# ThreatQuotient

**A Securonix Company**



## Intel 471 Vulnerability Reports CDF

**Version 2.0.0**

October 13, 2026

**ThreatQuotient**
20130 Lakeview Center Plaza Suite 400
Ashburn, VA 20147

**ThreatQ Supported**

**Support**
Email: tq-support@securonix.com
Web: https://ts.securonix.com
Phone: 703.574.9893

# Contents

**Warning and Disclaimer** ..... **3**
**Support** ..... **4**
**Integration Details** ..... **5**
**Introduction** ..... **6**
**Prerequisites** ..... **7**
Intel Requirement Custom Object ..... 7
**Installation** ..... **9**
**Configuration** ..... **10**
**ThreatQ Mapping** ..... **14**
Intel 471 Vulnerability Reports ..... 14
**Average Feed Run** ..... **20**
**Change Log** ..... **21**

# Warning and Disclaimer

ThreatQuotient, Inc. provides this document “as is”, without representation or warranty of any kind, express or implied, including without limitation any warranty concerning the accuracy, adequacy, or completeness of such information contained herein. ThreatQuotient, Inc. does not assume responsibility for the use or inability to use the software product as a result of providing this information.

Copyright © 2026 ThreatQuotient, Inc.
All rights reserved. This document and the software product it describes are licensed for use under a software license agreement. Reproduction or printing of this document is permitted in accordance with the license agreement.

# Support

This integration is designated as **ThreatQ Supported**.

**Support Email**: tq-support@securonix.com
**Support Web**: https://ts.securonix.com
**Support Phone**: 703.574.9893

Integrations/apps/add-ons designated as **ThreatQ Supported** are fully supported by ThreatQuotient's Customer Support team.

ThreatQuotient strives to ensure all ThreatQ Supported integrations will work with the current version of ThreatQuotient software at the time of initial publishing. This applies for both Hosted instance and Non-Hosted instance customers.

> ⚠ ThreatQuotient does not provide support or maintenance for integrations, apps, or add-ons published by any party other than ThreatQuotient, including third-party developers.

# Integration Details

ThreatQuotient provides the following details for this integration:

| | |
|---|---|
| **Current Integration Version** | 2.0.0 |
| **Compatible with ThreatQ Versions** | >= 6.12.1 |
| **Support Tier** | ThreatQ Supported |

# Introduction

The Intel 471 Vulnerability Reports CDF integration ingests curated vulnerability intelligence from the Intel 471 Verity471 platform into ThreatQ. Intel 471 Vulnerability Intelligence provides timely context on CVE weaponization, threat lifecycles, mass exploitation, active malware campaigns, and emerging threat actor activity observed in the criminal underground.

The integration provides the following feed:

- **Intel 471 Vulnerability Reports** - retrieves the Intel 471 Vulnerability Reports feed and ingests associated CVE indicators, Vulnerabilities, and Intel Requirements into ThreatQ.

The integration ingests the following system objects:

- Indicators (CVEs)
- Intel Requirements (custom object)
- Vulnerabilities

# Prerequisites

The following is required to install and run the integration:

- An Intel 471 Client ID and Client Secret.
- An Intel 471 API Key.

  To use the Intel 471 feeds, you must have Intel 471 API credentials with access to the Vulnerability Intelligence and Report APIs used by this integration. Generate credentials by signing in to the Intel 471 Developer Portal and creating a new application. After the application is created, use the provided Client ID and Client Secret to configure the integration in ThreatQ.

- The Intel Requirement Custom Object installed on your ThreatQ instance.

## Intel Requirement Custom Object

The integration requires the Intel Requirement custom object.

Use the steps provided to install the custom object.

When installing the custom objects, be aware that any in-progress feed runs will be cancelled, and the API will be in maintenance mode.

Use the following steps to install the custom object in ThreatQ v6:

1. Download the integration bundle from the ThreatQ Marketplace.
2. Unzip the bundle and locate the custom object files.

   The custom object files will typically consist of a JSON definition file, install.sh script, and a images folder containing the svg icons.

3. SSH into your ThreatQ instance.
4. Set your install pathway environment variable. This command will retrieve the install pathway from your configuration file and set it as variable for use during this installation process.

   ```
   INSTALL_CONF="/etc/threatq/platform/install.conf"

   if [ -f "$INSTALL_CONF" ]; then source "$INSTALL_CONF"
   ```

```
fi

MISC_DIR="${INSTALL_BASE_PATH:-/var/lib/threatq}/misc"
```

5. Navigate to the tmp folder using the environment variable:

```
cd $MISC_DIR
```

6. Upload the custom object files, including the images folder.

   The directory structure should resemble the following:

   - install.sh
   - <custom_object_name>.json
   - images (directory)
     - <custom_object_name>.svg

7. Run the following command:

```
kubectl exec -it deployment/api-schedule-run -n threatq --
sh /var/lib/threatq/misc/install.sh /var/lib/threatq/misc
```

   The installation script will automatically put the application into maintenance mode, move the files to their required directories, install the custom object, update permissions, bring the application out of maintenance mode, and restart dynamo.

8. Delete the install.sh, definition json file, and images directory from step 6 after the object has been installed as these files are no longer needed.

# Installation

The CDF requires the installation of the Intel Requirement custom object before installing the actual CDF. See the Intel Requirement section of this guide for more details. The custom object must be installed prior to installing the CDF. Attempting to install the CDF without the custom object will cause the CDF install process to fail.

Perform the following steps to install the integration:

The same steps can be used to upgrade the integration to a new version.

1. Log into https://marketplace.threatq.com/.
2. Locate and download the integration zip file.
3. Extract the integration files and install the Intel Requirement custom object if you have not done so already.
4. Navigate to the integrations management page on your ThreatQ instance.
5. Click on the **Add New Integration** button.
6. Upload the integration file using one of the following methods:
   - Drag and drop the file into the dialog box
   - Select **Click to Browse** to locate the integration file on your local machine

   ThreatQ will inform you if the feed already exists on the platform and will require user confirmation before proceeding. ThreatQ will also inform you if the new version of the feed contains changes to the user configuration. The new user configurations will overwrite the existing ones for the feed and will require user confirmation before proceeding.

The feed will be added to the integrations page. You will still need to configure and then enable the feed.

# Configuration

ThreatQuotient does not issue API keys for third-party vendors. Contact the specific vendor to obtain API keys and other integration-related credentials.

To configure the integration:

1. Navigate to your integrations management page in ThreatQ.
2. Select the **Commercial** option from the *Category* dropdown (optional).

   If you are installing the integration for the first time, it will be located under the **Disabled** tab.
3. Click on the integration entry to open its details page.
4. Enter the following parameters under the **Configuration** tab:

| PARAMETER | DESCRIPTION |
|---|---|
| **Client ID** | Required. Enter the Client ID for an application registered with Intel 471. To obtain credentials, sign in to the Intel 471 Developer Portal and create an application. Intel 471 provides a Client ID and Client Secret after the application is created. |
| **Client Secret** | Required. Enter the Client Secret associated with the configured Client ID. |
| **Exclude Historical CVEs** | Optional. Select this option to exclude CVEs with a **Historical** status from ingestion. Default: `False`. |
| **Text Filter** | Optional. Enter a text string to filter vulnerability reports. Default: None. |
| **GIR / PIR Filter** | Optional. Enter a comma-separated list of General Intelligence Requirement (GIR) identifiers to filter vulnerability reports. You can use the predefined `my_girs` or `company_pirs` values to dynamically filter reports based on user- or company-selected intelligence requirements. |

| PARAMETER | DESCRIPTION |
|---|---|
| | You can also combine specific GIR identifiers with these predefined values. Default: `None`. |
| **Risk Level Filter** | Optional. Select the risk levels used to filter vulnerability reports. If no values are selected, reports with all risk levels are included. Options: **Low**, **Medium**, and **High**. Default: **Medium**, **High**. |
| **Patch Status Filter** | Optional. Select the patch statuses used to filter vulnerability reports. If no values are selected, reports with all patch statuses are included. Options: **Unavailable**, **Some Available**, and **Available**. Default: `None`. |
| **Interest Level Filter** | Optional. Select the interest levels used to filter vulnerability reports. If no values are selected, reports with all interest levels are included. Options: **Disclosed Publicly**, **Researched Publicly**, and **Exploit Sought**. Default: `None`. |
| **Activity Location Filter** | Optional. Select the activity locations used to filter vulnerability reports. If no values are selected, reports with all activity locations are included. Options: **Private**, **Underground**, and **Open Source**. Default: `None`. |
| **Exploit Status Filter** | Optional. Select the exploit statuses used to filter vulnerability reports. If no values are selected, reports with all exploit statuses are included. Options: **Not Observed**, **Productized**, **Weaponized**, and **Available**. Default: `None`. |
| **Vendor Filter** | Optional. Enter a line-separated list of vendor names to filter vulnerability reports. Filtering occurs on the client side, and reports that do not match the specified vendors are excluded. Default: `None`. |
| **Selected Attributes** | Select the attributes to ingest for each vulnerability report, when available. Report descriptions include vulnerability summaries, relevant dates, and external references. CVSS scores are ingested only when the selected version is available. |

| PARAMETER | DESCRIPTION |
|---|---|
| | ◦ Status<br>◦ Type *(default)*<br>◦ Aliases *(default)*<br>◦ Risk Level *(default)*<br>◦ CVSSv2 Base Score<br>◦ CVSSv3 Base Score<br>◦ CVSSv3.1 Base Score<br>◦ CVSSv4 Base Score<br>◦ PoC Status<br>◦ Patch Status<br>◦ Interest Level<br>◦ Activity Location<br>◦ Exploit Status *(default)*<br>◦ Target Vendor *(default)*<br>◦ Target Product *(default)*<br>◦ First Seen<br>◦ Last Seen<br>◦ Last Updated At |
| **GIR / PIR Handling** | Select one or more methods for storing associated GIRs and company PIRs. Associated intelligence requirements are included in the report description regardless of the selected option.<br>◦ Store as Tags, for example, `ir-1.1.1`<br>◦ Store as Related Intel Requirement Objects *(default)*<br>◦ Store as Attributes, using **Intel Requirement** as the attribute name |
| **Include GIR Name When** | Select whether to include the GIR name when storing GIRs as attributes. When enabled, the value uses the format `Intel Requirement: 1.1.1 - <GIR Name>`. When |

| PARAMETER | DESCRIPTION |
|---|---|
| **Storing as Attributes** | disabled, the value uses the format `Intel Requirement: 1.1.1`. This parameter is displayed only when **Store as Attributes** is selected. Default: `True`. |
| **Save CVEs As** | Select whether to import CVEs as Vulnerability objects, Indicator objects, or both.<br>◦ Vulnerabilities *(default)*<br>◦ Indicators, using the **CVE** indicator type |

5. Review any additional settings, make any changes if needed, and click on **Save**.
6. Click on the toggle switch, located above the *Additional Information* section, to enable it.

# ThreatQ Mapping

## Intel 471 Vulnerability Reports

Intel 471 Vulnerability Reports feed retrieves the Intel 471 Vulnerability Reports feed and ingests associated CVE indicators, Vulnerabilities, and Intel Requirements into ThreatQ.

`GET https://api.intel471.cloud/integrations/intel-report/v1/reports/vulnerability/stream`

**Sample Response:**


```
{
    "count": 3905,
    "cursor_next": "...",
    "reports": [
        {
            "id":
"vulnerability--120c1004-241f-299d-1dbc-00ac1d6d52f6",
            "type": "vulnerability_report",
            "creation_ts": "2023-09-25T15:52:54Z",
            "released_ts": "2023-09-25T15:52:54Z",
            "last_updated_ts": "2025-07-10T04:40:04Z",
            "name": "CVE-2020-1234",
            "aliases": [],
            "status": "historical",
            "cve_type": "SSRF",
            "risk_level": "medium",
            "cvss": [
                {
                    "version": "2.0",
                    "score": -1.0
                },
                {
                    "version": "3.0",
                    "score": -1.0
                },
                {
                    "version": "3.1",
                    "score": -1.0
                },
                {
```

```
                    "version": "4.0",
                    "score": -1.0
                }
            ],
            "poc": "observed",
            "patch_status": "some_available",
            "body": "<p>CVE-2020-1234 is a server side request
forgery (SSRF) vulnerability impacting Acmesoft versions 8.5 to
8.7.11. A proof of concept (PoC) was observed publicly or in the
underground. Security researchers at the Cybersecurity and
Infrastructure Security Agency (CISA) claimed the vulnerability was
actively exploited in the wild.</p>",
            "underground_activity_summary_html": "<p>CVE-2020-1234
was productized. Intel 471 observed the actor NasyoneL post PoC code
for CVE-2020-1234.</p>",
            "counter_measures_html": "<p>The impacted vendor released
some patching information for impacted products and corresponding
versions. The impacted vendor addressed the vulnerability in a
security advisory.</p>",
            "interest_level": ["researched_publicly",
"disclosed_publicly"],
            "activity_location": [
                "location_underground",
                "location_opensource"
            ],
            "exploit_status": ["productized", "weaponized",
"available"],
            "sort_priority": 0,
            "vendor_name": "Acmesoft",
            "product_name": "Acmesoft Server",
            "sources": [
                {
                    "type": "internal",
                    "title": "[POST] Acmesoft Server Autodiscover
Servlet XXE / ProxyServlet SSRF Exploit",
                    "links": {
                        "verity_api": {
                            "href": "https://api.intel471.cloud/
integrations/sources/v1/forums/posts/stream?
thread_id=thread--1b4183c4-482e-51ec-c957-1628f0aed4df"
                        },
                        "verity_portal": {
```

```
                            "href": "https://verity.intel471.com/
sources/forum/forum--b051b854-ab58-5d61-aa83-0abc305b0fc1/thread/
thread--1b4183c4-482e-51ec-c957-1628f0aed4df"
                        }
                    }
                }
            ],
            "poc_links": [
                {
                    "title": "Exploit status link",
                    "links": {
                        "external": {
                            "href": "https://packetstormsecurity.com/
files/152487/Acmesoft-Server-Autodiscover-Servlet-XXE-ProxyServlet-
SSRF.html"
                        }
                    },
                    "type": "external"
                },
                {
                    "title": "CISA: KNOWN EXPLOITED VULNERABILITIES
CATALOG",
                    "links": {
                        "external": {
                            "href": "https://www.cisa.gov/known-
exploited-vulnerabilities-catalog"
                        }
                    },
                    "type": "external"
                }
            ],
            "patch_links": [
                {
                    "title": "Patch link",
                    "links": {
                        "external": {
                            "href": "https://wiki.acmesoft.com/wiki/
Acmesoft_Security_Advisories"
                        }
                    },
                    "type": "external"
                }
```

```
            ],
            "counter_measure_links": [],
            "classification": {
                "girs": [
                    {
                        "path": "2.1.2",
                        "name": "Software vulnerabilities"
                    },
                    {
                        "path": "2.2",
                        "name": "Exploit development"
                    },
                    {
                        "path": "2.2.1",
                        "name": "Proof-of-concept (PoC) exploit code"
                    }
                ]
            },
            "activity": {
                "first_seen_ts": "2023-09-25T15:52:54Z",
                "last_seen_ts": "2025-07-10T04:40:04Z"
            },
            "links": {
                "verity_api": {
                    "href": "https://api.intel471.cloud/integrations/
intel-report/v1/reports/vulnerability/
vulnerability--120c1004-241f-299d-1dbc-00ac1d6d52f6"
                },
                "verity_portal": {
                    "href": "https://verity.intel471.com/
vulnerabilities/vulnerability--120c1004-241f-299d-1dbc-00ac1d6d52f6"
                }
            }
        }
    ]
}
```

ThreatQuotient provides the following default mapping for this feed based on each record within the `.reports[]` array in the API response. Attributes and other contextual information may be subject to user-field filtering.

| FEED DATA PATH | THREATQ ENTITY | THREATQ OBJECT TYPE OR ATTRIBUTE KEY | PUBLISHED DATE | EXAMPLES | NOTES |
|---|---|---|---|---|---|
| `.name` | `Vulnerability.Value, Indicator.Value` | `CVE` | `.creation_ts` | `CVE-2026-12345` | `N/A` |
| `.status` | `*.Attribute` | `Status` | `.creation_ts` | `historical` | `N/A` |
| `.cve_type` | `*.Attribute` | `Type` | `.creation_ts` | `SSRF` | `Single-value attribute; updated on later feed runs.` |
| `.aliases[]` | `*.Attribute` | `Alias` | `.creation_ts` | `N/A` | `N/A` |
| `.risk_level` | `*.Attribute` | `Risk Level` | `.creation_ts` | `High` | `N/A` |
| `.poc` | `*.Attribute` | `PoC Status` | `.creation_ts` | `Observed` | `N/A` |
| `.patch_status` | `*.Attribute` | `Patch Status` | `.creation_ts` | `Some Available` | `N/A` |
| `.interest_level` | `*.Attribute` | `Interest Level` | `.creation_ts` | `Disclosed Publicly` | `N/A` |
| `.activity_location` | `*.Attribute` | `Activity Location` | `.creation_ts` | `Underground` | `N/A` |
| `.exploit_status` | `*.Attribute` | `Exploit Status` | `.creation_ts` | `Weaponized` | `N/A` |
| `.vendor_name` | `*.Attribute` | `Target Vendor` | `.creation_ts` | `Acme` | `N/A` |
| `.product_name` | `*.Attribute` | `Target Product` | `.creation_ts` | `Acmesoft Server` | `N/A` |
| `.classification.girs[]` | `*.IntelRequirement` | `N/A` | `.creation_ts` | `1.1.1 - Some Requirement` | `If user enables the Store as Related Intel Requirement Objects option in the user-fields.` |
| `.classification.girs[]` | `*.Attribute` | `Intel Requirement` | `.creation_ts` | `1.1.1 - Some Requirement` | `Multi-valued. If user enables the Store as Attributes (Name: Intel Requirement) option, each returned GIR/PIR is retained.` |
| `.classification.girs[]` | `*.Tag` | `N/A` | `.creation_ts` | `ir-1.1.1` | `If user enables the Store as Tags (Example: ir-1.1.1) option in the user-fields.` |
| `.activity.first_seen_ts` | `*.Attribute` | `First Seen` | `.creation_ts` | `2026-01-01T00:00:00Z` | `N/A` |
| `.activity.last_seen_ts` | `*.Attribute` | `Last Seen` | `.creation_ts` | `2026-01-01T00:00:00Z` | `N/A` |
| `.last_updated_ts` | `*.Attribute` | `Last Updated` | `.creation_ts` | `2026-01-01T00:00:00Z` | `Single-value attribute; updated on later feed runs.` |
| `.cvss[].score` | `*.Attribute` | `CVSSv{VERSION} Base Score` | `.creation_ts` | `8.3` | `N/A` |

| FEED DATA PATH | THREATQ ENTITY | THREATQ OBJECT TYPE OR ATTRIBUTE KEY | PUBLISHED DATE | EXAMPLES | NOTES |
|---|---|---|---|---|---|
| .* | *.Description | N/A | .creation_ts | N/A | Various fields are used in the description to give you rich context about the vulnerability. |

# Average Feed Run

Object counts and Feed runtime are supplied as generalities only - objects returned by a provider can differ based on credential configurations and Feed runtime may vary based on system resources and load.

| METRIC | RESULT |
|---|---|
| **Runtime** | 1 minute |
| **Vulnerabilities** | 8 |
| **Vulnerability Attributes** | 120 |
| **Intel Requirements** | 9 |

# Change Log

- **Version 2.0.0**
  - Reworked the integration to improve the effectiveness and usability of ingested intelligence.
  - Updated the integration to use the new Verity471 API and added configuration parameters to support the updated API.
  - Added support for ingesting data into ThreatQ as Intel Requirement custom objects.
  - Fixed an issue that caused duplicate **Updated At** attributes to be created during ingestion.
  - Updated the minimum ThreatQ version to 6.12.1.
- **Version 1.1.2**
  - Fixed a filter mapping error that would occur when the `title` for the `Poc Link Title, POC Link URL` attribute was not present.
- **Version 1.1.1**
  - Fixed a pagination bug.
- **Version 1.1.0**
  - Added 'Save CVE Data As' user configuration parameter.
- **Version 1.0.0**
  - Initial release.