

kaspersky

Kaspersky Data Feeds for QRadar importing utility

Product version: 2.1



Dear User,

Thank you for choosing Kaspersky as your security software provider. We hope that this document helps you to use our product.

Attention! This document is the property of AO Kaspersky Lab (herein also referred to as Kaspersky). All rights to this document are reserved by the copyright laws of the Russian Federation and by international treaties. Illegal reproduction and distribution of this document or parts hereof incur civil, administrative, or criminal liability under applicable law.

Any type of reproduction or distribution of any materials, including translations, is allowed only with the written permission of Kaspersky.

This document, and graphic images related to it, may be used for informational, non-commercial, and personal purposes only.

Kaspersky reserves the right to amend this document without additional notification.

Kaspersky assumes no liability for the content, quality, relevance, or accuracy of any materials used in this document to which rights are held by third parties, or for any potential harms associated with use of the document.

Registered trademarks and service marks used in this document are the property of their respective owners.

Document revision date: 5/27/2022

© 2022 AO Kaspersky Lab

<https://www.kaspersky.com>

<https://help.kaspersky.com>

<https://support.kaspersky.com>

About Kaspersky: <https://www.kaspersky.com/about/company>

Contents

About this document	4
About Kaspersky Data Feeds for QRadar importing utility	5
Distribution kit	6
Hardware and software requirements	6
Kaspersky Threat Data Feeds	7
About QRadar reference sets	8
Installing and configuring the utility	9
Custom event properties	9
Authorized services	12
Preparing the utility for use	13
Command-line options	14
Custom rules	15
Running the utility on a regular basis	19
Using the utility	20
Workflow	20
Increasing QRadar performance	22
Logging	23
Deleting reference sets automatically	24
Removing the utility	25
Alternative ways of checking events	27
Information about third-party code	28
Trademark notices	29

About this document

This document describes Kaspersky Data Feeds for QRadar® importing utility (hereinafter also referred to as *the utility*).

About Kaspersky Data Feeds for QRadar importing utility

Kaspersky Data Feeds for QRadar importing utility is a utility that imports indicators from Kaspersky Threat Data Feeds to the IBM® QRadar reference sets.

After the utility imports indicators from the feeds into the QRadar reference sets, the QRadar Custom Rules Engine (CRE) module can check if the incoming events contain these indicators. You can configure QRadar to respond in a specific way when the CRE module detects an indicator in the incoming event.

The utility is a Python® application provided by Kaspersky; it contains no binary files.

By installing and using the utility, you agree to the terms of the End User License Agreement (EULA). You can find the EULA in the license.txt file (see section "Distribution kit" on page [6](#)).

In this chapter

Distribution kit	6
Hardware and software requirements	6
Kaspersky Threat Data Feeds	7
About QRadar reference sets	8

Distribution kit

The utility is shipped as an archive. The following table describes the contents of the archive.

Table 1. Kaspersky Data Feeds for QRadar importing utility package contents

Item	Description
delete_ref_sets.py	Script for the automatic deletion of QRadar reference sets. Reference sets deletion is not included in the main functionality of the utility; the script is run separately (see section "Deleting reference sets automatically" on page 24).
download_feeds.py	Auxiliary file.
feed_downloader_for_qradar.py	Main file to run.
Kaspersky Data Feeds for QRadar importing utility.pdf	This documentation.
legal_notices.txt	Legal notices for the product and information about third-party code.
license.txt	End User License Agreement (EULA).
modules/apiclient.py	Auxiliary file.
modules/RestApiClient.py	Auxiliary file.
modules/__init__.py	Auxiliary file.
parse_feeds.py	Auxiliary file.
Release Notes.pdf	Release notes.
requirements.txt	Dependencies for Python 2.
requirements3.txt	Dependencies for Python 3.
settings.py	Configuration file.
utils.py	Auxiliary file.

Hardware and software requirements

The utility has the following system requirements.

Supported operating systems

The utility can run on the following operating systems:

- Linux® x64

Software requirements

The utility works with the following versions of Python:

- Python 2.7.18
- Python 3.7.0 or later

The dependencies for both versions are listed in requirements.txt and requirements3.txt, respectively.

RAM requirements

The utility requires at least 700 MB of RAM.

Kaspersky Threat Data Feeds

This section describes Kaspersky Threat Data Feeds that are processed by the utility.

The following feeds are processed:

- IP Reputation Data Feed—A set of IP addresses with context covering malicious hosts.
- APT Hash Data Feed—A set of hashes that cover malicious artifacts used by APT actors to conduct APT campaigns.
- APT IP Data Feed—A set of IP addresses that belong to the infrastructure used in APT campaigns.
- APT URL Data Feed—A set of domains that belong to the infrastructure used in APT campaigns.
- Botnet CnC URL Data Feed (exact)—A set of URLs and hashes with context that cover desktop botnet C&C servers and related malicious objects.
- Malicious Hash Data Feed—A set of file hashes with context that cover the most dangerous, prevalent, or emerging malware.
- Malicious URL Data Feed (exact)—A set of URLs with context that cover malicious websites and web pages.
- Mobile Botnet CnC URL Data Feed—A set of URLs with context that cover mobile botnet C&C servers.
- Mobile Malicious Hash Data Feed—A set of file hashes with context for detecting malicious objects that infect mobile Google™ Android™ and Apple® iPhone® devices.
- Phishing URL Data Feed (exact)—A set of URLs with context that cover phishing websites and web pages.
- Ransomware URL Data Feed—A set of URLs, domains, and hosts with context that cover ransomware links and websites.
- Vulnerability Data Feed—A set of file hashes with context that cover vulnerabilities in applications and cover exploits that use those vulnerabilities.
- IoT URL Data Feed—A set of URLs with context that cover malicious links used to download malware targeting Internet of Things-enabled devices.
- ICS Hash Data Feed—A set of hashes of malicious applications that are used to attack the ICS (Industrial Control Systems) infrastructure.

Demo feeds are also available. Demo feeds provide lower detection rates in comparison with their corresponding commercial versions. The following demo feeds are available:

- Demo Botnet CnC URL Data Feed—a demo version of Botnet CnC URL Data Feed.
- Demo IP Reputation Data Feed—a demo version of IP Reputation Data Feed.
- Demo Malicious Hash Data Feed—a demo version of Malicious Hash Data Feed.
- Demo APT Hash Data Feed—a demo version of APT Hash Data Feed.

- Demo APT URL Data Feed—a demo version of APT URL Data Feed.
- Demo APT IP Data Feed—a demo version of APT IP Data Feed.

About QRadar reference sets

QRadar uses *reference sets* to store data in a simple list format.

A reference set contains unique values that you can use in searches, filters, rule test conditions, and rule responses. Thus, you can use reference sets for storing indicators of compromise (IoCs). To determine whether a reference set contains a data element, use a rule (see section "Custom rules" on page [15](#)). For example, you can create a rule that detects that an IP address takes the user to a dangerous website.

Learn more about reference sets in the QRadar documentation

(<https://www.ibm.com/docs/en/qsip/7.3.3?topic=qradar-reference-sets-overview>).

Installing and configuring the utility

This section describes QRadar entities used by the utility and provides the steps to be performed before you use the utility.

In this chapter

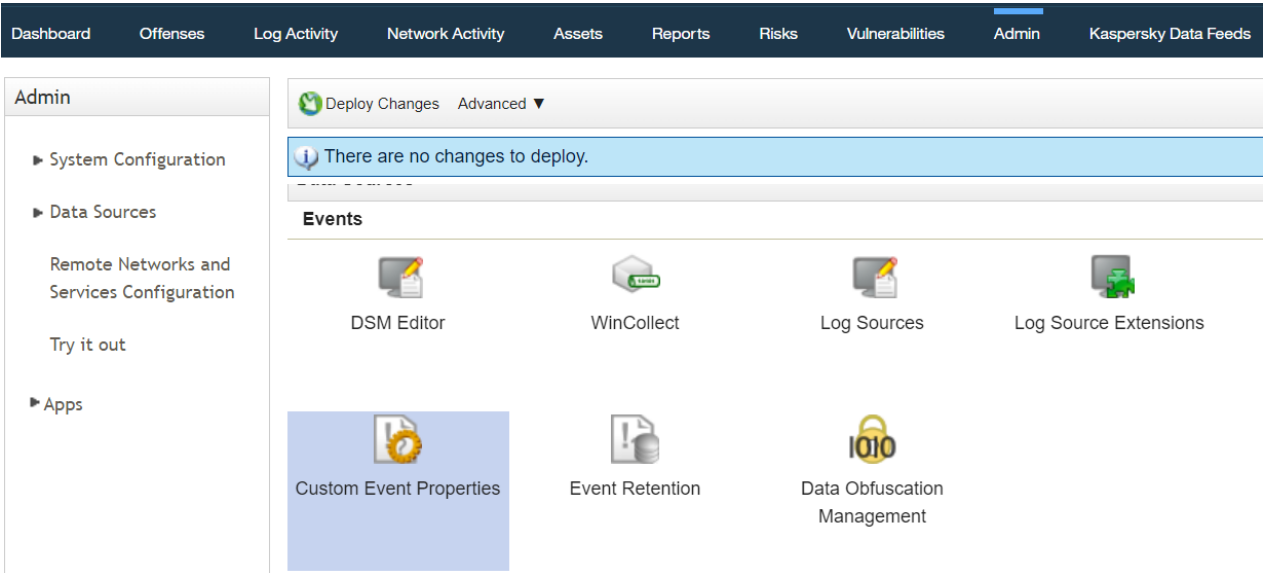
Custom event properties.....	9
Authorized services	12
Preparing the utility for use	13
Command-line options.....	14
Custom rules.....	15
Running the utility on a regular basis	19

Custom event properties

QRadar provides various event properties, based on regular expressions, for extracting information from events and checking the information against reference sets. Make sure that you have event properties for all indicators that you want to match against Kaspersky Data Feeds. If event properties for some indicators do not exist, create your own custom event properties.

Learn more about creating regex-based custom properties in the QRadar documentation (<https://www.ibm.com/docs/en/qsip/7.3.3?topic=siem-custom-event-flow-properties>).

- *To create a custom event property:*
1. In QRadar, select the **Admin** tab, go to the **Data sources** section, and under **Events**, click **Custom Event Properties**.



The **Custom Event Properties** dialog box opens.

2. Click the **Add** button.



The **Custom Event Property Definition** dialog box opens.

3. Specify the parameters of the new property.
 - a. Select the **Regex Based** option or make sure that the **Regex Based** option is selected. In the some versions of QRadar, this option is called **Extraction Based**.
 - b. In the **Property Definition** group box, select the **New Property** radio button.
 - c. Enter the name of the new property in the text field next to the **New Property** radio button. Examples of property names are listed in the table below.
 - d. In the **Field Type** drop-down list, select the field type for the property. Example property names that correspond to different field types are listed in the table below.
 - e. Select the **Parse in advance for rules, reports and searches** checkbox.
 - f. If needed, enter the description of the property in the **Description** text field.

Custom Event Properties

Property Type Selection

- ☒ Extraction Based:

Extraction based properties are created by matching a payload with a user supplied regex, JSON, LEEF, CEF, GenericList OR NameValuePair expression.
- ☐ Calculation Based:

Calculation based properties are created by choosing two numeric properties and an operator. The operator is applied to the two properties and returned as a numeric property.
- ☐ AQL Based:

AQL based properties are created by specifying the AQL expression that you want to evaluate against each event or flow in the system.

You can define custom properties from an event payload. Using the below options, you can test your RegEx entry that you wish to use to define your custom properties. If you navigated to this window from an event details window, the below options are populated with the payload of the event you were viewing.

Note: Custom fields are not indexed and therefore, could increase the time for reports, and/or searches to complete.

Test Field

Property Definition

- ☐ Existing Property:

Select a property...
- ☒ New Property:
- ☐ Parse in advance for rules, reports, and searches

Field Type:

AlphaNumeric

Description:

4. In the **Property Expression Definition** group box, specify the definition of the new property.
 - a. Select the **Enabled** checkbox.
 - b. In the **Log Source Type** and **Log Source** fields select the event source that the new event property will be applied to.

c. Do one of the following:

- Select the **Event Name** radio button and enter the name of the event that must be parsed into the field next to it.
- Select the **Category** radio button and select the category of the event in the drop-down list next to it.

d. In the **Extraction using** drop-down list, select **Regex**. Some versions of QRadar might not have this list. In this case, proceed to the next step.

e. In the **Regex** text field, specify the regular expression for extracting the corresponding indicator. Examples of regular expressions are listed in the table below.

f. In the **Capture Group** field, enter 1.

g. In the text field **Test Field** above the **Property Definition** group box, specify an example event for verification of the regular expression.

h. Verify the correctness of the regular expression by clicking the **Test** button.

```
Service|4.0|KL_TEST_URL|url=http://abc.cfde.fakess123.nu/index?abc=123 suser=EvalTestUserNameip=192.168.0.0
md5=AD5485FAC7FED74D112799600EDB2FBF sha1=A107F1046F5224FDB3A5826FA6F940A981FE65A1
sha256=B8FDAFA96DEC645BB54D4A4593C9BEAA555773F240EFE8CFF8717B20B2B93C5F act=VerificationTest eventId=101
```

Property Definition

☐ Existing Property: Select a property...

☒ New Property: url

☐ Parse in advance for rules, reports, and searches

Field Type: AlphaNumeric

Description:

Property Expression Definition

Enabled: ☒

Selection

Log Source Type: Select a log source type

Log Source: All

☒ Event Name: Please browse for an event Browse

☐ Category: High Level Category Any

Low Level Category Any

Extraction using Regex

Regex url=(?:https?:\W)?(?:[^\@\\n\\t]+@)?(?:\w\w\w\.)? ✓

Capture Group: 1 Test

5. Click **Save**.

The regular expressions specified below apply to events in the LEEF format that contain the fields `url`, `ip`, `md5`, `sha1`, and `sha256` (see the example event below). If you use a different event format, you need to edit these regular expressions to match it.

Table 2. Property names, field types, and regular expressions

Property name	Field type	Regular expression
Hash	AlphaNumeric	(?:md5 sha(?:1 256))=([a-zA-F0-9]{32,64})
URL	AlphaNumeric	url=(?:https?:\\/\ /)?(?:^\\n\\t *)
IP	IP	ip=((?:25[0-5] 2[0-4][0-9] [01]?[0-9][0-9]?\\.){3}(25[0-5] 2[0-4][0-9] [01]?[0-9][0-9]?))
Domain	AlphaNumeric	url=(?:https?:\\/\ /)?(?:[^\ /\\n\\t]+@)?(?:www\\.)?(?:[^\ /\\n\\t]+\\.)?([^\ /\\n\\t]{1,63}\\.[A-Za-z]{2,6})
Host	AlphaNumeric	url=(?:https?:\\/\ /)?(?:[^\ /\\n\\t]+@)?(?:www\\.)?([^\ /\\n\\t]+)

Example event

Fields from this example event in the LEEF format can be matched by the regular expressions from the table above. The fields are separated by the TAB characters.

```
May 2 16:41:40 KL_Verification_Tool LEEF:1.0|Kaspersky|Threat Feed
Service|4.0|KL_TEST_URL|url=http://abc.cfde.fakess123.nu/index?abc=123
suser=EvalTestUserNameip=192.168.0.0
md5=AD5485FAC7FED74D112799600EDB2FBBF
sha1=A107F1046F5224FDB3A5826FA6F940A981FE65A1
sha256=B8FDAFA96DEC645BB54D4A4593C9BEAA555773F240EFE8CFF8717B20B2B93C
5F act=VerificationTest eventId=101
```

Authorized services

The utility uses the QRadar RESTful API to interact with QRadar. To authenticate API calls to QRadar Console, the QRadar RESTful API uses either *authorized services* or QRadar users. This section describes how to add an authorized service and receive an authorization token associated with it.

The main difference between using a QRadar user login and password and using a token is the following: when you create a new user, it exists until you explicitly remove it, while a token is usually assigned a period during which it is valid.

The utility does not send sensitive data (user name, password, token) outside your organization. This data is only used to interact with the QRadar RESTful API.

► To add an authorized service:

1. In QRadar Console, select the **Admin** tab.
2. In the left navigation pane, click **System Configuration**.
3. In the right pane, under **User Management** click **Authorized Services**.

The **Manage Authorized Services** dialog box opens.

4. Click the **Add Authorized Service** button.
5. In the **Service Name** field, type a name for this authorized service (for example, *Kaspersky Data Feeds App*).
The name can be up to 255 characters in length.
6. In the **User Role** column, select the **Admin** user role to assign to this authorized service.
The user roles that are assigned to an authorized service determine the functions to which this service can gain access through the QRadar user interface.
7. In the **Security Profile** column, select the **Admin** security profile to assign to this authorized service.
The security profile determines the networks and log sources that this service can access through the QRadar user interface.
8. In the **Expiry Date** column, type or select a date when you want this service to expire. If a date of expiration is not required, select **No Expiry**.
9. Click **Create Service**.
A confirmation message appears containing a token field that you must copy into your vendor software to authenticate with QRadar.

Learn more about authorized service in the QRadar documentation

(<https://www.ibm.com/docs/en/qsip/7.3.3?topic=administration-managing-authorized-services>).

After you add an authorized service, QRadar notifies you whether the changes must be deployed.

► *To deploy the changes:*

1. In QRadar Console, select the **Admin** tab.
2. Click **Deploy Changes**.

Preparing the utility for use

The utility is shipped as an archive that contains several files (see section "Distribution kit" on page [6](#)).

► *To prepare the utility for use:*

1. Unpack the utility archive to any directory on your system.
This directory is referred to as `%utility_dir%` in this document.
2. Install the dependencies:
 - For Python 2.X.X, run:

```
pip install -r requirements.txt
```
 - For Python 3.X.X, run:

```
pip install -r requirements3.txt
```
3. Open the file `settings.py` for editing.

4. In the `FEEDS` dictionary, comment out Kaspersky Threat Data Feeds that you do not want to use.

The list of feeds that you can use is defined by your PEM certificate.

5. Specify the time period (in hours) for storing indicators in QRadar by using the `UPDATE_PERIOD_HOURS` variable.

It is recommended to specify a value close to the period for running the utility (see section "Running the utility on a regular basis" on page [19](#)).

If you want to change the value of `UPDATE_PERIOD_HOURS` after you have already imported indicators to QRadar reference sets:

- a. Delete the reference sets by using the QRadar GUI or by running `delete_ref_sets.py` (see section "Deleting reference sets automatically" on page [24](#)).
- b. Change the `UPDATE_PERIOD_HOURS` value.
- c. Run the utility.
6. If you need to increase QRadar performance, specify `MIN_POPULARITY` and `HASH_TYPES` values (see section "Increasing QRadar performance" on page [22](#)).
7. Save and close `settings.py`.
8. Run the utility manually (see section "Command-line options" on page [14](#)).

The necessary reference sets will be created in QRadar.

The utility converts URLs that contain colons (:), commas (,), or quotation marks (") to percent encoding before loading them to QRadar.

Command-line options

The utility is run from the command line as follows:

```
python %utility_dir%/feed_downloader_for_qradar.py [[-h|--help] | [-q|--qradar] <ip> [[-x|--proxy] <proxy_parameters>] [[-u|--user] <username> [-p|--password] <pwd> | [-t|--token] <token>] [[-f|--pem_file] <pem>] [-v|--verbose]] [-s|--split-by-popularity]]
```

The following table explains the command-line options.

Table 3. The utility command-line options

Option (short / full)	Description	Mandatory, default value
-q / --qradar	IP address or host name where QRadar Console is available.	Mandatory.
-x / --proxy	Proxy server connection string in the format <code>http[s]://user:password@host:port</code> . This proxy server will be used for downloading Kaspersky feeds from the WInfo server (https://winfo.kaspersky.com/).	Optional. If this option is not specified, no proxy is used.
-u / --user	Name of the user that has administrator privileges for access to the QRadar RESTful API.	You must specify either a user name and password or a token.
-p / --password	Password for access to the QRadar RESTful API.	You must specify either a user name and password or a token.
-t / --token	Authentication token for access to the QRadar RESTful API.	You must specify either a user name and password or a token.
-f / --pem_file	Path to the PEM-formatted certificate that will be used for downloading Kaspersky feeds.	Optional. By default, it is <code>%utility_dir%/feeds.pem</code> .
-v / --verbose	If specified, verbose logging is performed.	Optional.
-h / --help	If specified, a short description of the utility and how to use it is printed to the console.	Optional.
-s / --split-by-popularity	If specified, divides reference sets, created by the utility, by the value of the <code>popularity</code> field (see section "Increasing QRadar performance" on page 22).	Optional.

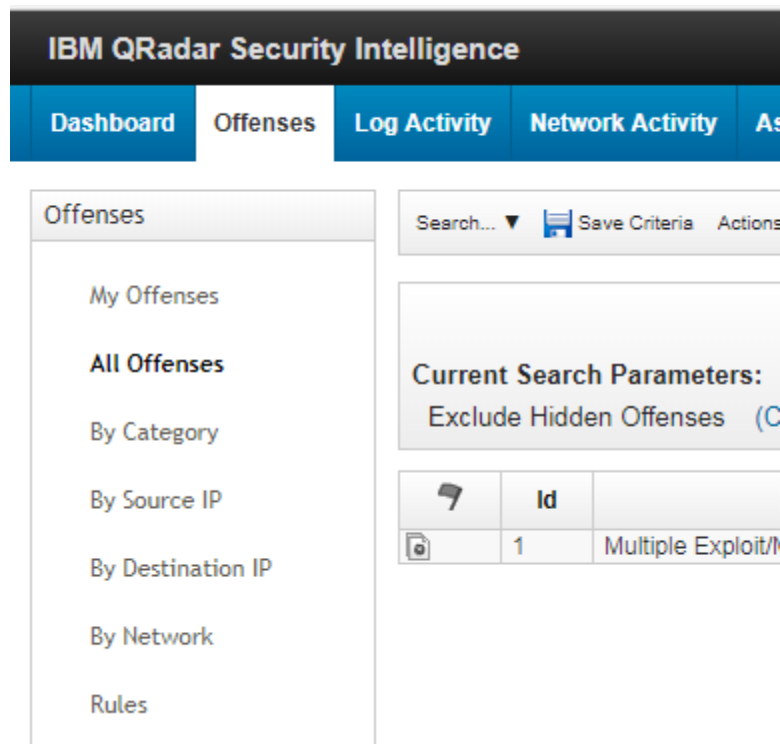
If the utility has successfully finished its work, it returns 0; otherwise, the return code is greater than 0. Therefore, you can check the return code and write a proper message to the console whether the work of the utility succeeded or failed.

Custom rules

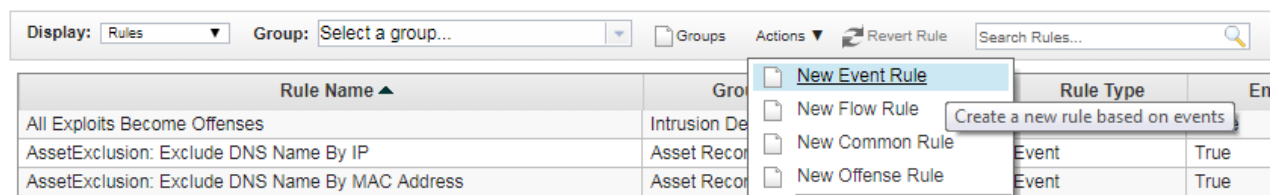
This section describes how to configure QRadar so that it will respond to incoming events. You configure QRadar by creating event rules after the utility successfully finishes its work for the first time.

► To create QRadar event rules:

- 1. In QRadar Console, select the **Offenses** tab, and then click **Rules** in the left navigation pane.



- 2. Click the **Actions** drop-down list, and then select **New Event Rule**.



The **Rule Wizard** starts.

3. Select the **Events** radio button, and then click **Next**.

Rule Wizard

Welcome to the Custom Rules Wizard!

Choose the source from which you want this rule to generate:

- ☒ **Events**
This rule type performs tests on fields that are unique to event records
- ☐ **Flows**
This rule type performs tests on fields that are unique to flow records
- ☐ **Events or Flows**
The rule type performs tests on fields that are common to both event or flow records
- ☐ **Offenses**
This rule type tests for when changes are made to existing offenses

Click Next to continue, or click Cancel to exit the wizard.

4. In the list, select the **when any of these event properties are contained in any of these reference set(s)** item, and then click the **Add test to rule** button (+) next to the item.

Rule Wizard: Rule Test Stack Editor

Which tests do you wish to perform on incoming events?

Test Group: All Export as Building Block

ref
+ when any of these event properties are contained in any of these reference set(s)
+ when any of these event properties is the key and any of these event properties is the value in any of these reference maps
+ Add test to rule these event properties is the key and any of these event properties is the value in any of these reference map or sets
+ when any of these event properties is the key of the first map and any of these event properties is the key of the second map and any of these event properties is the value in any of these reference map of maps
+ when Reference Table Key data matches any/all selected event properties and selected reference table column Select operator the value of selected event property

5. Specify the following parameters of the rule:
 - Rule name, in the **Apply** field.
For example, `KL_IP_Reputation_Danger`.
 - Detection system, in the system drop-down list.
You can select either the local or the global detection system.
 - If you select **Local**, all rules are processed by the event processor in which they were received and offenses are created only for the events that are processed locally.
 - If you select **Global**, all matching events are sent to QRadar Console for processing and, therefore, QRadar Console uses more bandwidth and processing resources.

- Fields containing indicators that must be checked against Kaspersky Data Feeds.

You can specify these fields by clicking **these event properties**. You can add necessary fields to QRadar beforehand (see section "Custom event properties" on page 9) if they are not already present there.

- Reference set.

You can specify the reference set (see section "About QRadar reference sets" on page 8) by clicking **these reference set(s)**.

Rule (Click on an underlined value to edit it)


Invalid tests are highlighted and must be fixed before rule can be saved.

Apply on events which are detected by the Local ▼ system
 and when any of these event properties are contained in any of these reference set(s)

Click **Next**.

6. Create the rule response by using the Wizard, as shown in the figure below.

Learn more about creating rule responses in the QRadar documentation (<https://www.ibm.com/docs/en/qsip/7.3.3?topic=siem-rules>).


Rule Wizard: Rule Response

Rule Action
 Choose the action(s) to take when an event occurs that triggers this rule

☐ Severity ▼
☐ Credibility ▼
☐ Relevance ▼
☐ Ensure the detected event is part of an offense
☐ Annotate event
☐ Drop the detected event

Rule Response
 Choose the response(s) to make when an event triggers this rule

☐ Dispatch New Event
☐ Email
☐ Send to Local SysLog
☐ Send to Forwarding Destinations
☐ Notify
☐ Add to a Reference Set
☐ Add to Reference Data
☐ Remove from a Reference Set
☐ Remove from Reference Data
☐ Trigger Scan
☐ Execute Custom Action

Response Limiter
 Use this section to configure the frequency with which you want this rule to respond

☐ Respond no more than time(s) per minute(s) ▼ per Rule ▼

Enable Rule
☒ Enable this rule if you want it to begin watching events right away.

7. Click **Finish** to save the rule.
8. Repeat the above steps for each reference set.

You can specify different responses for different reference sets.

Running the utility on a regular basis

Typically, the utility is run on a regular basis. This section describes how to use the cron utility for this purpose.

► *To configure periodic running of the utility:*

1. Create an empty log file `/tmp/kaspersky_feeds_for_qradar.log`.
2. Redirect log messages to `/tmp/kaspersky_feeds_for_qradar.log`, as described in Logging (on page [23](#)).
3. Run the following command for editing the crontab file:

```
crontab -e
```

4. Add the following string to the crontab file:

```
*/30 * * * * python %utility_dir%/feed_downloader_for_qradar.py
<command-line-options> || mail -s "KL feeds update failed"
email@example.com
```

(Substitute `%utility_dir%` with its real value and use a real email address for receiving failure notifications instead of `email@example.com`.)

The utility will run every 30 minutes and write its log messages to the `/tmp/kaspersky_feeds_for_qradar.log` file. If the work of the utility fails, an email with the subject `KL feeds update failed` will be sent to the specified email address.

Recommended intervals

The period for running the utility, which is specified in crontab, must have the following properties:

- Less than the value of the `UPDATE_PERIOD_HOURS` parameter in `feed_downloader_for_qradar.py`. The `UPDATE_PERIOD_HOURS` parameter means the lifetime of records in the reference sets created by the utility.
- Greater than the time needed by the utility to process all of the selected feeds. The utility needs up to three hours to processes all the feeds listed above. The utility needs about five minutes to process all the demo feeds.

We recommend that you update the reference sets as often as possible.

Using the utility

This section explains how to use the utility.

In this chapter

Workflow	20
Increasing QRadar performance	22
Logging	23
Deleting reference sets automatically.....	24
Removing the utility	25

Workflow

When you run the utility from the command line and specify the correct parameters, the utility downloads Kaspersky Threat Data Feeds from a Kaspersky server. Then it imports indicators from the feed to reference sets that are described in the table below.

Table 4. Reference sets and custom event properties

Data Feed	Reference set name	Custom event property
APT Hash Data Feed	Kaspersky APT Hash	Hash
APT IP Data Feed	Kaspersky APT IP	IP
APT URL Data Feed	Kaspersky APT URL	URL
APT URL Data Feed	Kaspersky APT Host	Host
APT URL Data Feed	Kaspersky APT Domain	Domain
Demo APT Hash Data Feed	Kaspersky Demo APT Hash	Hash
Demo APT IP Data Feed	Kaspersky Demo APT IP	IP
Demo APT URL Data Feed	Kaspersky Demo APT URL	URL
Demo APT URL Data Feed	Kaspersky Demo APT Host	Host
Demo APT URL Data Feed	Kaspersky Demo APT Domain	Domain
Botnet CnC URL Data Feed	Kaspersky Botnet CnC URL [POPULARITY_EN]	URL
Botnet CnC URL Data Feed	Kaspersky Botnet CnC Host [POPULARITY_EN]	Host

Data Feed	Reference set name	Custom event property
Botnet CnC URL Data Feed	Kaspersky Botnet CnC Domain [POPULARITY_EN]	Domain
Demo Botnet CnC URL Data Feed	Kaspersky Demo Botnet CnC URL [POPULARITY_EN]	URL
Demo Botnet CnC URL Data Feed	Kaspersky Demo Botnet CnC Host [POPULARITY_EN]	Host
Demo Botnet CnC URL Data Feed	Kaspersky Demo Botnet CnC Domain [POPULARITY_EN]	Domain
Mobile Botnet CnC URL Data Feed	Kaspersky Mobile Botnet CnC URL [POPULARITY_EN]	URL
Mobile Botnet CnC URL Data Feed	Kaspersky Mobile Botnet CnC Host [POPULARITY_EN]	Host
Mobile Botnet CnC URL Data Feed	Kaspersky Mobile Botnet CnC Domain [POPULARITY_EN]	Domain
ICS Hash Data Feed	Kaspersky ICS Hash [POPULARITY_EN]	Hash
IoT URL Data Feed	Kaspersky IoT URL	URL
IP Reputation Data Feed	Kaspersky IP Reputation [IP_CATEGORY] [THREAT_LEVEL]	IP
Demo IP Reputation Data Feed	Kaspersky Demo IP Reputation [IP_CATEGORY] [THREAT_LEVEL]	IP
Malicious URL Data Feed	Kaspersky Malicious URL [MASK_CATEGORY] [POPULARITY_EN]	URL
Malicious URL Data Feed	Kaspersky Malicious Host [MASK_CATEGORY] [POPULARITY_EN]	Host
Malicious URL Data Feed	Kaspersky Malicious Domain [MASK_CATEGORY] [POPULARITY_EN]	Domain
Malicious Hash Data Feed	Kaspersky Malicious Hash [POPULARITY_EN]	Hash
Demo Malicious Hash Data Feed	Kaspersky Demo Malicious Hash [POPULARITY_EN]	Hash
Mobile Malicious Hash Data Feed	Kaspersky Mobile Malicious Hash [POPULARITY_EN]	Hash
Phishing URL Data Feed	Kaspersky Phishing URL [POPULARITY_EN]	URL
Phishing URL Data Feed	Kaspersky Phishing Host [POPULARITY_EN]	Host
Phishing URL Data Feed	Kaspersky Phishing Domain [POPULARITY_EN]	Domain

Data Feed	Reference set name	Custom event property
Ransomware URL Data Feed	Kaspersky Ransomware URL	URL
Ransomware URL Data Feed	Kaspersky Ransomware Host	Host
Ransomware URL Data Feed	Kaspersky Ransomware Domain	Domain
Vulnerability Data Feed	Kaspersky Vulnerability Vulnerable Hash [SEVERITY]	Hash
Vulnerability Data Feed	Kaspersky Vulnerability Exploit Hash [SEVERITY]	Hash

Here, variables in brackets can have the following values:

- IP_CATEGORY—Malware, Spam, Tor Exit Node, Proxy, Phishing, Botnet CnC, Tor Node, Vpn, Test, Intrusion.

The list of possible values may change in further feed updates.

- MASK_CATEGORY—Malicious Redirect, Bot CnC, Exploit, Fraud, Malware, Mobile Malware.

The list of possible values may change in further feed updates.

- SEVERITY—Warning, Low, Medium, High, Critical.
- THREAT_LEVEL—Danger, Suspicious.

These values are determined depending on the `threat_score` field of the feed. If `threat_score` is less than 75, then the value is Suspicious, otherwise Danger.

- POPULARITY_EN—Rare, Very Rare, Average, Common, Very Common (present only if the utility is launched with the `--split-by-popularity` flag).

These values are determined depending on the `popularity` field of the feed. If `popularity` = 1, the value is Very Rare; 2—Rare; 3—Average; 4—Common; 5—Very Common.

If the `--split-by-popularity` flag is enabled and the utility has exported all feeds to QRadar, up to 232 reference sets are created. Without the `--split-by-popularity` flag, up to 96 reference sets are created.

Increasing QRadar performance

Sometimes the use of Kaspersky Data Feeds for QRadar with QRadar 7.4 and earlier shows performance degradation. The reason for this is a large number of indicators in some reference sets created by the utility.

You can increase QRadar performance as follows:

- Split reference sets by popularity with the `--split-by-popularity` option.
The utility allows you to split the reference sets created from Kaspersky Threat Data Feeds by the value of the `popularity` field. For each indicator in a feed, this field denotes the relative number of users per day who detected it. This field can have values from 1 (very rare indicators) to 5 (very common indicators). The reference sets that can be split by popularity are listed in the reference sets table and you can identify them by `[POPULARITY_EN]` in their names (see section "Workflow" on page [20](#)).
To enable splitting by popularity, run the utility with the `--split-by-popularity` option (see section "Command-line options" on page [14](#)).
- Change the `MIN_POPULARITY` and `HASH_TYPES` parameters in `%utility_dir%/settings.py`.
 - Increase the `MIN_POPULARITY` value. The utility only imports indicators whose `popularity` value is equal to or greater than the `MIN_POPULARITY` value. The higher the `MIN_POPULARITY` value, the fewer indicators the utility imports. Possible values for `MIN_POPULARITY` are from 1 (very rare indicators) to 5 (very common indicators). The default value for `MIN_POPULARITY` is 2. If `MIN_POPULARITY` is set to 0, the utility does not filter indicators by popularity.
 - Specify fewer hash types in `HASH_TYPES`. The default value for `HASH_TYPES` is `['MD5', 'SHA1', 'SHA256']`. You can remove one or two hash types from the list. For example, if you specify `['MD5']`, the utility only imports MD5 hashes of compromised files, and the SHA-1 and SHA-256 hashes are not imported.

Logging

By default, the utility logs its activity to the console. You can redirect log messages to a file.

► To redirect log messages to a file:

1. Open `%utility_dir%/settings.py` for editing.
2. Specify the `FILE` value in the `LOG_OUTPUT` variable:

```
LOG_OUTPUT = 'STDOUT'
```
3. Specify a full or relative path to the log file in the `LOG_FILENAME` variable:

```
LOG_FILENAME = 'qradar_kaspersky_feeds.log'
```
4. Save and close `settings.py`.

Log levels

The utility writes log messages at one of two log levels: brief or verbose. The log level is specified by the command-line option (see section "Command-line options" on page [14](#)).

If the brief log level is specified, the following information is written to the log:

- Information about the utility and the software environment:
 - Kaspersky Data Feeds for QRadar importing utility version
 - Python version (32-bit or 64-bit, version number)
 - Operating system (OS) version and bit

- Authentication method (login and password or token)
- List of feeds to be imported to QRadar
- For each downloaded feed:
 - Time when the download of the feed began
 - Time when the download of the feed finished
 - Size of the downloaded archive
- For each created reference set:
 - Name and other parameters of the reference set
- Message that the import of data to a reference set started
- Message that the import of data to a reference set finished
- Warnings and errors that occur during the use of RESTful API functions

If the verbose log level is specified, the following information is written to the log:

- Information that is written for the brief log level
- `UPDATE_PERIOD_HOURS`, `MIN_POPULARITY`, and `HASH_TYPES` values specified in `%utility_dir%/settings.py`
- Command-line parameters used for launching the utility

The user name, password, or token are replaced with the string `<private_data>`.
- Information about network requests made by the utility and the return codes of the requests

Deleting reference sets automatically

When importing indicators from the feeds, the utility can create a large number of reference sets (up to 232). Sometimes you may need to delete them all, for example, to change the `UPDATE_PERIOD_HOURS` parameter (see section "Preparing the utility for use" on page [13](#)). Doing it manually can be time-consuming.

Kaspersky provides you with a Python script that you can use to automatically delete reference sets created by the utility.

The script deletes reference sets containing the `Kaspersky` substring in their names. Before running the script, make sure that there are no such reference sets that you want to keep.

The script file named `delete_ref_sets.py` is included in the utility distribution kit. The dependencies for the script are listed in `%utility_dir%/requirements.txt` (Python 2) and `%utility_dir%/requirements3.txt` (Python 3).

The script cannot delete a reference set that is linked to a custom rule (see section "Custom rules" on page [15](#)). Be sure to unlink custom rules before running the script.

The script is run from the command line as follows:

```
python %utility_dir%/delete_ref_sets.py [-q|--qradar] <ip> [-t|--token]
<token>
```

The following table explains the command-line options.

Table 5. The script command-line options

Option (short / full)	Description
-q / --qradar	IP address or host name where QRadar Console is available. This option is mandatory.
-t / --token	Authentication token for access to the QRadar RESTful API. This option is mandatory.

Removing the utility

This section describes how to remove the utility.

After you have removed the utility files, you may also have to remove the following QRadar objects:

- Custom rules
- Custom event properties
- Reference sets
- Authorized services
- Users

Removing custom rules

The following procedure describes how to delete a custom rule.

► To delete a custom rule:

1. In QRadar, select the **Offenses** tab.
2. In the left navigation pane, click **Rules**.
3. Select a custom rule, click **Actions**, and in the drop-down list select **Delete**.

Removing custom event properties

The following procedure describes how to delete a custom event property.

► To delete a custom event property:

1. In QRadar, select **Admin**, go to the **Data sources** section, and under **Events** click **Custom Event Properties**.
The **Custom Event Properties** dialog box opens.
2. Select a custom event property and click the **Delete** button.

Removing reference sets

You can delete a reference set either manually by using the QRadar GUI or automatically (see section "Deleting reference sets automatically" on page [24](#)).

► *To delete a reference set by using the QRadar GUI:*

1. In QRadar, select **Admin** and under **System configuration**, select **Reference Set Management**.
The **Reference Set Management** dialog box opens.
2. Select a reference set and click the **Delete** button.

Removing authorized services

The following procedure describes how to delete an authorized service.

► *To delete an authorized service:*

1. In QRadar, select **Admin**, go to the **System configuration** section, and under **User Management** click **Authorized Services**.
The **Manage Authorized Services** dialog box opens.
2. Select an authorized service and click the **Delete** button.

Removing QRadar users

The following procedure describes how to delete a QRadar user.

► *To delete a QRadar user:*

1. In QRadar, select **Admin**, go to the **System configuration** section, and under **User Management** click **Users**.
The **User Management** dialog box opens.
2. Select a user and click the **Delete** button.

Alternative ways of checking events

The utility is designed so that you can check events by means of QRadar only, without having to use other software. At the same time, Kaspersky offers you another software product, Kaspersky CyberTrace, which has the following advantages:

- Performance of 5000 events per second without any impact on QRadar.
- High-speed capabilities for importing and exporting third-party indicators and Data Feeds.
- Kaspersky CyberTrace does not use the built-in capabilities of SIEM solutions for matching events against Data Feeds.
- No additional load on the SIEM solution and high-matching performance.
- The SIEM solutions are not designed for processing many indicators. Kaspersky CyberTrace does not have such limitations.
- The SIEM solutions are not designed for processing associated context for indicators. Kaspersky CyberTrace does not have such limitations.
- Complex matching logic with normalization of observables.
- Kaspersky CyberTrace allows searching for indicators in large sets of logs or to review historical data.

Due to a unique integration approach with SIEM solutions, Kaspersky CyberTrace helps to detect uncovered threats, measures which streams of threat intelligence are the most relevant, and provides Security Operations Center with a powerful tool for alerts triage:

- Dashboard with statistical data about detections and a breakdown of Threat Intelligence sources, taking into account false positives to highlight the best sources.
- Native integration of Kaspersky Threat Intelligence.
- Historical correlation (retrospective scan) for finding previously uncovered threats.
- Downloading feeds to a storage on a separate computer.
- Kaspersky Threat Feed App for QRadar that displays dashboards (<https://support.kaspersky.com/13854>).

Information about third-party code

Information about third-party code is contained in a file named legal_notices.txt of the distribution kit.

Trademark notices

Registered trademarks and service marks are the property of their respective owners.

Android and Google are trademarks of Google, Inc.

Apple and iPhone are trademarks of Apple Inc., registered in the U.S. and other countries.

IBM, ibm.com, and QRadar are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Python is a trademark or registered trademark of the Python Software Foundation.