QRadar Enhanced Offense Data Migration

Table of Contents

Release Notes

Version	Date	Notes
1.0.0	12/2020	Initial Release
1.1.0	07/2021	Support for Flows and QRoc
1.1.1	07/2021	Fixed selftest failing when using cafile
1.1.2	09/2020	Updated version of resilient-circuits to depend on
1.2.0	11/2021	Allow multiple QRadar instances

Overview

QRadar Enhanced Offense Data Migration

This app fetches the data associated with the QRadar Offense and provides live links back to QRadar, thereby simplifying case management.

Key Features

- Offense data available in a SOAR "QR Offense Details" tab as part of the Case to simplify reviewing information in one central/consistent location.
- Access to detailed Offense information by following the hotlink from the SOAR UI to QRadar Analyst Workflow.
- Centralize QRadar Offense IoC's associated with Security Events under Artifacts in order to use SOAR enabled integrations to enrich and remediate cases and provide visibility to the response team.

Requirements

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

SOAR platform

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

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

- SOAR platform >= `40.0.6554`.
- The app is in a container-based format (available from the AppExchange as a `zip` file).

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

- SOAR platform >= `40.0.6554`.
- 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 `resilient_circuits>=41.1.0`.
- If using an API key account, make sure the account provides the following minimum permissions:

Name	Permissions
Org Data	Read
Function	Read
Layouts	Read , Edit

The following SOAR 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 SOAR platform version. On the follow-on page, you can find the *App Host Deployment Guide* or *Integration Server Guide* by expanding **SOAR 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** support a proxy server.

QRadar Requirements

The app works with QRadar 7.4.0 or higher and requires the QRadar Analayst Workflow app 1.2 or higher to be installed on QRadar. The QRadar Analyst workflow app can be downloaded from the IBM App Exchange - https://exchange.xforce.ibmcloud.com/hub/extension/123f9ec5a53214cc6e35b1e4700b0806

Installation

Install

- To install or uninstall an App or Integration on the *SOAR 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	`localhost`	*QRadar host*
username	Yes	`admin`	*QRadar account username.*
qradarpassword	Yes	`password`	*Password associated with the QRadar account username*
qradartoken	Yes	`cb971c75-b2f9-4445-aaae-xxxxxxxxxxxx`	*SEC Token generated in QRadar*
verify_cert	Yes	`/path/to/cert`	*Path to certificate or specify `false` if using self signed certificate*
search_timeout	No	`300`	*Timeout for the AQL search to be specified in seconds*

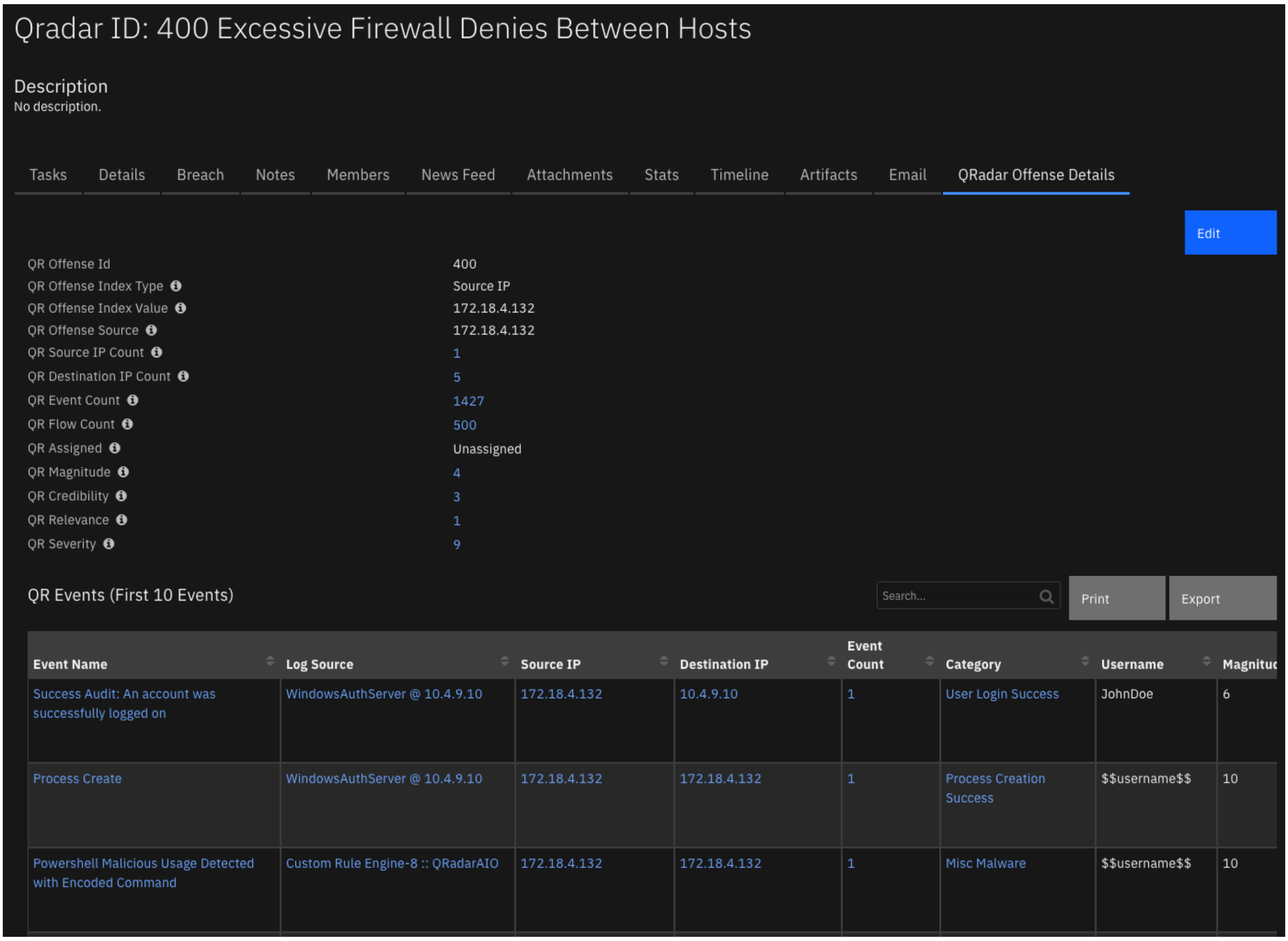
MSSP Configuration

For this app, Circuits needs to be run on the config org so that the tab is created in the config org via an API call and then afterwards, the config push is run to push to the child orgs .

Custom Layouts

Upon installation, this app adds a tab comprising of the custom fields and data tables to the Case management, if the Case has an associated Offense ID. Each of the fields and data tables have information associated with the Offense and a few have live links to QRadar Analyst Workflow. The data here is populated during the initial escalation of an Offense to a case.

All screenshots are examples of using the app with Cloud Pak.



Qradar ID: 400 Excessive Firewall Denies Between Hosts

Description
No description.

QR Offense Id	400
QR Offense Index Type	Source IP
QR Offense Index Value	172.18.4.132
QR Offense Source	172.18.4.132
QR Source IP Count	1
QR Destination IP Count	5
QR Event Count	1427
QR Flow Count	500
QR Assigned	Unassigned
QR Magnitude	4
QR Credibility	3
QR Relevance	1
QR Severity	9

QR Events (First 10 Events)

Event Name	Log Source	Source IP	Destination IP	Event Count	Category	Username	Magnitud
Success Audit: An account was successfully logged on	WindowsAuthServer @ 10.4.9.10	172.18.4.132	10.4.9.10	1	User Login Success	JohnDoe	6
Process Create	WindowsAuthServer @ 10.4.9.10	172.18.4.132	172.18.4.132	1	Process Creation Success	$$username$$	10
Powershell Malicious Usage Detected with Encoded Command	Custom Rule Engine-8 :: QRadarAIO	172.18.4.132	172.18.4.132	1	Misc Malware	$$username$$	10

Function - QRadar Offense Summary

Fetch QRadar Offense Details.

▶ Inputs:

Name	Type	Required	Example	Tooltip
`qradar_offense_id`	`text`	No	`-`	-
`qradar_query_type`	`text`	No	`-`	-

▶ Outputs:

```
results = {
    {
   "qrhost":"192.xxx.xxx.xx",
   "offenseid":"331",
   "rules_data":[
      {
         "actions":{
            "eventAnnotation":"None",
            "offenseAnnotation":"None",
            "credibility":"None",
            "ensureOffense":True,
            "offenseMapping":{
               "id":"0",
               "name":"Source IP",
               "__typename":"OffenseType"
            },
            "relevance":"None",
            "severity":"None",
            "drop":False,
            "__typename":"RuleActions"
         },
         "creationDate":"1146812107068",
         "enabled":True,
         "groups":[
            {
               "fullName":"Recon",
               "name":"Recon",
               "__typename":"Group"
            }
         ],
         "id":"100289",
         "modificationDate":"1592840490372",
         "name":"Local L2L Database Scanner",
         "notes":"Reports a scan from a local host against other local targets. At least 30 hosts were scanned in 10 minutes. ",
         "owner":"admin",
         "origin":"SYSTEM",
         "responses":{
            "newEvents":{
               "name":"Local Database Scanner Detected",
               "__typename":"RuleResponseEvent"
            },
            "email":"None",
            "log":False,
            "addToReferenceData":"None",
            "addToReferenceSet":"None",
            "removeFromReferenceData":"None",
            "removeFromReferenceSet":"None",
            "notify":False,
            "notifySeverityOverride":False,
            "selectiveForwardingResponse":"None",
            "customAction":"None",
            "__typename":"RuleResponse"
         },
         "tests":[
            {
               "group":"Event Property Tests",
               "negate":False,
               "text":"when the event context is Local to Local, Local to Remote",
               "uid":"1",
               "__typename":"RuleTest"
            },
            {
               "group":"Functions",
               "negate":False,
               "text":"when an event matches any of the following <BB>BB:PortDefinition: Database Ports</BB>",
               "uid":"3",
               "__typename":"RuleTest"
            },
            {
               "group":"Functions",
               "negate":False,
               "text":"when any of these <BB>BB:CategoryDefinition: Recon Events</BB> <BB>BB:CategoryDefinition: Suspicious Events with the same source IP more than 5 times</BB> across more than 29 destination IP within 10 minutes",
               "uid":"4",
               "__typename":"RuleTest"
            }
         ],
         "type":"COMMON",
         "__typename":"Rule"
      }
   ]
}
}
```

▶ Example Pre-Process Script:

```
inputs.qradar_offense_id= incident.properties.qradar_id
inputs.qradar_query_type = "offenserules"
```

▶ Example Post-Process Script:

```
link = "<a href=\"https://"+results.qrhost+"/console/ui/offenses?filter=
{0}%3B%3D%3B%3B{1}&page=1&pagesize=10\" target=\"_blank\">{2}</a>"

for event in results.rules_data:
  qradar_event = incident.addRow("qr_triggered_rules")
  qradar_event.rule_name = link.format("rules",event.id,event.name)
  qradar_event.rule_group = ", ".join(list(map(lambda x:x.name,list(filter(lambda x:x.name is not
None,event.groups))))) if len(event.groups)>0 else ""
  qradar_event.rule_type = event.type
  qradar_event.enabled = "True" if event.enabled else "False"
  qradar_event.response = "Yes" if event.responses.newEvents or event.responses.email or
event.responses.log or event.responses.addToReferenceData or event.responses.addToReferenceSet or
event.responses.removeFromReferenceData or event.responses.removeFromReferenceSet or
event.responses.notify or event.responses.notifySeverityOverride or
event.responses.selectiveForwardingResponse or event.responses.customAction else "No"
  qradar_event.date_created = event.creationDate
  qradar_event.last_modified = event.modificationDate
```

Function - QRadar Top Events

Search QRadar Top events for the given Offense ID.

▶ Inputs:

Name	Type	Required	Example	Tooltip
qradar_query	textarea	No	–	A qradar query string with parameters
qradar_query_param1	text	No	–	-
qradar_query_param2	text	No	–	-
qradar_query_param3	text	No	–	-
qradar_query_param4	text	No	–	-
qradar_query_param5	text	No	–	-
qradar_query_param6	text	No	–	-
qradar_query_type	text	No	–	-

▶ Outputs:

```
results = {
{
   "qrhost":"192.xxx.xxx.xx",
   "offenseid":"331",
   "events":[
      {
         "categoryname":"FTP Action Allowed",
         "magnitude":"9",
         "eventcount":"1",
         "eventtime":"1607458945836",
         "sourceipcount":"1",
         "destinationipcount":"1"
      },
      {
         "categoryname":"SFTP Login Succeeded",
         "magnitude":"6",
         "eventcount":"1",
         "eventtime":"1607458944884",
         "sourceipcount":"1",
         "destinationipcount":"1"
      },
      {
         "categoryname":"Firewall Deny",
         "magnitude":"8",
         "eventcount":"50",
         "eventtime":"1607458816101",
         "sourceipcount":"1",
         "destinationipcount":"50"
      },
      {
         "categoryname":"Network Sweep",
         "magnitude":"9",
         "eventcount":"1",
         "eventtime":"1607458807831",
         "sourceipcount":"1",
         "destinationipcount":"1"
      },
      {
         "categoryname":"Database Reconnaissance",
         "magnitude":"7",
         "eventcount":"1",
         "eventtime":"1607458796816",
         "sourceipcount":"1",
         "destinationipcount":"1"
      }
   ]
}
}
```

▶ Example Pre-Process Script:

```
inputs.qradar_query_param3 = incident.properties.qradar_id
inputs.qradar_query_type = "categories"
```

▶ Example Post-Process Script:

```
link = "<a href=\"https://"+results.qrhost+"/console/ui/offenses/{0}/events?filter={1}%3B%3D%3B%3B{2}&page=1&pagesize=10\" target=\"_blank\">{3}</a>"

for event in results.events:
  qradar_event = incident.addRow("qr_categories")
  qradar_event.category_name = link.format(results.offenseid,"category_name",event.categoryname,event.categoryname)
  qradar_event.magnitude = link.format(results.offenseid,"category_name",event.categoryname,event.magnitude)
  qradar_event.event_count = link.format(results.offenseid,"category_name",event.categoryname,event.eventcount)
  qradar_event.event_time =  event.eventtime
  qradar_event.sourceip_count = link.format(results.offenseid,"category_name",event.categoryname,event.sourceipcount)
  qradar_event.destinationip_count = link.format(results.offenseid,"category_name",event.categoryname,event.destinationipcount)
```

Script - Create Artifact from Destination IP info

Create artifact from Destination IP information for the selected row.

Object: qr_top_destination_ips

▶ Script Text:

```
#
# We create artifacts according to how they can be mapped to
# Resilient default artifacts. If you have custom artifacts, and would like
# to map them as well, please modify the following mapping dict.
#

type_mapping = {
    "Destination IP": "IP Address",
}

import re


artifact_types = rule.properties.select_to_create_artifact_from_destip

for type in artifact_types:
  if type in type_mapping:
    artifact_description = "QRadar Offense {0}".format(type)
    if type=="Destination IP":
      incident.addArtifact(type_mapping[type], re.sub("<[^<>]+>","",row.destination_ip["content"]),
artifact_description)
```

Script - Create Artifact from Source IP info

Create artifact from Source IP information for the selected row.

Object: qr_top_source_ips

▶ Script Text:

```
#
# We create artifacts according to how they can be mapped to
# Resilient default artifacts. If you have custom artifacts, and would like
# to map them as well, please modify the following mapping dict.
#

type_mapping = {
    "Source IP": "IP Address",
    "MAC": "MAC Address",
}

import re

artifact_types = rule.properties.select_to_create_artifact_from_sourceip

for type in artifact_types:
  if type in type_mapping:
    artifact_description = "QRadar Offense {0}".format(type)
    if type=="Source IP":
      incident.addArtifact(type_mapping[type],re.sub("<[^<>]+>","",row.source_ip["content"]),
artifact_description)
    elif type=="MAC":
      incident.addArtifact(type_mapping[type], row.mac, artifact_description)
```

Script - Create Artifact from Events info

Create artifact from the Events information of the selected row.

Object: qr_offense_top_events

▶ Script Text:

```
#
# We create artifacts according to how they can be mapped to
# Resilient default artifacts. If you have custom artifacts, and would like
# to map them as well, please modify the following mapping dict.
#

type_mapping = {
    "Source IP": "IP Address",
    "Destination IP": "IP Address",
    "Username": "User Account"
}

import re


artifact_types = rule.properties.select_to_create_artifact

for type in artifact_types:
  if type in type_mapping:
    artifact_description = "QRadar Offense {0}".format(type)
    if type=="Source IP":
      incident.addArtifact(type_mapping[type],re.sub("<[^<>]+>","",row.source_ip["content"]),
artifact_description)
    elif type=="Destination IP":
      incident.addArtifact(type_mapping[type], re.sub("<[^<>]+>","",row.destination_ip["content"]),
artifact_description)
    elif type=="Username":
      incident.addArtifact(type_mapping[type], row.username, artifact_description)
```

Script - Create Artifact from Assets info

Create artifact from Assets information for the selected row.

Object: qr_assets

▶ Script Text:

```
#
# We create artifacts according to how they can be mapped to
# Resilient default artifacts. If you have custom artifacts, and would like
# to map them as well, please modify the following mapping dict.
#

type_mapping = {
    "IP Address": "IP Address",
    "Name": "String",

}

import re


artifact_types = rule.properties.select_to_create_artifact_from_asset_info

for type in artifact_types:
  if type in type_mapping:
    artifact_description = "QRadar Offense {0}".format(type)
    if type=="IP Address":
     incident.addArtifact(type_mapping[type], row.ip_address["content"], artifact_description)
    elif type=="Name":
       incident.addArtifact(type_mapping[type], row.asset_name["content"], artifact_description)
```

Script - Create Artifact from Flows info

Create artifact from the Flows info of the selected row.

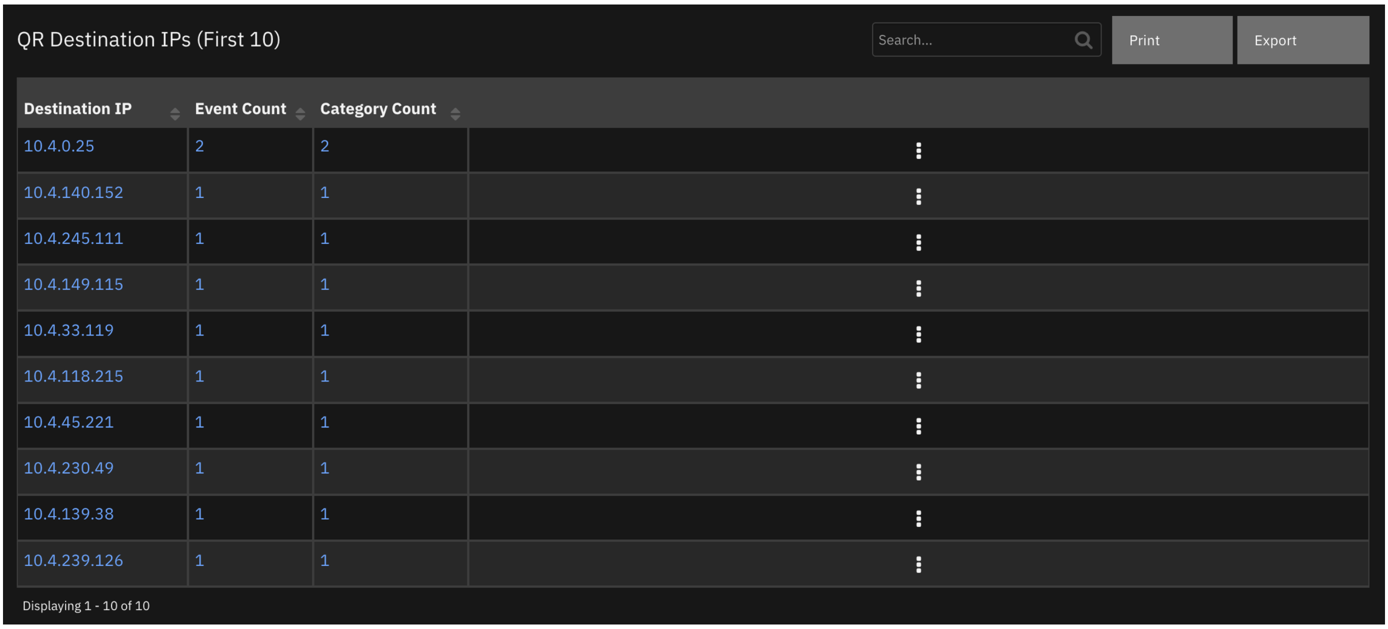
Object: qr_flows

▶ Script Text:

```
#
# We create artifacts according to how they can be mapped to
# Resilient default artifacts. If you have custom artifacts, and would like
# to map them as well, please modify the following mapping dict.
#

type_mapping = {
    "Source IP": "IP Address",
    "Destination IP": "IP Address",
    "Source Port": "Port",
    "Destination Port": "Port"

}

import re


artifact_types = rule.properties.select_to_create_artifact_from_flows_info

for type in artifact_types:
  if type in type_mapping:
    artifact_description = "QRadar Offense {0}".format(type)
    if type=="Source IP":
      incident.addArtifact(type_mapping[type],row.source_ip["content"], artifact_description)
    elif type=="Destination IP":
      incident.addArtifact(type_mapping[type],row.destination_ip["content"], artifact_description)
    elif type=="Source Port":
      incident.addArtifact(type_mapping[type],row.source_ip["content"], artifact_description)
    elif type=="Destination Port":
      incident.addArtifact(type_mapping[type],row.destination_ip["content"], artifact_description)
```

Data Table - QR Destination IPs (First 10)

The following is an example of QRadar Destination IP data table populated with the information related to Destination IPs associated with the Offense.



Destination IP	Event Count	Category Count
10.4.0.25	2	2
10.4.140.152	1	1
10.4.245.111	1	1
10.4.149.115	1	1
10.4.33.119	1	1
10.4.118.215	1	1
10.4.45.221	1	1
10.4.230.49	1	1
10.4.139.38	1	1
10.4.239.126	1	1

Displaying 1 - 10 of 10

API Name:

qr_top_destination_ips

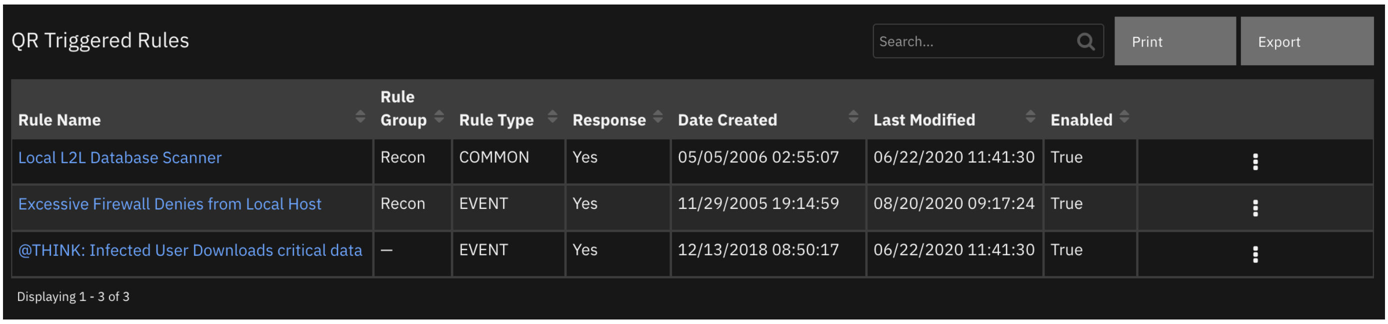
Columns:

Column Name	API Access Name	Type	Tooltip
Category Count	`category_count`	`textarea`	-
Destination IP	`destination_ip`	`textarea`	-

Column Name	API Access Name	Type	Tooltip
Event Count	`event_count`	`textarea`	-

Data Table - QR Triggered Rules

The following is an example of QRadar Triggered Rules data table populated with the information related to Contributing Rules for the Offense.


QR Triggered Rules

Rule Name	Rule Group	Rule Type	Response	Date Created	Last Modified	Enabled	
Local L2L Database Scanner	Recon	COMMON	Yes	05/05/2006 02:55:07	06/22/2020 11:41:30	True	⋮
Excessive Firewall Denies from Local Host	Recon	EVENT	Yes	11/29/2005 19:14:59	08/20/2020 09:17:24	True	⋮
@THINK: Infected User Downloads critical data	—	EVENT	Yes	12/13/2018 08:50:17	06/22/2020 11:41:30	True	⋮

Displaying 1 - 3 of 3

API Name:

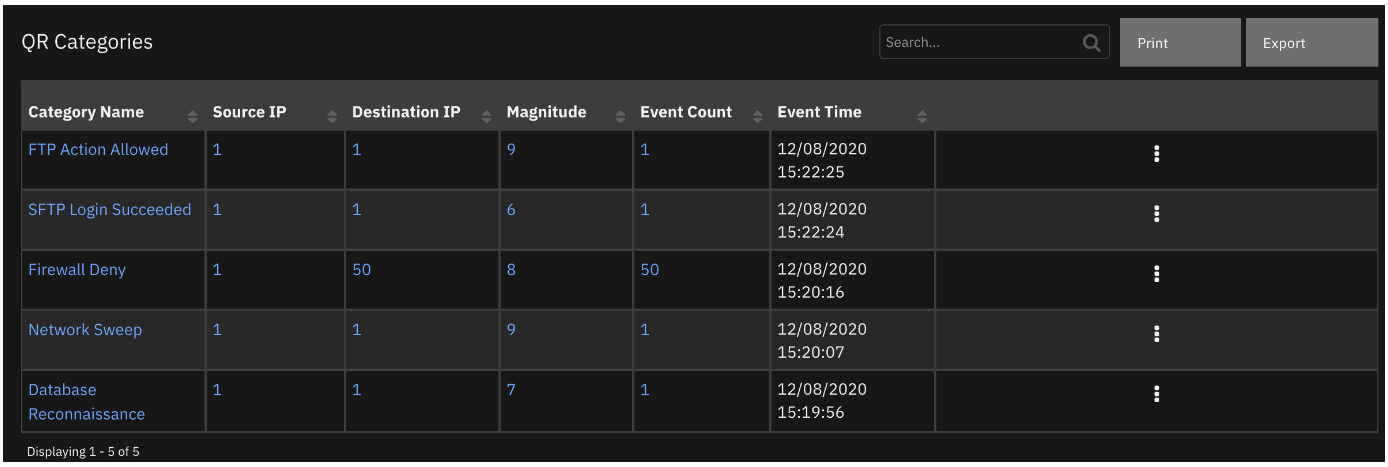
qr_triggered_rules

Columns:

Column Name	API Access Name	Type	Tooltip
Date Created	`date_created`	`datetimepicker`	-
Enabled	`enabled`	`text`	-
Last Modified	`last_modified`	`datetimepicker`	-
Response	`response`	`text`	-
Rule Group	`rule_group`	`text`	-
Rule Name	`rule_name`	`textarea`	-
Rule Type	`rule_type`	`text`	-

Data Table - QR Categories

The following is an example of QRadar Categories data table populated with the information related to Categories associated with the Offense.


QR Categories

Category Name	Source IP	Destination IP	Magnitude	Event Count	Event Time	
FTP Action Allowed	1	1	9	1	12/08/2020 15:22:25	⋮
SFTP Login Succeeded	1	1	6	1	12/08/2020 15:22:24	⋮
Firewall Deny	1	50	8	50	12/08/2020 15:20:16	⋮
Network Sweep	1	1	9	1	12/08/2020 15:20:07	⋮
Database Reconnaissance	1	1	7	1	12/08/2020 15:19:56	⋮

Displaying 1 - 5 of 5

API Name:

qr_categories

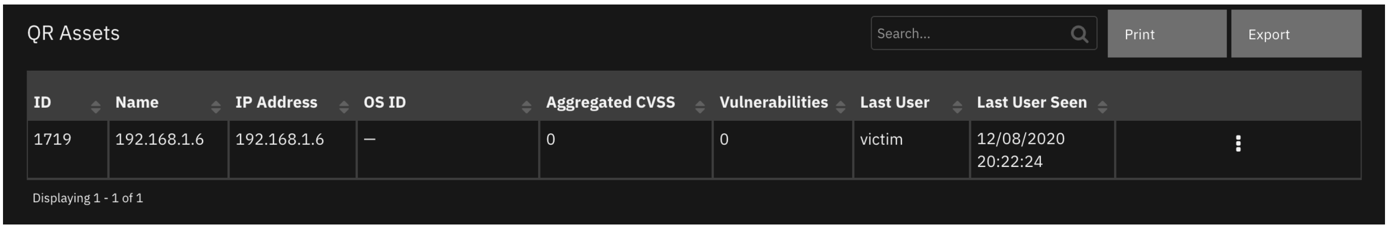
Columns:

Column Name	API Access Name	Type	Tooltip

Column Name	API Access Name	Type	Tooltip
Category Name	`category_name`	`textarea`	-
Destination IP	`destinationip_count`	`textarea`	-
Event Count	`event_count`	`textarea`	-
Event Time	`event_time`	`datetimepicker`	-
Magnitude	`magnitude`	`textarea`	-
Source IP	`sourceip_count`	`textarea`	-

Data Table - QR Assets

The following is an example of QRadar Assets data table populated with the Assets information related to the Offense.



QR Assets

ID	Name	IP Address	OS ID	Aggregated CVSS	Vulnerabilities	Last User	Last User Seen
1719	192.168.1.6	192.168.1.6	—	0	0	victim	12/08/2020 20:22:24

Displaying 1 - 1 of 1

API Name:

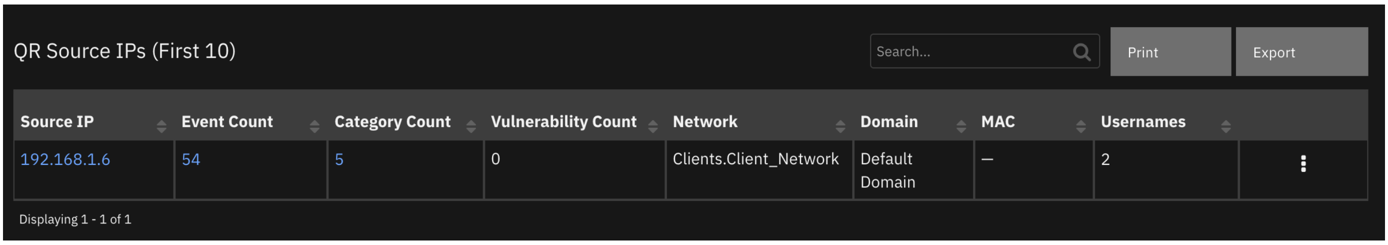
qr_assets

Columns:

Column Name	API Access Name	Type	Tooltip
Aggregated CVSS	`aggregated_cvss`	`textarea`	-
ID	`asset_id`	`textarea`	-
Name	`asset_name`	`textarea`	-
IP Address	`ip_address`	`textarea`	-
Last User	`last_user`	`textarea`	-
Last User Seen	`last_user_seen`	`datetimepicker`	-
OS ID	`operating_system`	`textarea`	-
Vulnerabilities	`vulnerabilities`	`textarea`	-

Data Table - QR Source IPs (First 10)

The following is an example of QRadar Source IP data table populated with the information related to Source IPs associated with the Offense.



QR Source IPs (First 10)

Source IP	Event Count	Category Count	Vulnerability Count	Network	Domain	MAC	Usernames
192.168.1.6	54	5	0	Clients.Client_Network	Default Domain	—	2

Displaying 1 - 1 of 1

API Name:

qr_top_source_ips

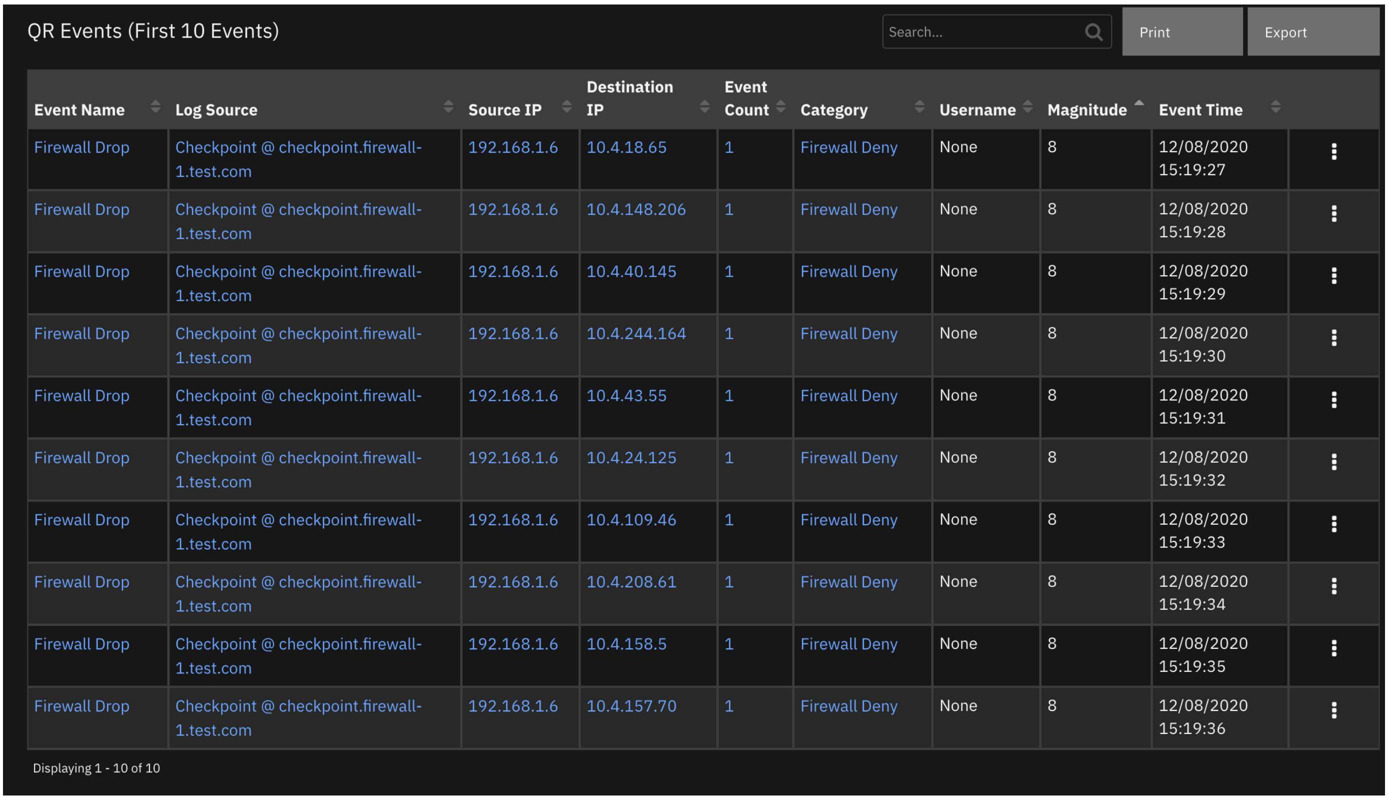
Columns:

Column Name	API Access Name	Type	Tooltip
Category Count	`category_count`	`textarea`	-

Column Name	API Access Name	Type	Tooltip
Domain	`domain`	`text`	-
Event Count	`event_count`	`textarea`	-
MAC	`mac`	`text`	-
Network	`network`	`text`	-
Source IP	`source_ip`	`textarea`	-
Usernames	`usernames`	`textarea`	-
Vulnerability Count	`vulnerability_count`	`number`	-

Data Table - QR Events (First 10 Events)

The following is an example of QRadar Events data table populated with the information related to first 10 events associated with the Offense.



QR Events (First 10 Events)

Event Name	Log Source	Source IP	Destination IP	Event Count	Category	Username	Magnitude	Event Time	
Firewall Drop	Checkpoint @ checkpoint.firewall-1.test.com	192.168.1.6	10.4.18.65	1	Firewall Deny	None	8	12/08/2020 15:19:27	⋮
Firewall Drop	Checkpoint @ checkpoint.firewall-1.test.com	192.168.1.6	10.4.148.206	1	Firewall Deny	None	8	12/08/2020 15:19:28	⋮
Firewall Drop	Checkpoint @ checkpoint.firewall-1.test.com	192.168.1.6	10.4.40.145	1	Firewall Deny	None	8	12/08/2020 15:19:29	⋮
Firewall Drop	Checkpoint @ checkpoint.firewall-1.test.com	192.168.1.6	10.4.244.164	1	Firewall Deny	None	8	12/08/2020 15:19:30	⋮
Firewall Drop	Checkpoint @ checkpoint.firewall-1.test.com	192.168.1.6	10.4.43.55	1	Firewall Deny	None	8	12/08/2020 15:19:31	⋮
Firewall Drop	Checkpoint @ checkpoint.firewall-1.test.com	192.168.1.6	10.4.24.125	1	Firewall Deny	None	8	12/08/2020 15:19:32	⋮
Firewall Drop	Checkpoint @ checkpoint.firewall-1.test.com	192.168.1.6	10.4.109.46	1	Firewall Deny	None	8	12/08/2020 15:19:33	⋮
Firewall Drop	Checkpoint @ checkpoint.firewall-1.test.com	192.168.1.6	10.4.208.61	1	Firewall Deny	None	8	12/08/2020 15:19:34	⋮
Firewall Drop	Checkpoint @ checkpoint.firewall-1.test.com	192.168.1.6	10.4.158.5	1	Firewall Deny	None	8	12/08/2020 15:19:35	⋮
Firewall Drop	Checkpoint @ checkpoint.firewall-1.test.com	192.168.1.6	10.4.157.70	1	Firewall Deny	None	8	12/08/2020 15:19:36	⋮

Displaying 1 - 10 of 10

API Name:

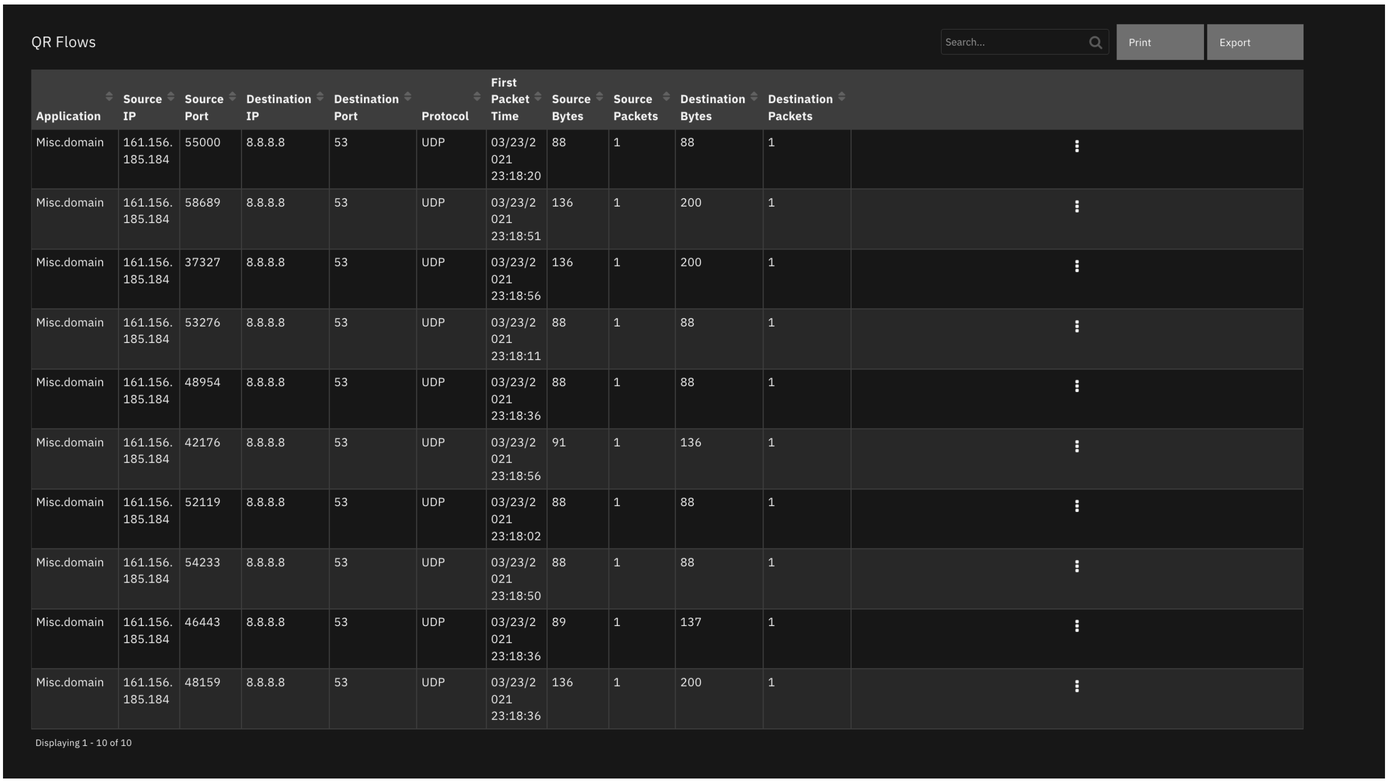
qr_offense_top_events

Columns:

Column Name	API Access Name	Type	Tooltip
Category	`category`	`textarea`	-
Destination IP	`destination_ip`	`textarea`	-
Event Count	`event_count`	`textarea`	-
Event Name	`event_name`	`textarea`	-
Event Time	`event_time`	`datetimepicker`	-
Log Source	`log_source`	`textarea`	-
Magnitude	`magnitude`	`text`	-
Source IP	`source_ip`	`textarea`	-
Username	`username`	`text`	-

Data Table - QR Flows

The following is an example of QRadar Flows data table populated with the information related to flows associated with the Offense.



QR Flows

Application	Source IP	Source Port	Destination IP	Destination Port	Protocol	First Packet Time	Source Bytes	Source Packets	Destination Bytes	Destination Packets
Misc.domain	161.156.185.184	55000	8.8.8.8	53	UDP	03/23/2021 23:18:20	88	1	88	1
Misc.domain	161.156.185.184	58689	8.8.8.8	53	UDP	03/23/2021 23:18:51	136	1	200	1
Misc.domain	161.156.185.184	37327	8.8.8.8	53	UDP	03/23/2021 23:18:56	136	1	200	1
Misc.domain	161.156.185.184	53276	8.8.8.8	53	UDP	03/23/2021 23:18:11	88	1	88	1
Misc.domain	161.156.185.184	48954	8.8.8.8	53	UDP	03/23/2021 23:18:36	88	1	88	1
Misc.domain	161.156.185.184	42176	8.8.8.8	53	UDP	03/23/2021 23:18:56	91	1	136	1
Misc.domain	161.156.185.184	52119	8.8.8.8	53	UDP	03/23/2021 23:18:02	88	1	88	1
Misc.domain	161.156.185.184	54233	8.8.8.8	53	UDP	03/23/2021 23:18:50	88	1	88	1
Misc.domain	161.156.185.184	46443	8.8.8.8	53	UDP	03/23/2021 23:18:36	89	1	137	1
Misc.domain	161.156.185.184	48159	8.8.8.8	53	UDP	03/23/2021 23:18:36	136	1	200	1

Displaying 1 - 10 of 10

API Name:

qr_flows

Columns:

Column Name	API Access Name	Type	Tooltip
Application	`application`	`textarea`	-
Source IP	`source_ip`	`textarea`	-
Source Port	`source_port`	`textarea`	-
Destination IP	`destination_ip`	`textarea`	-
Protocol	`protocol`	`textarea`	-
First Packet Time	`first_packet_time`	`textarea`	-
Source Bytes	`source_bytes`	`number`	-
Source Packets	`source_packets`	`number`	-
Destination Bytes	`destination_bytes`	`number`	-
Destination Packets	`destination_packets`	`number`	-

Custom Fields

Label	API Access Name	Type	Prefix	Placeholder	Tooltip
QR Offense Id	`qradar_id`	`text`	`properties`	-	-
QR Credibility	`qr_credibility`	`textarea`	`properties`	-	Indicates the integrity of the offense as determined by the credibility rating that is configured in the log source.

Label	API Access Name	Type	Prefix	Placeholder	Tooltip
QR Relevance	`qr_relevance`	`textarea`	`properties`	-	Indicates the importance of the destination. QRadar determines the relevance by the weight that the administrator assigned to the networks and assets.
QR Magnitude	`qr_magnitude`	`textarea`	`properties`	-	Indicates the relative importance of the offense. This value is calculated based on the relevance, severity, and credibility ratings.
QR Offense Index Type	`qr_offense_index_type`	`text`	`properties`	-	The type on which the QRadar Offense is indexed
QR Offense Source	`qr_offense_source`	`text`	`properties`	-	The source for the QRadar Offense
QR Event Count	`qr_event_count`	`textarea`	`properties`	-	The no. of events associated with the QRadar Offense
QR Flow Count	`qr_event_count`	`textarea`	`properties`	-	The no. of flows associated with the QRadar Offense
QR Destination IP Count	`qr_destination_ip_count`	`textarea`	`properties`	-	The no. of Destination IPs associated with the QRadar Offense
QR Offense Index Value	`qr_offense_index_value`	`text`	`properties`	-	The value by which QRadar Offense is indexed
QR Assigned	`qr_assigned`	`textarea`	`properties`	-	The analyst to whom the QRadar Offense is assigned to.
QR Severity	`qr_severity`	`textarea`	`properties`	-	Indicates the threat that an attack poses in relation to how prepared the destination is for the attack.
QR Source IP Count	`qr_source_ip_count`	`textarea`	`properties`	-	The no. of Source IPs associated with the QRadar Offense

Rules

Rule Name	Object	Workflow Triggered
Create artifact from Source IP info	qr_top_source_ips	`-`
QRadar Enhanced Data	incident	`qradar_offense_summary, qradar_triggered_rules, qradar_destination_ips, qradar_source_ips, qradar_categories, qradar_assets_information,example_of_searching_qradar_top_events_using_offense_id`
Create Artifact from Events info	qr_offense_top_events	`-`
Create Artifact from Assets info	qr_assets	`-`
Create artifact from Destination IP info	qr_top_destination_ips	`-`

The rule, QRadar Enhanced Data, is an automatic rule that triggers when a new incident with a qradar_id value is created, or an existing incident whose qradar_id value is updated. This rule triggers workflows as listed above and populates the Offense information in the custom fields and data tables. The rules for creating artifacts are menu item rules associated with the data tables. These rules can be executed at row level to generate artifacts from the column values. The workflows' input and post processing scripts can be customized for data retrieval and data presentation.

Troubleshooting & Support

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

For Support

This is a IBM supported App. For assistance - https://ibm.com/mysupport.