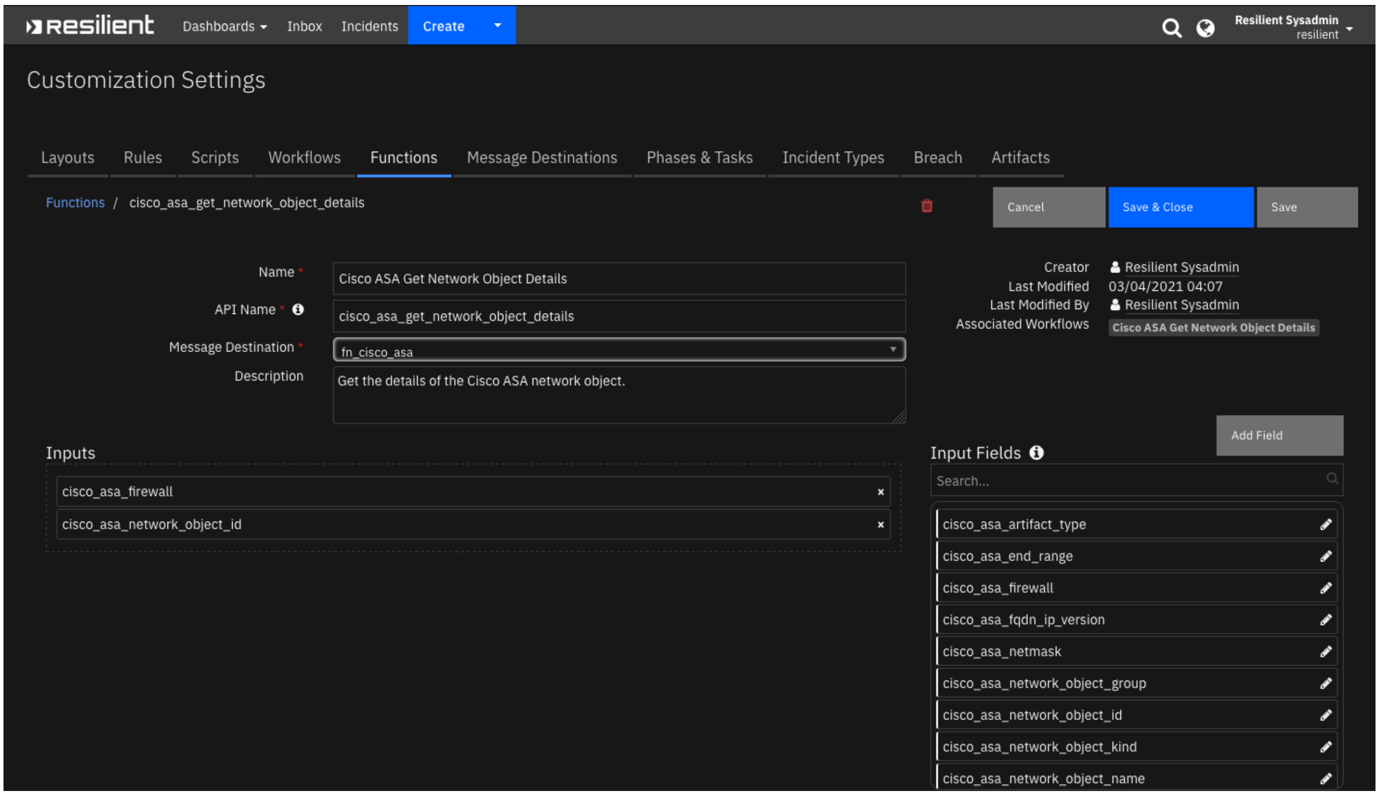
```
inputs.cisco_asa_firewall = row.cisco_asa_firewall
inputs.cisco_asa_network_object_group = row.cisco_asa_network_object_group
inputs.cisco_asa_network_object_kind = row.cisco_asa_network_object_kind
inputs.cisco_asa_network_object_value = row.cisco_asa_network_object_value
inputs.cisco_asa_network_object_id = row.cisco_asa_network_object_id
```

▶ Example Post-Process Script:

```
from java.util import Date

if results.success:
  text = "Removed"
else:
  text = "NotFound"

status_text = u"""<p style= "color:{color}">{status}</p>""".format(color="red",
status=text)
row['cisco_asa_status'] = helper.createRichText(status_text)
row["cisco_asa_query_date"] = Date()
```

Function - Cisco ASA Get Network Object Details

Get the details of the Cisco ASA network object.



Functions / cisco_asa_get_network_object_details

Name * Cisco ASA Get Network Object Details
API Name * cisco_asa_get_network_object_details
Message Destination * fn_cisco_asa
Description Get the details of the Cisco ASA network object.

Creator: Resilient Sysadmin
Last Modified: 03/04/2021 04:07
Last Modified By: Resilient Sysadmin
Associated Workflows: Cisco ASA Get Network Object Details

Inputs
- cisco_asa_firewall
- cisco_asa_network_object_id

Input Fields
- cisco_asa_artifact_type
- cisco_asa_end_range
- cisco_asa_firewall
- cisco_asa_fqdn_ip_version
- cisco_asa_netmask
- cisco_asa_network_object_group
- cisco_asa_network_object_id
- cisco_asa_network_object_kind
- cisco_asa_network_object_name

▶ Inputs:

Name	Type	Required	Example	Tooltip
`cisco_asa_firewall`	`text`	Yes	`-`	-
`cisco_asa_network_object_id`	`text`	No	`-`	-

▶ Outputs:

```
results = {
    # TODO: Copy and paste an example of the Function Output within this code block.
    # To view the output of a Function, run resilient-circuits in DEBUG mode and invoke
the Function.
    # The Function results will be printed in the logs: "resilient-circuits run --
loglevel=DEBUG"
}
```

▶ Example Pre-Process Script:

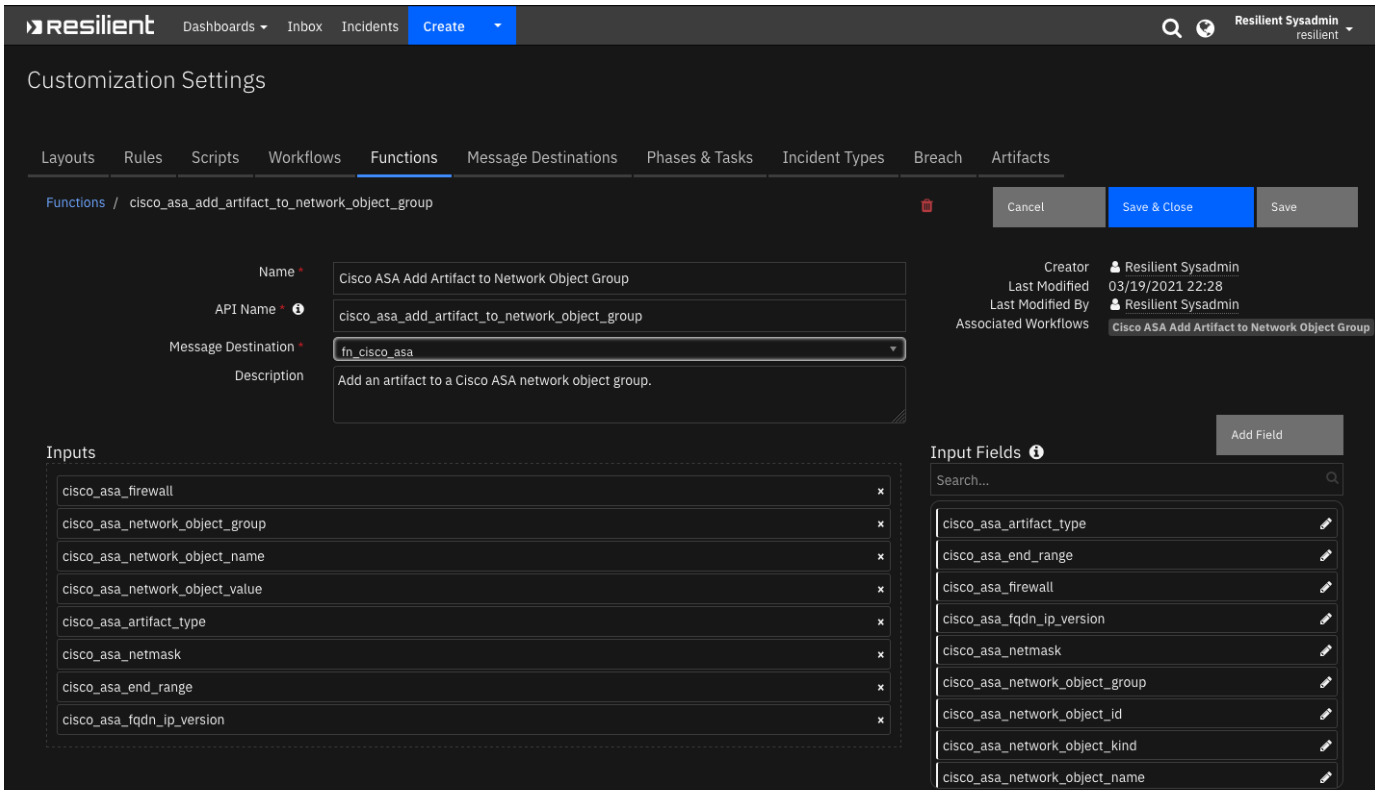
```
inputs.cisco_asa_firewall = row.cisco_asa_firewall
inputs.cisco_asa_network_object_id = row.cisco_asa_network_object_id
```

▶ Example Post-Process Script:

```
# Put the results json into a workflow property so we can call the
# convert_json_to_rich_text script to print readable formatted json in an incident note.
inputs = results.get("inputs")
firewall_id = inputs.get("cisco_asa_firewall")
object_id = inputs.get("cisco_asa_network_object_id")
header = u"Cisco ASA Firewall: {0} Network Object ID {1}".format(firewall_id, object_id)

json_note = {
              "version": "1.1",
              "header": header,
              "json": results.content,
              "sort": False
            }

workflow.addProperty('convert_json_to_rich_text', json_note)
```

Function - Cisco ASA Add Artifact to Network Object Group

Add an artifact to a Cisco ASA network object group.



Functions / cisco_asa_add_artifact_to_network_object_group

Name *	Cisco ASA Add Artifact to Network Object Group
API Name *	cisco_asa_add_artifact_to_network_object_group
Message Destination *	fn_cisco_asa
Description	Add an artifact to a Cisco ASA network object group.

Creator	Resilient Sysadmin
Last Modified	03/19/2021 22:28
Last Modified By	Resilient Sysadmin
Associated Workflows	Cisco ASA Add Artifact to Network Object Group

Inputs

- cisco_asa_firewall
- cisco_asa_network_object_group
- cisco_asa_network_object_name
- cisco_asa_network_object_value
- cisco_asa_artifact_type
- cisco_asa_netmask
- cisco_asa_end_range
- cisco_asa_fqdn_ip_version

Input Fields

Search...

- cisco_asa_artifact_type
- cisco_asa_end_range
- cisco_asa_firewall
- cisco_asa_fqdn_ip_version
- cisco_asa_netmask
- cisco_asa_network_object_group
- cisco_asa_network_object_id
- cisco_asa_network_object_kind
- cisco_asa_network_object_name

▶ Inputs:

Name	Type	Required	Example	Tooltip

Name	Type	Required	Example	Tooltip
`cisco_asa_artifact_type`	`text`	Yes	`-`	-
`cisco_asa_end_range`	`text`	No	`-`	-
`cisco_asa_firewall`	`text`	Yes	`-`	-
`cisco_asa_fqdn_ip_version`	`select`	No	`-`	-
`cisco_asa_netmask`	`text`	No	`-`	-
`cisco_asa_network_object_group`	`text`	Yes	`-`	-
`cisco_asa_network_object_name`	`text`	No	`-`	-
`cisco_asa_network_object_value`	`text`	Yes	`-`	-

▶ Outputs:

```
results = {
    # TODO: Copy and paste an example of the Function Output within this code block.
    # To view the output of a Function, run resilient-circuits in DEBUG mode and invoke
the Function.
    # The Function results will be printed in the logs: "resilient-circuits run --
loglevel=DEBUG"
}
```

▶ Example Pre-Process Script:

```
# Parse the firewall name and network object group from the colon separated string
# Or get the string from the text edit box if the use overrides the select list.
override = rule.properties.cisco_asa_firewall_network_object_group_pair_overide
if override is "" or override is None:
  firewall_group_pair = rule.properties.cisco_asa_firewall_network_object_group_pair
else:
  firewall_group_pair = override

# Parse the firewall group pair, which is a string in "firewall:network_object_group"
format
firewall_group_pair_list = firewall_group_pair.split(":")
inputs.cisco_asa_firewall = firewall_group_pair_list[0]
inputs.cisco_asa_network_object_group = firewall_group_pair_list[1]

# Get input from the artifact type and value
inputs.cisco_asa_network_object_value = artifact.value
inputs.cisco_asa_artifact_type = artifact.type

# Option params for IP netmask or end IP for IP range
inputs.cisco_asa_end_range = rule.properties.cisco_asa_end_range
if rule.properties.cisco_asa_ipv4_netmask:
  inputs.cisco_asa_netmask = rule.properties.cisco_asa_ipv4_netmask
elif rule.properties.cisco_asa_ipv6_prefix_length:
  inputs.cisco_asa_netmask = rule.properties.cisco_asa_ipv6_prefix_length

# FQDN version
if rule.properties.cisco_asa_fqdn_ip_version:
  inputs.cisco_asa_fqdn_ip_version = rule.properties.cisco_asa_fqdn_ip_version

# IPv4FQDN and IPv4Range require a name as input.
```

```
if rule.properties.cisco_asa_network_object_name_required:
  inputs.cisco_asa_network_object_name =
rule.properties.cisco_asa_network_object_name_required
else:
  inputs.cisco_asa_network_object_name = rule.properties.cisco_asa_network_object_name
```

▶ Example Post-Process Script:

```
from java.util import Date

if results.success:

  content = results.get("content")
  firewall = content.get("firewall")
  network_object_group = content.get("network_object_group")
  network_object_kind = content.get("network_object_kind")
  network_object_value = content.get("network_object_value")
  network_object_name = content.get("network_object_name")

  # Add each email as a row in the query results data table
  network_object_row = incident.addRow("cisco_asa_network_object_dt")
  network_object_row.cisco_asa_query_date = Date()
  network_object_row.cisco_asa_firewall = firewall
  network_object_row.cisco_asa_network_object_group = network_object_group
  network_object_row.cisco_asa_network_object_kind = network_object_kind
  network_object_row.cisco_asa_network_object_value = network_object_value
  network_object_row.cisco_asa_network_object_id = network_object_name

  # Update status field
  status_text = u"""<p style= "color:{color}">{status}</p>""".format(color="green",
status="Active")
  network_object_row.cisco_asa_status = helper.createRichText(status_text)
```

Script - Convert JSON to rich text v1.1

This script converts a json object into a hierarchical display of rich text and adds the rich text to an incident's rich text (custom) field or an incident note. A workflow property is used to share the json to convert and identify parameters used on how to perform the conversion.

Typically, a function will create the workflow property 'convert_json_to_rich_text', and this script will run after that function to perform the conversion.

Features:

- Display the hierarchical nature of json, presenting the json keys (sorted if specified) as bold labels
- Provide links to found URLs
- Create either an incident note or add results to an incident (custom) rich text field.

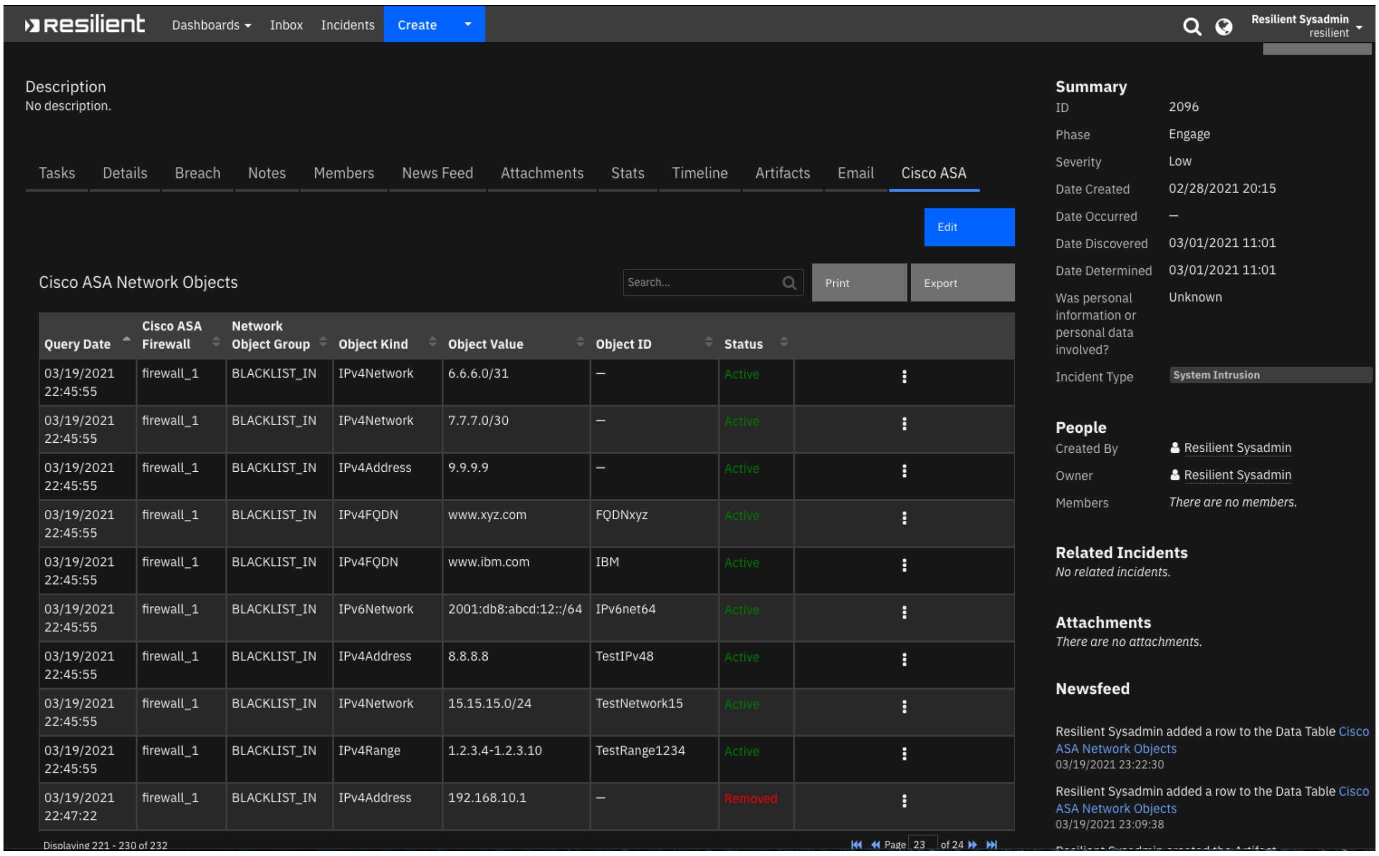
Object: incident

▶ Script Text:

```
# (c) Copyright IBM Corp. 2010, 2020. All Rights Reserved.
VERSION = 1.1
"""
```

```
  This script converts a json object into a hierarchical display of rich text and adds
the rich text to an incident's rich text (custom) field or an incident note.
  A workflow property is used to define the json to convert and identify parameters used
on how to perform the conversion.
  Typically, a function will create workflow property and this script will run after
that function to perform the conversion.
  Features:
    * Display the hierarchical nature of json, presenting the json keys as bold labels
    * Provide links to found URLs
    * Create either an incident note or add results to an incident (custom) rich text
field.

  In order to use this script, define a workflow property called:
convert_json_to_rich_text, to define the json and parameters to use for the conversion.
  Workflow properties can be added using a command similar to this:
  workflow.addProperty('convert_json_to_rich_text', {
    "version": 1.1,
    "header": "Artifact scan results for: {}".format(artifact.value),
    "padding": 10,
    "separator": u"<br />",
    "sort": True,
    "json": results.content,
    "json_omit_list": ["omit"],
    "incident_field": None
  })

  Format of workflow.property.convert_json_to_rich_text:
  {
    "version": 1.1, [this is for future compatibility]
    "header": str, [header line to add to converted json produced or None. Ex: Results
from scanning artifact: xxx. The header may contain rich text tags]
    "padding": 10, [padding for nested json elements, or defaults to 10]
    "separator": u"<br />"|list such as ['<span>','</span>'], [html separator between
json keys and lists or defaults to html break: '<br />'.
                                                If a list, then the data is brackets by
the pair specified]
    "sort": True|False, [sort the json keys at each level when displayed]
    "json": json, [required json to convert]
    "json_omit_list": [list of json keys to exclude or None]
    "incident_field": "<incident_field>" [indicates a builtin rich text incident field,
such as 'description'
                                          or a custom rich text field in the format:
'properties.<field>'. default: create an incident note]
  }
"""

import re

# needed for python 3
try:
    unicode("abc")
except:
    unicode = str


rc = re.compile(r'http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+#\?]|[!*\(\),]|(?:%[0-9a-fA-F][0-
9a-fA-F]))+')

class ConvertJson:
    """Class to hold the conversion parameters and perform the conversion"""

    def __init__(self, omit_keys=[], padding=10, separator=u"<br />", sort_keys=False):
        self.omit_keys = omit_keys
        self.padding = padding
        self.separator = separator
        self.sort_keys = sort_keys


    def format_link(self, item):
        """[summary]
          Find embedded urls (http(s)) and add html anchor tags to display as links
          Args:
              item ([string])

          Returns:
              [str]: None|original text if no links|text with html links
        """
        formatted_item = item
        if item and not isinstance(item, (int, bool, float)):
            list = rc.findall(item)
            if list:
                for link in list:
                    formatted_item = formatted_item.replace(link, u"<a target='blank'
href='{0}'>{0}</a>".format(link))

        return formatted_item

    def expand_list(self, list_value, is_list=False):
        """[summary]
          convert items to html, adding indents to nested dictionaries.
          Args:
              list_value ([dict|list]): json element

          Returns:
              [str]: html converted code
        """
        if not isinstance(list_value, list):
            return self.format_link(list_value)
        elif not list_value:
            return u"None<br>"

        try:
            items_list = []  # this will ensure list starts on second line of key label
            for item in list_value:
                if isinstance(item, dict):
                    result = self.convert_json_to_rich_text(item)
                    if is_list:
                        items_list.append(u"<li>{}</li>".format(result))
                    else:
                        items_list.append(result)
                elif isinstance(item, list):
                    items_list.append(self.expand_list(item, is_list=True))
                elif is_list:
                    items_list.append(u"<li>{}
</li>".format(self.format_link(unicode(item))))
                else:
                    items_list.append(self.format_link(unicode(item)))

            expand_list_result = self.add_separator(self.separator if not is_list else
u"",
                                                    items_list,
                                                    is_list=is_list)

            if is_list:
                return u"<ul>{}</ul>".format(expand_list_result)
            else:
                return u"<div style='padding:5px'>{}</div>".format(expand_list_result)
        except Exception as err:
            return str(err)

    def convert_json_to_rich_text(self, sub_dict):
        """[summary]
          Walk dictionary tree and convert to html for better display
          Args:
              sub_dict ([type]): [description]

          Returns:
              [type]: [description]
        """
        notes = []
        if sub_dict:
            if isinstance(sub_dict, list):
                expanded_list = self.expand_list(sub_dict, is_list=True)
                notes.append(self.add_separator(self.separator, expanded_list))
            else:
                keys = sorted (sub_dict.keys()) if self.sort_keys else sub_dict.keys()

                for key in keys:
                    if key not in self.omit_keys:
                        value = sub_dict[key]
                        is_list = isinstance(value, list)
                        item_list = [u"<strong>{0}</strong>: ".format(key)]
                        if isinstance(value, dict):
                            convert_result = self.convert_json_to_rich_text(value)
                            if convert_result:
                                item_list.append(u"<div style='padding:{}px'>{}
</div>".format(self.padding, convert_result))
                            else:
                                item_list.append(u"None<br>")
                        else:
                            item_list.append(self.expand_list(value, is_list=is_list))
                        notes.append(self.add_separator(self.separator,
u"".join(unicode(v) for v in item_list), is_list=is_list))

        result_notes = u"".join(notes)
        if isinstance(self.separator, list):
            return result_notes
        else:
            return result_notes.replace(
                u"</div>{0}".format(self.separator), u"</div>").replace(
                u"{0}</div>".format(self.separator), u"</div>"
            )  # tighten up result

    def add_separator(self, separator, items, is_list=False):
        """
        apply the separator to the data
        :param separator: None, str or list such as ['<span>', '</span>']
        :param items: str or list to add separator
        :return: text with separator applied
        """
        _items = items

        if not _items:
            return "<br>"

        if not isinstance(_items, list):
            _items = [_items]

        if isinstance(separator, list):
            return u"".join([u"{}{}{}".format(separator[0], item, separator[1]) for item
in _items])

        return u"{}{}".format(separator.join(_items), separator if not is_list else u"")

def get_properties(property_name):
    """
    Logic to collect the json and parameters from a workflow property.
    Args:
      property_name: workflow property to reference
    Returns:
      padding, separator, header, json_omit_list, incident_field, json, sort_keys
    """
    if not workflow.properties.get(property_name):
        helper.fail("workflow.properties.{} undefined".format(property_name))

    padding = int(workflow.properties[property_name].get("padding", 10))
    separator = workflow.properties[property_name].get("separator", u"<br />")
    if isinstance(separator, list) and len(separator) != 2:
        helper.fail("list of separators should be specified as a pair such as ['<div>',
'</div>']: {}".format(separator))

    header = workflow.properties[property_name].get("header")
    json_omit_list = workflow.properties[property_name].get("json_omit_list")
    if not json_omit_list:
        json_omit_list = []
    incident_field = workflow.properties[property_name].get("incident_field")
    json = workflow.properties[property_name].get("json", {})
    if not isinstance(json, dict) and not isinstance(json, list):
        helper.fail("json element is not formatted correctly: {}".format(json))
    sort_keys = bool(workflow.properties[property_name].get("sort", False))

    return padding, separator, header, json_omit_list, incident_field, json, sort_keys


## S T A R T
if 'workflow' in globals():
    padding, separator, header, json_omit_list, incident_field, json, sort_keys =
get_properties('convert_json_to_rich_text')

    if header:
        if isinstance(separator, list):
            hdr = u"{0}{1}{2}".format(separator[0], header, separator[1])
        else:
            hdr = u"{0}{1}".format(header, separator)
    else:
        hdr = u""

    convert = ConvertJson(omit_keys=json_omit_list, padding=padding,
separator=separator, sort_keys=sort_keys)
    converted_json = convert.convert_json_to_rich_text(json)
    result = u"{}{}".format(hdr, converted_json if converted_json else "\nNone")

    rich_text_note = helper.createRichText(result)
    if incident_field:
        incident[incident_field] = rich_text_note
    else:
        incident.addNote(rich_text_note)
```

Data Table - Cisco ASA Network Objects



Description
No description.

Cisco ASA Network Objects

Query Date	Cisco ASA Firewall	Network Object Group	Object Kind	Object Value	Object ID	Status
03/19/2021 22:45:55	firewall_1	BLACKLIST_IN	IPv4Network	6.6.6.0/31	–	Active
03/19/2021 22:45:55	firewall_1	BLACKLIST_IN	IPv4Network	7.7.7.0/30	–	Active
03/19/2021 22:45:55	firewall_1	BLACKLIST_IN	IPv4Address	9.9.9.9	–	Active
03/19/2021 22:45:55	firewall_1	BLACKLIST_IN	IPv4FQDN	www.xyz.com	FQDNxyz	Active
03/19/2021 22:45:55	firewall_1	BLACKLIST_IN	IPv4FQDN	www.ibm.com	IBM	Active
03/19/2021 22:45:55	firewall_1	BLACKLIST_IN	IPv6Network	2001:db8:abcd:12::/64	IPv6net64	Active
03/19/2021 22:45:55	firewall_1	BLACKLIST_IN	IPv4Address	8.8.8.8	TestIPv48	Active
03/19/2021 22:45:55	firewall_1	BLACKLIST_IN	IPv4Network	15.15.15.0/24	TestNetwork15	Active
03/19/2021 22:45:55	firewall_1	BLACKLIST_IN	IPv4Range	1.2.3.4-1.2.3.10	TestRange1234	Active
03/19/2021 22:47:22	firewall_1	BLACKLIST_IN	IPv4Address	192.168.10.1	–	Removed

Displaying 221 - 230 of 232 Page 23 of 24

Summary

ID	2096
Phase	Engage
Severity	Low
Date Created	02/28/2021 20:15
Date Occurred	–
Date Discovered	03/01/2021 11:01
Date Determined	03/01/2021 11:01
Was personal information or personal data involved?	Unknown
Incident Type	System Intrusion

People

Created By	Resilient Sysadmin
Owner	Resilient Sysadmin
Members	*There are no members.*

Related Incidents
No related incidents.

Attachments
There are no attachments.

Newsfeed

Resilient Sysadmin added a row to the Data Table Cisco ASA Network Objects
03/19/2021 23:22:30

Resilient Sysadmin added a row to the Data Table Cisco ASA Network Objects
03/19/2021 23:09:38

API Name:

cisco_asa_network_object_dt

Columns:

Column Name	API Access Name	Type	Tooltip
Cisco ASA Firewall	`cisco_asa_firewall`	`text`	-
Network Object Group	`cisco_asa_network_object_group`	`text`	-
Object ID	`cisco_asa_network_object_id`	`text`	-
Object Kind	`cisco_asa_network_object_kind`	`text`	-
Object Value	`cisco_asa_network_object_value`	`text`	-
Query Date	`cisco_asa_query_date`	`datetimepicker`	-
Status	`cisco_asa_status`	`textarea`	-

Rules

Rule Name	Object	Workflow Triggered
Cisco ASA: Get Network Object Group	incident	`cisco_asa_get_network_object_group`

Rule Name	Object	Workflow Triggered
Cisco ASA: Add IP Range to Network Object Group	artifact	`cisco_asa_add_artifact_to_network_object_group`
Cisco ASA: Add IP Address to Network Object Group	artifact	`cisco_asa_add_artifact_to_network_object_group`
Cisco ASA: Add FQDN to Network Object Group	artifact	`cisco_asa_add_artifact_to_network_object_group`
Cisco ASA: Get Network Object Details	cisco_asa_network_object_dt	`cisco_asa_get_network_object_details`
Cisco ASA: Remove Network Object from Network Object Group	cisco_asa_network_object_dt	`cisco_asa_remove_network_object_from_network_object_group`
Cisco ASA: Add IPv6Network to Network Object Group	artifact	`cisco_asa_add_artifact_to_network_object_group`
Cisco ASA: Add IPv4Network to Network Object Group	artifact	`cisco_asa_add_artifact_to_network_object_group`

Troubleshooting & Support

Refer to the documentation listed in the Requirements section for troubleshooting information.

For Support

This is a IBM Community provided App. Please search the Community https://ibm.biz/soarcommunity for assistance.