Illumio Core®

Version 23.6

REST API Developer Guide

March 2024

10000-100-23.6

Legal Notices

Product Version

PCE Version: 23.6

For the complete list of Illumio Core components compatible with Core PCE, see the Illumio Support portal (login required).

For information on Illumio software support for Standard and LTS releases, see Versions and Releases on the Illumio Support portal.

Resources

Legal information, see https://www.illumio.com/legal-information

Trademarks statements, see https://www.illumio.com/trademarks

Patent statements, see https://www.illumio.com/patents

License statements, see https://www.illumio.com/eula

Open source software utilized by the Illumio Core and their licenses, see Open Source Licensing Disclosures

Contact Information

To contact Illumio, go to https://www.illumio.com/contact-us

To contact the Illumio legal team, email us at legal@illumio.com

To contact the Illumio documentation team, email us at doc-feedback@illumio.com

Contents

Chapter 1

Overview of the Illumio REST API

This chapter contains the following topics:

The Illumio API is a RESTful API and uses JSON over HTTPS. JSON is used to encode all data transfer in both directions, so that everything sent to and everything received from the API gets encoded in JSON.

To work with Illumio API, you need to be authorized by an Illumio administrator and to have the appropriate credentials for authentication.

API Classification and Version

This chapter explains the distinction among the Illumio Public Stable, Public Experimental, and private APIs.

Public Stable APIs

The Public Stable APIs are generally available to all Illumio customers, are documented, and are stable. “Stable” means that Illumio will not introduce any further breaking changes to the API. If a breaking change is required, another version of the API will be introduced, and the previous version will continue to be supported for a minimum of six (6) months.

Public Experimental APIs

The Public Experimental APIs are generally available to all Illumio customers, are documented, but are subject to change from release to release. If you use experimental APIs, such as in scripts, be aware that some of them might change. Some of these APIs might be promoted to Public Stable at a future date, or could be made no longer available.

To help distinguish which APIs are "Public Experimental," this API guide uses orange color for headings inside these files.

Private APIs

In addition to the Public Stable or Public Experimental APIs, the Illumio Core includes additional Private APIs used by the PCE web console. The private Illumio APIs are not exposed to end-users, are not documented, or supported for use.

Illumio REST API Versions

Illumio REST APIs follow the release versions of other Illumio components, such as the PCE and VEN.

Illumio REST API Schema Files

Illumio REST API schema files follow the standard JSON schema form described at http://json-schema.org/. The file name convention is the Illumio REST API URL name with underscore rather than slashes + _ + operation + .schema.json. For example, for the login API, the payload schema file is named: `user_login_get.schema.json`.

REST API URIs

This section describes the URI syntax used with this API, which can be different depending on the REST call you are making and the types of Illumio resources on which you are operating.

API Version and Org HREF

The API version and organization HREF are two variables used in every call made to this API.

The current version of the Illumio Core REST API is version 2 (v2), which is represented in method URIs by the `[api_version]` variable. Version 1 (v1) is still supported.

> NOTE:
> The parameter tables and code examples in this document typically describe the v1 APIs, which in many cases are the same or very similar to the v2 APIs. For v2 API parameter tables, code examples, and authorization permissions, see the Illumio Core REST API Reference.

You can determine the organization HREF for the PCE when you use the login API to authenticate with the PCE and obtain a session token. In method URIs, this value is represented by the `[org_href]` variable.

In response to using the login API, the organization HREF is listed as shown, but depends on the version of the API you are using:

```
"orgs": [
    {
       "org_id": 2,
       "org_href": "/orgs/2",
```

Note that both `[api_version]` and `[org_href]` begin with a forward slash:

- `[api_version]` - `/api/v2`
- `[org_href]` - `/orgs/2`

For example, to get a collection of labels that exist inside an organization, construct the URI as follows, with the API version and the organization HREF shown in blue font:

```
GET [api_version][org_href]/labels
```

To get all of the API keys created by a specific user, construct the URI as follows, with the HREF path to the user shown in a blue font:

```
GET api/v2/orgs/1/api_keys
```

Port Number

The port number used in the code examples is 8443, which is the default. However, since the port number might be different depending on the implementation, ask your Illumio system administrator which port number to use when making calls to the Illumio Core REST API.

GET Collections URI Syntax

The base URI for Illumio REST API endpoint for GET collections:

```
GET http://[pce_hostname]:[port][api_version][org_href]/[api_endpoint]
```

> IMPORTANT:
> When making API calls, the `[pce_hostname]` or `[pce_hostname]:[port]` should not end with a forward slash ('/'). This is because `[api_version]` begins with a forward slash.

For example, the URI for getting a collection of workloads uses this syntax:

```
GET https://pce.my-company.com:8443/api/v2/orgs/1/workloads
```

In the rulesets API, you also have the ability to get all of the rules ("`sec_rules`") contained in a rule-set. The URI syntax for this operation is as follows:

```
GET http://[pce_hostname]:[port][api_version][object_href][api_endpoint]
```

For example:

```
GET [api_version][ruleset_href]/sec_rules
```

Non-GET Collections URI Syntax

For the non-GET methods of PUT, POST, and DELETE, the object HREF is listed as the endpoint, as shown here:

```
PUT [api_version][object_href]
```

The relative path of the `[api_version]` ("`api/v2/`") indicates that version 2 of the API is in use.

In the URI above, `[org_href]` is not added because it is included in the `[object_href]` string. For example, this is the `[object_href]` for a workload:

```
/orgs/2/workloads/3e3e17ce-XXXX-42b4-XXXX-1d4d3328b342
```

Another case is performing PUT, POST, or DELETE operations on the rules contained in a rule-set. The URI syntax is the same as a GET operation.

Security Policy Items and ":pversion"

This API operates on provisionable objects, which exist in either a `draft` (not provisioned) state or an `active` (provisioned) state.

Provisionable items include label groups, services, rulesets, IP lists, virtual services, firewall settings, enforcement boundaries, and virtual servers. For these objects, the URL of the API call must include the element called `:pversion`, which can be set to either `draft` or `active`.

Depending on the method, the API follows these rules:

- For GET operations – `:pversion` can be draft, active, or the ID of the security policy.
- For POST, PUT, DELETE – `:pversion` can be draft (you cannot operate on active items) or the ID if the security policy.

The URI for security policy items is as follows:

```
[pce_host][port][api_version][org_href]/sec_policy/draft/[api_endpoint]
```

REST API Limits

When making API calls, make sure that you take into account the allowed maximum number of calls per minute, returned objects, or total item count.

> IMPORTANT:Any tooling that parses the HTTP headers should be changed to allow case-insensitive header name matching in order to retain compatibility with future PCE releases. Refer to RFC 7230, section 3.2, "Header Fields," which states that field names should be case insensitive.

API Rate Limits and DOS Protection

The Illumio REST API is rate-limited and allows only a maximum of 500 requests per minute per user session or API key. The rate is set to maintain the PCE performance and service availability, and to prevent malicious attackers attempting to disrupt a service (for example, DoS attacks). If the set rate limit is reached, the call returns an HTTP error `429 Too many requests`.

Limits for Bulk Operations

In addition to the rate limits described above that are counted for all requests, the unpair workloads and delete traffic flows APIs have a rate limit of 10 calls per minute. There are also two

limits on the number of resources that can be operated on per call.

API Call and Endpoint	Request Rate Limit	Item Limit	Exposure
Unpair Workloads `PUT [api_version][org_href]/workloads/unpair`	10 per minute	1000 workloads per request	Public Stable

> NOTE:
> Illumio reserves the right to adjust the rate limit on the Illumio Secure Cloud for given endpoints at any time to ensure all clients receive a high-quality service.

Ruleset Rules Display Limit

The PCE web console supports up to 500 rules per ruleset. If you need to write more than 500 rules for a particular scope, create additional rulesets or use the Illumio Core REST API. Rulesets with more than 500 rules cannot be fully displayed in the PCE web console.

GET Collection Request Limits

By default, when you perform a synchronous GET request with this API, the maximum number of objects returned is 500.

Some GET APIs provide query parameters to help restrict the number of results, depending on the API. For example, the workloads API provides multiple query parameters for GET collections, such as `label`, `ip_address`, `policy_health`, and more.

If you wish to get more than 500 objects from a GET collection, use an asynchronous GET collection, which runs the request as an offline job. Job results can be downloaded after the job finishes.

Checking Total Item Count

To find out how many items exist for a given resource, such as whether there are more than 500 workloads in the PCE, first check the number of items using the `max_results` query parameter on a GET collection and then view the header of the response for the total item count for the resource.

If the total item count is less than 500, you can perform a regular GET collection for the results. If the total item count is more than 500, use an asynchronous GET collection.

For example, make the following GET call on a collection of workloads with the `max_results` query parameter set equal to 1, then check the header to see how many workloads exist in your organization.

> NOTE:
> When using multiple query parameters, enclose the URI, endpoint, and query_params in single quotes or double-quotes.

```
GET 'https://pce.mycompany.com:8443/api/v2/orgs/7/workloads?max_
results=1&managed=true'
```

You can check the HTTP response header for the '`X-Total-Count`' field, which indicates the total number of workloads. In this example, the total count shows 71 (highlighted in blue font), so a regular GET collection is appropriate. If the value were more than 500, then an asynchronous GET collection would be used.

```
Cache-Control →no-store
Content-Encoding →gzip
Content-Type →application/json
Date →Wed, 07 Sep 2016 14:01:00 GMT
ETag →W/"025cc8bfcXXXXXXXXX7900081e7c6cb"
Status →200 OK
Transfer-Encoding →chunked
Vary →Accept-Encoding
X-Matched-Count →71
X-Request-Id →d43a8ce9-XXXX-4453-XXXX-dde79XXX0fa8
X-Total-Count →71
```

Character Limits on Resource Names

When naming resources, the PCE has a 255 character limit for each name string. This JSON property is listed as '`name`' in the API.

For example, this 255 character limit applies when naming such things as workloads, labels, IP lists, and services

However, the PCE does not have a character limit for the description field that typically follows the name of a resource.

HTTP Requests and Responses

This section explains how to formulate HTTP requests and read HTTP responses.

HTTP Request Headers

Set an `Accept: application/json` header on all `GET` operations (optional for `DELETE` operations):

```
-H 'Accept: application/json'
```

Set a `Content-Type: application/json` header on `PUT` and `POST` operations:

```
-H 'Content-Type: application/json'
```

HTTP Request Body

Most of the parameters and data accompanying requests are contained in the body of the HTTP request. The Illumio REST API accepts JSON in the HTTP request body. No other data format is currently supported.

PUT Operations

Illumio REST API PUT operations modify a subset of attribute-value pairs for a specified resource. The attributes that are not specified in the PUT operation are left unmodified.

For example, to update a user's phone number (using the Users API) without modifying the user's address, call PUT with a request that only modifies the phone number, and only the phone number is changed.

Response Header Request-ID

The Illumio REST API provides a useful troubleshooting feature that returns a unique `Request-ID` in the HTTP response header on calls made with this API.

You can provide the `Request-ID` when opening Illumio support tickets, which are designed specifically for operations that produce errors. The `Request-ID` helps Illumio support to troubleshoot specific operations and errors

If you are using curl to make REST API calls to the PCE, you can specify the curl `-D` flag plus a file name to write the response header to a file.

Thie following example shows a curl command to get a collection of workloads that uses the `-D` flag to write the response header to a file named `temp_header`.

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/workloads -H "Accept:
application/json" -u $KEY:$TOKEN -D temp_header
```

The file contains the response header of the call (highlighted in **blue bold font**):

```
HTTP/1.1 200 OK
Server: nginx
Date: Wed, 09 Dec 2015 16:58:00 GMT
Content-Type: application/json
Content-Length: 2193032
Connection: keep-alive
Vary: Accept-Encoding
Vary: Accept-Encoding
Status: 200 OK
X-Total-Count: 1406
X-Matched-Count: 1406
ETag: "523d67cbd57b18d0e97bc8e7555142eb"
Cache-Control: max-age=0, private, must-revalidate
X-Request-Id:9722c8b5-94dc-4a50-853a-8e8f22266528
Cache-Control: no-store
Pragma: no-cache
```

Response Types

The HTTP response includes:

- An HTTP status code
- A response body that contains data in JSON format:
  - Your requested data if successful
  - An error code and message if there is an error

HTTP Status Codes – Success

The following table lists all expected success codes returned when you use the Illumio REST API:

HTTP Code	Description
`200 OK`	Successful operation where JSON body is returned
`201 Created`	Successful `POST` operation where an object was created
`204 No Content`	Operation succeeded and nothing was returned

HTTP Status Codes – Failure

All Illumio REST API methods (GET, POST, PUT, and DELETE) might fail with an error in the `400` range. The error code `400` usually means that either the resource is not available (such as trying to update a previously deleted label), or there is a mistake in the URL (such as specifying `/sh-labels` instead of `/labels`).

Other errors that might occur:

HTTP Code	Description
400 Bad Request	Something in the curl request was not correct, for example "curl **-X -i** GET" instead of "curl **-i -X** GET"
401 Authentication failure or HTTP/1.1 401 Unauthorized	For example, the user attempted to make an API call but forgot to log in, username or password were incorrect or missing, or a missing space before "**-u**"
403 Authorization failure	For example, the user is not authorized to make the call.
HTTP/1.1 403 Forbidden	For example, using the incorrect HTTP method (like using GET instead of POST), the incorrect `org_id` parameter was used
404 Invalid URL	
HTTP/1.1 404 Not Found	For example, an incorrect API version number /api/**v191**/, missing or incorrect **org_id**, `/orgs/{org_id}/`, wrong URL, or a misspelled endpoint.
404 Page not found	For example, the wrong org_id in the URI or missing blank space before an option dash, like before **-H 'Accept: application/json'**
405 Method not allowed	For example, if you are performing a POST on a resource that only allows PUT.
406 Invalid payload	The JSON request payload was constructed improperly.

Other Failure Codes

```
-bash: -H: command not found HTTP/1.1 401 Unauthorized
```

- This can be caused if more than one query parameter is used and the `URI` (including the query parameters) is not enclosed with single quotes or double quotes.
  Example:
  `'https://pce.my-company.com:8443/api/v2/orgs/2//workloads?managed=true&max_results=1'`

curl: (3) Illegal port number

- For example, a missing blank space between `-u uname:'pswd'` and the next option, for example `-H 'Accept: application/json'`.

parse error: Invalid numeric literal at line 1, column 9

- Can be caused by an incorrect curl command, for example including a path parameter that isn't allowed, like using `orgs/org_id` for an endpoint that doesn't use it. This is also a known JSON query bug caused by using `-i` in a curl command that uses json-query. To see the headers returned from the curl command, remove json-query from the curl command and use `-i`, for example "curl `-i` -X GET ..."

curl: (23) Failed writing body

- Can be caused by calling an endpoint that doesn't exist.

The property '#/' of type null did not match the following type: object in xxxxxxx.schema.json

- Can be caused by a missing or incomplete request body.

[{"token":"input_validation_error","message":"Input validation failed. Details: {The property '#/' of type NilClass did not match the following type: object in schema xxxxx.schema.json}"}]

- Is the wrong `-H` value being used? For example, is `-H 'Accept: application/json'` being used for a PUT or a POST instead of `-H 'Content-Type: application/json'`?

Request Calls Using Curl

This section explains how to use curl commands to work with Illumio APIs by defining some standard options and constants.

Curl Overview

Curl is a common command-line data transfer tool for making API calls and is especially useful in scripts written for automated tasks.

The syntax for using curl with the API for logging a user into the PCE is as follows:

```
curl -i -X <HTTP method> <uri_of_api> <header> -u $KEY:$TOKEN -Options
```

The syntax for using curl with the API for PUT operations using an API key for authentication is as follows:

```
curl -i -X PUT <URI of API> -H "Content-Type:application/json" -u $KEY:$TOKEN -d '
{ "json_property": "property_value", "json_property": "property_value" }'
```

For example:

```
curl -i -X PUT https://scp.illum.io:8443/api/v2/users/11/local_profile/password -H
"Content-Type:application/json" -u $KEY:$TOKEN -d '{ "current_password":
"NotMyReal_Old*96Password", "new_password": "NotMy*76New!pswd" }'
```

Curl-specific Options

For the curl examples provided in this API documentation, a few standard curl options are defined as follows.

The user and password to use for server authentication:

```
-u/--user <user:password>
```

For brevity, code examples typically use constants for `-u` username:'password' arguments. $TOKEN represents an authentication token (a string enclosed by single quotes to prevent it from unintentionally expanding):

```
-u $KEY:$TOKEN
```

(HTTP) Header to use when getting a web page:

```
-H/--header <header>
```

(HTTP) Specify a the HTTP method to use when communicating with the HTTP server:

```
-X/--request <command>
```

Example:

```
-X POST
```

(HTTP) Send the specified data in a POST request to the HTTP server in a way that emulates a user filling in an HTML form, and clicking **Submit:**

```
-d/--data <data>
```

Example API Call Using CURL

To get all of the API keys of a specific user using the user's session credentials:

```
curl -i -X GET https://scp.illum.io:8443/api/v2/users/11/api_keys -H "Accept:
application/json" -u $KEY:$TOKEN
```

Using Curl with json-query

When using json-query to format the output of curl commands, be aware that due to a json-query bug, this does not work with the curl `-i` option, which displays response headers. When you use the curl `-i` option, such as to see the total number of workloads when using `GET workloads`, you might get various error messages like `curl: (3) Illegal port number`. To work around this issue, remove the `-i` option and retry the curl command.

Chapter 2

Authentication and API User Permissions

This chapter contains the following topics:

To use the REST APIs, you must be an authorized Illumio user and have credentials to log into the PCE.

You get authorized to perform a specific job according to the privileges granted to you based on the role-based access control (RBAC) and implemented by the Illumio administrator.

The PCE has two types of credentials that you can use to authenticate with it and make REST API calls:

- API keys, which provide a persistent means of authenticating
- Session credentials, which provide a temporary means of authenticating

Required Permissions for API Users

To use the REST APIs, you must be an authorized Illumio user and have credentials to log into the PCE.

For authentication permissions for each REST API call, see the Illumio Core REST API Reference.

User Permissions and the API

Authentication to the PCE is based on three user roles that allow users to perform specific API operations:

- Organization owner: All `GET`, `POST`, `PUT`, and `DELETE` APIs
- Administrator: Most `GET`, `POST`, `PUT`, and `DELETE` APIs
- Read-only: `GET` only

The PCE also has two other kinds of roles:

- Unscoped: Not bound by label scopes
- Scoped: Bound by label scopes

Unscoped Roles

API Role Name	UI Role Name	Granted Access
`owner`	Global Organization Owner	Perform all actions: Add, edit, or delete any resource, organization setting, or user account.
`admin`	Global Administrator	Perform all actions except cannot change organization setting and cannot perform user management tasks.
`read_only`	Global Read Only	View any resource or organization setting. Cannot perform any operations.
`global_object_provisioner`	Global Policy Object Provisioner	Provision rules containing IP lists, services, and label groups, and manage security settings. Cannot provision rulesets, virtual services, or virtual servers, or add, modify, or delete existing policy items.

Scoped Roles

API Role Name	UI Role Name	Granted Access
`ruleset_manager`	Full Ruleset Manager	Add, edit, and delete all rulesets within the specified scope. Add, edit, and delete rules when the provider matches the specified scope. The rule consumer can match any scope.
`limited_ruleset_manager`	Limited Ruleset Manager	Add, edit, and delete all rulesets within the specified scope. Add, edit, and delete rules when the provider and consumer match the specified scope. Ruleset Managers with limited privileges cannot manage rules that use IP lists, user groups, label groups, or iptables rules as con-

API Role Name	UI Role Name	Granted Access
		sumers, or rules that allow internet connectivity.
`ruleset_provisioner`	Ruleset Provisioner	Provision rulesets within a specified scope. This role cannot provision virtual servers, virtual services, SecureConnect gateways, security settings, IP list, services, or label groups

Session Credentials

While API Keys provide a persistent means of authenticating with the PCE, session credentials provide a temporary means of authenticating so you can make Illumio REST API calls.

> IMPORTANT:Any tooling that parses the HTTP headers should be changed to allow case-insensitive header name matching in order to retain compatibility with future PCE releases. Refer to RFC 7230, section 3.2, "Header Fields," which states that field names should be case insensitive.

Choose a session token or an API key depending on your programming needs.

Session Credentials and Tokens

When you create session credentials, an `auth_username` and `session token` are returned that function as a temporary username and password for making API calls.

Session credentials are used to make all Illumio REST API calls that require authentication and are composed of an `auth_username` and a token. They expire after not being used for 30 minutes and reset for another 30 minutes if used within the 30-minute window.

The session token expires after 10 minutes of inactivity.

When to Use a Session Token

An `auth_username` and session token are useful for a one-time use of the API or for testing the API. To write a script that performs a one-time use of the API with a session token, use the Login API to create the `auth_username` and session token. Use those credentials for making other API calls in the script, and then once the script has run, the session token immediately expires when the user logs out.

What Does a Session Token Look Like?

When you authenticate with the PCE using the Login API, the response returns the credentials needed to make other API calls:

- Your username: `"auth_username": user_3`
- Your session token: `"session_token":"xxxxxxx563199f92af7b705ddca26854205b5233"`

To use the Illumio REST API:

1. Call `login_users/authenticate` using the e-mail address and password you used to create your PCE account to obtain an ***authentication token***.

   > NOTE:
   > The authorization token expires after 30 seconds, so have the next call formed and ready to paste onto the terminal window before calling `login_users/authenticate`.

2. Call `users/login` with the authentication token to obtain temporary session credentials.

Authenticate to Login Service

Before you can use the Illumio REST API to access the PCE, you need to use the Login Users API to authenticate with the Illumio Login Service and obtain an authentication token. This authentication token expires in 30 seconds.

The URL for the Illumio Login Service for Illumio Core Cloud users is:

- Login Server: `https://login.illum.io:443`
- PCE: `scp1.illum.io`

For SaaS customers the PCE URL can be different based upon their SaaS PCE:

- SCP1 & SCP2 (US)
- SCP3 UK only
- SCP4 APAC
- SCP5 (EMEA)

If you have deployed the PCE as software, then the hostname for the PCE is the value you defined for the '`pce_fqdn`' parameter in the `runtime_env.yml` file.

Once obtained, you can then pass the authentication token to the PCE you want to access using the Login API. Once you have authenticated with the PCE and obtained a session token, you can make other API calls or Create a User-based API Key for persistent API access to the PCE.

URI to Authenticate with the Login Service

```
POST [api_version]/login_users/authenticate
```

Create an Authentication Token for the Login Service

To create an authentication token and authenticate with the Login Service, you need to specify the Fully Qualified Domain Name (FQDN) of the PCE you want to access in the call.

Parameter	Description	Type	Required
`pce_fqdn`	Fully Qualified Domain Name (FQDN) of the PCE If you are using Illumio Core Cloud, then the FQDN for the PCE is `scp1.illum.io`. If you have deployed the PCE virtual appliance in your own network, then use the FQDN specified during installation of the PCE virtual appliance.	String	Yes

Curl Commands for Authentication

When you received your invitation, you used an e-mail and password to create your PCE account. Use these credentials now to make a call to authenticate.

If you haven't received an invitation, contact your Illumio administrator.

Example (local users only, use SAML ID for remote users):

- `joe_user@example.com` (username)
- `password` (password)

You also need the FQDN of the Login Server plus the FQDN of the PCE host you want to access:

- The Login Server FQDN for Illumio Core Cloud users is `https://login.illum.io:443`
- The PCE FQDN is `scp1.illum.io`

> NOTE:
> The authorization token that is returned (`auth_token`) expires after being idle for 30 seconds, so be ready to call `GET users/login` to create session credentials immediately after making the call to `login_users/authenticate`.

Retrieve a Token

This curl example shows how SaaS local users can use the Illumio Login Service (SAML ID for Remote Users)

```
curl -i -X POST https://login.illum.io:443/api/v2/login_users/authenticate?pce_
fqdn=scp1.illum.io -u joe_user@example.com:'password' -H "Content-Type:
application/json"
```

Illumio on-premises solutions do not use a login server, so the curl command will look like this:

```
curl -i -X POST -u joe_user@my-company.com:password https://pce.my-
company.com:8443/api/v2/login_users/authenticate?pce_fqdn=pce.my-company.com -H
"Content-Type: application/json"
```

Response Body to Authenticate with Login Service

The response for the Login Users API is an authentication token (in blue font):

```
{ "auth_token":"xxxxxxxxxxxxxxxxxxxxxxw89QutJ5WLntqz5jUrI2guA1rZJXKfcbwuF" }
```

Use Login API to Create Session Credentials

Unless you're using persistent API credentials, every time you want to access the Illumio REST API, you must authenticate with the PCE using an *auth username* and a *session token*. To create these session credentials, call `GET /users/login` with the authentication token previously returned by a call to `POST /login_users/authenticate`.

URI

```
GET [api_version]/users/login
```

Parameters

Login Service authentication token you obtained using the Login Users API.

Login Users API JSON Schema

This API uses the Illumio Core schema `users_login_get.schema.json`.

Create Session Token

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/users/login -H
"Authorization: Token token=ntqz5jUrI2guA1XzUiLCJlbmMiOiJBMTI4Q0JDLUhZJ"
```

Response Body

`GET /users/login` returns a temporary `auth_username` and `session_tokes` shown below in blue. These are used in the curl examples as `$KEY:$TOKEN` respectively (if you're not using persistent API credentials).

Example: `-u user_4:'xxxxxxxx628f5773c47b72dbcd437b4a10d85a06a'`


```
{
    "full_name": "Buford T. Justice",
    "local": true,
    "type": "local",
    "href": "/users/4",
    "auth_username": "user_4",
    "inactivity_expiration_minutes": 10,
    "start": "2017-10-12 16:49:49 UTC",
    "time_zone": "America/Los_Angeles",
    "last_login_ip_address": "209.37.96.18",
    "last_login_on": "2020-10-12T16:49:49.000Z",
    "certificate": {
        "expiration": "2020-11-27T03:09:00.000Z",
        "generated": false
    },
    "login_url": "https://devtest166.ilabs.io:8443/login",
    "orgs": [
        {
            "org_id": 1,
            "org_href": "/orgs/1",
            "display_name": "illum.io",
            "role_scopes": [
                {
                    "role": {
                        "href": "/orgs/1/roles/owner"
                    },
                    "scope": [],
                    "href": "/orgs/1/users/4/role_scopes/4"
                }
            ]
        }
    ],
```

```
    "session_token": "xxxxxxxx628f5773c47b72dbcd437b4a10d85a0",
    "version_tag": "60.1.0-9701f78bef46f521e3d6dd98f70cd8c220940885",
    "version_date": "Tue Sep 12 11:12:46 2020 -0700",
    "product_version": {
        "version": "17.1.1",
        "build": "6168",
        "long_display": "17.1.1-6168",
        "short_display": "17.1.1"
    }
}
```

API Call Using Session Credentials

Once you obtain an `auth_username` and session token from the PCE, you use them to make API calls.

For example, if you wanted to use this session token to get a collection of labels in an organization using the Labels API, the curl command can be written as shown below, using the following authentication:

- auth_username: `user_3`
- Session Token: `xxxxxxx563199f92af7b705ddca26854205b5233`

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/3/labels -H "Accept:
application/json" -u user4:'xxxxxxxx628f5773c47b72dbcd437b4a10d85'
```

Optional Features

This API was introduced to help avoid issues with misconfigured DNS, which can cause problems with VEN connectivity. Likewise, misconfiguring DHCP can cause problems with IP addresses.

You need a key to invoke /optional_features API to enable `editable_dns_client_rule` or `editable_dhcp_client_rule`. Such a key involves a portion that is tightly controlled so that it cannot be randomly generated.

Once the key is generated, it cannot be used in more than one place, which means that an API call provided to customer #1 cannot be replayed at customer #2 who must request their own key.

An example of the generated key:

```
secret =
               '...'   # value embedded in code

data = Base64.strict_encode64({
       'pce_fqdn'   => Illumio::RuntimeEnvironment.pce_fqdn,
       'org_id'   => xorg_id,
       'optional_feature'   =>
       'editable_dns_client_rule' ,
       'not_valid_after'   => Time.now.utc.iso8601
})

key = data + OpenSSL::HMAC.hexdigest( 'SHA256' , secret, data)
```

Optional Feature Schema: `optional_feature.schema.json`

```
{
       "$schema": "http://json-schema.org/draft-04/schema#",
       "type": "object",
       "additionalProperties": false,
       "description": "PCE Feature",
       "required": [
              "name",
              "enabled"
              ],
       "properties": {
              "name": {
                     "type": "string",
                     "description": "The name of the feature"
              },
              "preview": {
                     "type": "boolean",
                     "description": "Is this a preview feature"
              },
              "enabled": {
                     "type": "boolean",
                     "description": "Is this feature enabled"
              }
       }
}
```

Get the optional features collection: `optional_features_get`

```
{
	"$schema": "http://json-schema.org/draft-04/schema#",
	"type": "array",
	"items": {
			"$ref": "optional_feature.schema.json"
		}
	}
```

Set the optional features for an organization: `optional_features_put`

```
{
	"$schema": "http://json-schema.org/draft-04/schema#",
	"type": "array",
	"items": {
	    "oneOf": [
	    {
	    "type": "object",
	    "additionalProperties": false,
	    "required": [
	        "name",
	        "enabled"
	    ],
	    "properties": {
	        "name": {
	        "description": "Name of the feature",
	        "type": "string",
	        "enum": [
	                "ip_forwarding_firewall_setting",
	                "ui_analytics",
	                "illumination_classic"
	                ]
	        },
	    "enabled": {
	        "description": "Enable or disable this feature",
	        "type": "boolean"
	        }
	    }
```

```
            },
                {
            "type": "object",
            "additionalProperties": false,
                "required": [
                "name",
                "enabled"
                ],
            "properties": {
                "name": {
                        "description": "Name of the feature",
                        "type": "string",
                        "enum": [
                                "editable_dns_client_rule",
                                "editable_dhcp_client_rule"
                        ]
                        },
                "enabled": {
                        "description": "Enable or disable this feature",
                        "type": "boolean"
                        },
                "key": {
                        "description": "Key required to enable the feature. Contact Illumio Support
more details.",
                        "type": "string"
                }
            }
        }
```

Setting Optional Features

Analytics opt-out

The property `configurable_label_dimension` was added so that the UI users can determine if an organization has enabled the user analytics.

Analytics is opt-in by default. If it has been disabled, the UI shows not to track analytics for that organization.

To set or clear the optional analytics feature, use:

```
{
name: "ui_analytics", enabled: false|true
}
```

Illumination Classic opt-out

The property `illumination_classic` is added to enable or disable use of the that feature.

To set or clear the optional Illumination Classic feature, use:

```
{
name: "illumination_classic", enabled: false|true
}
```

Label-Based Network Detection

The APIs

- `POST /api/v2/orgs/{org_id}/networks`
- `PUT /api/v2/orgs/{org_id}/networks/:network_id`

require that one of the following optional features is enabled :

- `label_based_network_detection`
- `cidr_network_detection_enabled`

In addition, both APIs are implementing the input validation on payload content:

- If the cidrs field is provided, the optional feature `cidr_network_detection_enabled` must be set.
- If the scopes field is provided, the optional feature `label_based_network_detection` must be enabled.

The example response for the API `optional_features_put` with the `label_based_network_detection` enabled:

```
            "illumination_classic",
               "ransomware_readiness_dashboard",
               "per_rule_flow_log_setting",
```

```
            "lightning_default",
            "label_based_network_detection"
            ]
      },
      "enabled": {
```

labels_editing_warning_for_enforcement_mode

In releases 23.2.10 and 23.4, for the required property `name` a new optional feature flag for label editing was added: `labels_editing_warning_for_enforcement_mode`.

To enable or disable this flag, use the following CURL command:

```
curl -u ${your_api_key}: ${your_api_secret} -H "Content-Type: application/json" -X
PUT -d '[{"name":"labels_editing_warning_for_enforcement_mode","enabled":true}]'
https://${your_pce_server}:8443/api/v2/orgs/${your_ord_id}/optional_features
```

windows_outbound_process_enforcement

In release 23.5, an optional feature flag for Windows outbound process was added: `windows_outbound_process_enforcement`.

This feature flag can be enabled or disabled using the following CURL command:

```
curl -u ${your_api_key}: ${your_api_secret} -H "Content-Type:
application/json" -X PUT -d '[{"name":"windows_outbound_process_
enforcement","enabled":true}]' https://${your_pce_
server}:8443/api/v2/orgs/${your_ord_id}/optional_features
```

where you can define the part of the command: `"enabled":true` or `"enabled":false`.

API Keys

API keys provide a persistent means of authenticating with the PCE and are recommended for scriptwriting.

API keys can be used to make API calls to access the PCE. The API Key usage is limited to Servers & Endpoints (Illumio Core) APIs.

All API keys are organization-based.

There are two categories of API keys:

- User-Based API Keys

  These keys are based on specific user accounts so that users can make API calls to the PCE.

- Service Account-based API Keys

  These API keys are based on a service account instead of a user account.

Working with API Keys

When you create an API key, you receive an `api_username` and `secret`, which function as the username and password for making API calls. An API key is permanent and does not expire (except when deleted).

> IMPORTANT:
> Any tooling that parses the HTTP headers should be changed to allow case-insensitive header name matching in order to retain compatibility with future PCE releases. Refer to RFC 7230, section 3.2, "Header Fields," which states that field names should be case insensitive.

Use API keys to write scripts that run automatically, without requiring a human user to authenticate the API call. Unless you are a read-only user, you can create multiple API keys and make API calls in your scripts.

You can also create different API keys for different functions. For example, you might use an API key for scripting automatic workload pairing, and another API key for collecting system events from Illumio.

When you create an API key, the response returns both the `auth_username` and the secret needed for authenticating other API calls:

- API username: `"auth_username":"api_xxxxxxxxxx29"` (represented in the code examples in this document as `$KEY`)
- API key secret: `"secret":"xxxxxxx5048a6a85ce846a706e134e-f1d4bf2ac1f253b84c1bf8df6b83c70d95"` (represented in the code examples in this document as `$TOKEN`)

Get a Collection of all API keys

You can now get a list of all API keys, both user-based and service account-based.

To query API keys regardless of their type, use this API:

```
GET /api/v2/orgs/:xorg_id/api_keys
```

Query Parameters for API Keys

Some parameters have been renamed or deprecated to allow differentiation between the type `user`and `service_account`:

- query parameter `name`is retained for the type `service_account`
- query parameter `name`is changed to `username` for the type `user`
- query parameter `service_account_name` was deprecated and consolidated to `name`
- query parameter `api_key_name` was deprecated and removed as not needed

Special Characters in API Calls

If a `username`or `name` in an API call contains special characters, these have to be encoded for the call to be successful.

For example, for a service account name `sa&1`, instead of

```
api/v2/orgs/1/api_keys?type=service_account&name=sa&1
```

enter the call as

```
api/v2/orgs/1/api_keys?type=service_account&name=sa%261
```

User-Based API Keys

This Public Stable API allows you to manage API keys and make API calls to the PCE.

Working with User-based API Keys

User API Key Methods

Functionality	HTTP	URI
Get a collection of API keys	`GET`	`[api_version][user_href]/api_keys`
Get an individual API key	`GET`	`[api_version][api_key_href]`
Create an API key	`POST`	`[api_version][user_href]/api_keys`
Update an API key	`PUT`	`[api_version][api_key_href]`
Delete an API key	`DELETE`	`[api_version][api_key_href]`

Parameters

Parameter	Description	Type
`user_id`	The user ID in the form of an HREF (e.g., '`users/6`') of the user who created the API key.	String
`api_key_id`	This is the actual API key ID. Used only for DELETE.	String
`org_id`	Organization ID	Integer
`max_results`	Maximum number of api keys to return.	Integer
`username`	Username of the user to filter by	String
`name`	(POST, PUT, GET) The key name - just a label to be used	String
`role`	Role URI (JSON-encoded string) to filter on	Sring
`state`	State of api keys - active or expired	Sring
`type`	Type of principal - User or Service Account	String

Response Properties

Property	Description	Type
`key_id`	This is the actual API key ID. Use this query parameter only for a GET instance call.	String
`auth_username`	Username required for authentication	String
`created_at`	Timestamp when this key was first created (RFC 3339)	String
`name`	The key name - just a label to be used	Sring
`href`	URI of the key	date/time

List User-Based API Keys

When you GET an individual API key or a collection of API keys, the response only returns those API keys created by the user that has authenticated with the PCE for the session.

This API gets one API key or a collection of API keys that a specific user has created. To get a single API key, you need to know the API key's URI, which is returned in the form of an HREF path when you create an API key, as well as the HREF of the user who created the key.

You can query the user API keys as follows:

```
GET /api/v2/orgs/:xorg_id/api_keys?type=user
```

You can also query the user-based API keys based on their expiration:

```
GET /api/v2/orgs/:xorg_id/api_keys?type=user&state=active
```

```
GET /api/v2/orgs/:xorg_id/api_keys?type=user&state=expired
```

```
GET [api_version][user_href]/api_keys
```

Curl Command to Get a Key

The API key is identified in the form of an HREF path property:

```
"/users/11/api_keys/a034248fbcdd60b4"
```

```
 curl -i -X GET https://pce.my-company.com:8443/api/v2/users/11/api_
keys/a034248fbcdd60b4 -H "Accept: application/json" -u $KEY:$TOKEN
```

Get a Collection of Keys

To use an API key, store the key and secret safely. Anyone with access to both has access to your organization's API.

Due to security concerns, external users are not allowed to create an API Key even if their roles allow it.

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/users/11/api_keys -H
"Accept: application/json" -u $KEY:$TOKEN
```

Response Body

An API key is represented by its HREF path, as shown here:

```
/users/29/api_keys/1e9bb1787883639d5
```

For example:

```
[
  {
```

```
    "href": "/users/29/api_keys/1e9bb1787883639d5",
    "key_id": "1e9bb1787883639d5",
    "auth_username": "api_1e9bb1787883639d5",
    "created_at": "2020-01-27T01:30:22.274Z",
    "name": "my_api_key",
    "description": "my_scripting_key"
  },
  {
    "href": "/users/29/api_keys/1793df73a99255f7e",
    "key_id": "1793df73a99255f7e",
    "auth_username": "api_1793df73a99255f7e",
    "created_at": "2016-03-14T16:20:43.603Z",
    "name": "MyKey",
    "description": "My Special Key"
  }
]
```


Get All Labels with a Key

If you use an API key to get a collection of labels in an organization, and your API key uses these credentials:

- `api_xxxxxxx64fcee809` is the API key
- `xxxxxxx09137412532289d6ecd10bc89c6b1f608c9a85482e7a573` is the secret (API key password)

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/3/labels -H "Accept: application/json" -u api_xxxxxxx64fcee809:'xxxxxxx09137412532289d6ecd10bc89c6b1f608c9a85482e7a573'
```

Session and persistent (API key) credentials are represented in this document as the constants `$KEY:$TOKEN` (with no spaces).

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/3/labels -H "Accept: application/json" -u $KEY:$TOKEN
```

Create a User-based API Key

This API creates a unique API key and returns an API key ID and secret, which you can use to get, update, or delete the key, and to make other API calls.

To create an API key, you first need to authenticate either using a session token or another API key. To obtain a session token, use the Users API and authenticate with the PCE. You will receive your user ID, user HREF, and a session token that you can use when you call this API to create an API key.

> IMPORTANT:
> If you use an API key, safely store the key and the secret. Anyone with access to both will have access to the API for your organization.

> IMPORTANT:
> Due to security concerns, external users are not allowed to create an API Key even if their roles allow it.

URI

```
POST [api_version][user_href]/api_keys
```

An example user HREF looks like this:

```
/users/99
```

```
{
     "name": "my_api_key",
     "description": "my_scripting_key"
}
```

To create a user-based API key

In this curl command, the user authentication (`-u`) uses the session credentials returned from calling the Login API to log in a user. The API key is passed as a JSON object formatted inside of double quotes in the command:

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/users/14/api_keys -H
"Content-Type:application/json" -u user_14:'xxxxxxx563199f92af7b705ddca2685'-d "{
"name":"my_api_key","description":"my_scripting_key" }"
```

Response Body

This example shows the response from creating an API key, which you can use for making other API calls. These values do not expire. The `auth_username` functions as the username, and the secret functions as the password when making other API calls:

```
{
     key_id: "xxxxxxx6654188229"
     secret: "xxxxxxxxxxxa6a85ce846a706e134ef1d4bf2ac1f253b84c1bf8df6b83c70d95"
     auth_username: api_xxxxxxx6654188229
}
```

These values can now be use authenticate with the API as follows:

- Username: `api_xxxxxxx29api_xxxxxxx6654188229`
- Password: `xxxxxxxxxxxa6a85ce846a706e134ef1d4bf2ac1f253b84c1bf8df6b83c70d95`

Use the Console

You can also create API keys in the Illumio Console with the **User** Menu (at the logged-in user's ID, in the upper right corner of the Console UI).

1. In the drop-down **User** menu, select **My API Keys**.

   A list of configured API keys is displayed.
   If no API keys are configured, the message "No API Keys" is displayed.
2. To add a new API key, click **Add**.
3. In the **Create API Key** pop-up window, enter a name for the API key in the Name field. Optionally, enter a description in the Description field.
4. Click **Save** to save your API key or click **Cancel** to close the pop-up window without saving your changes.
5. When the **API Key Created** window appears, click the **>** button next to "Show credentials" to display the credentials for your API key.

   The following information is displayed:

   - **Key ID**: The unique ID of the API key
   - **Authentication Username**: The username that authenticates the API calls
   - **Secret**: The password for the API key
6. Click **Download Credentials** to download the credentials as a text file. Make sure that you have saved the credential information before clicking **Done**.

After you click **Done**, the API Keys page displays a summary of your new API key, including the following information:

- Name
- Description
- Key ID
- Authentication Username
- Created On date

> NOTE:
> The credential information is displayed only once. Make sure to save it in a secure location because it is used to access the API for your organization. If the credential information is lost, you must create a new API key.

Update a User-Based API Key

This API allows you to update an API key name or description. To make this call, you need the API key URI, which is returned in the form of an HREF path when you Create a User-based API Key.

Update a User-based API Key

```
PUT [api_version][api_key_href
```

Example Payload

```
{
     "name": "my_api_key1",
     "description": "my_scripting_key v2"
}
```

```
 curl -i -X POST https://pce.my-company.com:8443/api/v2/users/99/api_
keys/a034248fbcdd60b4 -H "Content-Type:application/json" -u $KEY:$TOKEN -d '{
"name": "my_key_1", "description": "my_scripting_key v2" }'
```

Delete a User-based API Key

To delete an API key, you need the unique API key ID, which is returned in the form of an HREF path property when you either create a new API key, or when you get a single or a collection of API keys.

URI to Delete a User-based API Key

```
DELETE [api_version][api_key_href]
```

Service Account-based API Keys

Service account-based APIs allow for the creation and management of API keys based on a service account. You can manage the expiration of service account-based API keys.

> NOTE:**NOTE**: When Service Accounts were introduced, the following restriction was explicitly added: a Service Account cannot be used to operate on service accounts and on user-related resources. In release 23.4.0, this restriction has been removed: a Service Account `api_key` can be used to perform most of the operations like the user `api_key`, except for API's that require user context. There were no changes to the APIs to support this restriction removal.

Service accounts are always organization-based and specific to a PCE. While creating a service account, users create their permissions, and an `api_key` implicitly gets created. Deleting a service account removes its permissions and all associated API keys.

Methods

Functionality	HTTP	URI
List all service account API keys	`GET`	`[api_version][org_href]/api_keys?type=service_account`
To retrieve a specific service account	`GET`	`[api_version][org_href]/service_accounts/:service_account_id`
To retrieve all API keys, regardless of the account	`GET`	`api_keys`
Create a new service account API key	`POST`	`[api_version][org_href]/service_accounts`
To create a new service account API key after performing the required validation	`POST`	`[api_version][org_href]/service_accounts/:service_account_id/api_keys`
Update a service account	`PUT`	`[api_version][org_href]/service_accounts/:service_account_id`
Delete a service account API key	`DELETE`	`[api_version][org_href]/service_accounts/:service_account_id/api_keys/:key_id`

Functionality	HTTP	URI
Delete a service account including any associated API keys	DELETE	[api_version][org_href]/service_account-s/:service_account_id

Parameters for Service-based API Keys

Parameter	Description	Type	Required
org_id	Organization ID.(GET, POST, PUT, DELETE)	Integer	Yes
max_results	Maximum number of service accounts to return (GET)	Integer	No
name	Name of service account to filter by	String	No(GET) Yes (POST)
role	Role URI (JSON-encoded string) to filter on(GET)	String	No
service_ account_id	Service account UUID (GET account info, DELETE, PUT, POST)	String	Yes
api_key_id	API Key ID (DELETE)	String	Yes

Response Properties for Service-based API Keys

Property	Description	Type	Required
name	Service account name	String	Yes
href			Yes
created at	Timestamp when this service account was first created (RFC 3339)	date/time	Yes
updated_at	Timestamp when this service account was last updated	date/time	Yes
created_by	User who originally created this service account Required: href	Object	Yes
updated_by	User who last updated this service account	Struing	Yes
permissions	List of permissions: required: role, scope	Array	Yes
role	Reference to common/orgs_roles.schema.json		
scope	Reference to org_scope.schema.json		
api_keys	List of associated api_keys		Yes

Property	Description	Type	Required
	Reference to `api_keys_get.schema.json`		
`api_key`	required: "expires_in_seconds" "type": "integer", "minimum": -1, "maximum": 2147483647	Object	
`access_restriction`	Access restriction assigned to the keys created under this service_account	Object, Null	No
`href`	URI of service_account	String	
`api_keys`	List of associated api_keys Reference to `api_keys_get.schema.json`	Array	No
`expires_in_seconds`	Validity of the api_key, in seconds "type": "integer", "minimum": -1, "maximum": 2147483647	String	No
`last_login_on`	Timestamp when this key was last used	date/time	
`account`	required: "href": Associated identity "type": Type of the account "name": Name of the account		 Yes Yes Yes
`service_account_id`	Service account UUID (GET account info, DELETE, PUT, POST)	String	No
`api_key_id`	API Key ID (DELETE)	String	Yes

Parameters for api_keys_get (all API keys)

Parameter	Description	Type	Required
`key_id`	Key ID	String	Yes
`auth_username`	Username required for authentication	String	Yes

Parameter	Description	Type	Required
name	The key name - just a label to be used	String	Yes
role	Role URI (JSON-encoded string) to filter on(GET)	String	No
service_account_id	Service account UUID (GET account info, DELETE, PUT, POST)	String	No
api_key_id	API Key ID (DELETE)	String	No

Response Properties for api_keys_get (all API keys)

Property	Description	Type	Required
key_id	Key ID	String	Yes
auth_username	Username required for authentication	String	Yes
created at	Timestamp when this service account was first created (RFC 3339)	date/time	Yes
name	Service account name	String	Yes Yes
href	URI of the key	String	Yes
state	State of the api_key	String	No
expires_in_seconds	Validity of the api_key, in seconds	String	No
created_by	User who originally created this api key	String	No
last_login_on	Timestamp when this key was last used Examle: "last_login_on": "2023-04-22T03:54:25Z"	date/time	
account	required: "href": Associated identity "type": Type of the account "name": Name of the account		 Yes Yes Yes
access_restriction	Access restriction assigned to the keys created under this service_account	Object, Null	

Property	Description	Type	Required
`permissions`	List of permissions: required: `role`, `scope`	Array	Yes
`role`	Reference to `common/orgs_roles.schema.json`		
`scope`	Reference to `org_scope.schema.json`		

Service account-based API keys' expiration is defined by owners who can specify the default expiration time.

The key expiration time is specified with a default value specified in the settings, where the expiration date of an existing API key cannot be modified.

When an API request is authenticated by an expired API key, the request is rejected and the audit event triggered by this failure includes the API key's Key ID and the expired status of the API Key. The details also include the expiration date and the `last_used_at` date.

Query Keys by Expiration

To retrieve the API keys based on the expiration (active or expired) used these APIs:

```
GET /api/v2/orgs/:xorg_id/api_keys?type=service_account&state=expired
```

This query lists all expired API keys.

```
GET /api/v2/orgs/:xorg_id/api_keys?type=service_account&state=active
```

This query lists all active API keys.

Settings

Settings for service account-based API keys specify the default expiration period for service account keys and retention period for expired keys.

The Public Experimental APIs that manage API keys settings are based on the role of the organization administrator (`this_org_admin`) and are ad follows:

- ```
  GET /api/v1/orgs/:xorg_id/settings
  ```

  Support for viewing api_key settings for an organization.
  ```

- `PUT /api/v1/orgs/:xorg_id/setting`

Support for updating `api_key` settings for an organization.
API key expiration is now set between `-1` and `2147483647` seconds and expired key retention is a minimum of `0` seconds.

The `settings_put.schema.json` schema looks as follows:

```
{
       "$schema": "http://json-schema.org/draft-04/schema#",
       "type": "object",
       "additionalProperties": false,
       "properties": {
          "max_api_key_expiration_in_seconds": {
               "description": "Validity of api_key in seconds; -1 specifies api_keys never
expire",
               "type": "integer",
               "minimum": -1,
               "default": 7776000,
               "maximum": 2147483647
           },
         "expired_api_keys_retention_in_seconds": {
               "description": "Retention of expired api_keys in the database",
               "type": "integer",
               "default": 7776000,
               "minimum": 0,
               "maximum": 31536000
           },
          "advanced_ruleset_display": {
               "description": "When true, the UI will display rulesets in advanced mode.
This means that scopes will be displayed for any unscoped rulesets, including
newly added rulesets.",
               "type": "boolean",
               "default": true
               },
           "ven_maintenance_token_required": {
               "description": "Identifies if the tampering protection for the \'EN and
endpoints is enabled or not.",
               "type": "boolean",
```

```
                "default": false
                }
        }
}
```

The new property `ven_maintenance_token` identifies if the tampering protection for the VEN and endpoints is enabled. The default is "not enabled".

Create a new Service Account API Key

Request

```
{
        "name": "key3",
        "description": "testing key 3",
        "access_restriction": {
                "href": "/orgs/1/access_restrictions/2"
                },
        "permissions": [
                {
                "role": { "href": "/orgs/1/roles/ruleset_manager"},
                "scope": [
                {
                        "label": {
                        "href": "/orgs/1/labels/9",
                        "key": "env",
                        "value": "Development"
                }
        }
        ]
},
{
                "role": { "href": "/orgs/1/roles/owner" },
                "scope":  []
        }
],
        "api_key": {
        "expires_in_seconds": 86400
```

```
        }
}
```

Response

```
              {
        "name": "service_account1",
        "description": "testing service_account",
        "href": "/orgs/1/service_accounts/33ed7e04-9b25-4c9a-a031-a6b1bd437807",
        "access_restriction": {
                "href": "/orgs/1/access_restrictions/2"
                },
        "permissions": [
                {
                "href": "/orgs/1/permissions/84e5541f-3349-41c9-8fdb-9756faf96baa",
                "role": {"href": "/orgs/1/roles/ruleset_manager"
                        },
                "scope": [
                        {
                        "label": {
                        "href": "/orgs/1/labels/9"
                }
        }
        ]
},
{
                "role": {
                        "href": "/orgs/1/roles/owner"
                },
                "scope": []
                }
        ],
        "api_key": {
                "auth_username": "api_135c247aa6e3b654e",
        "secret": "ab80cc497f7556e0cd72703c5229d814322c301d14d2d8d8c7060d516990097b"
        }
}
```

REST API Users

This Public Stable API allows you to log your User into the PCE so you can get a session token to access other Illumio Core REST API calls. This API is your starting point for interacting with the PCE using the REST API.

Users API Methods

Functionality	HTTP	URI
Authenticate to the Illumio Login Service and obtain a single-use authentication token.	POST	[api_version]/login_users/authenticate
Create a new user.	POST	[api_version][users]
Log in a user and obtain a session token.	GET	[api_version]/users/login
Log out a user and destroy the session token.	PUT	[api_version][user_href]/logout
Get a user's information.	GET	[api_version][user_href]
Update user's information.	PUT	[api_version][user_href]
Change a user's password (a local, non-SSO user).	PUT	[api_version]/login_users/[user_href]/password

Log Into the PCE

URI to Log In User

```
GET [api_version]/users/login
```

For step-by-step instructions about how to authenticate to the PCE and use `GET /users/login` in conjunction with other methods, see Authentication and API User Permissions.

Log Out and Destroy Credentials

This API logs users out of the PCE and destroys the temporary session credentials used to log them in.

> NOTE:
> This `PUT /logout` call is not used with persistent API credentials.

URI to Log Out a User

```
PUT [user_href]/logout
```

Request Body

The request body is an empty JSON object.

```
{}
```

Log Out a User

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/authentication_
services/password_policy -H "Content-Type: application/json" -u $KEY:$TOKEN -d '
{"require_type_symbol": true, "expire_time_days": 90}
```

Get User Information

This API gets specific information about a user, such as when a user logged into the Illumio PCE, the IP address from where the user logged in, the user's name, and password.

URI to Get User Information

```
GET [user_href]
```

Properties

Property	Description	Type
`href`	URI of the user.	String
`username`	Username used for authentication.	String
`last_login_on`	When the user logged on.	String
`last_login_ip_address`	The IP address of the system where the user has logged into the PCE.	String
`login_count`	The number of times the user has logged in.	Integer
`full_name`	Full name of a user as listed in the PCE web console.	String
`time_zone`	User's timezone IANA Region name.	String
`locked`	Indicates if a user account is locked or not. True = locked.	Boolean
`effective_groups`	A list of group names to which the user belongs.	String
`local profile`	Local user profile	Object
`updated_at`	Date when user account information was last updated in the system.	String
`created_at`	Date when the user account was created in the system.	String

Property	Description	Type
type	Indicates if the user account is authenticated by the PCE (local) or by a third party SAML-based identity management system (external)	String
one_time_password	The time-based one-time password for two-factor authentication. This password is required in addition to username and password for authentication.	String

Request Example

```
GET https://pce.my-company.com:8443/api/v2/users/5
```

Get a User's Information

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/users/14 -H "Accept:
application/json" -u $KEY:$TOKEN
```

Response Body

In this response, the user is represented in the system by an HREF path property ("href": "/users/14") that can be used when you want to update the user information.

```
{
    "href": "/users/14",
    "type": "local",
    "effective_groups": [],
    "id": 14,
    "username": "joe.user@pce.my-company.com",
    "full_name": "Ralph W. Emerson",
    "time_zone": "America/Los_Angeles",
    "locked": false,
    "login_count": 75,
    "last_login_ip_address": "xxx.37.96.18",
    "last_login_on": "2020-08-17T15:42:25.732Z",
    "local_profile": {
        "pending_invitation": false
    },
    "created_at": "2019-10-26T05:24:08.735Z",
    "updated_at": "2019-08-17T15:55:40.130Z"
}
```

Create a New User

This API creates a new local user.

URI to Create a New User

```
POST [api_version][users]
```

Request Body

Property	Description	Type	Required
`full_name`	User's full name.	String	No
`username`	`username` is an e-mail address such as `user@example.com`	String	Yes
`type`	User's type, such as user authenticated as local.	String	Yes
`time_zone`	The user's timezone IANA region name.	String	No

Create a User

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/users/users
```

Possible Responses

When you execute the command to update a user, you can receive one of these three messages:

- `204 success`: A new local user was created successfully.
- `406`: Validation error such as `invalid`.
- `501`: The user is created, but the invitation e-mail failed. The new user cannot register or sign-up. If you receive this message, you need to create another local user.

Resend Invitation for a Local User

To resend the invitation to a new local user after an e-mail notification failure, use the following URI:

```
PUT /users/:user_id/local_profile/reinvite
```

Update User Information

This API updates an Illumio API user's account information.

URI to Update User's Information

```
PUT [api_version][user_href]
```

Request Body

The request body is an empty JSON object.

```
{}
```

If you attempt to use a PUT with that URL without a payload, the 406 error shows `No payload provided for PUT request`.

Property	Description	Type	Optional
`full_name`	User's full name	String	Yes
`time_zone`	The user's time zone IANA region name	String	Yes

Log Out a User

Use PUT to log out a user:

```
    "logout": {
                "http_method": "PUT",
                "path": "/users/:id/logout",
                "summary": "Logout a specific user and destroy the access token",
```

Curl Command to log out a User

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/users/12345678/logout -H
"Content-Type: application/json" -u $KEY:$TOKEN
```

where "12345678" is the user ID.

Change the User Password

This API method allows currently authenticated users to change their login password.

- The call must be made **by the user currently authenticated** in the session; even an administrator cannot change another user's password.
- An API key is not used with this API.
- The user's login name (typically the user's e-mail address) and login password are used for authentication.
- The user's five most recent passwords cannot be used.

URI to Change the User's Password

```
PUT [api_version]/login_users/[user_href]/password
```

Request Body

Property	Description	Type	Required
`password`	User's new password must meet these requirements: • Have a minimum of 8 characters • Have at least 1 capital letter • Have at least 1 lowercase letter • Have at least 1 number • Not match previously used passwords	String	Yes

Example Request Body

```
{
     "password":"'new_password'"
}
```

Change the User's Password

```
curl -u 'username'@'company'.com:'existing_password' -X PUT
https://'company'.com:8443/api/v2/login_users/me/password -H "Content-type:
application/json" -d '{"password":"'new_password'"}' -i
```

Possible Responses

When you execute the command to change a password, you can receive one of these three messages:

- `204 success`: The password was changed successfully.
- `406`: Validation error such as `invalid`.
- `501`: Password is changed, but e-mail notification failed.

LDAP Authentication

This Public Experimental API provides user authentication with the PCE using LDAP with OpenLDAP and Active Directory.

LDAP authentication comes in addition to the two previously available methods:

- API Keys, which provide persistent authentication, and
- Session credentials, which provide temporary authentication.

Prerequisites and Limitations

Before configuring LDAP for authentication with the PCE, it is important to provide the required prerequisites and review any limitations.

Determine Your User Base DN

Before you map your LDAP settings to PCE settings, determine your user base Distinguished Name (DN). The DN is the location in the directory where authentication information is stored.

If you don't have this information, contact your LDAP administrator for assistance.

When configuring the PCE to work with LDAP, be aware of the following:

- PCE uses LDAP protocol version 3 ("v3").
- Supported LDAP distributions include OpenLDAP 2.4 and Active Directory.
- Supported LDAP protocols include LDAP, LDAPS, or LDAP with STARTTLS.

Limitations

These are the current limitations for LDAP authentication:

- Any locally created user has precedence over an LDAP user of the same name. For example, if the LDAP server has a user with a username attribute (such as `cn` or `uid`) of ***johndoe*** and the default PCE user of the same name is present, the PCE user takes precedence. Only the local password is accepted. On login, the roles mapped to the local user will be in effect. To work around this limitation, you must delete the specific local user.
- LDAP and SAML single sign-on authentication methods cannot be used together. In this release of the PCE, an organization can either use LDAP or SAML single sign-on for authenticating external users.
- This release enables LDAP configuration via REST APIs only.

LDAP Authentication for the PCE

The PCE supports user and role configuration for LDAP users and groups. You can configure up to three LDAP servers and map users and user groups from your LDAP servers to PCE roles.

For information about configuring multiple LDAP servers, see How the PCE Works with Multiple LDAP Servers.

Before you configure LDAP, review the LDAP prerequisites and considerations topic in this document.

Authentication Precedence

PCE local authentication takes precedence over any external systems. The PCE authenticates a user in the following order:

a. The PCE first attempts local authentication. If the account is expired or otherwise fails, the PCE does not try to log in by using LDAP authentication.

b. If the local user does not exist, the PCE attempts LDAP login (if enabled).

Configuration Steps

To configure the PCE to work with LDAP, perform these steps:

1. Enable the PCE to use LDAP authentication. See Enable LDAP Authentication.
2. Set up an LDAP configuration. See Configure LDAP Authentication.

   When searching for LDAP users, the PCE follows the order in which the servers were configured. The configurable request timeout is 5 seconds by default. Once the request time expires, the PCE attempts to connect to the next server in the configuration.

   For example, assume that you configure three LDAP servers in this order: A, B, and C. The PCE will search the servers in that same order. If it finds a user on server A, it stops even if the same user also exists on servers B and C. The PCE will try to use A's credentials for that user, but if it fails to connect to A, it searches the remaining servers: first B, The search proceeds following the expiration of the connection timeout.
3. Map your LDAP groups to one or more PCE roles. See Mapping Group Membership to User Roles.

Mapping Group Membership to User Roles

First, configure the PCE to use LDAP authentication. Second, map PCE roles to that server's groups.

When a user attempts to log in, the PCE queries the server(s) to find that user. It grants the user permissions based on any roles associated with the LDAP groups to which the user belongs.

You have the following options for changing user permissions:

- For a group of users, remap the LDAP group to a different PCE role.
- For an individual user, move the user to an LDAP group mapped to a different PCE role using the LDAP server.

You can also perform these user management activities:

- Add a user to a PCE role:
  - On the PCE, map the PCE role to an LDAP group.
  - On your LDAP server, add the user to that LDAP group.
- Remove a user from a PCE role by removing it from the corresponding LDAP group on your LDAP server.

Users can have memberships in several roles. In that case, they have access to all the capabilities available for any of these roles. For example, a user is a member of both the **docs** and **eng** groups and **docs** group is mapped to "Ruleset Manager" while the **eng** group is mapped to "Ruleset Provisioner." In this case, the user obtains all permissions assigned both to the "Ruleset Manager" and "Ruleset Provisioner" roles.

> NOTE:The PCE checks LDAP membership information when a user attempts to log in. You do not need to reload the authentication configuration when adding or removing users.

See the PCE Web Console Guide for information about the mapping from external groups to PCE user roles.

Set up the PCE for LDAP Authentication

The PCE supports LDAPS and LDAP with STARTTLS. To use the PCE with secure LDAP with SSL/TLS certificates, add the certificate chain to the local certificate store on the PCE.

Using REST APIs for LDAP Configuration in the PCE

The following table provides an overview of the REST APIs you have available to configure the PCE for LDAP Authentication. For information about the parameters for these REST APIs, see LDAP Configuration Parameters and REST API Schema Files.

APIs for LDAP Configuration

PCE APIs	HTTP | URI
Retrieve the PCE authentication settings	`GET [api_version]/authentication_settings`
Update the PCE authentication settings	`PUT [api_version]/authentication_settings`
Retrieve the LDAP configuration	`GET [api_version]/authentication_settings/ldap_configs`
Get instance	`GET [api_version]/authentication_settings/ldap_con-`

PCE APIs	HTTP | URI
	`figs/:uuid`
Create an LDAP configuration	`POST [api_version]/authentication_settings/ldap_configs`
Update an LDAP configuration	`PUT [api_version]/authentication_settings/ldap_configs/:uuid`
Delete an LDAP configuration	`DELETE [api_version]/authentication_settings/ldap_configs/:uuid`
Verify the connection to the LDAP server	`POST [api_version]/authentication_settings/ldap_configs/:uuid/verify_connection`

LDAP Configuration Parameters

API Property Name	Type	Required	Description
`pce_fqdn`	String	No	Regional PCE member FQDN for Supercluster. For non-supercluster deployment, it is the FQDN of the PCE cluster.
`name`	String	No	Friendly name of the LDAP server
`address`	String. Format: hostname or ipv4	Yes	IP address or hostname of the LDAP server
`port`	Integer	Yes	Port number of the LDAP server - 636 for LDAPS or 389 for STARTTLS
`authentication_method`	Enum	Yes	• `LDAP`: Clear text connection • `LDAPS`: LDAP over SSL/TLS Protocol • `STARTTLS`: LDAP over SSL/TLS Protocol with handshake establishment before Secure connection:
`request_timeout_seconds`	Integer	No	Number of seconds to wait for a response; default 5 seconds. Possible values: 1-60
`bind_distinguished_`	String	No	Distinguished name (DN) used

API Property Name	Type	Required	Description
name			to bind to the LDAP server
bind_password	String	No	Bind DN password. Only applicable for POST or PUT operations; attribute will not be returned for GET instance or collection APIs.
is_bind_password_set	boolean	No	Flag to indicate whether Bind DN password is configured. Adding this flag because the API does not return the bind password and there is a need to indicate if the password has been set for the bind_distinguished_name. Only applicable for GET operation
user_base_distinguished_name	String	Yes	Base DN to search for users
user_distinguished_name_pattern	String	No	Pattern used to create a DN string for a user during login; for example, uid=*,ou=people, where * will be replaced with the username
user_base_filter	String	No	Search filter used to query the LDAP tree for users
username_attribute	String	Yes	Attribute on a user object that contains the username; for example, uid, sAMAccountName, userPrincipalName
full_name_attribute	String	No	Attribute on a user object that contains the full name; for example, cn, commonName, displayName
user_memberof_attribute	String	No	Attribute on a user object containing group membership information;

API Property Name	Type	Required	Description
			for example, memberOf, isMemberOf
`insecure_disable_tls_certificate_verification`	boolean	No	Specifies whether to verify the server certificate when establishing an SSL connection to the LDAP server; default false

Enable LDAP Authentication

This section explains how to use API to enable the PCE for LDAP authentication. You must enable the LDAP preview feature in the PCE before invoking this API. For the steps to enable this preview feature, see Enabling the LDAP Authentication Preview.

URI

```
PUT /api/v2/authentication_settings
```

Request Body

Property	Data Type	Required	Description
authentication_type	enum	Yes	The type of authentication

Enum Item	Purpose
Local	Local DB authentication
SAML	SAML authentication enabled
RADIUS	RADIUS authentication enabled
LDAP	LDAP authentication enabled

Example Payload to Configure LDAP Authentication

```
{
"authentication_type": "LDAP",
}
```

Response Code

The following response codes can be returned:

- 200 indicates success
- 403 indicates the user is not an org owner
- 406 indicates invalid parameters

Configure LDAP Authentication

This API creates the configuration for an LDAP server in the PCE. For information about the request parameters, see LDAP Configuration Parameters.

URI

```
POST /api/v2/authentication_settings/ldap_configs
```

Request body for a multi-node cluster

```
{
       "name" :  "ldap 1" ,
       "address" :  "ldap-1.mycompany.com " ,
       "port" :  "10636" ,
       "authentication_method" :  "LDAPS" ,
       "request_timeout_seconds" : 4,
       "bind_distinguished_name" :  'CN=admin,CN=Users,DC=mycompany,DC=com' ,
       "bind_password" :  'test1234' ,
       "user_base_distinguished_name" :  'DC=mycompany,DC=com' ,
       "username_attribute" :  'sAMAccountName' ,
       "full_name_attribute" :  'cn' ,
       "user_memberof_attribute" :  'memberof',
}
```

Request body for a supercluster

```
{
       "pce_fqdn" :  "devmr01" ,
       "name" :  "ldap 1" ,
       "address" :  "ldap-1.mycompany.com" ,
       "port" :  "10636" ,
       "authentication_method" :  "LDAPS" ,
       "request_timeout_seconds" : 4,
       "bind_distinguished_name" :  'CN=admin,CN=Users,DC=mycompany,DC=com' ,
       "bind_password" :  'test1234' ,
       "user_base_distinguished_name" :  'DC=mycompany,DC=com' ,
       "username_attribute" :  'sAMAccountName' ,
       "full_name_attribute" :  'cn' ,
       "user_memberof_attribute" :  'memberof' ,
}
```

Response Code

The following response codes can be returned:

- 204 indicates success
- 403 indicates the user is not an org owner
- 406 indicates invalid parameters

Configure Secure LDAP

In the process of configuring an LDAP server in the PCE, you need to configure LDAP for SSL authentication.

You can Secure LDAP with SSL/TLS Certificates using these three methods:

- Use PCE Web UI to Configure Secure LDAP.
- Install LDAP TLS Certificates to the PCE System CA Store from the PCE Command-Line.
- Configure LDAP for SSL authentication using REST APIs

Configure LDAP for SSL authentication

The following APIs are used to configure LDAP for SSL:

- `GET /authentication_settings/ldap_configs`
- `GET /authentication_settings/ldap_configs/:uuid`
- `POST /authentication_settings/ldap_configs`
- `PUT /authentication_settings/ldap_configs/:uuid`

The required property is `tls_ca_bundle`.

To manage TLS CA bundle for LDAP authentication use these APIs:

- `GET /login_proxy_ldap_configs`
- `POST /login_proxy_ldap_configs`
- `PUT /login_proxy_ldap_configs/update`

Update LDAP configuration

This section outlines how to update the LDAP server configuration in the PCE. For information about the request parameters, see LDAP Configuration Parameters.

URI

```
PUT /api/v2/authentication_settings/ldap_configs/:uuid
```

(uuid indicates the LDAP server configuration uui)

Request Body

```
{
        "address" :  "ldap-1.mycompany.com" ,
        "bind_password" :  "qw3r!y123!!" ,
       "full_name_attribute" :  "displayName" ,
        "port" : 636,
        "insecure_disable_tls_certificate_verification": true
}
```

Response Code

The following response codes can be returned:

- 204 indicates success
- 403 indicates the user is not an org owner
- 404 indicates LDAP configuration not found or an attempt to update LDAP configuration in another domain
- 406 indicates invalid parameters

Delete LDAP Server Configuration

This API deletes the configuration for an LDAP server in the PCE. For information about the request parameters, see LDAP Configuration Parameters Overview.

URI

```
DELETE /api/v2/authentication_settings/ldap_configs/:uuid
```

`uuid` indicates the LDAP server configuration uuid

Request Body

None

Response Code

The following response codes can be returned:

- 204 indicates success
- 403 indicates the user is not an org owner

- 404 indicates LDAP configuration not found or an attempt to update LDAP configuration in another domain
- 406 indicates invalid parameters

Test LDAP Server Connectivity

This section outlines the use of the API to verify the connectivity for a configured LDAP server in the PCE.

URI

```
POST /api/v2/authentication_settings/ldap_configs/:uuid/verify_connection
```

(`uuid` indicates the LDAP server configuration uuid)

Request Body

none

Response Body

If a server connection is verified successfully:

```
{
       "verified" :  true
}
```

If the server connection verification fails:

```
{
       "verified" :  false ,
       "errors" : [
       {
       "token" :  "ldap_server_verification_failure" ,
       "message" :  "LDAP server verification failure:  LDAP server error message"
       }
]
```

Response Code

The following response codes can be returned:

- 200 indicates success
- 403 indicates the user is not an org owner
- 404 indicates LDAP configuration not found

Use Cases

Configure LDAP for SSL authentication

Use case 1:

Retrieve all LDAP configurations for the domain.

1. Request format:`GET /api/v2/authentication_settings/ldap_configs`
2. Possible parameters (drawn from REST API conventions):
    - Required: none
    - Optional: none
3. Request Body: none
4. Response format: JSON
5. Response Code: 200 success

Use case 2:

Create LDAP server configuration.

1. Request format: POST /api/v2/authentication_settings/ldap_configs
2. Possible parameters (drawn somewhat from REST API Conventions):
    - Required: none
    - Optional: none
3. Request Body:

    Single-PCE

```
{
        "name": "ldap 1",
        "address": "ldap-1.ilabs.io",
        "port": "10636",
```

```
        "authentication_method": "LDAPS",
        "request_timeout_seconds": 4,
        "bind_distinguished_name": 'CN=admin,CN=Users,DC=ilabs,DC=io',
        "bind_password": 'test1234',
        "user_base_distinguished_name": 'DC=ilabs,DC=io',
        "username_attribute": 'sAMAccountName',
        "full_name_attribute": 'cn',
        "user_memberof_attribute": 'memberof',
        "tls_ca_bundle": "
        -----BEGIN CERTIFICATE-----
        MIIDhTCCAm2gAwIBAgIQYx+dZzQPBLdN6e8uqW2ByDANBgkqhkiG9w0BAQ0FADBJ
        ..............................................................
        -----END CERTIFICATE-----
        -----BEGIN CERTIFICATE-----
        MIIF7TCCBNWgAwIBAgITEgAAAEg0ToOKIywtOQAAAAAASDANBgkqhkiG9w0BAQ0F

.....................................................................................
.......
        -----END CERTIFICATE-----"
       }
```

Supercluster

```
{
        "pce_fqdn": "devmr01",
        "name": "ldap 1",
        "address": "ldap-1.ilabs.io",
        "port": "10636",
        "authentication_method": "LDAPS",
        "request_timeout_seconds": 4,
        "bind_distinguished_name": 'CN=admin,CN=Users,DC=ilabs,DC=io',
        "bind_password": 'test1234',
        "user_base_distinguished_name": 'DC=ilabs,DC=io',
        "username_attribute": 'sAMAccountName',
        "full_name_attribute": 'cn',
        "user_memberof_attribute": 'memberof',
        "tls_ca_bundle": "-----BEGIN CERTIFICATE-----
        MIIDhTCCAm2gAwIBAgIQYx+dZzQPBLdN6e8uqW2ByDANBgkqhkiG9w0BAQ0FADBJ
```

```
        -----END CERTIFICATE-----
        -----BEGIN CERTIFICATE-----
        MIIF7TCCBNWgAwIBAgITEgAAAEg0ToOKIywtOQAAAAAASDANBgkqhkiG9w0BAQ0F
        -----END CERTIFICATE-----"
}
```

4. Response format: JSON
5. Response Code:

- 204 success
- 403 not an org owner
- 406 invalid params

Use case 3:

Update LDAP server configuration:

1. Request format: `PUT /api/v2/authentication_settings/ldap_configs/:uuid`
2. Possible parameters (drawn somewhat from REST API Conventions):
    - Required: `uuid - LDAP server configuration UUID`
    - Optional: none
3. Request Body:

```
{
"tls_ca_bundle":"
-----BEGIN CERTIFICATE-----
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE------
----END CERTIFICATE-----"
}
```

4. Response format: JSON
5. Response Codes:

- 204 success
- 403 not an org owner

- 404 LDAP config not found or attempt to update LDAP config in another domain
- 406 invalid params

REST API Schema Files

The following schema files for LDAP configuration are available in 19.3.5:

- `ldap_config.schema.json`
- `authentication_settings_ldap_configs_get.schema.json`
- `authentication_settings_ldap_configs_post.schema.json`
- `authentication_settings_ldap_configs_put.schema.json`
- `authentication_settings_ldap_configs_verify_connection_post.schema.json`
- `authentication_settings_get.schema.json`
- `authentication_settings_put.schema.json`

Sample Responses

`GET /authentication_settings`

```
{
       "authentication_type" : "LDAP"
}
```

Single-PCE: `GET /authentication_settings/ldap_configs`

```
[
       {
       "href":"/authentication_settings/ldap_configs/acf577c8-839a-4828-90f6-
797bfc1b54d1",
       "pce_fqdn":"test.io",
       "name":"mycompany",
       "address":"ldap-1.mycompany.com",
       "port":389,
       "authentication_method":"LDAP",
       "request_timeout_seconds":5,
       "bind_distinguished_name":"john.doe@mycompany.com",
```

```
        "is_bind_password_set":true,
        "user_base_distinguished_name":"OU=Users,OU=mycompany
Employees,DC=mycompany,DC=com",
        "user_distinguished_name_pattern":null,
        "user_base_filter":"(&(objectcategory=person)(objectclass=user))",
        "username_attribute":"userPrincipalName",
        "full_name_attribute":"cn",
        "user_memberof_attribute":"memberOf",
        "insecure_disable_tls_certificate_verification":false,
        "created_at":"2019-03-07T23:30:13.046Z",
        "updated_at":"2019-03-07T23:30:13.046Z",
        "created_by":{
                "username":"john.doe@mycompany.com"
        },
        "updated_by":{
                "username":"john.doe@mycompany.com"
        }
    },
]
```


Supercluster: `GET /authentication_settings/ldap_configs`


```
[
        {
        "pce_fqdn":"devmr01",
        "href":"/authentication_settings/ldap_configs/8501dff7-cd3f-4c01-9057-
f2b9b1486348",
        "name":"ldap 1",
        "address":"ldap-1.mycompany.com",
        "port":389,
        "authentication_method":"STARTTLS",
        "is_bind_password_set":false,
        "user_base_distinguished_name":"DC=ilabs,DC=io",
        "user_distinguished_name_pattern":null,
        "username_attribute":"sAMAccountName",
        "full_name_attribute":"cn",
        "user_memberof_attribute":"memberof",
        "insecure_disable_tls_certificate_verification":false,
```

```
        "created_at":"2018-11-30T18:38:36.634Z",
        "updated_at":"2018-11-30T18:38:36.634Z",
        "created_by":{
                "username":"john.doe@mycompany.com"
        },
        "updated_by":{
                "username":"john.doe@mycompany.com"
        }
]
```

Chapter 3

Asynchronous GET Collections

This chapter contains the following topics:

When using the standard synchronous GET method on more than the maximum allowed number of 500 resources, only the *latest* 500 results are returned.

To GET all the results when the number of resources exceeds 500, specify in the header that the call is asynchronous (“async”), which then executes the request as an offline job.

Overview of Async GET Requests

An asynchronous job collects all matching records and downloads them as a single job. You can configure a script to continuously poll the job until it is done and then download the results of the job using the job `Location` HREF listed in the response.

Collection vs. Instance

GET collection methods return HREF path properties for each individual resource. Perform other REST operations on individual instances of these resources (such as POST, PUT, and DELETE) using the HREF to identify the resources on which to operate.

For example, the response body for the API to get a collection of labels returns a list of labels, where each one is identified as an HREF path. In this instance, the general syntax for the API call looks like this:

```
GET https://scp.illum.io:8443[api_version][org_href]labels
```

`[org_href]` identifies the organization from which you want to get a collection of labels.

A single label instance in the response is identified by its HREF path:

```
{
    href: "/orgs/2/labels/8"
    key: "env"
    value: "Prod"
    created_at: "2020-01-22T18:24:33Z"
    updated_at: "2020-01-22T18:24:40Z"
    created_by: {
       href: "/users/9"
    }
    updated_by: {
       href: "/users/9"
    }
}
```

To perform other operations on this label (href: "`/orgs/2/labels/8`"), you can provide this HREF in the API call to operate on this label instance.

For example:

```
PUT https://scp.illum.io:8443/api/v2/orgs/2/labels/8
```

Async GET Supported APIs

These APIs support async GET collections:

Description	Resource Type	Exposure
agents/update	`GET [api_version][org_href]/agents` `GET [api_version][org_href]/agents/update`	Experimental Experimental
audit_log_events	`GET [api_version][org_href]/audit_log_events`	Experimental
auth_security_principals	`GET [api_version][org_href]/auth_securiuty_principals`	Experimental
authentication_settings/ password_policy	`GET [api_version][org_href]/authentication_settings/ password_policy`	Experimental
datafiles	`GET [api_version][org_href]/datafiles`	Experimental
events	`GET [api_version][org_href]/events`	Experimental

Description	Resource Type	Exposure
jobs	`GET [api_version][org_href]/jobs`	Experimental
labels	`GET [api_version][org_href]/labels`	Both
network_devices/ network_endpoints	`GET [api_version][org_href]/network_devices/network_endpoints`	Experimental
network_enforcement_nodes	`GET [api_version][org_href]/network_enforcement_nodes`	Experimental
node_available	`GET [api_version][org_href]/node_available`	Both
Pairing Profiles	`GET [api_version][org_href]/pairing_profiles`	Experimental
permissions	`GET [api_version][org_href]/permissions`	Experimental
security_principals	`GET [api_version][org_href]/sec_policy/draft/security_principals`	Experimental
system_events	`GET [api_version][org_href]/system_events`	
vulnerability_reports	`GET [api_version][org_href]/vulnerability_reports`	Experimental
sec_policy/draft/		
allow	`GET [api_version][org_href]/sec_policy/draft/allow`	Experimental
dependencies	`GET [api_version][org_href]/sec_policy/draft/dependencies`	Experimental
ip_lists	`GET [api_version][org_href]/sec_policy/draft/ip_lists`	Both
label_groups	`GET [api_version][org_href]/sec_policy/draft/label_groups`	Experimental
label_groups/member-of	`GET [api_version][org_href]/sec_policy/draft/label_groups/member-of`	Experimental
modified_objects	`GET [api_version][org_href]/sec_policy/draft/modified_objects`	Experimental
pending	`GET [api_version][org_href]/sec_policy/draft/pending`	Experimental
rule_sets	`GET [api_version][org_href]/sec_policy/draft/rule_sets`	Both
rule_sets/sec_rule	`GET [api_version][org_href]/sec_policy/draft/rule_sets/sec_rules`	Both
services	`GET [api_version][org_href]/sec_policy/draft/services`	Both
virtual_service	`GET [api_version][org_href]/sec_policy/draft/virtaual_services`	Both
settings/		

Description	Resource Type	Exposure
settings	`GET [api_version][org_href]/settings`	
syslog/destinations	`GET [api_version][org_href]/-settings/syslog/destinations`	Experimental
workloads	`GET [api_version][org_href]/settings/workloads`	Experimental
users/		
users	`GET [api_version][org_href]/users`	Stable
api_keys	`GET [api_version][org_href]/users/api_keys`	Both
orgs	`GET [api_version][org_href]/users/orgs`	Experimental
login	`GET [api_version][org_href]/users/login`	Stable
workloads/		
workloads/	`GET [api_version][org_href]/workloads`	Both
interfaces	`GET [api_version][org_href]/workloads/interfaces`	Both

Async Job Operations

To create the asynchronous GET job request, set the following preference:

```
-H 'Prefer: respond-async'
```

Setting this preference executes the request during low-traffic times as an asynchronous job in the background, which lightens network traffic loads.

Workflow

The workflow for requesting an asynchronous bulk job consists of the following tasks:

1. Create the asynchronous GET job request.
2. Poll the job until the status is "Done" or "Failed."
3. Obtain the HREF of the completed request job.
4. Use the HREF to get the results of the request job.

Create an Async Job Request

This example demonstrates a request for an asynchronous collection of labels.

> NOTE:
> Use query parameters for a filtered job request, such as to return only the environment labels: `.../labels?key=env`

URI to Create a Job Request

```
GET [api_version]/labels
```

The asynchronous collection header is highlighted in **blue bold** font:

```
curl -i -X GET 'https://pce.my-company.com:8443/api/v2/orgs/1/labels' -H 'Accept:
application/json' -H 'Prefer: respond-async' -u $KEY:$TOKEN
```

Response with a Job Status

The response is `202 - Accepted`, which includes `Location`, the header `Retry-After` and an empty body:

```
Server: nginx
Date: Thu, 14 Jan 2020 23:16:52 GMT
       "location": https://pce.my-company.com:8443/api/v2/orgs/1/jobs/d1775367-1951-
4707-aa2e-37a0b9076d31",
       Retry-After: 5
       Transfer-Encoding: chunked
       Connection: keep-alive
       Status: 202 Accepted
       Cache-Control: no-cache
       X-Request-Id: 36aae8ce-82ed-4a6a-8a76-77d2df78daff
```

Poll the Job

After submitting the job request, poll the job using the suggested `Retry-After` time to determine when the job is complete.

URI to Get the Status of the Job

The following example demonstrates how to poll the job to determine its status.

```
GET [api_version][org_href]/jobs/[href]
```

Poll the HREF provided in the Location field of the response using the duration specified in `Retry-After` until the status is either `done` or `failed`.

```
curl -i -X GET 'https://pce.my-company.com:8443/api/v2/orgs/1/jobs/[href]' -H
'Accept: application/json' -u $KEY:$TOKEN
```

Async Job Response Properties

The following table defines the properties returned in the response:

Property	Description	Type	Required
`href`	HREF for resource	String	Yes
`status`	The current state of the job, to the effect of its success, failure, etc	String	Yes
`job_type`	An arbitrary designator for the job type or kind, typically supplied by the job requestor	String	No
`result`	The result produced by the job, typically a URI (with 'href' sub-property), or an error in case of failure.	Object, Null	No
`requested_at`	Time PCE received request	Date-time	No
`requested_by`	The URI of the user who requested this job	Object (HREF, required)	No
`terminated_at`	The time (rfc3339 timestamp) at which this job terminated, either successfully or failingly.",	Date-time	No
`created_by`	Creator of request	Object (HREF, required)	No

Async Job Status

If the job status is `running`, the response body includes the following results:

```
{
  "href": "/orgs/1/jobs/43f6e9e3-6a68-4481-87c6-18fd096dafbe",
  "job_type": ":illumio/async_requests",
  "description": "/orgs/1/labels",
  "result": {
  },
  "status": "running",
  "requested_at": "2020-01-14 23:16:52.303166",
  "requested_by": {
    "href": "/users/1"
```

```
  }
}
```

Get Async Job Results

The following example demonstrates how to get job results.

URI to Get Async Job Results

```
GET [api_version][org_href]/datafiles/[href]
```

Curl Command to Get Async Job Results

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/1/datafiles/[href] -H
'Accept: application/json' -u $KEY:$TOKEN
```

Response Body with Request Results

When the job is complete, use the HREF in the `Result` field to obtain the results:

```
{
  "href": "/orgs/1/jobs/43f6e9e3-6a68-4481-87c6-18fd096dafbe",
  "job_type": ":illumio/async_requests",
  "description": "/orgs/1/labels",
  "result": {
    "href": "/orgs/1/datafiles/[href]"
  },
  "status": "done",
  "requested_at": "2020-01-14 23:16:52.303166",
  "terminated_at": "2020-01-14 23:17:05.223047",
  "requested_by": {
    "href": "/users/1"
  }
}
```

Poll the Query Job Status

After submitting the job request, poll the job using the suggested "`Retry-After`" duration to determine when the job is complete.

The PCE has four possible status options for the job:

- `Pending`: Waiting to start
- `Running`: In progress
- `Done`: Complete (successful/unsuccessful)
- `Failed`: Unable to complete (exceeded time limit)

Get Jobs

Specify the maximum number of jobs to return with the `max_results` query parameter.

Specify the type of job to return with the `job_type` query parameter.

URI to Get the Status of All Jobs

```
GET [api_version]/jobs
```

Curl Command to Get All Job Status

```
curl -i -X GET 'https://pce.my-company.com:8443/api/v2/orgs/1/jobs' -H 'Accept:
application/json' -u $KEY:$TOKEN
```

Get a Job

URI to Get the Status of a Job

```
GET [api_version]/jobs/[href]
```

Curl Command to Get a Job Status

```
curl -i -X GET 'https://pce.my-company.com:8443/api/v2/orgs/1/jobs/[href]' -H
'Accept: application/json' -u $KEY:$TOKE
```

Response - Updated Job

If the job is still running, the response includes a status of `"running"`, as highlighted in blue below:

```
  {
  "href": "/orgs/1/jobs/43f6e9e3-6a68-4481-87c6-18fd096dafbe",
  "job_type": ":illumio/async_requests",
  "description": "/orgs/1/labels",
  "result": {
  },
```

```
  "status": "running",
  "requested_at": "2016-01-14 23:16:52.303166",
  "requested_by": {
    "href": "/users/1"
  }
}
```

Delete a Job

URI to Delete a Job

```
DELETE [api_version]/jobs/[href]
```

Curl Command to Delete a Job

```
curl -i -X DELETE 'https://pce.my-company.com:8443/api/v2/orgs/1/jobs/[href]' -u
$KEY:$TOKEN
```

Get the Job Results

This example demonstrates how to get job results after polling job returns a status of `"done"`.

The `uuid` path parameter is required. The `filename` path parameter is optional, it specifies the file-name to save the job.

URI to Get Job Results

```
GET [api_version][org_href]/datafiles/[uuid]
```

Curl Command to Get Job Results

```
curl -i -X GET 'https://yourcompany.com:1234/api/v2/orgs/1/datafiles/[uuid]' -H
'Accept: application/json' -u $KEY:$TOKEN
```

Response with Results of Request

```
{
  "href": "/orgs/1/jobs/43f6e9e3-6a68-4481-87c6-18fd096dafbe",
  "job_type": ":illumio/async_requests",
```

```
  "description": "/orgs/1/labels",
  "result": {
    "href": "/orgs/1/datafiles/[uuid]"
  },
  "status": "done",
  "requested_at": "2016-01-14 23:16:52.303166",
  "terminated_at": "2016-01-14 23:17:05.223047",
  "requested_by": {
    "href": "/users/1"
  }
}
```

Chapter 4

PCE Management

This chapter contains the following topics:

As an Illumio administrator, use the APIs listed in this chapter to manage the Policy Compute Engine (PCE).

You can manage many aspects of the PCE through APIs, from authentication and passwords to PCE health.

Product Version

This API returns the current version of the PCE software.

URI to Get Product Version

```
GET [api_version]/product_version
```

Curl Command to Get Product Version

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/product_version -H "Accept:
application/json" -u $KEY:$TOKEN
```

Example Response

The response body has a format similar to this example:

```
{
  "version": "19.3.0",
  "build": 12864,
  "long_display": "19.3.0-12864",
  "short_display": "19.3.0"
}
```

Authentication Settings

This Public Experimental API gets or updates the authentication settings for the login domain (organization).

These new APIs with the included `saml_configs` setting provide customers an option to sign authN requests.

API Methods

HTTP	URI	Functionality
GET	[api_version]/authentication_settings	Get authentication settings
PUT	[api_version]/authentication_settings	Update authentication settings
GET	[api_version]/authentication_settings/saml_configs	Gets all SAML configurations where `any_org_owner` is authorized to use it. The response now includes the PCE signing certificates that will be used by IdP for the SAML authN request signature validation.

HTTP	URI	Functionality
GET	[api_version]/authentication_settings/saml_configs/:uuid	Get the specified SAML configuration. The response now includes the PCE signing certificates that will be used by IdP for the SAML authN request signature validation.
PUT	[api_version]/authentication_settings/saml_config/:uuid	Update the specified SAML configuration. API has been enhanced to enable/disable the signing of a SAML authN request.
POST	[api_version]/authentication_settings/saml_configs/:uuid/pce_signing_cert	Generate a new certificate for signing SAML AuthN requests.

Get Authentication Settings

Curl Command to Get Authentication Settings

The `org/:org_id/` path parameter is not specified in this command.

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/authentication_settings -H
"Accept: application/json" -u $KEY:$TOKEN
```

Example Default Response

```
200 OK

{ "authentication_type":"Local" }
```

Update Authentication Settings

Curl Command to Update Authentication Settings

The `org/:org_id/` path parameter is not specified in this command.

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/authentication_
settings/password_policy -H "Content-Type: application/json" -d '{"authentication_
settings": "SAML"}' u $KEY:$TOKEN
```

Request Properties

Parameter	Description	Type	Required
`Local`	Local authentication.	String	No
`SAML`	Authentication with SAML.	String	No

Example Request Body

```
{"authentication_settings": "SAML"}
```

Password Policy

This Public Experimental API gets or updates the domain password policy.

A default password policy is created automatically when a new login domain (organization) is created. There is only one password policy per login domain, so the same password policy applies to all users.

Password Policy Methods

Functionality	HTTP	URI
Get the password policy	`GET`	`[api_version]/authentication_settings/password_policy`
Update the password policy	`PUT`	`[api_version]/authentication_settings/password_policy`

Curl Command Get the Password Policy

The `org/:org_id/` path parameter is not specified in this command.

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/authentication_
services/password_policy -H "Accept: application/json" -u $KEY:$TOKEN
```

Example Default Response: 200 OK

```
{
  "require_type_number": true,
  "require_type_lowercase": true,
  "require_type_uppercase": true,
  "require_type_symbol": false,
  "min_characters_per_type": 1,
  "min_length": 8,
```

```
  "min_changed_characters": 1,
  "history_count": 1,
  "expire_time_days": 0,
  "updated_at": "2019-09-20T03:40:00Z",
  "updated_by": null
}
```

Response Parameters

Parameter	Description	Type	Req
`require_type_number`	If `true`, the password must contain a numerical digit.	Boolean	Yes
`require_type_lowercase`	If `true`, the password must contain a lowercase letter.	Boolean	Yes
`require_type_uppercase`	If `true`, the password must contain an uppercase letter.	Boolean	Yes
`require_type_symbol`	If `true`, the password must contain a symbol, for example: ! @ # $ % ^ * ? \u0026 \u003c \u003e	Boolean	Yes
`min_characters_per_type`	Minimum number of characters for each character type.	Integer	Yes
`min_length`	Minimum password length.	Integer	Yes
`min_changed_characters`	Minimum number of changed characters for a new password. Minimum: 1 Maximum: 4	Integer	Yes
`history_count`	Number of old passwords to remember. Minimum: 1 Maximum: 24	Integer	Yes
`expire_time_days`	Number of days until the password expires. A value of 0 (zero) means the password never expires. Minimum: 0 Maximum: 99	Integer	Yes
`updated_at`	RFC-3339 date-time timestamp of when the password policy was last updated. Automatically recorded by the system.	date-time String	Yes

Parameter	Description	Type	Req
updated_by	The username of the person that last updated this password policy (null for the default password policy). Automatically recorded by the system.	String	Yes

Update Password Policy

Curl Command Update the Password Policy

The org/:org_id/ path parameter is not specified in this command.

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/authentication_
services/password_policy -H "Content-Type: application/json" -u $KEY:$TOKEN -d '
{"require_type_symbol": true, "expire_time_days": 90}
```

Parameters

At least three of the four available character types must be true, otherwise a 406 Not Acceptable error message is returned.*

Parameter	Description	Type	Required
require_type_number	If true, the password must contain a numerical digit.	Boolean	*
require_type_lower-case	If true, the password must contain a lowercase letter.	Boolean	*
require_type_upper-case	If true, the password must contain an uppercase letter.	Boolean	*
require_type_symbol	If true, the password must contain a symbol, for example: ! @ # $ % ^ * ? \u0026 \u003c \u003e	Boolean	*
min_characters_per_type	Minimum number of characters for each character type.	Integer	No
min_length	Minimum password length.	Integer	No
min_changed_characters	Minimum number of changed characters for new passwords. Minimum: 1 Maximum: 4	Integer	No
history_count	Number of old passwords to remember. Minimum: 1	Integer	No

Parameter	Description	Type	Required
	Maximum: 24		
`expire_time_days`	Number of days password expires. A value of 0 (zero) means the password never expires. Minimum: 0 Maximum: 99	Integer	No

Example Request Body

Only the parameters to change must be included in the request body.

```
{
  "require_type_number": true,
  "require_type_lowercase": true,
  "require_type_uppercase": true,
  "require_type_symbol": true,
  "min_characters_per_type": 1,
  "min_length": 8,
  "min_changed_characters": 1,
  "history_count": 1,
  "expire_time_days": 90
}
```

Supercluster Leader

The Supercluster Leader Public Stable API method checks each PCE in a Supercluster and indicates which PCE is the leader.

Supercluster Leader API

This call is typically made by a customer's Global Server Load Balancer (GSLB) to monitor the health of the leader.

Possible results:

- If the API returns an HTTP 202 response, the cluster where you made this call is the leader.
- If the API returns an HTTP 404 response, then the cluster where you made this call is a member.

Get Supercluster Leader

```
GET [api_version]/supercluster/leader
```

Curl Command Get Supercluster Leader

```
curl -i  -X GET https://pce.my-company.com:8443/api/v2/orgs/3/supercluster/leader
-H "Accept: application/json" -u $KEY:$TOKEN
```

PCE Health

The Public Stable Health Check API displays health information about a 4X2 Supercluster or a PCE virtual appliance.

> NOTE:
> This API is only available for Illumio Core PCE installed on-premises and is not available for Illumio Cloud customers.

About PCE Health API

With this API, you can see the following health information:

- How long the PCE has been running, its runlevel, and overall health (normal, warning, or error).
- Each node hostname, IP address, uptime, runlevel, and whether the PCE software is running properly.
- Each node type (core or data), and which data node is the database replica and which is the primary database. The replication delay for the database replica is also displayed.
- Information about PCE service alerts, such as the number of degraded or failed services in the cluster, so you can see where service failures have occurred.

PCE Health API Method

Functionality	HTTP	URI
Check the health of the PCE.	`GET`	`[api_version]/health`

Check PCE Health

URI to Check PCE Health

```
GET [api_version]/health
```

Curl Command Check PCE Health

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/health -H 'Accept:
application/json' -u $KEY:'TOKEN'
```

PCE Health Response Properties

Property	Description	Type
`status`	Current health status of the PCE. Possible values: • `normal`: When a PCE health is a normal state it means: ◦ All required services are running. ◦ All nodes are running. ◦ CPU usage of all nodes is less than 95%. ◦ Memory usage of all nodes is less than 95%. ◦ Disk usage of all nodes is less than 95%. ◦ Database replication lag is less than or equal to 30 seconds. • `warning`: When PCE health is in a warning state, it means: ◦ One or more nodes are unreachable. ◦ One or more optional services are missing, or one or more required services have been degraded. ◦ The CPU usage of any node is greater than or equal to 95%. ◦ Memory usage of any node is greater than or equal to 95%. ◦ Disk usage of any node is greater than or equal to 95%. ◦ Database replication lag is greater than 30 seconds. • `critical`: A PCE is considered to be in a critical state when one or more required services are missing. If a PCE enters a critical state, it might not be possible to authenticate to the PCE or get an API response depending on which services are missing from the PCE.	String
`type`	The type of PCE:	String

Property	Description	Type
	• `standalone`: Indicates that this PCE is an on-premises 2x2 or 4x2 PCE cluster. Or one of the following types: • `leader`: Indicates that this PCE is the leader of a Supercluster. • `member`: Indicates that this PCE is a member of a Supercluster.	
`fqdn`	The fully qualified domain name (FQDN) of the PCE.	String
`available_seconds`	The length of time that this PCE has been available, measured in seconds.	Number
`notifications`	Health warnings related to the PCE, which contain the following properties: • `status`: Severity status of this notification. Possible values include: `normal`, `warning`, or `critical`. • `token`: Description of the notification. • `message`: Notification message.	
`listen_only_mode_enabled_at`	Indicates when listen-only mode was enabled for this PCE. For information about enabling or disabling listen-only mode for a PCE, see the *PCE Administration Guide*.	String
`nodes`	The nodes that comprise your PCE cluster. For each node of your PCE, this API call returns the following properties: • `hostname`: The node hostname. • `ip_address`: The node IP address. • `runlevel`: (Number) The current runlevel of the PCE software on the node. For more information about runlevels and their usage, see the *PCE Administration Guide*. • `uptime_seconds`: Seconds since this node has been restarted. • `cpu`: Percentage of the node CPU being used. Includes the following two sub-properties: ◦ `status`: Either `normal`, `warning`, or `critical`. ◦ `percent`: (Number) Percentage of the node CPU being used. • `disk`: Percentage of the node's disk that is being used. Includes the following two sub-properties: ◦ `status`: Either `normal`, `warning`, or `critical`.	String

Property	Description	Type
	◦ `percent`: (Number) Percentage of the node disk being used. • `memory`: Percentage of the node's memory that is being used. Includes the following two sub-properties: ◦ `status`: Either `normal`, `warning`, or `critical`. ◦ `percent`: (Number) Percentage of the node disk being used. • `services`: The status of all PCE services running on the node. Possible status for PCE services include: ◦ `running`: The service is fully running and operational. ◦ `not running`: The service has stopped running. ◦ `partial`: The service is running but in a partial state. ◦ `optional` ◦ `unknown` • `generated_at`:Timestamp when this information was generated.	
`network`	**PCE 2x2 or 4x2 Deployment** For a PCE 2x2 or 4x2 deployment, the `network`property provides latency information between the database primary and database replica data nodes in your PCE for policy and traffic data. This property also indicates which data node in your PCE is the database primary database and which is the database replica. This `type` of database replication is called `intracluster` in the REST API. Sub-properties include: `replication`: The category of properties that provide database replication latency information for a PCE cluster. (For a PCE Supercluster, this information is provided for each PCE in the Supercluster.) • `type`: Type of replication. `intracluster` for a PCE 2x2 or 4x2 deployment. • `details`: Includes the following properties: ◦ `database_name`: Either `agent` for policy data or `traffic` for traffic data. ◦ `primary_fqdn`: The FQDN of the database primarynode. ◦ `replica_fqdn`: FQDN of the replica database node.	Array

Property	Description	Type

	• `value`: The amount of replication lag between the primary and database replica for both policy and traffic data.
	  ◦ `status`: Either `normal`, `warning`, or `critical`.
	  ◦ `lag_seconds`: The amount of lag measured in seconds between the primary and replica databases for both policy and traffic data.
	
	Supercluster Deployment
	
	If you have deployed a PCE Supercluster, the PCE health call also returns information about the database replication between the PCE you are currently logged into and all other PCEs in the Supercluster.
	
	In a Supercluster deployment, the security policy provisioned on the leader is replicated to all other PCEs in the Supercluster. Additionally, all PCEs in the Supercluster (leader and members) replicate copies of each workload's context, such as IP addresses, to all other PCEs in the Supercluster.
	
	This other `type` of database replication for a Supercluster is called `intercluster` in the REST API, and information is provided for all PCEs in the Supercluster.
	
	Properties include:
	
	`replication`: The category of properties that provide database replication latency
	information for a PCE cluster.
	
	• `type`: Type of replication. `intercluster` for a PCE Supercluster deployment.
	• `details`: Includes the following properties:
	  ◦ `fqdn`: The FQDN of the primary database of the other PCEs listed in this section.
	• `value`: The amount of replication lag between the PCE you are logged into
	and one of the other PCEs in the Supercluster.
	  ◦ `status`: Either `normal`, `warning`, or `critical`.
	  ◦ `lag_seconds`: The amount of lag measured in seconds between the PCE you are logged into	

Property	Description	Type
	and the other PCE listed in this section.	
generated_at	The timestamp of when the information was generated.	String

PCE Health Response

Example response returned from the PCE Health API.

```
[
    {
        "status": "normal",
        "type": "standalone",
        "fqdn": "pce.mycompany.com",
        "available_seconds": 84133,
        "notifications": [],
        "listen_only_mode_enabled_at": null,
        "nodes": [
            {
                "hostname": "pce_core1.mycompany.com,
                "ip_address": "192.0.1.0",
                "type": "core",
                "runlevel": 5,
                "uptime_seconds": 2051301,
                "cpu": {
                    "status": "normal",
                    "percent": 7
                },
                "disk": [
                    {
                        "location": "disk",
                        "value": {
                            "status": "normal",
                            "percent": 17
                        }
                    }
                ],
                "memory": {
                    "status": "warning",
                    "percent": 85
                },
```

```
            "services": {
                "status": "normal",
                "services": {
                    "running": [
                        "agent_background_worker_service",
                        "agent_service",
                        "agent_traffic_service",
                        "auditable_events_service",
                        "collector_service",
                        "ev_service",
                        "executor_service",
                        "fluentd_source_service",
                        "login_service",
                        "memcached",
                        "node_monitor",
                        "search_index_service",
                        "server_load_balancer",
                        "service_discovery_server",
                        "traffic_worker_service",
                        "web_server",
                    ]
                }
            },
            "generated_at": "2020-03-03T19:38:52+00:00"
        },
        }
    ],
    "network": {
        "replication": [
            {
                "type": "intracluster",
                "details": {
                    "database_name": "agent",
                    "primary_fqdn": "bkhorram-qa-6node-v0-pce-1-dbase0"
                },
                "value": {
                    "status": "normal",
                    "lag_seconds": 0
```

```
                    }
                },
                {
                    "type": "intracluster",
                    "details": {
                        "database_name": "traffic",
                        "primary_fqdn": "bkhorram-qa-6node-v0-pce-1-dbase0"
                    },
                    "value": {
                        "status": "normal",
                        "lag_seconds": 0
                    }
                }
            ]
        },
        "generated_at": "2020-03-03T19:38:52+00:00"
    }
]
```

Node Availability

This Public Stable API method allows the Load Balancer to monitor the health of the PCE core nodes in a 2x2 or 4x2 cluster. This feature is only available if the PCE is deployed as software in your datacenter.

> NOTE:
> This API call does not require authentication.

URI to Check Node Availability

```
GET [api_version]/node_available
```

Check Node Availability

`-X GET` and authentication are not required for this method. The curl `-v` flag provides verbose output.

```
curl -v https://pce.my-company.com:8443/api/v2/node_available
```

Or, you can use `-i -X GET` to return a `200 OK` status if the node is available:

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/node_available
```

Returns `200 OK` if the core node is healthy, and it can see at least one of each service running in the PCE cluster.
Otherwise, it returns a `404` error.

For example, if the PCE is healthy and accessible, the response is `200 OK`.

Health Check from a Load Balancer

In a production deployment, customers run health checks from a Load Balancer. The actual request syntax varies, but here is a sample command for Infoblox:

```
GET /api/v2/node_available HTTP/1.1
```

Support Bundle Requests

Several APIs have been introduced to provide a mechanism to generate a support bundle on each node, including a time range and possibly additional options.

API Methods

Functionality	HTTP	URI
Return the collection of PCE support bundle requests:	`GET`	`[api_version][org_href]/support_bundle_requests`
Return a specific PCE support bundle request:	`GET`	`[api_version][org_href]/support_bundle_requests/:uuid`
Create a PCE support bundle request	`POST`	`[api_version][org_href]/support_bundle_requests`
Delete the PCE support bundle request	`DELETE`	`[api_version][org_href]/support_bundle_requests/:uuid`

Query Parameters

Property	Description	Type	Required
`org_id`	Organization ID	Integer	Yes
`ending_at`	Time at which to exclude entries	String	No
`include_logs`	Set to true if logs are to be included	Boolean	No
`starting_at`	Start date for log filtering	Sring	No
`requested_at`	Time support bundle requested	string(date-time)	Yes

Properties for Support Bundle Requests

Property	Description	Type	Required
`href`	URI of this request Reference to `common/href_object.schema.json`		Yes
`name`	The name of the support bundle	String	Yes
`download_url`	URI of associated report file Reference to `common/href_object.schema.json`		Yes
`requested_at`	Time support bundle requested	String (date-time)	Yes
`completed_at`	Time support bundle completed	String, Null(date-time)	Yes
`status`	A status annunciator indicating the state of this request	String	Yes
`include_logs`	Set to true if logs are to be included	Boolean	Yes
`starting_at`	(GET, POST) Start date for log filtering	String, Null(date-time)	Yes
`ending_at`	End date for log filtering	String, Null(date-time)	Yes

Example for POST

```
{
        "include_logs": true,
        "starting_at": null,
        "ending_at": null
}
```

No Op

The No Op Public Stable API makes a call to the PCE without performing any operations. This API is used to check connectivity to and from the PCE.

Use this API to verify that new authentication credentials are working after creating a new set of keys.

URI for No Op

```
GET [api_version]/noop
```

Curl Command for No Op

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/noop -H "Accept:
application/json" -u $KEY:'TOKEN'
```

Events

This Public Experimental API gets a collection of events or an individual event.

> NOTE:
> Starting with Illumio Core 18.2, use this Events API instead of Audit Events.

Events include logging a user in or out of the PCE, granting a role to a user, pairing or unpairing a workload, creating a label, ruleset, or IP list.

Event Types

For a complete list of JSON events, descriptions, CEF/LEEF success events, and CEF/LEEF failure events, see the *Events Administration Guide*.

Event API Methods

Functionality	HTTP	URI
Get a collection of events	`GET`	`[api_version][org_href]/events`
Get an individual event	`GET`	`[api_version][event_href]`

Get Events

This API gets a collection of events or a specific event identified by an event ID (in the form of a UUID).

Get Events Collection

When getting a collection of events, be aware of the following caveats:

- Use the `max_results` query parameter to increase the maximum number of events returned.

- The largest value accepted for `max_results` is 10000. To return more than 10000 events, use an Asynchronous GET Collection.

URI to Get a Collection of Events

```
GET [api_version][org_href]/events
```

URI to Get an Individual Event

```
GET [api_version][event_href]
```

Query Parameters

Parameter	Description	Type
`xorg_id`	Organization ID in which the event occurred.	Integer
`created_by`	Information about the person, agent, or system that created the event. Created by *system*: • `system`: Appears only if the event was generated by the PCE. Created by *user* properties: • `href`: URI of the user who created the event. • `username`: The user's name (usually formatted as an e-mail address). Created by *workload* properties: • `href`: URI of the agent on the workload that initiated the event. • `hostname`: The hostname of the workload.	String
`event_type`	Type of the event specified by the `event_type` query parameter if given. If no query parameters are given, all event types are returned. For types of events returned from a GET call, see the response properties table below.	String
`max_results`	Maximum number of events to return. The default is 100, and the maximum is 10000.	Integer
`severity`	Severity level of the events retrieved. Values include: • Warning (`warning`): A warning that the event is likely to occur if	String

Parameter	Description	Type
	action is not taken. • Error (`err`) • Information (`info`): Normal operational messages, which can be harvested for reporting and measuring throughput; for example, a user pairing or unpairing workloads in the PCE web console.	
`status`	Status of the event, either `success` or `failure`.	String
`timestamp [gte]`	Event start timestamp in RFC 3339 format.	String
`timestamp [lte]`	Event end timestamp in RFC 3339 format.	String

Response Properties

Parameter	Description	Type
`event_type`	Type of the event specified by the `event_type` query parameter if given. If no query parameters are given, all event types are returned. For types of events returned from a GET call, see the response properties table below.	String
`status`	Status of the event; usually a mapping of `api_status_code` to a generic result string; nil if no action. For presentation purposes only.	String
`severity`	Severity level of the events retrieved. Values include: • Warning (`warning`): A warning that the event is likely to occur if action is not taken. • Error (`err`) • Information (`info`): Normal operational messages, which can be harvested for reporting and measuring throughput; for example, a user pairing or unpairing workloads in the PCE web console.	String
`created_by`	Information about the person, agent, or system that created the event. Created by *system*: • `system`: Appears only if the event was generated by the PCE. Created by *user* properties: • `href`: URI of the user who created the event. • `username`: The user's name (usually formatted as an e-mail address).	String

Parameter	Description	Type
	Created by *workload* properties: • `href`: URI of the agent on the workload that initiated the event. • `hostname`: The hostname of the workload.	

Curl Command to Get an Event

You need the ID of the system event you want to get, which is the number at the end of its HREF path property: `"/2/events/68632"`.

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/events/12345 -H
"Accept: application/json" -u $KEY:$TOKEN
```

Curl Command Get Event Collection

In this example, only two events are returned because of`max_events=2`.

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/events?max_results=2
-H "Accept: application/json" -u $KEY:$TOKEN
```

Example Response

```
[
  {
    "href": "/orgs/1/events/xxxxxxx-5f59-46ab-8f18-xxxxxxxxx",
    "timestamp": "2019-09-03T01:xx:xx.xxxZ",
    "pce_fqdn": "pce.my-company.com",
    "created_by": {
      "agent": {
        "href": "/orgs/1/agents/xxx",
        "hostname": "xxx-xxxxx-xxxx"
      }
    },
    "event_type": "agent.clone_detected",
    "status": null,
    "severity": "info",
    "action": null,
    "resource_changes": [],
    "notifications": [
      {
```

```
        "uuid": "xxxxxxx-e04b-43bc-a64a-xxxxxxxxxx",
        "notification_type": "agent.clone_detected",
        "info": {
          "agent": {
            "href": "/orgs/1/agents/xxx",
            "name": null,
            "hostname": "xxx-xxxxx-xxxx"
          }
        }
      }
    ]
  },
  {
    "href": "/orgs/1/events/xxxxxxx-60a2-4db4-b0f4-xxxxxxxxxx",
    "timestamp": "2019-09-03T0x:xx:xx.xxxZ",
    "pce_fqdn": "pce.my-company.com",
    "created_by": {
      "agent": {
        "href": "/orgs/1/agents/xxx",
        "hostname": "xxx-xxxxx-xxxx"
      }
    },
    ]
  }
]
```


Organization Settings

For Organization Settings parameters, properties, JSON request and response bodies, and example curl commands, see "Organization Settings" in the Illumio Core REST API Reference.

Get Events Settings

Returns events settings information.

For parameters, properties, JSON response body, and example curl command, see "Get Events Settings" in the Illumio Core REST API Reference.

Example JSON Response Body for Get Events Settings

```
{
    "audit_event_retention_seconds": 180,
    "audit_event_min_severity": "informational",
    "format": "JSON"
}
```

Update Events Settings

For parameters, properties, JSON request body, and example curl command, see "Update Events Settings" in the Illumio Core REST API Reference.

Example JSON Request Body for Update Events

```
{
    "audit_event_retention_seconds": 90,
    "audit_event_min_severity": "informational"
}
```

Syslog Destinations

Use this API to specify a local syslog location and/or one or more remote syslog locations.

Get all Syslog Destinations

Returns all syslog destination information.

For parameters, properties, JSON response body, and example curl command, see "Get Syslog Destinations" in the Illumio Core REST API Reference.

Example JSON Response Body with Local and Remote Syslog Location Information

```
[
    {
        "href": "/api/v2/orgs/1/settings/syslog/destinations/xxxxxxxx-xxxx-xxxx-
xxxx-xxxxxxxxxxxx",
        "pce_scope": [ "remote-my-company0.com", "remote-my-company1.com" ],
        "type": "remote_syslog",
        "description": "remotesyslog",
        "audit_event_logger": {
            "configuration_event_included": true,
```

```
            "system_event_included": false,
            "min_severity": "warning"
        },
        "traffic_event_logger": {
            "traffic_flow_allowed_event_included": true,
            "traffic_flow_potentially_blocked_event_included": true,
            "traffic_flow_blocked_event_included": true
        },
        "node_status_logger": {
            "node_status_included": true
        },
        "remote_syslog": {
            "address"  : "my-company-20.com",
            "port"     : 12345,
            "protocol" : 6,
            "tls_enabled"     : false,
            "tls_verify_cert" : false
        }
    }
]
```

Get a Specified Syslog Destination

Returns information about one syslog destination.

For parameters, properties, JSON response body, and example curl command, see "Get a Syslog Destination" in the Illumio Core REST API Reference.

Example JSON Response Body with Remote Syslog Location Information

```
{
    "href": "/api/v2/orgs/1/settings/syslog/destinations/xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxxx",
    "pce_scope": [ "remote-my-company0.com", "remote-my-company1.com" ],
    "type": "remote_syslog",
    "description": "remotesyslog",
    "audit_event_logger": {
        "configuration_event_included": true,
        "system_event_included": false,
        "min_severity": "warning"
```

```
    },
    "traffic_event_logger": {
        "traffic_flow_allowed_event_included": true,
        "traffic_flow_potentially_blocked_event_included": true,
        "traffic_flow_blocked_event_included": true
    },
    "node_status_logger": {
        "node_status_included": true
    },
    "remote_syslog": {
        "address"  : "my-company-20.com",
        "port"     : 12345,
        "protocol" : 6,
        "tls_enabled"     : false,
        "tls_verify_cert" : false
    }
}
```

Create a Syslog Destination

Creates a local and remote syslog destination.

For parameters, properties, JSON request body, and example curl command, see "Create a Syslog Destination" in the Illumio Core REST API Reference.

Example JSON Request Body to Create a Remote Syslog Destination

```
{
    "pce_scope": [ "my-company0.com", "my-company1.com", "my-company2.com" ],
    "type": "remote_syslog",
    "description": "remote syslog",
    "audit_event_logger": {
        "configuration_event_included": true,
        "system_event_included": false,
        "min_severity": "warning"
    },
    "traffic_event_logger": {
       "traffic_flow_allowed_event_included": true,
       "traffic_flow_potentially_blocked_event_included": true,
       "traffic_flow_blocked_event_included": true
```

```
    },
    "node_status_logger": {
        "node_status_included": true
    },
    "remote_syslog": {
        "address"  : "my-company-20.com",
        "port"     : 12345,
        "protocol" : 6,
        "tls_enabled"     : false,
        "tls_verify_cert" : false
    }
}
```

Update a Syslog Destination

Updates a local and a remote syslog destination.

For parameters, properties, JSON request body, and example curl command, see "Update a Syslog Destination" in the Illumio Core REST API Reference.

Example JSON Request Body to Update a Syslog Destination

```
{
    "href": "/api/v2/orgs/1/settings/syslog/destinations/xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxxx",
    "pce_scope": [ "my-company0.com", "my-company1.com", "my-company2.com"  ],
    "type": "remote_syslog",
    "description": "localhost syslog",
    "audit_event_logger": {
        "configuration_event_included": true,
        "system_event_included": true,
        "min_severity": "informational"
    },
    "traffic_event_logger": {
       "traffic_flow_allowed_event_included": true,
       "traffic_flow_potentially_blocked_event_included": true,
       "traffic_flow_blocked_event_included": true
    },
    "node_status_logger": {
        "node_status_included": false
```

```
    },
    "remote_syslog": {
        "address"  : "my-company-20.com",
        "port"     : 67890,
        "protocol" : 6,
        "tls_enabled"     : false,
        "tls_verify_cert" : false
    }
}
```

Delete a Syslog Destination

Deletes a syslog destination.

For parameters, properties, and example curl command, see "Delete a Syslog Destination" in the Illumio Core REST API Reference.

Container Clusters

The Illumio Core uses three groups of APIs to manage container clusters:

- Container Cluster API (GET, POST, PUT, DELETE)
- Container Cluster Workload Profiles API (GET, POST, PUT, DELETE)
- Container Cluster Service Backend API (GET)

Container Cluster API

A container cluster object is used to store all the information about a Kubernetes cluster in the PCE by collecting telemetry from Kubelink. Each Kubernetes cluster maps to one container cluster object in the PCE.

Use these methods to get, create, update, or delete container clusters:

Functionality	HTTP	URI
Get the list of container clusters	GET	[api_version][org_href]/container_clusters
Get the specified container cluster	GET	[api_version][org_href]/container_clusters/:uuid
Create a container cluster	POST	[api_version][org_href]/container_clusters
Update the specified container cluster	PUT	[api_version][org_href]/container_clusters/:uuid
Delete the specified container cluster	DELETE	[api_version][org_href]/container_clusters/:uuid

Query Parameters for the GET Method

Use the following required and optional parameters:

Parameter	Description	Type	Required
`href`	URI of the container cluster.	String	Yes
`name`	User assigned name of the container cluster.	String	Yes
`description`	User-assigned description of the container cluster.	String	Yes
`nodes`		Array	No
`machine_id`	This parameter has the following property: • `pod_subnet`: The pod subnet	Object String	Yes
`manager_type`	Manager of the container cluster (and version).	String	No
`network_type`	Type of network.	String	No
`last_connected`	Date-time format.	String	No
`online`	Online: true/false.	Boolean	No
`errors`	The object `error_type` has the following properties: • `audit_event`: ◦ `href` • `duplicate_ids` • `error_type`	Array Object String Array String String	No
`kubelink_version`	Kubelink software version.	String	No
`pce_fqdn`	PCE FQDN for this container cluster; used only in Super-cluster.	String	No
`cluster_mode`	The new property `cluster_mode` was added in 23.5.10 to describe the cluster mode for container cluster:	String	

Query Parameters for the POST and PUT Methods

Use the following parameters:

Parameter	Description	Type	Required
`name`	User-assigned name of the cluster	String	Yes
`description`	User-assigned description of the cluster	String	No

Curl Examples and Responses

Curl Command for GET

```
curl --request GET --url https://pce.my-
company.com:8443/api/v2/orgs/1/container_clusters --header 'authorization: Basic
YXBpXzE2YjBkYjI0MjJhZGNkYWU5OjA5ZmRjNjA4MDhiMzExZTc2Y2UyNzNmOWNiN2ZhMTA5OTdkMWNlMD
AzZmMzOTQ1ZGMxYzEwZGzlmZjM='
```

Example Response for GET

```
[
 {
  "href":"/orgs/1/container_clusters/445bfa9b-4de4-4c09-9705-496eb04b190f",
      "pce_fqdn":null,
      "name":"k8s2",
      "description":"",
      "manager_type":"Kubernetes v1.16.2",
      "last_connected":"2019-10-28T22:48:31.228Z",
      "kubelink_version":"2.0.0-master.96e58b",
      "online":true,
      "nodes":
      [
       {
        "name":"node1",
        "pod_subnet":"10.233.64.0/24"
         },
        {
         "name":"node2",
        "pod_subnet":"10.233.65.0/24"
         },
       {
        "name":"node3",
        "pod_subnet":"10.233.66.0/24"
        }
      ],
      "errors":[]
        },
{
```

```
 "href":"/orgs/1/container_clusters/ad678193-8e2f-402b-a864-4947dcc0c6d7",
     "pce_fqdn":null,
     "name":"Openshift 3.11",
     "description":"",
     "manager_type":"Openshift v3.11.43",
     "last_connected":"2019-10-28T22:50:30.201Z",
     "kubelink_version":"1.0.0-master.a81280",
     "online":true,
     "nodes":
       [
        {
         "name":"ip-172-31-19-198.us-west-2.compute.internal",
         "pod_subnet":"10.128.0.0/23"
         },
        {
         "name":"ip-172-31-20-168.us-west-2.compute.internal",
         "pod_subnet":"10.131.0.0/23"
         },
        {
         "name":"ip-172-31-22-56.us-west-2.compute.internal",
         "pod_subnet":"10.130.0.0/23"
         },
        {
         "name":"ip-172-31-27-241.us-west-2.compute.internal",
         "pod_subnet":"10.129.0.0/23"
         }
     ],
     "errors":[]
        },
{
 "href": "/orgs/1/container_clusters/bef57e90-97d4-4744-a129-5d35aa12b21b",
     "pce_fqdn":null,
     "name":"k8s3 Cluster",
     "description":"Flannel Vx Lan",
     "manager_type":"Kubernetes v1.13.2",
     "last_connected":"2019-10-28T22:47:59.122Z",
     "kubelink_version":"EYE-60264",
     "online":true,
```

```
      "nodes":
       [
         {
          "name":"k8s3master",
          "pod_subnet":"10.244.0.0/24"
          },
         {
          "name":"k8s3minion1",
          "pod_subnet":"10.244.2.0/24"
          },
         {
          "name":"k8s3minion2",
          "pod_subnet":"10.244.1.0/24"
          }
      ],
      "errors":[]
        },
{
 "href":"/orgs/1/container_clusters/d7d62400-7650-4407-ae9b-71803dbb1324",
      "pce_fqdn":null,
      "name":"k8s1 v4",
      "description":"",
      "manager_type":"Kubernetes v1.12.4",
      "last_connected":"2019-10-24T23:58:55.795Z",
      "kubelink_version":"EYE-61567",
      "online":false,
      "nodes":
        [
         {
          "name":"k8s1master",
          "pod_subnet":"10.244.0.0/24"
           },
         {
          "name":"k8s1minion1",
          "pod_subnet":"10.244.2.0/24"
          },
         {
          "name":"k8s1minion2",
```

```
            "pod_subnet":"10.244.1.0/24"
            }
            ],
         "errors":[]
        }
]
```

Curl Example for POST

```
curl --request POST --url https://pce.my-company.com:8443/api/v2/orgs/1/container_
clusters --header 'authorization: Basic
jI0MjJhZGNkYWU5OjA5ZmRjNjA4MDhiMzExZTc2Y2UyNzNmOWNiN2ZhMTA5OTdkMWNlMDAzZmMzOTQ1ZGM
xYzEwZGJhZTg5NzlmZjM=' --header 'content-type: application/json' --data '{"name":
"test","description": "test"}'
```

Curl Example for PUT

```
curl --request PUT --url https://pce.my-company.com:8443/api/v2/orgs/1/container_
clusters/1b851d4b-f22d-47be-b744-f3c2dca490a0 --header 'authorization: Basic
YXBpXzE2YjBkYjI0MjJhZGNkYWU5OjA5ZmRjNjA4MDhiMzExZTc2Y2UyNzNmOWNiN2ZhMTA5OTdkMWNlMD
AzZmMzOTQ1ZGMxYzEwZGJhZTg5NzlmZjM=' --header 'content-type: application/json' --
data '{"name": "test","description": "test"}'
```

Example Response for POST

```
{
       "href": "/orgs/1/container_clusters/1b851d4b-f22d-47be-b744-f3c2dca490a0",
               "pce_fqdn": null,
               "name": "test",
               "description": "test",
               "manager_type": null,
               "last_connected": null,
               "kubelink_version": null,
               "online": false,
               "nodes": [],
               "errors": [],
               "container_cluster_token": "1_
```

```
0dfec0acb8e4bc53e052874874da0c24e7ac98da3b3954e3c9ea6f9860722e84"
}
```

Container Cluster Workload Profiles

When you install an Illumio VEN on a container cluster, all pods in the container cluster are unmanaged or not visible in the PCE. However, all namespaces that exist on the container clusters are reported by Kubelink and made visible via the Container Container Workload Profiles API.

Each container workload profile maps to a Kubernetes namespace and can be either managed or unmanaged. The default state for a profile is unmanaged.

Use these methods to get, create, update, or delete container cluster workload profiles:

Functionality	HTTP	URI
Get the list of container cluster workload profiles	`GET`	`GET /orgs/:xorg_id/container_clusters/: container_cluster_id/container_workload_ profiles`
Create container cluster workload profiles	`POST`	`POST /orgs/:xorg_id/container_clusters/: container_cluster_id/container_workload_ profiles`
Update the specified container cluster workload profile	`PUT`	`PUT /orgs/:xorg_id/container_clusters/: container_cluster_id/container_workload_ profiles/:container_workload_profile_id`
Supports the UI feature for bulk update of container workload profiles	`PUT`	`PUT /orgs/:xorg_id/container_clusters/: container_cluster_id/container_workload_ profiles_ update`
Delete the specified container cluster workload profile	`DELETE`	`DELETE /orgs/:xorg_id/container_clusters/: container_cluster_id/container_workload_ profiles/:container_workload_profile_id`

Query Parameters for Container Workload Methods

Parameter	Description	Type	Required
`org_id`	Organization ID	Integer	Yes
`container_cluster_id`	Cluster UUID	String	Yes
`assign_labels`	(GET) List of lists of label URIs, encoded as a JSON string (POST, PUT) Assigned labels	String	No No
`enforcement_mode`	(GET) Filter by enforcement mode. (PUT) workload enforcement mode	String	No No

Parameter	Description	Type	Required
`linked`	Filter by linked container workload profiles.	Boolean	No
`managed`	Filter by managed state	Boolean	No
`max_results`	Maximum number of container workloads to return..	Integer	No
`name`	(GET) Name string to match. Supports partial matches. (POST) A friendly name given to a profile if the namespace is not user friendly	String	No YES
`namespace`	Namespace string to match. Supports partial matches.	String	No
`visibility_level`	Filter by visibility level	String	No

Response Properties for Container Workload Methods

Parameter	Description	Type
`href`	`Container Workload Profile URI`	String
`enforcement_mode`	Reference to `common/workload_enforcement_mode.schema.json`	
`managed`	If the namespace is managed or not	Boolean
`max_results`	Maximum number of container workloads to return..	Integer
`name`	A friendly name given to a profile if the namespace is not user friendly	String, Null
`namespace`	Namespace name	String, Null
`container_workload_profile_id`	Container workload profile UUID	String
`labels`	Labels to assign to the workload that matches the namespace. Reference to `common/label_restrictions.schema.json`	

Curl Examples and Responses

Curl example for GET

```
curl --request GET --url https://pce.my-
company.com:8443/api/v2/orgs/1/containermeters iun one table and verified with the
Quick Reference._clusters/445bfa9b-4de4-4c09-9705-496eb04b190f/container_workload_
profiles --header 'authorization: Basic
NjA4MDhiMzExZTc2Y2UyNzNmOWNiN2ZhMTA5OTdkMWNlMDAzZmMzOTQ1ZGMxYzEwZGJhZTg5NzlmZjM='
--header 'content-type: application/json'
```

Curl Example for POST

```
curl --request POST --url https://pce.my-company.com:8443/api/v2/orgs/1/container_
clusters/445bfa9b-4de4-4c09-9705-496eb04b190f/container_workload_profiles --header
'authorization: Basic
A5ZmRjNjA4MDhiMzExZTc2Y2UyNzNmOWNiN2ZhMTA5OTdkMWNlMDAzZmMzOTQ1ZGMxYzEwZGJhZTg5Nzlm
ZjM=' --header 'content-type: application/json' --data '{"name":
"test","description": "test","assign_labels": [{"href":
"/orgs/1/labels/1"}],"mode": "full","log_traffic": true}'
```

Curl Example for PUT

```
curl --request PUT --url https://pce.my-company.com:8443/api/v2/orgs/1/container_
clusters/445bfa9b-4de4-4c09-9705-496eb04b190f/container_workload_
profiles/219b49c3-3bb5-4fc0-9913-b76398105e35 --header 'authorization: Basic
mRjNjA4MDhiMzExZTc2Y2UyNzNmOWNiN2ZhMTA5OTdkMWNlMDAzZmMzOTQ1ZGMxYzEwZGJhZTg5NzlmZjM
=' --header 'content-type: application/json' --data '{"name":
"test","description": "test","assign_labels": [{"href":
"/orgs/1/labels/1"}],"mode": "full","log_traffic": true}'
```

Example Response for GET


```
                [
    {
        "href": "/orgs/10/container_clusters/974aec34-e8e7-478d-9ca2-
90ebb3642edc/container_workload_profiles/5454cc84-d6be-4e6c-ac62-465f9504fac0",
        "namespace": "openshift-host-network",
        "enforcement_mode": "visibility_only",
```

```
        "visibility_level": "flow_summary",
        "managed": true,
        "assign_labels": [
            {
                "href": "/orgs/10/labels/128"
            },
            {
                "href": "/orgs/10/labels/225"
            }
        ],
        "labels": [
            {
                "key": "loc",
                "assignment": {
                    "href": "/orgs/10/labels/128",
                     "value": "AWS"
                }
            },
            {
                "key": "env",
                "assignment": {
                    "href": "/orgs/10/labels/225",
                    "value": "OCP4.6"
                }
        }
    ],
        "linked": true,
        "created_at": "2021-08-25T18:11:52.665Z",
        "created_by": {
             "href": "/orgs/10/container_clusters/974aec34-e8e7-478d-9ca2-
90ebb3642edc"
        },
        "updated_at": "2021-08-25T18:11:52.665Z",
        "updated_by": {
            "href": "/orgs/10/container_clusters/974aec34-e8e7-478d-9ca2-
90ebb3642edc"
        }
    }
```

```
]
```

Examples for container_workload_profiles/update

Example Request

```
            {
       "container_workload_profiles": [
       {
               "href": "url_to_some_container_workload_profile"
               },
               {
               "href": "url_to_other_container_workload_profile"
       }
       ],
       "labels": [
       {
               "key": "role",
               "assignment": {
               "href": "url_to_label"
               }
       }
       ],
       "enforcement_mode": 2,
               "visibility_level": "flow_summary",
               "managed": true
}
```

Example Response

- For success: Response code 204; Response body: none
- If an error occurred on any of the input records:
    - Response code 406;
    - Response body:

```
                                [
        {
        "token": "input_validation_error",
        "message": "....., record_index=>1, ..., unmanaged_container_workload_
profile_labels, ..."
                # message contains index of failed record and specific
error message
        },
        ...
]
```

Label Restrictions

Kubernetes pods and services running in a namespace (Kubernetes) or project (OpenShift) must be labeled (RAEL) to be included in policy within Illumio Core. The container workload profile defines how labels will be assigned to pods and services within a namespace.

Illumio labels can be statically assigned from the PCE or defined in the Kubernetes manifest files using annotations. For each label key (RAEL), the PCE administrator can define four options:

1. No label will be assigned.
2. One label will be assigned from PCE.
3. A restricted list of labels can be assigned from Kubernetes using annotations. Label restrictions prevent misuse of Illumio labels by the people managing the Kubernetes platform and makes sure the labels inherit the policy they should be receiving.
4. Any label can be assigned from Kubernetes.

You can set role labels for the following APIs:

- `PUT /api/v2/orgs/:xorg_id/container_clusters/<:cluster_id>/container_workload_profiles`
- `POST /api/v2/orgs/:xorg_id/container_clusters/<:cluster_id>/container_workload_profiles`

Examples

Set an empty Role label

```
            {
      "labels": [
```

```
                {"key": "role", "assign": {} }
        ]
}
```

Set a Location label

```
PUT /api/v2/orgs/1/container_clusters/65d1f197-938a-49ef-9343-
6f55ec76fd90/container_workload_profiles/afe4661a-03ef-462f-ada6-ce7334aa9704

{
        "labels": [
                { "key": "loc", "restriction": {"href": "/orgs/1/labels/221"} }
        ]
}
```

Set an allow list for the Environment label

Allow a list of Environment labels to be assigned using Kubernetes:

```
PUT /api/v2/orgs/1/container_clusters/65d1f197-938a-49ef-9343-
6f55ec76fd90/container_workload_profiles/afe4661a-03ef-462f-ada6-ce7334aa9704

{
        "labels": [
                { "key": "env", "restriction": [{"href": "/orgs/1/labels/176"}, {"href":
"/orgs/1/labels/302"}, {"href": "/orgs/1/labels/303"}] }
        ]
}
```

Allow any value for the Application label

```
PUT /api/v2/orgs/1/container_clusters/65d1f197-938a-49ef-9343-
6f55ec76fd90/container_workload_profiles/afe4661a-03ef-462f-ada6-ce7334aa9704

{
        "labels": [
                { "key": "app", "restriction": [] }
```

```
      ]
}
```

Multiple ways to assign or allow labels used together in one Container Workload Profile

```
PUT /api/v2/orgs/1/container_clusters/65d1f197-938a-49ef-9343-
6f55ec76fd90/container_workload_profiles/afe4661a-03ef-462f-ada6-ce7334aa9704

{
       "labels": [
              {"key": "role", "assign": {} },
              {"key": "app", "restriction": [] },
              {"key": "env", "restriction": [{"href": "/orgs/1/labels/176"}, {"href":
"/orgs/1/labels/302"}, {"href": "/orgs/1/labels/303"}] },
              {"key": "loc", "assign": {"href": "/orgs/1/labels/221"} }
       ]
}
```

Result for the above example:

- `role`: No label will be set; it is an explicit statement (you don't want a `role` label to be assigned).
- `app`: Any value can be set in the annotations for the `app` label key (provided the value exists in PCE).
- `env`: Only the values specified in the allowlist can be set in the annotations for the `env` label key.
- `loc`: The value of the `loc` label key is assigned to the value defined in the payload.

Label Assignment Configuration

To clear the label assignment option and go back to the default option (any labels passed at runtime using Kubernetes annotations will be allowed), 2 options:

Option 1: explicit statement

```
            {
       "labels": [
              { "key": "role", "restriction": [] }
```

```
        ]
}
```

Option 2: empty payload

```
            {
        "labels": []
}
```

Backend Services Associated with Container Clusters

Kubernetes services are represented as virtual services in the Illumio policy model. For the services in Kubernetes, Kubelink creates virtual services in the PCE and reports the list of Replication Controllers, DaemonSets, and ReplicaSets responsible for managing the pods that support the services.

When there is a match between the Replication Controller and ReplicaSet managing a pod, the PCE creates a binding between the virtual service and the container workload.

The Service Backend represents a match between a virtual service and an application type, such as Deployment or ReplicaSet.

Use this method to get the service backend:

Functionality	HTTP	URI
Get data about the service backend	`GET`	`GET /orgs/1/container_clusters/:container_cluster_id/service_backends`

Properties for Backend Services

Parameters	Description	Type	Required
`name`	The name of the container cluster backend.	String	Yes
`kind`	The type (or kind) of the container cluster backend.	String	Yes
`updated_at`	The time (rfc339 timestamp) at which the container cluster backend was updated.	String	Yes
`created_at`	The time (rfc339 timestamp) at which the container cluster backend was created.	String	Yes
`virtual_services`	Includes the following properties:	Object	Yes

Parameters	Description	Type	Required
	• `href`:The URI to the associated virtual service • `name`: The virtual service name	String String	

Curl Examples

Curl Example for GET

```
curl --request GET --url https://pce.my-company.com:8443/api/v2/orgs/1/container_
clusters/445bfa9b-4de4-4c09-9705-496eb04b190f/service_backends --header
'authorization: Basic
YzE2YjBkYjI0MjJhZGNkYWU5OjA5ZmRjNjA4MDhiMzExZTc2Y2UyNzNmOWNiN2ZhMTA5OTdkMWNlMDAzZm
MzOTQ1ZGMxYzEwZGJhZTg5NzlmZjM='
```

Example Response for GET

```
            [
    {
        "name": "58687784f9",
        "kind": "replicasethash",
        "namespace": "kube-system",
        "updated_at": "2020-10-25T20:07:39.741Z",
        "created_at": "2020-10-25T20:07:39.741Z",
        "virtual_service": {
        "href": "/orgs/1/sec_policy/draft/virtual_services/926c2f63-bcd8-42f1-
8811-165b34f84334",
        "name": "coredns-k8s2-kube-system"
        }
    },
    {
        "name": "556b9ff8f8",
        "kind": "replicasethash",
        "namespace": "kube-system",
        "updated_at": "2020-10-25T20:07:39.768Z",
        "created_at": "2020-10-25T20:07:39.768Z",
        "virtual_service": {
        "href": "/orgs/1/sec_policy/draft/virtual_services/58b0df03-1151-464e-
8352-069e3ad0d7ed",
        "name": "kubernetes-dashboard-k8s2-kube-system"
```

```
        }
    }
]
```

Kubernetes APIs

Kubernetes Workload Endpoints

Customers have been requiring to see the details of Kubernetes workloads in PCE so that they can write policies and troubleshoot any issues.

Two new endpoints have been created for Kubernetes workloads:

`GET /api/v2/orgs/:xorg_id/kubernetes_workloads`

This API lists all new Kubernetes Workloads in separate tab/page with separate sorts and filters.

It contains required properties such as `name`, `kind`, `namespace`, as well as optional properties `href`, `labels`, `enforcement_mode`,`visibility_level`, `container_workload_profile`, `container_cluster`, `security_policy_applied_at`, `security_policy_sync_state`, `created_at`, `k8s_label`, and `k8s_annotations`.

```
        {
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "required": [
            "name",
            "kind",
            "namespace"
    ],
    "properties": {
            "href": {
                    "description": "URI of the container workload",
                    "type": "string"
            },
            "name": {
                    "description": "Container workload name",
                    "type": "string"
            },
```

```
                "namespace": {
                        "description": "k8s namespace where this k8s Workload belongs to",
                        "type": "string"
                },
                "kind": {
                        "description": "k8s resource kind, e.g. Deployment",
                        "type": "string"
                },
                "labels": {
                        "type": "array",
                        "items": {
                        "$ref": "../common/label_optional_key_value.schema.json"
                   }
                },
                "enforcement_mode": {
                        "$ref": "../common/workload_enforcement_mode.schema.json"
                },
                "visibility_level": {
                        "$ref": "../common/workload_visibility_level.schema.json"
                },
                "container_workload_profile": {
                        "$ref": "container_clusters_container_workload_profiles_get.schema.json"
                },
                "container_cluster": {
                        "$ref": "container_clusters_get.schema.json"
                },
                "security_policy_applied_at": {
                        "description": "Last reported time when policy was processed by CLAS to the
workload (UTC)",
                        "type": [
                         "string",
                         "null"
                        ],
                        "format": "date-time"
                },
                "security_policy_sync_state": {
                        "description": "Current state of security policy",
                           "type": "string"
```

```
        },
        "created_at": {
                "description": "RFC 3339 timestamp at which this record was created",
                "format": "date-time",
                "type": "string"
        },
        "updated_at": {
                "description": "RFC 3339 timestamp at which this record was updated",
                "format": "date-time",
                "type": "string"
        },
        "k8s_labels": {
                "type": "array",
                "items": {
                "type": "object",
                "required": [
                   "key",
                   "value"
                ],
                "properties": {
                   "key": {
                   "type": "string"
                },
                "value": {
                "type": "string"
                }
             }
           }
        },
        "k8s_annotations": {
                "type": "array",
                "items": {
                "type": "object",
                "required": [
                   "key",
                   "value"
                ],
                "properties": {
```

```
                        "key": {
                        "type": "string"
                    },
                    "value": {
                        "type": "string"
                    }
                }
            }
        }
    }
}
```

For this API, these changes have been made in release 23.5.0:

- two arrays have been removed, `k8s_labels` and s`k8s_annotation`, and replaced with the property `metadata`

```
        "metadata": {
               "$ref": "
               ../common/kubernetes_workloads_metadata.schema.json"
```

- HREF description has been changed from `URI of the container workload`, to `URI of the kubernetes workload`.

`GET /api/v2/orgs/:xorg_id/kubernetes_workloads/:kubernetes_workload_uuid`

This API provides a detailed page for the specified Kubernetes workload with custom K8S attributes.

`common non_empty_label_scopes.schema.json`

This new common schema provides a collection of assigned list of labels. Minimum number is one.

```
{
       "$schema": "http://json-schema.org/draft-04/schema#",
       "description": "Collection of assigned list of labels",
       "type": "array",
```

```
        "items": {
                "$ref": "labels.schema.json",
                "minItems": 1
        },
        "uniqueItems": true,
        "minItems": 1
}
```

common kubernetes_workloads_metadata

The new common schema `kubernetes_workloads_metadata` was added in release 23.5.0 that is referenced from `kubernetes_workload_get`.

It provides Kubernetes properties such as labels, annotations, and external service's UID.

```
              {
        "$schema": "http://json-schema.org/draft-04/schema#",
        "description": "k8s object metadata",
        "additionalProperties": false,
        "type": "object",
        "properties": {
        "labels": {
           "description": "k8s key/value pairs attached to object that specify
identifying attributes",
           "type": "object"
            },
        "annotations": {
           "description": "k8s key/value pairs representing arbitrary non-identifying
metadata of object",
           "type": "object"
           },
        "external_service_uid": {
           "description": "k8s object uid of external traffic service (NodePort or
LoadBalancer)",
            "type": "string"
           }
     }
}
```

For more information, see the Illumio Core for Kubernetes and OpenShift guide.

Access Restrictions and Trusted Proxy IPs

To employ automation for managing the PCE environment, you can use API Keys created by an admin user and automate the PCE management tasks. Illumio provides a way to restrict the usage of these API keys by IP address so that you can block API requests coming in from non-allowed IP addresses.

Access Restrictions

Access restrictions are configurable entities and contain a list of up to 8 IPv4 IP addresses or CIDR blocks that specify the source IP addresses of the allowed clients. Only the global Org Owner can manage access restrictions in the organization while other roles can neither edit nor view them.

The following rules apply to access restrictions:

- Each access restriction can be applied to either one or both:
  - API requests authenticated by API keys
  - API requests authenticated by `Username/Password` credentials
- The global Org Owners can edit an access restriction after it has been created by modifying the allowed IP list or the options. They can also assign access restrictions to Local and External Users. The API supports the update of access restrictions for a list of users.
- Access restrictions are leader-owned configuration objects that are replicated to all super-cluster regions.
- Access restrictions are enforced as follows:
  - To enforce an API request, determine the user account for that API request using the API key or the user session token and then find the access restriction that is configured for that user. If there is no access restriction assigned to the user, the API request proceeds.
  - If the client IP address for an API request does not satisfy the corresponding user's access restrictions, the request is rejected with a 401 error message.
  - Access restrictions are not enforced on some URLs (node_available, static JS/CSS content).
- When a request is rejected due to unsatisfied access restrictions, it generates an Event that specifies a failure caused by an invalid source IP address, including the actual IP address and an appropriate error code (403).

Assignment to Users

Each Access Restriction is a configuration object that specifies a set of allow-list IP addresses or CIDR blocks, designating the allowed client IP address. It also specifies the types of API accesses that are restricted (those authenticated by API Keys or those authenticated by user session tokens).

The Org Owners create and manage access restrictions in their organizations so that there are maximum of 50 access restrictions per organization. The Org Owners can assign a single access restriction to each Local or External User (by default, a user has no access restriction assigned).

Access Restriction Methods

Functionality	HTTP	URI
Get a list of access restrictions	`GET`	`/api/v2/orgs/<org_id>/access_restrictions`
Get a specific access restriction	`Get`	`/api/v2/orgs/<org_id>/access_restrictions/<id>`
Create an access restriction	`POST`	`/api/v2/orgs/<org_id>/access_restrictions`
Update an access restriction. Same schema as for `POST`, but fields such as `name` or`ips` might not be required	`PUT`	`/api/v2/orgs/<org_id>/access_restrictions/<id>`
The `DELETE` endpoint should return an error if the specified `access_restriction` is referenced by any User or Group. The existing `access_restrictions` from all Users and Groups must be removed before they can be deleted.	`DELETE`	`/api/v2/orgs/<org_id>/access_restrictions/<id>`

Return Values for Access Restriction

These are the return values for the Access Restriction methods:

Property	Method	Description	Required
`href`	`GET`	URI of access restriction	Yes
`name`	`GET,`	User assigned name of the access restric-	`(No GET)(Yes`

Property	Method	Description	Required
	POST,	tion	POST)
description	GET, POST	User assigned description of the access restriction	No
ips	GET, POST	Array of ip addresses or CIDR blocks	Yes
enforcement_exclu-sions	GET, POST	The types of API access methods that are excluded from access restriction enforcement	No

Manage Access Restrictions

Create an Access Restrictions

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/1/access_restrictions/
```

Response

```
{
       "name": "sample Access Restriction payload",
       "description": "example",
       "ips": [ "192.168.33.1/16" ],
       "enforcement_exclusions": [ "user_sessions" ]
}
```

Read an Access Restriction

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/1/access_restrictions/
```

Update an Access Restriction

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/1/access_restrictions/1
```

```
{
       "name": "modified Access Restriction payload",
       "description": "example",
       "ips": [ "192.168.33.1/16" ],
```

```
        "enforcement_exclusions": [ "user_sessions" ]
}
```

Delete the Access Restriction

```
curl -i -X DELETE https://pce.my-company.com:8443/api/v2/orgs/1/access_
restrictions/1
```

Curl Command to associate an Access Restriction with an Org Auth Sec Principal (PUT)

```
curl -i -X -PUT https://pce.my-company.com:8443/api/v2/orgs/1/auth_security_
principals/76a0607b-6961-4c74-a98a-8b10775c8a9b
```

```
{
        "name": "test.user@illumio.com",
        "display_name": "test",
        "type": "user",
        "access_restriction": {
        "href": "/orgs/1/access_restrictions/1"
}
}
```

Trusted Proxy IPs

When a client is connected to the PCEs haproxy server, this connection can traverse one or more load balancers or proxies. Therefore, the source IP address of a client connection to haproxy might not be the actual public IP address of the client.

Proxies and intermediaries often use the `X-Forwarded-For` header (and some other custom headers, like X-Client-IP) to pass along the client IP address. The value of this header is a comma-separated list of one or more IP addresses, where the source IP address seen by the most recent proxy is at the end of the list.

The client IP address used for API requests and Web UI connections comes from the value of the `X-Forwarded-For` header that haproxy sets on the back-end request to the webservice. It is set to the one of these values:

- Value of the `X-Forwarded-For` header on the incoming request (when `trust_upstream_x_forwarded_for` is `true`)

- Source IP address of the client connection to haproxy (when `trust_upstream_x_forwarded_for` is `false`)

Configurable trusted proxy IPs allow Org Owners to configure a list of IPv4 addresses or CIDR blocks that are considered trusted for setting a client's `X-Forwarded-For` header. Using this setting, the Org Owner can designate the trusted proxies/intermediaries and the PCE will consider all others to be un-trusted for the purpose of setting the `X-Forwarded-For` header.

The haproxy is configured to always put the client's source IP address in the `X-Real-IP` header on the back-end request and to pass along any `X-Forwarded-For` headers that are in the front-end request.

Trusted Proxy IP Methods

Functionality	HTTP	URI
Get a list of trusted IP proxies	`GET`	`/api/v2/orgs/<org_id>/-settings/trusted_proxy_ips`
Interservice API for fetching an orgs' `trusted_proxy_ips` settings, so that it may be cached locally. It uses the same schema as the GET endpoint above; it receives the org_id as a quer input	`GET`	`/api/v2/org_trusted_proxy_ips?org_id=<id>`
Update `trusted_proxy_ips` settings for a given org, with the same schema as the GET endpoint (except without the `max_trusted_proxy_ips_per_region` property)	`PUT`	`/api/v2/orgs/<org_id>/-settings/trusted_proxy_ips`

Trusted Proxy IPs

These are the return values for the Trusted Proxy methods:

Parameter	Method	Description	Req
`max_trusted_proxy_ips_per_region`	`GET`	Maximum number of Trusted Proxy IPs allowed for each PCE	Yes
`trusted_proxy_ips`	`GET, PUT`	IPs or CIDRs trusted (per-region) for handling clients' X-Forwarded-For header> Required:	Yes

Parameter	Method	Description	Req
		`pce_fqdn`: FQDN of PCE region, or null if not in supercluster `ip`: IP address or CIDR trusted for handling the clients' header X-Forwarded-For	

Manage Trusted Proxy IPs

Read a Trusted Proxy IP

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/1/access_restrictions/
```

Update a Trusted Proxy IP

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/1/settings/trusted_
proxy_ips/
```

```
{
        "trusted_proxy_ips": [
                {
                        "pce_fqdn": null,
                        "ip": "66.151.147.0/24"
                },
                {
                        "pce_fqdn": null,
                        "ip": "192.168.34.0/24"
                }
        ]
}
```

Organization Access

Changes to the organization access introduced a new common schema:

common ipv4_ipv6_subnet

```
{
       "$schema": "http://json-schema.org/draft-04/schema#",
       "type": "string",
       "oneOf": [
              { "format": "ipv4" },
              { "format": "ipv6" }
       ]
}
```

This common schema is replacing the one that is now deleted: `common ipv4_subnet`

```
{
       "$schema": "http://json-schema.org/draft-04/schema#",
       "type": "string",
       "pattern": "^((25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\\.)
                      {3}(25[0-5]|2[0-4][0-9]|[01]?[0-9]
                      [0-9]?)(\\/(3[0-2]|[0-2]?[0-9]))?$"
}
```

Three organization access APIs have been changed to substitute `common/ipv4_subnet.schema` with `common/ipv4_ipv6_subnet.schema`:

- orgs_access_restrictions_post
- orgs_access_restrictions_put

```
{
       "properties": {
           "ips": {
               "items": {
                   "$ref": {
                       "__old": "../common/ipv4_subnet.schema.json",
                       "__new": "../common/ipv4_ipv6_subnet.schema.json"
                   }
               }
```

```
            }
        }
}
```

settings_trusted_proxy_ips_put

```
{
       "properties": {
           "trusted_proxy_ips": {
               "items": {
                   "properties": {
                       "ip": {
                       "$ref": {
                       "__old": "../common/ipv4_subnet.schema.json",
                       "__new": "../common/ipv4_ipv6_subnet.schema.json"
                   }
               }
           }
       }
    }
```

Chapter 5

Provisioning

This chapter contains the following topics:

Use the Public Stable Provisioning API to implement all current changes to your security policy, such as additions, changes, and deletions.

The Public Experimental Provisioning API supplies information about unprovisioned changes to security policy items.

Finally, the Policy Update Mode API controls when policy updates are applied to workloads.

Provisioning (public stable)

This Public Stable API provisions all current changes (additions, changes, and deletions) to your security policy.

This API can also return a collection of provisioning versions or an individual provisioning version.

To get information about unprovisioned changes to security policy items, find provisioning dependencies, delete unprovisioned security policy items, revert the last provisioned items, and check whether a security rules exists that allows communications between two workloads, see Provisioning.

Provisioning API Methods

Functionality	HTTP	URI
Provision the current set of modified security policy items	POST	[api_version][org_href]/sec_policy
Get a list of all provisioned security policy versions	GET	[api_version][org_href]/sec_policy
Get a specific version of a provisioned security policy	GET	[api_version][sec_policy_version_href]

Provision All Items

Policy item additions, modifications, and deletions must be provisioned before they take effect on workloads.

URI to Provision All Items

```
POST api_version][org_href]/sec_policy
```

Provision All Items

This example passes a provisioning comment using the curl `-d` option (lowercase d) followed by the comment `'{"update_description":"make active"}'`. This operation provisions all draft policy items.

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/2/sec_policy -H
"Content-Type: application/json" -u $KEY:$TOKEN -d '{"update_description":"make
active"}'
```

Response

After provisioning the draft security policy, the response provides information related to the operation, including the version HREF of the provisioning.

You can use a provision history HREF to get all modified items for a particular version.

The response also indicates how many workloads were affected, when the provisioning was done, which user did it, and any message that was provided.

```
{
     "href": "/orgs/2/sec_policy/80",
     "commit_message": null,    "version": 80,
```

```
      "workloads_affected": 3,
      "object_counts": 3,
      "created_at": "2020-26T21:48:46.446Z",
      "created_by": { "href": "/users/18" }
}
```

Provision Individual Items

Curl Example

The request body uses `update_description` instead of `commit_message`, and instead of `entities`, define an array of pending HREFs for each method as appropriate.

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/2/sec_policy -H
"Content- Type:application/json" -u $KEY:$TOKEN -d '{"change_subset":{"rule_sets":
[{"href": "/orgs/2/sec_policy/draft/rule_sets/843"}], "ip_lists": [{"href":
"/orgs/2/sec_policy/draft/ip_lists/151"}]}, "update_description":"Provisioning a
ruleset and an ip list"}'
```

Request Body Prototype

The security policy POST request body has this format. Only define the methods used in the call and don't include any unused methods in the request body.

```
{
       "update_description": "string",
       "change_subset": {
              "label_groups": [
              {
                     "href": "string"
              }
       ],
       "services": [
              {
                     "href": "string"
              }
       ],
       "rule_sets": [
              {
                     "href": "string"
```

```
        }
    ],
    "ip_lists": [
        {
            "href": "string"
        }
    ],
    "virtual_services": [
        {
            "href": "string"
        }
    ],
    "firewall_settings": [
        {
            "href": "string"
        }
    ],
    "enforcement_boundaries": [
        {
            "href": "string"
        }
    ],
    "secure_connect_gateways": [
        {
            "href": "string"
        }
    ],
    "virtual_servers": [
        {
            "href": "string"
        }
    ]
```

Restore the Previous Security Policy

This API creates draft changes of the previous security policy's changes. When this API is called, there should not be any draft changes present in the PCE.

Curl Command to Restore the Security Policy

```
curl -i -X POSThttps://pce.my-company.com:8443/api/v2/orgs/1/sec_
policy/127/restore -H "Content-Type: application/json" -u $KEY:$TOKEN -d {}
```

Get All Provision Versions

This method gets the full history of all provisioned security policy versions.

URI to Get All Provisioned Versions

```
GET [api_version][org_href]/sec_policy
```

Get the Provision Versions

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/1/sec_
policy/127/restore -H "Content-Type: application/json" -u $KEY:$TOKEN -d {}
```

Response

Note that field `selective_enforcement_rules` was renamed to `enforcement_boundaries` in the `object_counts` property.

```
{
        "href": "string",
        "version": "string",
        "workloads_affected": 0,
        "commit_message": "string",
        "object_counts": {
                "rule_sets": 0,
                "ip_lists": 0,
                "services": 0,
                "virtual_services": 0,
                "label_groups": 0,
                "virtual_servers": 0,
                "firewall_settings": 0,
                "secure_connect_gateways": 0,
                "enforcement_boundaries": 0
        },
```

```
        "created_at": "string",
        "created_by": {
                "href": "string"
        }
}
```

Get an Individual Provision Version

This method gets a specific version of a provisioned policy.

Every time security policy is provisioned, it gets a unique version ID, which takes the form of an HREF. This HREF can be obtained from a GET of all security policy provisioned versions and then used in this call.

URI to Get an Individual Version of a Provisioned Policy

```
GET  [api_version][sec_policy_version_href]
```

Curl Command to Get Version

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/sec_policy/79 -H
"Accept: application/json" -u $KEY:$TOKEN
```

Response

```
{
        "href": "string",
        "version": "string",
        "workloads_affected": 0,
        "commit_message": "string",
        "object_counts": {
                "rule_sets": 0,
                "ip_lists": 0,
                "services": 0,
                "virtual_services": 0,
                "label_groups": 0,
                "virtual_servers": 0,
                "firewall_settings": 0,
                "secure_connect_gateways": 0,
                "enforcement_boundaries": 0
```

```
        },
        "created_at": "string",
        "created_by": {
                "href": "string"
        }
}
```

Provisioning

This Public Experimental API gets information about unprovisioned changes to security policy items (rulesets, IP lists, security settings, labels and label groups, services, virtual services, and user groups). You can also find provisioning dependencies, delete unprovisioned security policy items, revert the last provisioned items, and check whether a security rule exists that allows communications between two workloads.

To provision security policy items and get information about one or more provisioned items, see Provisioning (public stable).

Provisioning API Methods

Functionality	HTTP	URI
Get the collection of modified (`draft`) security policy items pending provisioning.	`GET`	`[api_version][org_href]/sec_policy/pending`
Check whether a rule exists between two workloads that allows communication.	`GET`	`[api_version][sec_policy_version_href]/allow`
Get the collection of all policy items that were modified in a specific version of a security policy.	`GET`	`[api_version][sec_policy_version_href]/modified_objects`
Delete all unprovisioned security policy item modifications (all unprovisioned `draft` changes) pending provisioning.	`DELETE`	`[api_version][org_href]/sec_policy/pending`
Revert a specified list of pending uncommitted security policy items. This method allows you to select specific items to revert.	`PUT`	`[api_version][org_href]/sec_policy/delete`
Determine if a specific set of objects can be provisioned, or if they are dependent on other objects that	`POST`	`[api_version]/sec_policy/draft/dependencies`

Functionality	HTTP	URI
need to be provisioned as well.		
Used to see the policy impact before provisioning. This API is referencing `sec_policy_change_subset.schema.json`, which contains the property `change_subset`	`POST`	`[api_version]/sec_policy/impact`

Provisionable Policy Items

The following security policy items all require provisioning before they can take effect on managed workloads (workloads with a VEN installed on them). The total sum of these policy items constitutes the security policy.

- **IP Lists**: IP addresses, IP ranges, and CIDR blocks allowed to access managed workloads.
- **Label Groups**: Labels can be managed in label groups.
- **Rulesets**: Policy item that includes labels and rules to define permitted communication between workloads and between groups.
- **Pairing Profiles**: A Pairing Profile applies certain properties to workloads as they pair with the PCE, such as labels and workload policy states.
- **Security Settings**: General network security settings, such as ICMP echo reply, allow or disable IPv6, and connectivity settings.
- **Services**: Definitions or discovery of existing services on your workloads.
- **Virtual Servers**: Allows rules that allow communication with workloads managed by a load balancer.
- **Virtual Services**: A virtual service is a single service (a port/protocol set) that can be used directly in a rule as a single entity. labels that represent multiple virtual services can also be used to write rules.
- **Enforcement Boundaries:** Facilitate the implementation of allow-lists by narrowing the scope for segmentation so that users can reach a high level of system maintainability using a simple policy mode.

When the security policy is provisioned, the PCE recalculates any changes made to policy configurations and then transmits those changes to the VENs installed on the workloads.

Policy Provisioning States

This API operates on provisionable objects, which exist in either a `draft` (not provisioned) state or an `active` (provisioned) state.

Provisionable items include label groups, services, rulesets, IP lists, virtual services, firewall settings, enforcement boundaries, and virtual servers. For these objects, the URL of the API call must include the element called `:pversion`, which can be set to either `draft` or `active`.

Depending on the method, the API follows these rules:

- For GET operations – `:pversion` can be draft, active, or the ID of the security policy.
- For POST, PUT, DELETE – `:pversion` can be draft (you cannot operate on active items) or the ID if the security policy.

Get All Items Pending Provisioning

This method gets a list of all modified policy items pending provisioning.

URI to Get All Policy Items Pending Provisioning

This API allows the user to view a list of all policy objects pending provisioning bucketed by type. The UI uses this to generate the "draft changes" page.

```
GET [api_version][org_href]/sec_policy/pending
```

Get Items Pending Provisioning

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/sec_
policy/pending -H "Accept:application/json" -u $KEY:$TOKEN
```

Response

```
---------
       ],
       "virtual_services": [
              {
                     "name": "string",
                     "href": "string",
                     "updated_by": null,
                     "updated_at": "2021-05-03T00:24:56Z",
                     "update_type": "create",
                     "caps": [
```

```
                        "write"
                        ]
                }
        ],
        "
                enforcement_boundaries
                ": [
                {
                        "name": "string",
                        "href": "string",
                        "updated_by": null,
                        "updated_at": "2021-05-03T00:24:56Z",
                        "update_type": "create",
                        "caps": [
                        "write"
                        ]
                }
        ]
}
```

The field `selective_enforcement_rules` was replaced with `enforcement_boundaries`.

Revert All Items Pending Provisioning

This method reverts (undoes) the current set of unprovisioned security policy modifications (all unprovisioned draft changes).

```
DELETE [api_version][org_href]/sec_policy/pending
```

Revert all items pending provisioning

```
curl -i -X DELETE https://pce.my-company.com:8443/api/v2/orgs/2/sec_policy/pending
-u $KEY:$TOKEN
```

Revert a List of Items Pending Provisioning

This API allows the user to revert a subset of policy objects via the change_subset field. via the `change_subset` field.

The field `selective_enforcement_rules` was replaced with `enforcement_boundaries`.

Revert a Specific List of Items Pending Provisioning

```
PUT [api_version][org_href]/sec_policy/delete
```

```
{
"change_subset": {
"label_groups": [
        {
                "href": "string"
        }
],
"services": [
        {
                "href": "string"
        }
],
"rule_sets": [
        {
                "href": "string"
        }
],
"ip_lists": [
        {
                "href": "string"
        }
],
"virtual_services": [
        {
                "href": "string"
        }
],
"firewall_settings": [
        {
                "href": "string"
        }
],
"secure_connect_gateways": [
        {
```

```
                        "href": "string"
                }
        ],
        "virtual_servers": [
                {
                        "href": "string"
                }
        ],
        "enforcement_boundaries": [
                {
                        "href": "string"
                }
        ]
    }
}
```

If an empty request body is given,

```
{}
```

then all objects will be reverted.

Curl Command to Revert a Pending Rule

```
            curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/1/sec_
policy/delete -H "Accept: application/json" -H "Content-Type: application/json" -u
api_
1fc24761346777702:'26c55be6892762b65f27aacc795076767f16ffcd7e9fde323a307e5fd286eb8
d' -d '{"change_subset":{"rule_sets":[{"href":"/orgs/1/sec_policy/draft/rule_
sets/3"}]}}'
```

Get Security Policy Dependencies

This public experimental API allows the user to determining the provisioning (or revert) dependencies for a particular policy object. The response JSON is also bucketed by object, and has the same schema change.

URI to Get Specific Security Policy Dependencies

```
POST /sec_policy/draft/dependencies
```

Security Policy Properties

Parameter	Description
`change_subset`	Defines a hash of provisionable or revertible objects identified by their HREFs. Includes label groups, services, rulesets, IP lists, virtual services, and virtual servers. Each individual object of a specific type (for example, `rule_sets`) is represented in the request body as an array of HREFs for those object types. For `POST /api/v2/orgs/:xorg_id/sec_policy/impact`: • If provided, the impact will be calculated only on `change_subset`. • If missing, the impact will be calculated on all of the pending items.
`operation`	Determines if there are dependencies for *provisioning* or *reverting* the specified objects: • `commit`: Specify this value to check for dependencies before *provisioning* an object. • `revert`: Specify this value to check for dependencies before *reverting* an object that is in a `draft` state.
Sub properties of `change_subset` that represent provisionable objects	
`label_groups`	List of label groups in the draft state to check for provisioning dependencies identified by label group HREF.
`services`	List of services in the draft state to check for provisioning dependencies identified by service HREF.
`rule_sets`	List of rulesets in the draft state to check for provisioning dependencies identified by rule_set HREF.
`ip_lists`	List of IP lists in the draft state to check for provisioning dependencies, identified by IP list HREF.
`virtual_services`	List of virtual services in the draft state to check for provisioning dependencies identified by virtual service HREF.

Parameter	Description
	Reference to `common/href_object.schema.json`
`virtual_servers`	List of virtual servers in the draft state that you want to check for provisioning dependencies identified by virtual server HREF. Reference to `common/href_object.schema.json`
`firewall_settings`	Reference to `common/href_object.schema.json`
`enforcement_boundaries`	Reference to `common/href_object.schema.json`

Request Body

```
            {
       "operation": "commit",
       "change_subset": {
               "enforcement_boundaries":
[
               {
               "href": "/orgs/2/sec_policy/draft/enforcement_boundaries/51"
               }
           ]
       }
}
```

Check for Provisioning Dependencies

```
            curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/7/sec_
policy/draft/dependencies -H "Content-Type: application/json" -u $KEY:$TOKEN -d '
{"operation":"commit", "change_subset": {"rule_sets":[{"href":"/orgs/1/sec_
policy/draft/rule_sets/9"}, {"href":"/orgs/1/sec_policy/draft/rule_sets/3"}],
"virtual_services": [{"href":"/orgs/1/sec_policy/draft/virtual_services/xxxxxxxx-
adeb-4895-8ff2-60c5b9833d9e"}, {"href":"/orgs/1/sec_policy/draft/virtual_
services/xxxxxxxx-12bc-4cfa-99ef-330c399bc78c"}]}'
```

Response

The response indicates that the field `selective_enforcement` was replaced with `enforcement_boudaries` following the change in the request.

```
        "$ref": "../common/href_object.schema.json"
        }
},
-       "selective_enforcement_rules":
{
+       "enforcement_boundaries":
{
                "type": "array",
                "items": {
                "$ref": "../common/href_object.schema.json"
```

If there are no dependencies for either commit or revert, the response returns an empty array.

```
            []
```

Get Modified Items in a Provisioned Version

This method gets a collection of all modified policy items in a specific version of the security policy.

Every time the security policy is provisioned, it gets a version, which takes the form of an HREF. The HREF can be obtained when getting all provisioned versions of your security policy. You can use that provision version HREF when calling this method.

URI to Get All Modified Items in a Specific Provisioned Version

```
GET [api_version][sec_policy_version_href]/modified_objects
```

Curl Command Example

```
curl -X GET /orgs/{org_id}/sec_policy/{pversion}/modified_objects -u $KEY:$TOKEN -
H 'Accept: application/json'
```

Response (similar to the following)

```
            {
        "update_type": null,
        "object_type": null,
        "href": null,
        "name": "string",
        "updated_at": "2021-05-03T00:24:56Z",
        "updated_by": null,
}
```


Required properties `updated_at` and `updated_by` have been added and the properties `modified_by` and `modified_at` have been deleted.

Get Rules Allowing Communication

This method gets a list of all rules that allow communication between two workloads (and other entities) for a specific version of a provisioned security policy.

By default, the maximum number returned on a GET collection with this API is 500. If you want to get more than 500 results, use an Asynchronous GET Collection.

Check for Rules Between Workloads

```
GET /api/v2/orgs/{org_id}/sec_policy/{pversion}/allow
```

Query Parameters

Provide query parameters in the URI that specify the source workload IP address or HREF, the service HREF, and the destination workload HREF. You can obtain a workload HREF with a GET call on the Workloads API.

Parameter	Description	Type	Required
`org_id`	Organization	Integer	Yes
`pversion`	Security policy version	String	Yes

`src_external_ip`
OR
`src_workload`
	The external IP of the source workload
	or
	The URI of the source workload
		String	No

`dst_`	The external IP of the des-	String	No

Parameter	Description	Type	Required

`external_ip`
OR
`dst_work-load`
	tination workload
	OR
	The URI of the destination work-load		

`service`	The specific service to check	String	No
`port`	The specific port number to check	Integer	No
`protocol`	The specific protocol number to check	Integer	No

Curl Command to Get Rules Between Workloads

The workloads and the service are identified by their HREFs:

```
            curl -X GET /orgs/{org_id}/sec_policy/{pversion}/allow -u $KEY:$TOKEN
-H 'Accept: application/json'
```

Response

```
            [
      {
      "href": "string",
      "enabled": true,
      "description": "string",
      "service": {
              "href": "string"
      },
      "ub_service": null,
      "sec_connect": true,
      "providers": [
      {
              "actors": "string",
              "label": {
              "href": "string"
              },
              "agent": {
                      "href": "string"
```

```
                },
                "workload": {
                        "href": "string"
                },
                "bound_service": {
                        "href": "string"
                },
                "virtual_server": {
                        "href": "string"
                },
                "ip_list": {
                        "href": "string"
                }
        }
        ],
        "consumers": [
                {
                "actors": "string",
                "label": {
                        "href": "string"
                },
                "agent": {
                        "href": "string"
                },
                "workload": {
                        "href": "string"
                },
                "bound_service": {
                        "href": "string"
                },
                "ip_list": {
                        "href": "string"
                }
        }
]
}
```

Example for POST /api/v2/orgs/1/sec_policy/impact

Each of the allowed properties such as `ip_lists`, `label_groups`, and `services` can be included in the request body of the POST call and the response schema defines the format and values of this API request for the example in the request body.

`sec_policy_impact_post_response.schema.json`

```
            {
        "$schema": "http://json-schema.org/draft-04/schema#",
        "type": "object",
        "required": ["num_sets", "num_managed_workloads", "num_container_workloads",
                                "num_unmanaged_workloads"],
        "properties": {
            "num_sets": {
                "description": "number of affected sets",
                "type": "integer"
            },
            "num_virtual_servers": {
                "description": "number of affected virtual servers",
                "type": "integer"
            },
            "num_managed_workloads": {
                "description": "number of affected workloads of type Workload",
                "type": "integer"
            },
            "num_container_workloads": {
                "description": "number of affected workloads of type ContainerWorkload",
                "type": "integer"
            },
            "num_unmanaged_workloads": {
                "description": "number of affected unmanaged workloads",
                "type": "integer"
           },
            "all_workloads_optimization": {
                "description": "flag to indicate if all-workloads-optimization has been used",
                "type": "boolean"
            }
        }
}
```

Policy Update Mode

This Public Experimental API controls when policy updates are applied to workloads.

Overview of Policy Update Mode

The PCE has two policy update options:

- **Adaptive**: Apply policy changes as soon as you provision.
- **Static**: Apply policy changes at a later time, such as during a scheduled maintenance window.

By default, the PCE policy update mode is set to `Adaptive`, but you can configure `Static` policy update mode for certain sets of workloads identified by scopes. Workloads that share the same labels configured for static policy update scope *receive* policy changes from the PCE, but those changes *will not be applied* until a user or an orchestration system instructs the PCE to apply those changes.

Configuring static policy update mode requires defining a scope that contains one or more environment, application, or location labels and role labels. If a label type is not defined in the scope, that label type is interpreted as `All`. For example, if the policy update scope is
`Application = Checking, Location = China,`
the PCE interprets the scope as
`Application = Checking, Location = China, Environment = All.`

Methods

Functionality	HTTP	URI
Get the current policy update mode for your organization	`GET`	`[api_version][org_href]/sec_policy/draft/firewall_settings`
Change the policy update mode for your organization	`PUT`	`[api_version][org_href]/sec_policy/draft/firewall_settings`

Policy Update Parameters

Parameter	Description	Type	Required
`org_id`	Organization	Integer	Yes
`pversion`	Security Policy Version	String	Yes
`max_results`	Maximum number of policy objects to return (per type)	Integer	No

Policy Update Properties

The current `firewall_settings` resource specifies a combination of IPsec / IKE authentication method (PSK or certificate) for SecureConnect and Machine Authentication.

Parameter	Description	Type	Required
`update_type`	Type of update	String	Yes
`static_policy_scopes`	Scopes that have static policy application mode Reference to `common/rule_set_scopes_get.schema.json`		No
`max_results`	Maximum number of policy objects to return (per type)	Integer	No
`ike_authentication_type`	IKE authentication type to use for IPsec (SecureConnect and Machine Authentication)	String	No
`allow_captive_portal_outbound`	Defines whether or not to open the endpoint firewall to all outbound traffic when a captive portal scenario is discovered by the VEN	Boolean	No
`containers_inherit_host_policy_scopes`	Workloads that match the scope will apply the policy it receives both to itself and the containers hosted by itl. Reference to `common/rule_set_scopes_get.schema.json`		
`blocked_connection_reject_scopes`	Scopes whose blocked connection action will be reject Reference to `common/rule_set_scope_get.schema.json`		
`loopback_interfaces_in_policy_scopes`	Workloads that match the scope will apply policy on loopback interfaces and the loopback interface's IPs will be distributed to peers. Reference to `common/rule_set_scope_get.schema.json`		

Get Policy Update Mode

You can use this method to get the current policy update mode settings for your organization, which is part of your PCE security settings. This method contains a variable (**:pversion**) that can be used to return the security settings with `active` (currently provisioned) or `draft` state for your organization.

URI To Get Policy Update Mode

```
GET [api_version][org_href]/sec_policy/draft/firewall_settings
```

Draft or Active Policy Update Mode

Variable	Description
`:pversion`	Allows you to get: • `active`: The currently provisioned security settings, including policy update mode • `draft`: The draft state of any changed security settings that have not yet been provisioned, including policy update mode

Curl Command Get Active Policy Update Mode

This curl example gets the active (currently provisioned) security settings for your organization, which includes the policy update mode settings.

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/7/sec_
policy/active/firewall_settings -H "Accept: application/json" -u $KEY:$TOKEN
```

Response Body

The `static_policy_scopes` property in the response (in **blue**) indicates that two static scopes have been configured for policy update.

Each scope is defined as a JSON array of labels, which includes an Application, Environment, and a Location label. The labels in the scope are identified by their HREFs.

```
{
  "href": "/orgs/7/sec_policy/active/firewall_settings",
  "created_at": "2015-10-23T22:01:01.151Z",
  "updated_at": "2017-09-02T19:08:55.623Z",
  "deleted_at": null,
  "created_by": { "href": "/users/0"  },
  "updated_by": { "href": "/users/14" },
  "deleted_by": null,
  "update_type": null,
  "allow_dhcp_client": true,
```

```
    "log_dropped_multicast": true,
    "log_dropped_broadcast": false,
    "allow_traceroute": true,
    "allow_ipv6": true,
    "allow_igmp": false,
    "track_flow": true,
    "system_rule_log_flow": false,
    "allow_path_mtu_discovery": true,
    "network_detection_mode": "single_private_brn",
    "static_policy_scopes": [
      [
        { "label": { "href": "/orgs/7/labels/83" } },
        { "label": { "href": "/orgs/7/labels/86" } },
        { "label": { "href": "/orgs/7/labels/94" } }
      ],
      [
        { "label": { "href": "/orgs/7/labels/82"  } },
        { "label": { "href": "/orgs/7/labels/100" } },
        { "label": { "href": "/orgs/7/labels/89"  } },
        { "label": { "href": "/orgs/7/labels/94"  } }
      ]
    ],
      "secure_connect_certs": {
      "default_issuer_name_match": "test",
      "scoped_certificates": []
    }
}
```


Change Policy Update Mode

The Change Policy Update Mode sets your organization's draft policy update mode, which might include adding or removing a policy scope.

The draft state of your policy update mode can be modified, but not the currently active (provisioned) version. First, change to the draft policy update mode, and then provision those changes.

URI To Change Policy Update Mode

```
PUT [api_version][org_href]/sec_policy/draft/firewall_settings
```

Request Properties

Property	Description	Type	Required
`static_policy_scopes`	A set of up to four labels, one or more of the type Application, Environment, Role, and Location. Each label in the policy scope is identified by its HREF, nested in a JSON array. Before updating the organization policy update mode, make sure you have the exact set of labels you want to use and their HREFs.	JSON array of strings	Yes

Request Body

This example shows the request body for two policy update scopes. The first has a single label scope, and the second scope has a set of three labels.

```
{
  "static_policy_scopes": [
    [
      { "label": { "href": "/orgs/1/labels/8" } }
    ],
    [
      { "label": { "href": "/orgs/1/labels/2"  } },
      { "label": { "href": "/orgs/1/labels/8"  } },
      { "label": { "href": "/orgs/1/labels/11" } }
    ]
  ]
}
```

Curl Command to Update Policy Update Mode

```
curl -i -X PUT https://pce.my-
company.com:8443/api/v2/orgs/7/firewall_settings -H "Content-Type:
application/json" -u $KEY:$TOKEN -d '{"static_policy_scopes":[[{"label":
```

```
{"href":"/orgs/1/labels/8"}}],[{"label":{"href":"/orgs/1/labels/2"}},{"label":
{"href":"/orgs/1/labels/8"}},{"label":{"href":"/orgs/1/labels/11"}}]]}'
```

Response

The response for a successful change to your policy update mode is an HTTP `204 No Content Operation`. No data is returned.

Remove all Static Policy Scopes

To remove all static policy scopes, pass an empty JSON array:

```
            PUT [api_version][org_href]/sec_policy/draft/firewall_settings {
"static_policy_scopes": [] }
```

> NOTE:When all static policy scopes are removed, the policy update mode is set to Adaptive.

Curl Command to Remove Static Policy Scopes

```
            curl -i -X PUT https://pce.my-
company.com:8443/api/v2/orgs/7/firewall_settings -H "Content-Type:
application/json" -u $KEY:$TOKEN -d '{"static_policy_scopes":[]}'
```

Virtual Server Filtering

Filtering of the discovered virtual servers and draft virtual servers endpoints makes it easier to manage large numbers of virtual servers.

The existing Public Experimental API endpoints for virtual servers have been changed to support the required filters and associated UI operations. You can now filter a discovered virtual server collection by:

- name
- SLB (API uses href as per conventions)
- VIP: IP, proto, port (any or all)
- virtual server href

Virtual Server Endpoints

New filters have been added for the following existing endpoints:

- `GET /orgs/:xorg_id/discovered_virtual_servers`
- `GET /orgs/:xorg_id/sec_policy/:pversion/virtual_servers`

NOTE:These Interface endpoints are available only for API version V2.

New Filters for Virtual Servers

Discovered Virtual Servers

Filter	URI Example	Notes
`name`	`/discovered_virtual_servers?name=myvip`	Supports partial and incomplete matches
`slb`	`/discovered_virtual_servers?slb=/orgs/1/slbs/<uuid>`	
`vip`	`/discovered_virtual_servers?vip=10.1`	Supports suffix matches, e.g. 10.1 matches any IP address that starts with "10.1", "10.100", ... but not "110.x"
`vip-proto`	`/discovered_virtual_servers?vip_proto=6`	
`vip_port`	`/discovered_virtual_servers?vip_port=80`	
`has_virtual_server`	`/discovered_virtual_servers?has_virtual_server=true`	The `virtual_server_mode` and `virtual_server_labels` MUST be used with `has_virtual_server=true`, otherwise an error will be raised.
`virtual_server_mode`	`/discovered_virtual_servers?virtual_server_mode=enforced`	Options for this filter are "unmanaged" or "enforced"
`virtual_server_labels`	`/discovered_virtual_servers?virtual_server_labels=[[/orgs/1/labels/2, /orgs/1/labels/3], [/orgs/1/labels/4]]` (JSON encoded array of arrays)	
`virtual_server`	`/discovered_virtual_servers?virtual_server=/orgs/1/sec_policy/draft/virtual_servers/<uuid>`	

Virtual Servers

Filter	URI Example	Notes
name	/virtual_servers?name=myvip	Supports partial and incomplete matches
slb	/virtual_servers?slb=/orgs/1/slbs/<uuid>	
vip	/virtual_servers?vip=10.1	Supports suffix matches, e.g. 10.1 matches any IP address that starts with "10.1", "10.100", ... but not "110.x"
vip-proto	/virtual_servers?vip_proto=6	
vip_port	/virtual_servers?vip_port=80	
mode	/virtual_servers?mode=enforced	Options for this filter are "unmanaged" or "enforced"
labels	/virtual_servers?[[/orgs/1/labels/2, /orgs/1/labels/3], [/orgs/1/labels/4]] (JSON encoded array of arrays)	
discovered_virtual_server	/virtual_servers?discovered_virtual_server=/orgs/1/discovered_virtual_servers/<uuid>	

Schema Changes

discovered_virtual_servers

The following object has been added to the schema:

```
{
[... existing fields ...]
"virtual_server" : {
      "href": "/orgs/1/sec_policy/draft/virtual_servers/fbae7cd2-04c3-4d7b-a628-
2d69a9d64a71" ,
      "update_type" : "create", # or "update", "delete", null
      "mode": "enforced", # or "unmanaged"
      "labels" [
           { "href": "/orgs/1/labels/2", "key": "role", "value": "database"},
           { "href": "/orgs/1/label/12", "key": "env", "value": "production"}
```

```
      ]
    }
}
```

virtual_servers

The "mode" and "vip_port" fields have been added to the "discovered_virtual_server sub-object" to reflect the result of filtering.

```
{
    [... existing fields ...]
    "discovered_virtual_server" : {
        "dvs_identifier" :  "5111ecf75c61544720d800cce97a624d" ,
        "href" :  "/orgs/1/discovered_virtual_servers/c1cd1f00-7b48-4c43-a099-
f758ac1a9b40" ,
        "mode" :  "snat" ,
        "name" :  "Common/vip1" ,
        "vip_port" : {
            "port" :  "80" ,
            "protocol" :  6 ,
            "vip" :  "10.0.0.109"
        }
    }
}
```

slb_config

This schema has been deprecated. It was used for `nfc` (Network Function Controller), which is now deprecated.

Request and Response Examples

Discovered Virtual Servers

Curl Command for Discovered Virtual Servers

```
curl -i -u api_
1bbac8b7295e9b512:343461267jks009651245343461267jks00965124b27074fa181f1edb3bb4a3
https://2x2testvc27.ilabs.io:8443/api/v2/orgs/1/discovered_virtual_servers
```

Response Body

```
[{
    "href": "/orgs/1/discovered_virtual_servers/52044aea-14db-4510-a1c6-
00231230034",
       "dvs_identifier": "96803bd07185cd093dd800231230034",
       "name": "Common/QL_VIP_1",
       "nfc": {
              "href": "/orgs/1/nfcs/0bcf6c3d-f588-44c7-a269-00231230034"
              },
       "slb": {
              "href": "/orgs/1/slbs/84a1cd93-142f-480d-b9f8-00231230034"
              },
       "vip_port": {
              "vip": "172.16.27.88",
              "protocol": 6,
              "port": "8080"
              },
       "local_ips": ["172.16.26.18", "172.16.27.18"],
       "mode": "snat",
       "snat_type": "snat_pool",
       "snat_pool_ips": ["172.16.26.27", "172.16.26.18", "172.16.27.18"],
       "service_checks": [{
              "protocol": 1
              }],
       "created_at": "2021-02-26T08:32:02.131Z",
       "updated_at": "2021-02-26T08:32:02.131Z",
       "created_by": {
              "href": "/orgs/1/nfcs/0bcf6c3d-f588-44c7-a269-00231230034"
```

```
                },
        "updated_by": {
                "href": "/orgs/1/nfcs/0bcf6c3d-f588-44c7-a269-00231230034"
                }
                }, {
        "href": "/orgs/1/discovered_virtual_servers/073c40ec-7357-44f4-a66d-
002312300349",
        "dvs_identifier": "b679034796cdde929a000231230034",
        "name": "Common/QL_VIP_2",
        "nfc": {
                "href": "/orgs/1/nfcs/0bcf6c3d-f588-44c7-a269-00231230034"
                },
        "slb": {
                "href": "/orgs/1/slbs/84a1cd93-142f-480d-b9f8-00231230034"
                },
        "vip_port": {
                "vip": "172.16.27.71",
                "protocol": 6,
                "port": "8080"
                },
        "local_ips": ["172.16.26.18", "172.16.27.18"],
        "mode": "snat",
        "snat_type": "snat_pool",
        "snat_pool_ips": ["172.16.26.28", "172.16.26.18", "172.16.27.18"],
        "service_checks": [{
                "protocol": 1
                }],
        "created_at": "2021-02-26T08:32:02.177Z",
        "updated_at": "2021-02-26T08:32:02.177Z",
        "created_by": {
                "href": "/orgs/1/nfcs/0bcf6c3d-f588-44c7-a269-00231230034"
                                },
        "updated_by": {
                "href": "/orgs/1/nfcs/0bcf6c3d-f588-44c7-a269-00231230034"
                }
        }
]
```

Response Body, another example


```
[
   {
       "href": "/orgs/1/discovered_virtual_servers/5db1ce10-263a-44fb-8c0c-
a2312dfb2e6a",
       "dvs_identifier": "the_test_dvs-1",
       "name": "Test DVS No. 1",
       "vip_port": {
              "vip": "30.55.148.143",
              "protocol": 6,
              "port": "8001"
       },
       "local_ips": [
              "10.0.0.1"
       ],
       "mode": "snat",
       "slb": {
              "href": "/orgs/1/slbs/8798cea8-1fd4-40e3-a2f1-adae6f094766"
       },
       "nfc": {
              "href": "/orgs/1/nfcs/7c6ae23f-5532-41b0-9b1e-cf9c689de0ae"
       },
       "network_enforcement_node": {
              "href": "/orgs/1/network_enforcement_nodes/7c6ae23f-5532-41b0-9b1e-cf9c689de0ae"
       },
       "snat_type": "snat_local_ips",
       "service_checks": [],
       "created_at": "2022-09-09T22:19:49.915Z",
       "updated_at": "2022-09-09T22:19:49.915Z",
       "created_by": {
              "href": "/users/0"
       },
       "updated_by": {
              "href": "/users/0"
       },
       "virtual_server": {
              "href": "/orgs/1/sec_policy/draft/virtual_servers/ba700305-29b9-4d6a-8bed-
c8476753c327",
```

```
                "update_type": null,
                "mode": "enforced",
                "labels": [
                {
                   "href": "/orgs/1/labels/14",
                   "key": "role",
                   "value": "LBL-ROLE-1"
                },
                {
                   "href": "/orgs/1/labels/15",
                   "key": "loc",
                   "value": "LBL-LOC-1"
                }
        ]
}
{
        "href": "/orgs/1/discovered_virtual_servers/de5f454e-e95b-40c8-a128-
fc27a1bed840",
        "dvs_identifier": "the_test_dvs-2",
        "name": "Test DVS No. 2",
        "vip_port": {
                "vip": "34.92.48.237",
                "protocol": 6,
                "port": "8002"
                },
        "local_ips": [
                "10.0.0.2"
        ],
        "mode": "snat",
        "slb": {
                "href": "/orgs/1/slbs/8798cea8-1fd4-40e3-a2f1-adae6f094766"
        },
        "nfc": {
                href": "/orgs/1/nfcs/7c6ae23f-5532-41b0-9b1e-cf9c689de0ae"
        },
        "network_enforcement_node": {
                "href": "/orgs/1/network_enforcement_nodes/7c6ae23f-5532-41b0-9b1e-cf9c689de0ae"
        },
```

```
        "snat_type": "snat_local_ips",
        "service_checks": [],
        "created_at": "2022-09-09T22:19:49.919Z",
        "updated_at": "2022-09-09T22:19:49.919Z",
        "created_by": {
                "href": "/users/0"
        },
        "updated_by": {
                "href": "/users/0"
        },
        "virtual_server": {
                "href": "/orgs/1/sec_policy/draft/virtual_servers/e1502bf3-0992-4167-b84f-
eaebd73cc2d7",
                "update_type": null,
                "mode": "enforced",
                "labels": [
                {
                        "href": "/orgs/1/labels/28",
                        "key": "role",
                        "value": "LBL-ROLE-2"
                        },
                        {
                        "href": "/orgs/1/labels/29",
                        "key": "loc",
                        "value": "LBL-LOC-2"
                }
            ]
        }
 }
```


Curl Command for Virtual Servers

```
curl -i -u api_
1bcab8b7295e9b512:343461267jks00965124500jkjdmnwe00231230034dfd256124fa181f1edb3bb
4a3 https://2x2testvc27.ilabs.io:8443/api/v2/orgs/1/sec_policy/draft/virtual_
servers
```

Response Body


```
[{
    "href": "/orgs/1/sec_policy/draft/virtual_servers/5c7aeb96-56e2-4af8-8b4e-
00231230034",
       "created_at": "2021-02-26T08:38:15.298Z",
       "updated_at": "2021-02-26T08:39:21.676Z",
       "deleted_at": null,
       "created_by": {
              "href": "/users/1"
              },
       "updated_by": {
              "href": "/users/1"
              },
       "deleted_by": null,
       "update_type": null,
       "name": "Common/QL_VIP_1",
       "description": "",
       "discovered_virtual_server": {
            "href": "/orgs/1/discovered_virtual_servers/52044aea-14db-4510-a1c6-
00231230034"
            },
       "dvs_name": "Common/QL_VIP_1",
       "dvs_identifier": "96803bd07185cd093dd800231230034",
       "labels": [{
              "href": "/orgs/1/labels/1185",
              "key": "role",
              "value": "Database_VIP_1"
       }, {
              "href": "/orgs/1/labels/1178",
              "key": "app",
              "value": "Application_1"
       }, {
              "href": "/orgs/1/labels/1176",
              "key": "loc",
              "value": "test_place_1"
       }, {
              "href": "/orgs/1/labels/1174",
              "key": "env",
```

```
                "value": "Production"
        }],
        "service": {
                "href": "/orgs/1/sec_policy/draft/services/1"
                },
        "providers": [{
                "label": {
                        "href": "/orgs/1/labels/1183",
                        "key": "role",
                        "value": "Web"
                }
        }, {
                "label": {
                        "href": "/orgs/1/labels/1178",
                        "key": "app",
                        "value": "Application_1"
                }
        }, {
                "label": {
                        "href": "/orgs/1/labels/1176",
                        "key": "loc",
                        "value": "test_place_1"
                }
        }, {
                "label": {
                        "href": "/orgs/1/labels/1174",
                        "key": "env",
                        "value": "Production"
                }
        }],
        "mode": "unmanaged"
}]
```

Virtual Server Discoveries

Virtual server discovery happens passively once the Server Load Balancer (SLB) is configured and the Network Enforcement Node (NEN) receives the SLB configuration changes. However, users might want to be able to run virtual server discovery on demand.

The new schema `network_enforcement_nodes_virtual_server_discovery_jobs_put.s-chema.json` is used to create a virtual server discovery job request that contains the slb_name and virtual server ip_address and port. NEN picks up the request, launches the discovery of the virtual server information, and posts the results back.

Discovery Job On-demand

Use the following API:

`POST /api/v2/orgs/1/network_enforcement_nodes/virtual_server_discovery_jobs`

where the required properties are:

`slb_name`

- Description: Name of the SLB to interrogate.
- Format: String

`virtual_server_infos`

- Description: An array of virtual_server_info objects consisting of virtual_server port and IP address
- Format: Array of Objects

Sample for Request:

```
{
       "$schema": "http://json-schema.org/draft-04/schema#",
       "description": "Details of Virtual Servers to discover",
       "type": "object",
       "additionalProperties": false,
       "required": ["slb_name", "virtual_server_infos"],
       "properties": {
       "slb_name": {
       "description": "Name of SLB to interrogate"
       "type": "string"
},
       "virtual_server_infos": {
       "description": "IP address and port info of Virtual Servers to discover",
       "type": "array",
       "additonalProperties": false,
       "minItems": 1,
       "items": {
```

```
        "type": "object",
        "required": ["ip_address", "port"],
        "properties": {
                "ip_address": {
                "description": "Virtual Server IP address",
                "type": "string"
        },
        "port": {
                "description": "Virtual Server port",
                "type": "integer"
                }
        }
}
```

Sample Response

```
{
"$schema": "http://json-schema.org/draft-04/schema#",
"description": "Details of Virtual Servers discovery job",
"type": "object",
"additionalProperties": false,
"properties": {
        "href": {
        "description": "URI of Virtual Servers discovery job",
        "type": "string"
        }
}
```

Check the Status of Discovery Job

To find out the results of the discovery request use the following command:

```
GET /api/v2/orgs/1/network_enforcement_nodes/virtual_server_discovery_jobs/:job_
uuid
```

```
 {
        "$schema": "http://json-schema.org/draft-04/schema#",
        "description": "Details of Virtual Servers discovery job",
```

```
       "type": "object",
       "additionalProperties": false,
       "required": ["status", "created_at", "created_by"],
       "properties": {
            "href": {
            "description": "URI of the requested discovery job",
            "type": "string"
            }
            "status": {
            "description": "The current state of the request",
            "type": "string",
            "enum": ["pending", "running", "done"]
            },
            "created_at": {
            "description": "The time (rfc3339 timestamp) at which this job was
created",
            "type": "string",
            "format": "date-time"
            },
            "completed_at": {
            "description": "The time (rfc3339 timestamp) at which this job was
completed",
            "type": "string",
            "format": "date-time"
            },
            "created_by": {
            type": "object",
               "required": ["href"],
               "properties": {
               "href": {
               "description": "User who originally created this Virtual Server discovery job",
               "type": "string"
               }
          }
      },
     "connection_state": {
        "description": "Status of most recent connection to the SLB device",
        "type": "string",
```

```
        "enum": ["pending", "successful", "cannot_resolve", "cannot_connect",
"bad_credentials", "bad_certificate", "bad_request", "dup_device"]
       },
    "virtual_server_infos": {
        "description": "Information of individual virtual server discovered",
        "type": "array",
        "minItems": 1,
        "items": {
        "type": "object",
        "additionalProperties": false,
        "properties": {
        "ip_address": {
               "description": "Virtual server IP address",
               "type": "string"
       },
       "port": {
               "description": "Virtual server port",
               "type": "integer"
               },
        "discovered_virtual_server": {
               "description": "Discovered Virtual Server. Null indicates not found",
               "type": "object",
               "required": ["href"],
               "properties": {
               "href": {
               "description": "URI of Discovered Virtual Server",
               "type": "string"
           }
      }
}
```

If a virtual server is discovered, the response might look as follows:

```
{
       "status" : "done",
       "created_at" : "2021-7-19T07:20:50.52Z",
       "created_by" : {
               "href" : "api/v2/orgs/1/users/1"
       },
```

```
       "connection_state" : "successful",
       "completed_at : "2021-7-19T07:20:54.97Z",
       "virtual_server_infos" : [
           { "ip_address" : "10.2.4.54",
           "port" : 443,
           "discovered_virtual_server" : {
           "href" : "api/v2/orgs/1/discovered_virtual_servers/7a597ef0-6609-4927-9eee-
ce403517d850"
           { "ip_address" : "10.23.23.2",
           "port" : 8443,
           discovered_virtual_server" : {
           "href" : "api/v2/orgs/1/discovered_virtual_servers/6a597ef0-6609-4927-9eee-
ce403517d850"
           }
      ]
}
```

If the connection was not established, the response might look as follows:

```
{
       "status" : "done",
       "connection_state" : "cannot_connect",
       "created_at" : "2021-7-19T07:20:50.52Z",
       "completed_at : "2021-7-19T07:20:54.97Z",
       "created_by" : {
              "href" : "api/v2/orgs/1/users/1"
       }
}
```

Chapter 6

Rulesets and Rules

This chapter contains the following topics:

Illumio security policy includes three rule types: intra-scope rules, extra-scope rules, and custom iptables rules. The scope of a ruleset determines which workloads receive the ruleset's rules:

- Intra-scope rules allow communication between providers and consumers within a specific scope.
- Extra-scope rules permit communication between applications. You can write rules so that consumers within or outside a specified scope can access the providers within a scope. For extra-scope rules, the labels used in the scope must match the labels used by the provider.
- Custom iptables rules are needed for your applications as part of the rules managed by the PCE. These rules help preserve any configured iptables from native Linux host configurations by allowing you to include them with the rules for your policy.

You can combine multiple types of rules in a single ruleset.

Rulesets

This Public Stable API gets, creates, updates, and deletes rulesets. Rulesets contain rules and scopes, which define where the rules apply.

Ruleset API Methods

Functionality	HTTP	URI
Get a collection of rule-sets	`GET`	`[api_version][org_href]/sec_policy/rule_sets`
Get a specified ruleset	`GET`	`[api_version[org_href]/sec_policy/rule_sets/rule_set_id]`
Create a ruleset	`POST`	`[api_version][org_href]/sec_policy/rule_sets`
Update a specified rule-set	`PUT`	`[api_version][org_href]/sec_policy/rule_sets/rule_set_id]`
Delete a specified rule-set	`DELETE`	`[api_version][org_href]/sec_policy/rule_sets/rule_set_id]`

Active vs. Draft

This API operates on provisionable objects, which exist in either a `draft` (not provisioned) state or an `active` (provisioned) state.

Provisionable items include label groups, services, rulesets, IP lists, virtual services, firewall settings, enforcement boundaries, and virtual servers. For these objects, the URL of the API call must include the element called `:pversion`, which can be set to either `draft` or `active`.

Depending on the method, the API follows these rules:

- For GET operations – `:pversion` can be draft, active, or the ID of the security policy.
- For POST, PUT, DELETE – `:pversion` can be draft (you cannot operate on active items) or the ID if the security policy.

Ruleset Components

Rulesets are the core of the Illumio Core policy model, and consist of the following elements:

- **Scopes**: Sets of labels (application, environment, and location) that define the boundaries of the rules in a ruleset. If the workloads specified in the rules share the same labels in a ruleset scope, then those workloads and their communications are governed by the rules of the ruleset.

A scope can contain zero or more application, environment, and location labels. A scope can also contain one or more label groups.

If the scope is an empty array (`[]`), then the scope includes all applications, environments, and locations.

If one of the label types is not specified, then all instances of that type are permitted. For example, if application labels are omitted but environment and location labels are present, then all applications are within the scope.

A label type cannot be used in a rule unless the scope for the label type is "All." For example, to use a location label, the scope would have to be an empty array (`[]`), or if there is an application label and an environment label in the scope, the location label cannot be defined in the scope.

A ruleset is not limited to a single scope. A rule can contain multiple scopes depending on the needs of the security policy.

> IMPORTANT:
> Role labels are not used in scopes, but can be used in rules. Never use a role label in a scope.

- **Rules**: A security rule consisting one or more providers (provides a service over a port and protocol), one or more consumers (consumes the service offered by the provider), and one or more services. A provider or consumer can be an individual workload, a role label that represents multiple workloads, IP lists, and so on.

Example Ruleset Scope

Each label in a scope is identified by its HREF. For example, this is the JSON representation of a single ruleset scope with three labels.

Each label must have a different key (role, app, loc, or env). Duplicate label keys are allowed in a scope only if they are in a label group.

```
{
  "scopes": [
    [
      {"label": {"href": "/orgs/7/labels/105"}},
      {"label": {"href": "/orgs/7/labels/88"}},
      {"label": {"href": "/orgs/7/labels/98"}}
    ]
```

```
  ]
}
```

Ruleset Rules

> NOTE:The common schema `consuming_security_principals` has been replaced by two other APIs: `consuming_security_principals_get` and `consuming_security_principals_put`

Ruleset rules define the allowed communication between workloads, or between workloads and IP lists.

For information, see Rules.

Get Rulesets

This method gets all of the rulesets in your organization. This method gets those rulesets that are in the “draft” policy state, which means the current state of rulesets that have not been provisioned.

By default, the maximum number returned on a GET collection of rulesets is 500.

> NOTE:
> To return more than 500 rulesets, use an Asynchronous GET Collection.

URI to Get a Collection of Rulesets

`pversion`: Contains provisionable objects, which exist in either a `draft` (not provisioned) or `active` (provisioned) state. .

```
GET [api_version][org_href]/sec_policy/:pversion/rule_sets
```

URI to Get an Individual Ruleset

```
[api_version[org_href]/sec_policy/rule_sets/rule_set_id]
```

Query Parameters

You can use the following query parameters to restrict the results of the query to get a collection of rulesets.

Parameter	Description	Type	Required
`org_id`	Organization	Integer	Yes
`pversion`	Security Policy Version	String	Yes
`rule_set_id`		Integer	Yes
`name`	Name of the rulesets to filter, which must be unique. This parameter supports partial matches.	String	No
`description`	Description of Rule Set(s) to return. Supports partial matches	String	No
`external_data_set`	The data source from which the resource originates. For example, if ruleset information is stored in an external database.	String, Null	No
`external_data_reference`	A unique identifier within the external data source. For example, if ruleset information is stored in an external database.	String, Null	No
`enabled`	Enabled flag	Boolean	No
`update_type`	Type of update	String	No
`scopes`	Rule set scopes • `label`: label URI • `label_group`: label group URI	Array String	No

`rules`	Array of rules in this rule set
	Required properties:
	`enabled`: Enabled flag
	`description`
	`external_data_set`
	`external_data_reference`
	`ingress_service`
		Object	No

Properties

Property	Description	Type	Required
`enabled`	Enabled flag	Boolean	Yes
`name`	Name of the rulesets to filter. This parameter supports partial matches.	String	Yes
`scopes`	Reference to `common/rule_set_scopes_get.schema.json`		Yes
`rules`	Array of rules in this rule set		Yes

Property	Description	Type	Required
	Reference to `sec_policy_rule_sets_sec_rules_get.schema.json`		
`created_at`	Timestamp when this rule set was first created	String	Yes
`updated_at`	Timestamp when this rule set was last updated	String	Yes
`deleted_at`	Timestamp when this rule set was deleted	String, Null	Yes
`created_by`	User who originally created this rule set	Object	No
`updated_by`	User who last updated this rule set	Object	No
`deleted_by`	User who deleted this rule set	Object, Null	No
`update_type`	Type of update Reference to `common/sec_policy_update_type.schema.json`		No
`external_data_set`	The data source from which the resource originates. For example, if ruleset information is stored in an external database.	String	No
`external_data_reference`	A unique identifier within the external data source. For example, if ruleset information is stored in an external database.	String	No
`ip_tables_rules`	Array of iptables rules in this rule set. Reference to `common/ip_tables_rules_get.schema.json`		No

Create a Ruleset

This method creates an individual ruleset. The PCE web console supports up to 500 rules per ruleset.

> NOTE:
> To write more than 500 rules for a particular ruleset, create additional rulesets, or use the Illumio Core REST API (rulesets with more than 500 rules are not fully displayed in the PCE web console).

URI to Create a Ruleset

```
POST [api_version][ruleset_href]
```

Properties for POST

Property	Description	Type	Required
`name`	Name of the new ruleset, which must be unique.	String	Yes
`scopes`	Reference to `common/rule_set_scopes_put.schema.json`		Yes
`rules`	Reference to `sec_policy_rule_sets_sec_rules_post.schema. json`		No
`ip_tables_ rules`	Array of custom iptables rules in this rule set. Reference to `common/ip_tables_rules_post.schema.json`		No
`external_data_set`	External data set identifier	String, Null	No
`external_data_reference`	External data reference identifier.	String, Null	No
`enabled`	Enabled flag	Boolean	Yes
`scopes`	Reference to `common/rule_set_scopes_put.schema.json`		Yes
`rules`	Array of rules in this rule set Reference to `sec_policy_rule_sets_sec_rules_post.schema.json`		Yes

Update a Ruleset

To update an individual ruleset, you need the HREF of the ruleset you want to update, which can be obtained when you get a collection or an individual ruleset.

If you want to add a single rule to an existing ruleset, use `PUT /api/v1/orgs/1/sec_policy/draft/rule_sets/123/sec_rules`.

Properties for PUT

Property	Description	Type	Required
`name`	Name of the ruleset to update, must be unique	String	No
`external_data_set`	External data set identifier	String, Null	No
`external_data_reference`	External data reference identifier.	String, Null	No
`enabled`	Enabled flag	Boolean	Yes
`update_type`	Type of update		No

Property	Description	Type	Required
	Reference to `common/sec_policy_update_type.schema.json`		
`scopes`	Reference to `common/rule_set_scopes_put.schema.json`		No
`rules`	Array of rules in this rule set Required properties: "href" "enabled", "providers", "consumers", "ingress_services", "resolve_labels_as"	Object	No
`consumers`	Reference to `sec_policy_rule_sets_sec_rules_consumers_put.schema.json`		
`consuming_security_principals`	Reference to `common/ consuming_security_principals_put.schema.json`		
`network_type`	Reference to `common/ rule_network_type.schema.json`		
`use_workload_subnets`	Reference to `sec_ rule_use_workload_subnets.schema.json`		

Delete a Ruleset

To delete an individual ruleset, you need the HREF of the ruleset you want to delete, which can be obtained when you get a collection of rulesets.

URI to Delete an Individual Ruleset

```
DELETE [api_version][ruleset_href]
```

Examples

Get a Ruleset

```
$curl -X GET https://pce.my-company.com:8443/api/v2/orgs/1/sec_policy/draft/rule_sets -H "Accept: application/json" -u api_
```

```
1c2618a67847c94b8:98c76f7a4563f29cd78b3392684cd5ec09534bafe5197fe8e901d95561bdd8f5
| jq
```

Response


```
[
	{
		"href": "/orgs/1/sec_policy/draft/rule_sets/1",
		"created_at": "2023-04-05T23:08:32.578Z",
		"updated_at": "2023-04-05T23:08:32.632Z",
		"deleted_at": null,
		"created_by": {
			"href": "/users/0"
		},
		"updated_by": {
			"href": "/users/0"
		},
		"deleted_by": null,
		"update_type": null,
		"name": "Default",
		"description": null,
		"enabled": true,
		"scopes": [
		[]
		],
		"rules": [
		{
			"href": "/orgs/1/sec_policy/draft/rule_sets/1/sec_rules/1",
			"created_at": "2023-04-05T23:08:32.599Z",
			"updated_at": "2023-04-05T23:08:32.632Z",
			"deleted_at": null,
			"created_by": {
				"href": "/users/0"
			},
			"updated_by": {
				"href": "/users/0"
			},
			"deleted_by": null,
			"update_type": null,
```

```
                "description": "Allow outbound connections",
                "enabled": true,
                "providers": [
                        {
                        "ip_list": {
                        "href": "/orgs/1/sec_policy/draft/ip_lists/1"
                        }
                        }
                ],
                "consumers": [
                {
                "actors": "ams"
                }
                ],
        "consuming_security_principals": [],
                "sec_connect": false,
                "stateless": false,
                "machine_auth": false,
                "unscoped_consumers": false,
                "network_type": "brn",
                "use_workload_subnets": [],
                "ingress_services": [
                {
                        "href": "/orgs/1/sec_policy/draft/services/1"
                }
                ],
                "egress_services": [],
                "resolve_labels_as": {
                "providers": [
                "workloads"
                ],
                "consumers": [
                "workloads"
                ]
                }
                }
                ],
        "ip_tables_rules": [],
```

```
                        "caps": [
                        "write",
                        "provision"
                        ]
                        },
                {
                        "href": "/orgs/1/sec_policy/draft/rule_sets/3",
                        "created_at": "2023-04-05T23:50:05.591Z",
                        "updated_at": "2023-04-06T19:03:49.947Z",
                        "deleted_at": null,
                        "created_by": {
                                "href": "/users/1"
                        },
                        "updated_by": {
                                "href": "/users/1"
                        },
                        "deleted_by": null,
                        "update_type": null,
                        "name": "ruleset1"
"               "description": "",
                "enabled": true,
                "scopes": [
                []
                ],
        "rules": [
                {
                "href": "/orgs/1/sec_policy/draft/rule_sets/3/sec_rules/9",
                "created_at": "2023-04-06T00:58:55.061Z",
                "updated_at": "2023-04-06T00:58:55.088Z",
                "deleted_at": null,
                "created_by": {
                        "href": "/users/1"
                },
                "updated_by": {
                        "href": "/users/1"
                },
                "deleted_by": null,
                "update_type": null,
```

```
            "description": "",
            "enabled": true,
            "providers": [
            {
            "label": {
                    "href": "/orgs/1/labels/14"
            },
            "exclusion": false
            }
            ],
            "consumers": [
            {
            "label": {
                    "href": "/orgs/1/labels/15"
            },
            "exclusion": false
            }
            ],
    "consuming_security_principals": [],
            "sec_connect": true,
            "stateless": false,
            "machine_auth": false,
            "unscoped_consumers": false,
            "network_type": "brn",
            "use_workload_subnets": [],
            "ingress_services": [
                    {
                    "href": "/orgs/1/sec_policy/draft/services/9"
            },
            {
            "port": 23000,
            "proto": 6
            }
            ],
            "egress_services": [],
            "resolve_labels_as": {
            "providers": [
            "workloads"
```

```
                ],
                "consumers": [
                "workloads"
                ]
                }
        }
                ],
                "ip_tables_rules": [],
                "caps": [
                "write",
                "provision"
                ]
        }
]
```

Create a Ruleset

```
$curl -u api_
1c2618a67847c94b8:98c76f7a4563f29cd78b3392684cd5ec09534bafe5197fe8e901d95561bdd8f
5-X POST -H 'Content-Type: application/json' -d '
{"name":"ruleset3","description":"","scopes":[[{"exclusion":false,"label":
{"href":"/orgs/1/labels/14"}}]]}'https://2x2testvc168.ilabs.io:8443/api/v2/orgs/1/
sec_policy/draft/rule_sets | jq
```

Response

```
{
        "href": "/orgs/1/sec_policy/draft/rule_sets/16",
        "created_at": "2023-04-06T18:46:34.718Z",
        "updated_at": "2023-04-06T18:46:34.727Z",
        "deleted_at": null, "created_by": {
                "href": "/users/1"
                },
        "updated_by": {
                "href": "/users/1"
                },
        "deleted_by": null,
        "update_type": "create",
        "name": "ruleset3",
```

```
        "description": "",
        "enabled": true, "scopes": [
                [
                {
                "label": {
                        "href": "/orgs/1/labels/14"
                },
                "exclusion": false
                }
                ]
        ],
        "rules": [],
                "ip_tables_rules": [], "caps": [
                "write",
                "provision"
        ]
}
```

Update a Ruleset

```
$curl -w "%{http_code}" -u api_
1c2618a67847c94b8:98c76f7a4563f29cd78b3392684cd5ec09534bafe5197fe8e901d95561bdd8f5
-X PUT -H 'Content-Type: application/json' -d '{"scopes":[[{"label":
{"href":"/orgs/1/labels/14"}},{"label":{"href":"/orgs/1/labels/15"}}]]}'
https://2x2testvc168.ilabs.io:8443/api/v2/orgs/1/sec_policy/draft/rule_sets/14 |
jq
```

Response

The ruleset was successfully updated:

```
204
```

Rules

This Public Stable API creates, updates, and deletes individual rules in rulesets . It also gets a collection of rules from a ruleset.

Providers and Consumers

The Illumio Core allowlist policy model uses rules to define the allowed communications between two or more workloads, or between workloads and other entities, such as IP lists, virtual servers, and the internet.

The fundamental structure of a rule (except custom iptables rules) consists of a provider, a service that the provider makes available over a network port and protocol, and a consumer of that service.

Rules API Methods

Functionality	HTTP	URI
Get rules	`GET`	`sec_policy_rule_sets_sec_rules`
Get rules for provides	`GET`	`sec_policy_rule_sets_sec_rules_providers`
Get rules for consumers	`GET`	`sec_policy_rule_sets_sec_rules_consumers`
Update rules	`PUT`	`sec_policy_rule_sets_sec_rules`
Update rules for providers	`PUT`	`sec_policy_rule_sets_sec_rules_providers`
Update rules for consumers	`PUT`	`sec_policy_rule_sets_sec_rules_consumers`
Create rules	`POST`	`sec_policy_rule_sets_sec_rules`
Delete an individual rule	`DELETE`	`sec_rule_href`

Active vs Draft

This API operates on provisionable objects, which exist in either a `draft` (not provisioned) state or an `active` (provisioned) state.

Provisionable items include label groups, services, rulesets, IP lists, virtual services, firewall settings, enforcement boundaries, and virtual servers. For these objects, the URL of the API call must include the element called `:pversion`, which can be set to either `draft` or `active`.

Depending on the method, the API follows these rules:

- For GET operations – `:pversion` can be draft, active, or the ID of the security policy.
- For POST, PUT, DELETE – `:pversion` can be draft (you cannot operate on active items) or the ID if the security policy.

Rule Types

There are three types of rules:

- **Intra-scope rules**: Allow communication between providers and consumers within a specific scope.

- **Extra-scope rules**: Rules that go beyond the scope of the ruleset to which they belong. In this rule type, the workloads, labels or IP list in the consumers part of the rule are not constricted by the scope of the ruleset. This type of rule is used when you want specific rules that allow providers to offer a service to other workloads or groups that are not within the boundaries of the ruleset scope.
- **Custom iptables rules**: Used to configure custom iptables rules on Linux workloads; for example, to preserve existing native Linux host iptables rules by including them in a ruleset.

> NOTE:
> The PCE web console can only display up to 500 rules per ruleset. To write more than 500 rules for a particular scope, consider splitting the rules across multiple rulesets, otherwise users won't be able to view them all in the PCE web console.

Rule Type JSON Specification

To define a rule as either intra-scope or extra-scope, specify if the rule is “scoped” or “not scoped” by defining the `'unscoped_consumers'` property:

- When a rule has `unscoped_consumers: false`, this defines an intra-scope rule, which means both its providers and consumers are bound by the ruleset scope.
- When a rule has `unscoped_consumers: true`, this defines an extra-scope rule, which means its providers are bound by the ruleset scope, but the consumers are ***not*** bound by the ruleset scope.

Intra-Scope Rule Example

> NOTE:The common schema `consuming_security_principals` has been replaced by two other APIs: `consuming_security_principals_get` and `consuming_security_principals_put`

This rule illustrates an intra-scope rule because it has its `unscoped_consumers` property set to `false`:

```
{
  "rules": [
    {
      "enabled": true,
      "providers": [ {"label": {"href": "/orgs/1/labels/2"} } ],
      "consumers": [ {"label": {"href": "/orgs/1/labels/1"} } ],
```

```
      "consuming_security_principals": [],
      "ingress_services": ["href": "/orgs/1/sec_policy/draft/services/20"],
      "resolve_labels_as": {
        "providers": ["workloads"],
        "consumers": ["workloads"]
      },
      "sec_connect": false,
      "unscoped_consumers": false
    }
  ]
}
```

Stateless Rules

A rule can be configured to have stateless packet filtering (`"stateless": true`). This means that the VEN instructs the host firewall to *not* maintain persistent connections for all sessions. This type of rule is typically used for datacenter “core services” such as DNS and NTP.

A stateless rule can have these consumer types:

- Any IP list plus all workloads
- A label (one of a specific type)
- An individual item (such as an individual workload)

An attempt to add more consumers, or one not supported, will return an error.

A PCE can only have a maximum of 100 stateless rules. If an implementation requires more than 100 stateless rules, contact your Illumio Professional Services Representative for more information.

> NOTE:
> This property has an API exposure level of
> Public Experimental, which means it is not intended
> for production use and might change in future releases.
> For more information, see API Classification and Version.

Get Rules

This API gets a collection of rules or gets an individual rule from a ruleset.

Before you can get rules from a ruleset with this API, you need to obtain the ruleset HREF, which is returned when you Get a Collection of Rulesets.

Query Parameters to Get a Collection of Security Rules from a Ruleset

Parameter	Description	Type	Required
`org_id`	Organization	Integer	Yes
`pversion`	Security policy version -- `draft`(not provisioned) or `active` (provisioned)	String	Yes
`rule_set_id`	Ruleset ID	Integer	Yes
`external_data_reference`	A unique identifier within the external data source. For example, if this rule information is stored in an external database.	String	No
`external_data_set`	The data source from which the resource originates. For example, if this rule information is stored in an external database.	String	No
`labels`	List of lists of label URIs, encoded as a JSON string	String	No
`max_results`	Maximum number of Rule Sets to return	Integer	No
`name`	Name of Rule Set(s) to return. Supports partial matches	String	No

Query Parameters to Get an Individual Security Rule from a Ruleset

Parameter	Description	Type	Required
`org_id`	Organization	Integer	Yes
`pversion`	Security policy version -- `draft`(not provisioned) or `active` (provisioned)	String	Yes
`rule_set_id`	Ruleset ID	Integer	Yes

Create Rules

This API allows you to create one or more rules inside a specific ruleset.

URI to Create a Rule

```
POST [api_version][rule_set_href]/sec_rules
```

Properties

Property	Description	Type	Required
`enabled`	Indicates if the rule is enabled or disabled.	Boolean	Yes
`providers`	Entities that can be used as a provider in a rule. Reference to `sec_policy_rule_sets_sec_rules_providers_put.schema.json`		Yes

Property	Description	Type	Required
consum	Entities that can be used as a consumer in a rule. Reference to sec_policy_rule_sets_sec_rules_consumers_put.schema.json		Yes
ingress_services	Reference to sec_rule_ingress_services.schema.json		Yes
resolve_labels_as	Reference to sec_rule_resolve_labels_as.schema.json		Yes
sec_connect	Indicates whether a secure connection is established. If set to true, then the rule will use SecureConnect IPsec encryption for all traffic allowed by the rule.	Boolean	No
stateless	Whether packet filtering is stateless for the rule. If set to true, then the rule's packet filtering is stateless. This means that the VEN will instruct the host firewall to not maintain persistent connections for a session. This type of rule is typically used for datacenter “core services” such as DNS and NTP. You can only create a total of 100 stateless rules in your PCE. If you need more than 100 stateless rules in your Illumio policy, contact your Illumio Professional Services Representative for more information.	Boolean	No
machine_auth	Whether machine authentication is enabled. If set to true, then machine authentication is used for the rule, meaning that any hosts defined in the rule have been configured for the PKI-based machine authentication. Before using this property, your PCE must already be configured for machine authentication. See the PCE Administration Guide for information on configuring machine authentication for the PCE.	Boolean	No
consuming_security_principals	Reference to common/consuming_security_principals_put.schema.json		
unscoped_consumers	Set the scope for rule consumers to All	Boolean	
network_type	Reference to common/rule_network_type.schema.json		

Property	Description	Type	Required
use_workload_subnets	Reference to sec_rule_use_workload_subnets.schema.json		

Update Rules

This API updates an individual rule inside a ruleset.

URI to Update Rules

```
PUT [api_version][sec_rule_href]
```

The request body and JSON payload is the same as that for Create Rules.

Delete a Rule

This API deletes an individual rule inside a ruleset.

URI to Delete a Rule

```
DELETE [api_version][sec_rule_href]
```

Curl Command to Delete Rule

The curl command for deleting a rule can be structured as follows:

```
curl -i -X DELETE https://pce.my-company.com:8443/api/v2/orgs/sec_
policy/draft/rule_sets/152/sec_rules/124 -H "Accept: application/json" -u
$KEY:$TOKEN
```

Rule Search

This Public Experimental method searches for rules across all rulesets. This method is especially useful when your organization has large numbers of rules organized in rulesets. For example, your organization has 192,000 rules organized across 650 rulesets and you needed to know how many rules applied for SNMP (UDP 161). You can't easily find this information without using this method.

NOTE:
Rule search concurrent requests are now increased to 12 searches on 2x2s and 4x2s.

URI to Search for Rules

```
POST sec_policy_rule_search
```

Attributes for Rule Search

You can search for `Workloads` and IP lists by `href`. The `ingress_services` field accepts either an HREF or an object containing port/protocol/process name/service name, but not service_ports or windows_services sub-resource.

To search by `providers` and `consumers`, you can using the following attributes:

Actor Name	Actor Value Type	Required Keys	Providers	Consumers
`actors`	String	N/A	True	True
`labels`	JSON Object	HREF	True	True
`label_group`	JSON Object	HREF	True	True
`workload`	JSON Object	HREF	True	True
`virtual_service`	JSON Object	HREF	True	True
`virtual_server`	JSON Object	HREF	True	False
`ip_list`	JSON Object	HREF	True	True

Examples for Rule Search

Curl Command Examples for Rule Search

```
$ curl -u API_ID:API_SECRET -X POST -H 'Content-Type: application/json' -d '
{"providers": [{"label": {"href": "/orgs/1/labels/2"}}],"consumers": [{"label":
{"href": "/orgs/1/labels/1"}}]}'https://dev6.ilabs.io:8443/api/v2/orgs/1/sec_
policy/draft/rule_search
```

```
$ curl -u API_ID:API_SECRET -X POST -H 'Content-Type: application/json' -d '
{"providers": [{"workload": {"href": "/orgs/1/workloads/4ce873d3-2e5d-4f06-82f5-
4b1e0ec9ceb2"}}]}'https://dev6.ilabs.io:8443/api/v2/orgs/1/sec_policy/draft/rule_
search
```

```
$ curl -u API_ID:API_SECRET -X POST -H 'Content-Type: application/json' -d '
{"ingress_services": [{"href": "/orgs/1/sec_
policy/draft/services/1"}]}'https://dev6.ilabs.io:8443/api/v2/orgs/1/sec_
policy/draft/rule_search
```

```
$ curl -u API_ID:API_SECRET -X POST -H 'Content-Type: application/json' -d '
{"ingress_services": [{"port": 11000, "to_port": 12000, "proto":
6}]}'https://dev6.ilabs.io:8443/api/v2/orgs/1/sec_policy/draft/rule_search
```

Examples

Get a Rule

```
$curl -X GET https://pce.my-company.com:8443/api/v2/orgs/1/sec_policy/active/rule_
sets/ -H "Accept: application/json" -u api_
1c2618a67847c94b8:98c76f7a4563f29cd78b3392684cd5ec09534bafe5197fe8e901d95561bdd8f5
| jq
```

Response

```
[
       {
       "href": "/orgs/1/sec_policy/active/rule_sets/1",
       "created_at": "2023-04-05T23:08:32.578Z",
       "updated_at": "2023-04-05T23:08:32.632Z",
       "deleted_at": null, "created_by": {
              "href": "/users/0"
              },
       "updated_by": {
              "href": "/users/0"
              },
       "deleted_by": null,
       "name": "Default",
       "description": null,
       "enabled": true, "scopes": [ []
              ],
       "rules": [
              {
```

```
        "href": "/orgs/1/sec_policy/active/rule_sets/1/sec_rules/1",
"created_at": "2023-04-05T23:08:32.599Z",
"updated_at": "2023-04-05T23:08:32.632Z",
"deleted_at": null, "created_by": {
        "href": "/users/0"
        },
"updated_by": {
        "href": "/users/0"
        },
"deleted_by": null,
"description": "Allow outbound connections",
"enabled": true,
"providers": [ {
"ip_list": {
        "href": "/orgs/1/sec_policy/active/ip_lists/1"
        }
        }
],
"consumers": [ {
"actors": "ams"
        }
        ],
"consuming_security_principals": [],
"sec_connect": false,
"stateless": false,
"machine_auth": false,
"unscoped_consumers": false,
"network_type": "brn",
"use_workload_subnets": [], "ingress_services": [
        {
        "href": "/orgs/1/sec_policy/active/services/1" }
        ],
"egress_services": [],
"resolve_labels_as": {
"providers": [
"workloads"
        ],
"consumers": [
```

```
        "workloads"
                ]
        }
}
],
```

Create a Rule

```
curl -u api_
1c2618a67847c94b8:98c76f7a4563f29cd78b3392684cd5ec09534bafe5197fe8e901d95561bdd8f5
-X POST -H 'Content-Type: application/json' -d '{"providers":[{"label":
{"href":"/orgs/1/labels/14"}}],"consumers":[{"label":
{"href":"/orgs/1/labels/15"}}],"enabled":true,"ingress_services":
[{"href":"/orgs/1/sec_policy/draft/services/9"},
{"proto":6,"port":23000}],"network_type":"brn","consuming_security_principals":
[],"sec_connect":true,"machine_auth":false,"stateless":false,"unscoped_
consumers":false,"description":"","use_workload_subnets":[],"resolve_labels_as":
{"consumers":["workloads"],"providers":["workloads"]}}'
https://2x2testvc168.ilabs.io:8443/api/v2/orgs/1/sec_policy/draft/rule_sets/3/sec_
rules | jq
```

```
{
        "href": "/orgs/1/sec_policy/draft/rule_sets/3/sec_rules/9",
        "created_at": "2023-04-06T00:58:55.061Z",
        "updated_at": "2023-04-06T00:58:55.088Z",
        "deleted_at": null, "created_by": {
                "href": "/users/1"
                },
        "updated_by": {
                "href": "/users/1"
                },
        "deleted_by": null,
        "update_type": "create",
        "description": "",
        "enabled": true, "providers": [
                {
                "label": {
                        "href": "/orgs/1/labels/14"
```

```
                },
                "exclusion": false
                        }
                        ],
        "consumers": [
                {
                "label": {
                        "href": "/orgs/1/labels/15"
                },
                "exclusion": false
                }
        ],
        "consuming_security_principals": [],
                "sec_connect": true,
                "stateless": false,
                "machine_auth": false,
                "unscoped_consumers": false,
                "network_type": "brn",
        "use_workload_subnets": [], "ingress_services": [
                {
                "href": "/orgs/1/sec_policy/draft/services/9"
                }, {
                        "port": 23000,
                        "proto": 6
                        }
                        ],
        "egress_services": [],
        "resolve_labels_as": {
        "providers": [
                "workloads"
                ],
        "consumers": [
                "workloads"
                ]
        }
}
```

Update a Rule

```
curl -w "%{http_code}" -u api_
1c2618a67847c94b8:98c76f7a4563f29cd78b3392684cd5ec09534bafe5197fe8e901d95561bdd8f5
-X PUT -H 'Content-Type: application/json' -d '{"providers":
[{"exclusion":false,"label":{"href":"/orgs/1/labels/14"}}],"consumers":
[{"exclusion":false,"label":
{"href":"/orgs/1/labels/15"}}],"enabled":true,"ingress_services":
[{"href":"/orgs/1/sec_policy/draft/services/9"},
{"proto":6,"port":25000}],"network_type":"brn","consuming_security_principals":
[],"sec_connect":true,"machine_auth":false,"stateless":false,"unscoped_
consumers":false,"description":"","use_workload_subnets":[],"resolve_labels_as":
{"providers":["workloads"],"consumers":["workloads"]}}'
https://2x2testvc168.ilabs.io:8443/api/v2/orgs/1/sec_policy/draft/rule_sets/3/sec_
rules/3 | jq
```

Response

The rule was successfully uodated:

```
204
```

Rule Hit Count

The Rule Hit Count feature is configured so that only certain VENs can compute the rule hit counts and send the rule ID info over to the PCE.

Enabling Rule Hit Count

The Rule Hit Count feature is disabled by default on all the VENs and on the PCE.

To use the Rule Hit Count feature, you first need to enable it on the PCE and the relevant VENs.

Enable Rule Hit Count on a VEN

Use the following API to enable the feature on a VEN on all scopes:

PUT api/v2/orgs/:xorg_id/sec_policy/draft/firewall_settings

```
{
        "rule_hit_count_enabled_scopes":[[]]
```

```
}
```

This is a sample API that can be used to enable the feature on specific scopes. In this example, it enables the features on all VENs with labels 7 and 12.

```
 {
       "rule_hit_count_enabled_scopes": [
          [
               {
                  "label": {
                       "href": "/orgs/1/labels/7"
               }
           },
           {
                  "label": {
                       "href": "/orgs/1/labels/12"
                  }
               }
          ]
     ]
}
```

Commit or provision these DRAFT changes.

POST /api/v2/orgs/:xorg_id/sec_policy

```
{
       "update_description":"Enable rule hit count",
       "change_subset": {
               "firewall_settings":
               [
                   {
                       "href":"/orgs/1/sec_policy/draft/firewall_settings"
                   }
               ]
       }
}
```

Disable the feature Rule Hit Count on all VENs:

PUT api/v2/orgs/:xorg_id/sec_policy/draft/firewall_settings

The property `rule_hit_count_enabled_scopes` was added to this API:

```
  {
          "rule_hit_count_enabled_scopes": {
}
```

Enable Rule Hit Count on a PCE

Use the following API to enable the feature on a PCE:

PUT /api/v2/orgs/:xorg_id/report_templates/rule_hit_count_report

```
{
       "enabled": true
}
```

Generating Rule Hit Count Reports

A Rule Hit Count report can be either a scheduled report generated on a recurrent basis or an ad-hoc report.

To generate the Rule Hit Count report, two new schemas have been introduced: rule_hit_count_report_params and rule_set_lists:

rule_hit_count_report_params

The new schema returns the rule hit count statistics for all the rules in a ruleset during the specified time-range.

```
   {
       "$schema": "http://json-schema.org/draft-04/schema#",
       "description": "Returns the rule hit count stats for all the rules in a
                       ruleset during the specified time-range",
       "type": "object",
       "additionalProperties": false,
```

```
        "required": [
                "report_time_range",
                "rule_sets"
        ],
        "properties": {
                "report_time_range": {
                "description": "Time range the report is built across",
                "type": "object",
                "oneOf": [
                        {
                        "$ref": "report_time_range_definitions.schema.json#/
                                definitions/custom_date_range"
                        },
                        {
                        "$ref": "report_time_range_definitions.schema.json#/
                                definitions/last_num_days"
                }
                ]
                },
                "rule_sets": {
                        "$ref": "rule_set_lists.schema.json"
                        },
                "max_results": {
                        "description": "maximum number of rules to return in
                                the specified time-range
                                in descending order of rule creation time",
                        "minimum": 0,
                        "maximum": 200000,
                        "type": "integer"
                }
        }
}
```

rule_set_lists

This schema returns the rule hit count statistics for all the rules in a ruleset during the specified time-range.

```
{
	"$schema": "http://json-schema.org/draft-04/schema#",
	"description": "Returns the rule hit count stats for all the rules in a
			ruleset during the specified time-range",
	"type": "array",
	"items": {
		"type": "object",
		"additionalProperties": false,
		"required": [
			"href"
		],
		"properties": {
			"href": {
			    "description": "HREF of the ruleset",
			    "type": "string"
			}
		}
	}
}
```

Generate an Ad-hoc Report

The following API can be used to create a report for the last x number of days. In the example, It generates a rule hit count report for the last 30 days for all rule sets.

POST /api/v2/orgs/:xorg_id/reports

```
 {
	"report_template": {
		"href": "/orgs/1/report_templates/rule_hit_count_report"
	},
	"description": "My first rule hit count report",
	"report_parameters": {
		"report_time_range": {
			"last_num_days": 30
		},
		"rule_sets": []
	},
```

```
        "send_by_email": true
}
```

The example response is such as the following:

```
{
        "href": "/orgs/1/reports/d1b80240-ffa5-4e99-b2a0-c3d4946efe03",
        "report_template": {
                "href": "/orgs/1/report_templates/rule_hit_count_report",
                "name": "Rule Hit Count Report"
        },
        "description": "My first rule hit count report",
        "created_at": "2023-11-03T07:52:04.018Z",
        "updated_at": "2023-11-03T07:52:04.018Z",
        "progress_percentage": 0,
        "generated_at": null,
        "status": "pending",
        "report_parameters": {
                        "report_time_range": {
                            "last_num_days": 30
                        },
                        "rule_sets": []
                },
        "send_by_email": true,
        "created_by": {
                "href": "/users/1"
        },
        "updated_by": {
                "href": "/users/1"
        }
}
```

To create a report for a custom date range, use the following:

```
 {
        "report_template": {
                "href": "/orgs/1/report_templates/rule_hit_count_report"
        },
```

```
        "description": "My first rule hit count report",
        "report_parameters": {
                "report_time_range": {
                        "start_date": "2023-10-03T00:00:00Z",
                        "end_date": "2023-11-03T23:59:59Z"
                },
                "rule_sets": []
        },
        "send_by_email": true
}
```

Check the Status of the Report

Use a GET API and the HREF from the POST response to check the status of the report:

GET /api/v2/orgs/:xorg_id/reports/:report_uuid

```
 {
        "href": "/orgs/1/reports/d1b80240-ffa5-4e99-b2a0-c3d4946efe03",
        "report_template": {
                "href": "/orgs/1/report_templates/rule_hit_count_report",
                "name": "Rule Hit Count Report"
        },
        "description": "My first rule hit count report",
        "created_at": "2023-11-03T07:52:04.018Z",
        "updated_at": "2023-11-03T07:52:05.233Z",
        "progress_percentage": 100,
        "generated_at": "2023-11-03T07:52:05.233Z",
        "status": "done",
        "report_parameters": {
                "rule_sets": [],
                "report_time_range": {
                   "last_num_days": 30
                }
        },
        "send_by_email": true,
        "created_by": {
                "href": "/users/1"
        },
```

```
        "updated_by": {
                "href": "/users/1"
        }
}
```

Download the Report

When the status of the report is completed, it is emailed to the user who created the report if the option `send_by_email` is set.

Once the status of the report is set to "**done**", the report can be downloaded using the download API as follows:

GET /api/v2/orgs/:xorg_id/reports/:report_uuid/download

Here's a sample response that can be saved as CSV

```
    Rule HREF,Rule Name,Rule Set HREF,Rule Set Name,Rule Hit Count,Days Since Last
Hit,
               Last Updated Timestamp,Last Updated By,Start Date,End Date
/orgs/1/sec_policy/active/rule_sets/1/sec_rules/23,"",/orgs/1/
               sec_policy/active/rule_sets/1,Default,0,-1,2023-08-07T22:55:37-07:00,
              /users/1,2023-10-04T00:00:00Z,2023-11-02T23:59:00Z
/orgs/1/sec_policy/active/rule_sets/1/sec_rules/21,"",/orgs/1/sec_policy/active/
               rule_sets/1,Default,0,-1,2023-07-25T04:48:09-07:00,
              /users/1,2023-10-04T00:00:00Z,2023-11-02T23:59:00Z
/orgs/1/sec_policy/active/rule_sets/1/sec_rules/19,"",/orgs/1/sec_policy/active/
               rule_sets/1,Default,0,1,2023-07-25T04:35:31-07:00,
              /users/1,2023-10-04T00:00:00Z,2023-11-02T23:59:00Z
/orgs/1/sec_policy/active/rule_sets/1/sec_rules/8,"",/orgs/1/sec_policy/active/
               rule_sets/1,Default,0,-1,2023-07-21T16:34:08-07:00,
              /users/1,2023-10-04T00:00:00Z,2023-11-02T23:59:00Z
/orgs/1/sec_policy/active/rule_sets/1/sec_rules/3,"",/orgs/1/sec_policy/active/
               rule_sets/1,Default,0,1,2023-07-20T04:22:23-07:00,
             /users/1,2023-10-04T00:00:00Z,2023-11-02T23:59:00Z
/orgs/1/sec_policy/active/rule_sets/1/sec_rules/1,Allow outbound connections,/
               orgs/1/sec_policy/active/rule_sets/1,Default,0,1,2023-07-25T04:52:39-07:00,
             /users/1,2023-10-04T00:00:00Z,2023-11-02T23:59:00Z
/orgs/1/sec_policy/active/enforcement_boundaries/5,my test deny rule with iplists,
```

```
              "","",0,-1,2023-07-20T03:00:05-07:00,
             /users/1,2023-10-04T00:00:00Z,2023-11-02T23:59:00Z
/orgs/1/sec_policy/active/enforcement_boundaries/3,ransomware_deny_rule2,"",
              "",0,1,2023-06-30T17:16:38-07:00,
             /users/1,2023-10-04T00:00:00Z,2023-11-02T23:59:00Z
/orgs/1/sec_policy/active/enforcement_boundaries/1,ransomware deny rule,"",
              "",0,-1,2023-06-07T23:32:07-07:00,
             /users/1,2023-10-04T00:00:00Z,2023-11-02T23:59:00Z
```

Schedule a Recurrent Report

To create a recurring report, you need to create a report schedule. In this example, the report named "Monthly Rule Hit Count Report" is generated for the last 30 days, and will be sent via email to the person who requested the report.

Create a Report Schedule

```
 {
       "report_template": {
              "href": "/orgs/1/report_templates/rule_hit_count_report"
       },
       "report_parameters": {
              "report_time_range": {
                  "last_num_days": 30
              },
              "rule_sets": []
       },
       "send_by_email": true,
       "report_generation_frequency": "monthly",
       "name": "Monthly Rule Hit Count Report",
}
```

Other API Changes to Support the Rule Hit Count Feature

sec_policy_label_groups_get

The property `rule_hit_count_enabled_scopes` was added.

```
"properties": {
       "rule_hit_count_enabled_scopes": {
              "description": "Label Group is referenced by Rule Hit
                     Count Enabled Scopes",
              "type": "boolean"
              }
```

sec_policy_firewall_settings_get

The property `rule_hit_count_enabled_scopes` was added.

```
 {
       "properties": {
              "rule_hit_count_enabled_scopes": {
                  "description": "Workloads that match the scope
                      will have rule hit count enabled",
                  "$ref": "../common/rule_set_scopes_get.schema.json"
              }
       }
}
```

- report_schedules_get
- report_schedules_put
- report_schedules_post
- reports_post
- report_templates_get

```
{
       "$ref": "rule_hit_count_report_params.schema.json"
}
```

In all these listed APIs, a reference to the schema `rule_hit_count_report_params` was added.

Custom iptables Rules

This Public Stable API allows you to leverage preexisting iptables rules on Linux workloads and add them as rules to rulesets.

You can use the rules API to create custom iptables rules in situations where your Linux workloads have preexisting iptables rules configured that you would like to keep in addition to rules you create using Illumio Core.

If you configured iptables on Linux workloads before using Illumio Core, when you pair a workload, the VEN assumes control of the iptables to enact policy and disables any pre-programmed iptables. To solve this, you can use the Rules API to leverage your own iptables rule configurations in a ruleset.

Custom iptables Rules

These terms clarify the relationship between your iptables rules and Illumio Core rules:

- **iptables**: Linux host configuration before the VEN is installed
- **Rules**: Configurations in the PCE that define the allowed communication between two or more workloads or other entities (IP lists, labels representing multiple workloads, and label groups)
- **Custom iptables rules**: PCE rules that leverage your iptables rule configurations that get programmed on your workloads by the VEN and managed by the PCE

How Custom iptables Rules Work

Custom iptables rules in the PCE consist of a list of predefined iptables statements and the entities that receive the rule definitions. Each rule can have a list of iptables configurations, which allows you to group a sequence of rules for a specific function. Custom iptables rules are programmed after the Illumio PCE generates the iptables rules and they are provisioned.

Before custom iptables rules are sent to the VEN, they are checked for any unsupported tokens (such as names of firewall chains already in use by Illumio, matching against IP sets, and semicolons). If an unsupported token is included, the rule cannot be saved or provisioned.

If the VEN fails to apply a custom iptables rule because of a missing package or an incorrectly formatted rule:

- Error is reported to the PCE and is logged as two audit events:
  “Firewall config failure” (`fw_config_failure`) and
  “Failed to apply policy changes” (`policy_deploy_failed`).

- The error is displayed in the VEN health status.
- The new policy is not used and the last known successful policy is used instead.

For policy distribution and enforcement, the VEN creates a custom chain that contains the rules for each table or chain in the iptables. Each custom chain is appended to the end of its corresponding chain in the correct table. When the VEN requests the policy, the `iptables` command is sent, including where the chain should be placed.

For security reasons, custom iptables rules only support rules in the `mangle`, `nat`, and `filter` tables.

The following table describes the permitted actions for each iptables type:

Table Name	Chain Names	Custom Rules
`raw`	prerouting, output	No
`mangle`	prerouting, input, output, forward, postrouting	Yes
`nat`	prerouting, output, postrouting	Yes
`filter`	input, output, forward	Yes
`security`	input, output, forward	No

Create a Custom iptables Rule

This method allows you to create a rule that can contain custom iptables.

Create a Custom iptables Rule

```
POST [api_version/[rule_set_href]/sec_rules
```

Query Parameters

Parameter	Description	Type	Required
`name`	Ruleset name (must be unique)	String	Yes
`scopes`	Scope for ruleset, which consists of a list of labels, with each list having at least one application, environment, and/or location label	Array	Yes
`external_data_set`	External data set identifier	String	No
`external_data_reference`	External data reference identifier	String	No
`enabled`	Whether the ruleset is enabled or not	Boolean	Yes

Parameter	Description	Type	Required
`rules`	Standard (non-iptables) rules	String	Yes
`iptables_rules`	Rules that use iptables (see following table for properties)	String	Yes

Custom iptables_rules Properties

Property	Description	Type	Required
`enabled`	Whether the rule is currently enabled	Enum	Yes
`ip_version`	Whether IPv4 or IPv6 is used	String	Yes
`description`	Description of ruleset	String	No
`actors`	Entities that receive the ruleset.	String	Yes
`statements`	Rules for iptables (table, chain name, and parameters), which consist of the following elements: • `table_name`: Name of iptables table, which is `nat`, `mangle`, or `filter` • `chain_name`: Name of iptables chain, which is `prerouting`, `input`, `output`, `forward`, or `postrouting` • `parameters`: Remaining iptables rules (excluding table name and chain name)	String	Yes

For more information on rules, see Rulesets.

Request Body

In this example, a ruleset named `test_ipt_rs` is created that contains two iptables rules.

> NOTE:
> Each iptables rule can contain multiple statements.

```
          {
"name": "test_ipt_rs",
"enabled": true,
  "scopes": [
[
  { "label": { "href": "/orgs/1/labels/24" } },
  { "label": { "href": "/orgs/1/labels/27" } },
  { "label": { "href": "/orgs/1/labels/21" } }
],
```

```
  ],
    "ip_tables_rules": [
        {
            "enabled": true,
            "actors": [{"label": { "href": "/orgs/1/labels/11" }}],
            "statements": [
                {
                    "table_name": "mangle",
                    "chain_name": "PREROUTING",
                    "parameters": "-i eth0 -p tcp --dport 2222 -j MARK --set-mark
2222"
                },
                {
                    "table_name": "nat",
                    "chain_name": "PREROUTING",
                    "parameters": "-i eth0 -p tcp -m mark --mark 2222 -j REDIRECT
--to-port 3333"
                },
                {
                    "table_name": "filter",
                    "chain_name": "INPUT",
                    "parameters": "-i eth0 -p tcp -m mark --mark 2222 -j ACCEPT"
                }
            ],
            "ip_version": "4"
        },
        {
            "enabled": true,
            "actors": [{ "actors": "ams" }],
            "statements": [
                {
                    "table_name": "nat",
                    "chain_name": "POSTROUTING",
                    "parameters": "-o eth1 -s 192.0.2.10! -d 198.51.100.0/24 -j
MASQUERADE"
                }
            ],
            "ip_version": "4"
```

```
        }
    ]
}
```

Create Custom iptables Rule

```
            curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/2/sec_
policy/draft/rule_sets -H "Content-Type:application/json" -u $KEY:$TOKEN-d '
{"name":"test_ipt_rs","enabled":true,"scopes":[{[],[]}],"ip_tables_rules":
[{"enabled":true,"actors":[{"label":{"href":"/orgs/1/labels/11"}}],"statements":
[{"table_name":"mangle","chain_name":"PREROUTING","parameters":"-i eth0 -p tcp --
dport 2222 -j MARK --set-mark 2222"},{"table_name":"nat","chain_
name":"PREROUTING","parameters":"-i eth0 -p tcp -m mark --mark 2222 -j REDIRECT --
to-port 3333"},{"table_name":"filter","chain_name":"INPUT","parameters":"-i eth0 -
p tcp -m mark --mark 2222 -j ACCEPT"}], "ip_version":"4"},
{"enabled":true,"actors":[{"actors":"ams"}],"statements":[{"table_name":"nat",
"chain_name":"POSTROUTING","parameters":"-o eth1 -s 10.0.0.2 ! -d 172.17.0.0/16 -j
MASQUERADE"}], "ip_version":"4"}]}'
```

Response Body

Property	Description	Type
href	Identifier for the resource	String

Response

```
            {
  "href": "/orgs/1/sec_policy/draft/rule_sets/17",
  "created_at": "2023-02-24T23:19:01.020Z",
  "updated_at": "2023-02-24T23:19:01.020Z",
  "deleted_at": null,
  "created_by": {
    "href": "/users/1"
  },
  "updated_by": {
    "href": "/users/1"
  },
```

```
"deleted_by": null,
"name": "test_ipt_rs",
"description": null,
"enabled": true,
"scopes": [
  [
    { "label": { "href": "/orgs/1/labels/24" } },
    { "label": { "href": "/orgs/1/labels/27" } },
    { "label": { "href": "/orgs/1/labels/21" } }
  ],
  [
    { "label": { "href": "/orgs/1/labels/15" } },
    { "label": { "href": "/orgs/1/labels/16" } },
    { "label": { "href": "/orgs/1/labels/17" } }
  ]
 ],
],
"rules": [],
"ip_tables_rules": [
  {
    "href": "/orgs/1/sec_policy/draft/rule_sets/17/ip_tables_rules/20",
    "created_at": "2023-02-24T23:19:01.280Z",
    "updated_at": "2023-02-24T23:19:01.280Z",
    "deleted_at": null,
    "created_by": {
      "href": "/users/1"
    },
    "updated_by": {
      "href": "/users/1"
    },
    "deleted_by": null,
    "description": null,
    "enabled": true,
    "actors": [
      {
        "actors": "ams"
      }
    ],
```

```
    "ip_version": "4",
    "statements": [
      {
        "table_name": "nat",
        "chain_name": "POSTROUTING",
        "parameters": "-o eth1 -s 192.0.2.0 ! -d 198.51.100.0/24 -j MASQUERADE"
      }
    ]
  },
  {
    "href": "/orgs/1/sec_policy/draft/rule_sets/17/ip_tables_rules/18",
    "created_at": "2023-02-24T23:19:01.229Z",
    "updated_at": "2023-02-24T23:19:01.229Z",
    "deleted_at": null,
    "created_by": {
      "href": "/users/1"
    },
    "updated_by": {
      "href": "/users/1"
    },
    "deleted_by": null,
    "description": null,
    "enabled": true,
    "actors": [
      {
        "label": {
          "href": "/orgs/1/labels/11",
          "key": "loc",
          "value": "test"
        }
      }
    ],
    "ip_version": "4",
    "statements": [
      {
        "table_name": "filter",
        "chain_name": "INPUT",
        "parameters": "-i eth0 -p tcp -m mark --mark 2222 -j ACCEPT"
```

```
        },
        {
          "table_name": "nat",
          "chain_name": "PREROUTING",
          "parameters": "-i eth0 -p tcp -m mark --mark 2222 -j REDIRECT --to-port
3333"
        },
        {
          "table_name": "mangle",
          "chain_name": "PREROUTING",
          "parameters": "-i eth0 -p tcp --dport 2222 -j MARK --set-mark 2222"
        }
      ]
    }
  ]
}
```

Machine Authentication

This Public Experimental API allows you to configure unmanaged workloads and rules for machine authentication in case you configured the PCE to use machine authentication.

Before you start writing rules, you need to complete the following tasks:

- Configure an unmanaged (no VEN) workload that you want to use machine authentication on with the client certificate X.509 Subject distinguished name (`distinguished_name`) issued from the CA. If you are using machine authentication with managed workloads (with VENs installed), you do not need to set this property.
- Configure rules for machine authentication by setting the `machine_auth` flag to true on each rule. You can also optionally set SecureConnect (`sec_connect`) if you want the traffic data to be encrypted using IPsec.

Once you have done these two tasks, you can use these unmanaged workloads in machine authentication-based rules.

Configure Machine Authentication

The machine authentication workload property for the certificate distinguished name is required for those hosts or systems where you have not installed a VEN, such a laptop or other server whose IP address is unknown or changes often.

You can set the `distinguished_name` when you first create (POST) the unmanaged workload, which is passed in the JSON request payload.

> NOTE:
> For information on how to create an unmanaged workload, see Create an Unmanaged Workload.

URI to Configure Machine Authentication on an Unmanaged Workload

Use this URI to configure machine authentication when you create a new unmanaged workload:

```
POST [api_version][org_href]/workloads
```

If you want to enable machine authentication on an existing unmanaged workload, you need to know the workload HREF, which can be obtained from the command GET on a collection of Workloads.

The workload HREF is highlighted in blue:

```
/orgs/7/workloads/XXXXXXX-9611-44aa-ae06-fXXX8903db65
```

Use this URI to configure machine authentication for an existing unmanaged workload:

```
PUT [api_version][workload_href]
```

Request Parameter

Parameter	Description
`distinguished_name`	The X.509 Subject distinguished name, used if you want this unmanaged workload to use machine authentication when communicating with other hosts.

Request Body

```
{
   "distinguished_name": "CN=ACCVRAIZ1, OU=PKIACCV, O=ACCV, C=ES"
}
```

Curl Command Enable Machine Authentication

```
curl -i -X PUT https://pce.my-company.com/api/v2/orgs/7/workloads/XXXXXXX-9611-
44aa-ae06-fXXX8903db65 -H "Content-Type:application/json" -u $KEY:$TOKEN -d '
{"distinguished_name": "CN=ACCVRAIZ1, OU=PKIACCV, O=ACCV, C=ES"}'
```

Configure Machine Authentication on Rule

For a rule to use machine authentication, you need to configure it on the rule when you create or update it.

URI to Configure Machine Authentication for a Rule

Use this URI to configure machine authentication for a new rule:

```
POST [api_version][rule_set_href]/sec_rules
```

If you want to enable machine authentication on an existing rule, you need to know the HREF of the rule. For example:

```
/orgs/3/sec_policy/draft/rule_sets/152/sec_rules/124
```

Use this URI to configure machine authentication for an existing rule:

```
PUT [api_version][sec_rule_href]
```

Request Parameters

Parameter	Description
`machine_auth`	Optional boolean flag to enable machine authentication for the rule. When set to true, machine authentication is enabled for the rule.
`sec_connect`	Optional boolean flag to enable SecureConnect (host-to-host traffic encryption) for the rule.

Request Body

This example shows the JSON payload for updating a rule to enable machine authentication, but with SecureConnect disabled.

```
{
    "providers": [{"label": {"href": "/orgs/1/labels/1"}}],
```

```
    "sec_connect": false,
    "consumers": [{
        "actors": "ams"
    }],
    "consuming_security_principals": [],
    "unscoped_consumers": false,
    "description": "",
    "ingress_services": [{"proto": 6}],
    "resolve_labels_as": {
      "providers": ["workloads"],
      "consumers": ["workloads"]
    },
    "enabled": true,
    "machine_auth": true
}
```

Configure Machine Authentication for Rule

```
curl -i -X PUT https://pce.my-company.com/api/v2/orgs/1/sec_policy/draft/rule_
sets/152/sec_rules/124 -H "Content-Type:application/json" -u $KEY:$TOKEN -d '
{"providers":["{"label": {"href":"/orgs/1/labels/1"}}], "sec_connect":false,
"consumers":[{"actors":"ams"}],"consuming_security_principals":[], "ingress_
services": [{"proto": 6}], unscoped_consumers":false, "description":","resolve_
labels_as":{"providers":["workloads"],"consumers":
["workloads"]},"enabled":true,"machine_auth":true}'"consumers":
[{"actors":"ams"}],"consuming_security_principals":[], "ingress_services":
[{"proto": 6}], unscoped_consumers":false, "description":","resolve_labels_as":
{"providers":["workloads"],"consumers":["workloads"]},"enabled":true,"machine_
auth":true}'
```

Enforcement Boundaries

In the Illumio Core 21.2.0 release, Illumio introduced Enforcement Boundaries, a new feature to speed your journey toward Zero Trust.

The Illumio security policy model is based on the principle of Zero Trust. Achieving Zero Trust security is possible with Illumio Core because it bases security policy on an allowlist model. From a security perspective, creating policy based on allowlists is the preferred method and has the advantage of specifying what you trust explicitly. However, you can encounter situations when you need more flexibility in segmenting your data centers. The solution is to introduce a new set

of rules that determine where segmentation rules apply. These rules are referred to as Enforcement Boundaries in Illumio Core.

Enforcement Boundaries can block traffic from communicating with workloads you specify, while still allowing you to progress toward a Zero Trust environment.

For more information about deploying Enforcement Boundaries in your data center, see Policy Enforcement in the *Security Policy Guide*.

Selective Enforcement vs. Enforcement Boundaries

The introduction of Enforcement Boundaries resulted in changes to *What's New in This Release*.the REST API. This topic describes the major changes. For a description of all changes due to Enforcement Boundaries, see Enforcement Boundaries in the "Illumio Core REST API in 21.2" chapter of

Documentation Update: In Illumio Core 21.2, this topic for Enforcement Boundaries replaces the Illumio Core 20.2.0 topic for Selective Enforcement.

The APIs with the endpoints `enforcement_boundaries` replace the APIs with the endpoints `selective_enforcement_rules`. Specifically, the APIs for Enforcement Boundaries replace the APIs used for Selective Enforcement as follows:

- `sec_policy_selective_enforcement_rules_get.schema.json` has been replaced with `sec_policy_enforcement_boundaries_get.schema.json`
- `sec_policy_selective_enforcement_rules_post.schema.json` has been replaced with `sec_policy_enforcement_boundaries_post.schema.json`
- `sec_policy_selective_enforcement_rules_put.schema.json` has been replaced with `sec_policy_enforcement_boundaries_put.schema.json`

Changes to the Policy Modes

In addition to the changes for Enforcement Boundaries, the policy modes changed in Illumio Core 20.2.0 and later releases in the following ways.

The existing common schema `workload_modes.schema.json` is DEPRECATED:

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "description": "DEPRECATED AND REPLACED (Use enforcement_mode instead)",
    "type": "string",
    "enum": ["idle", "illuminated", "enforced"]
}
```

The common `workload_enforcement_mode.schema.json` is added.

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "description": "Workload enforcement mode",
    "type": "string",
    "enum": ["idle", "visibility_only", "full", "selective"]
}
```

The following list compares the policy modes in Illumio Core 20.2.0 to 21.2.0:

- `idle` is the same
- `illuminated` (build, test) = `visibility_only`
- `enforced` = `full`
- `selective`: Added by `workload_enforcement_mode.schema.json`

Enforcement Boundaries in the REST API

The RBAC roles Global Org Owner and Global Admin can manage Enforcement Boundaries without restrictions.

You can only use Enforcement Boundaries with managed workloads. You cannot apply Enforcement Boundaries to NEN-controlled or other unmanaged workloads.

One or more ports on a workload are enforced ("port enforcement") while leaving the remaining ports unenforced. Instead of configuring workloads directly, enforcement is controlled using policies.

Workloads have to be placed in `selective` mode when using Enforcement Boundaries for them. Therefore, to use an Enforcement Boundary, you need to perform two separate configurations:

- Set the workload policy state to `selective`.
- Create security policy with a scope that includes the workload.

Enforcement Boundaries Methods

Functionality	HTTP	URI
View the configured enforcement boundaries	`GET`	`[api_version][org_href]/sec_policy/:version/enforcement_boundaries:/id`
Edit the specified enforcement boundary	`PUT`	`[api_version][org_href]/sec_policy/:version/enforcement_boundaries/:id`
Create a new enforcement boundary	`POST`	`[api_version][org_href]/sec_policy/:version/enforcement_boundaries`
Delete the specified enforce-	`DELETE`	`[api_version][org_href]/sec_policy/:version/enforcement_boundaries/:id`

Functionality	HTTP	URI
ment boundary		

Enforcement Boundaries Parameters

Parameter	Method	Description	Type	Required
`org_id`	`GET, PUT, POST, DELETE`	Organization ID	Integer	Yes
`pversion`	`GET, PUT, POST, DELETE`	Security Policy Version	String	Yes
`labels`	`GET`	List of lists of label URIs, encoded as a JSON string	String	No
`max_results`	`GET`	Maximum number of Rule Sets to return	Integer	No
`name`	`GET`	Filter by name supports partial matching	String	No
`service`	`GET`	Service URI	String	No
`service_ports.port`	`GET`	Specify port or port range to filter results. The range is from -1 to 65535.	String	No
`service_ports.proto`	`GET`	Protocol to filter on	Integer	No
`enforcement_boundary_id`	`PUT`	Enforcement boundary ID	Integer	Yes

Enforcement Boundaries Properties

Property	Method	Description	Type	Required
`href`	`GET`	URI of the selective enforcement rule	String	Yes
`name`	`GET, PUT, POST`	Name of the selective enforcement rule	String	Yes
`providers`	`GET, PUT, POST`	`label` .....Label URI. Required parameter is `href`. `label_group` .....Label group URI. Required parameter is `href`.	Array	Yes

Property	Method	Description	Type	Required

		`ip_list`
		.....IP List URI. Required parameter is `href`.
		`actors`
		.....Label group URI. Required parameter is `href`.		

`consumers`	`GET`, `PUT`, `POST`	`label`
		.....Label URI. Required parameter is `href`.
		`label_group`
		.....Rule actors are all workloads ('ams').
		`ip_list`
		.....IP List URI. Required parameter is `href`.
		`actors`
		.....Rule actors are all workloads ('ams').
			Array	Yes

`ingress_services`	`GET`, `PUT`, `POST`	Collection of services that are enforced `port`: Port number, or the starting port of a range. If unspecified, this will apply to all ports for the given protocol. minimum: 0, maximum: 65535 `to_port`: Upper end of port range; this field should not be included if specifying an individual port. minimum: 0, maximum: 65535 `proto`: Transport protocol (numeric) enum: 6,17	Array	Yes
`created_at`	`GET`	Timestamp when this Enforcement Boundary was first created. Format `date-time`	String date/time	No
`updated_at`	`GET`	Timestamp when this Enforcement Boundary was last updated. Format `date-time`	String date/time	No
`deleted_at`	`GET`	Timestamp when this Enforcement Boundary was deleted	String date/time	No
`created_by`	`GET`	User who originally created this Enforcement	String	No

Property	Method	Description	Type	Required
		Boundary Required parameter `href`.		
`updated_by`	`GET`	User who last updated this Enforcement Boundary Required parameter `href`.	String	No
`deleted_by`	`GET`	User who deleted this Enforcement Boundary Required parameter `href`.	String	No
`update_type`	`GET`	Type of update	String	No
`enabled`	`POST`, `PUT`	For `POST`: The optional `enabled` boolean field can be provided in the payload. If it is not provided, the newly created enforcement boundary object is enabled by default. For `PUT`: The optional boolean value for the enabled field in the payload is: `"enabled": true`	Boolean	No

Get Enforcement Boundaries

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/1/sec_
policy/draft/enforcement_boundaries -H "Accept: application/json" -u $KEY:$TOKEN
```

Response

In this response, the former scope property is replaced with providers, and another property consumers was added. The required properties are: name, providers, consumers, and `ingress_services`(formerly `enforced_service`).

```
{
"href":"/orgs/1/sec_policy/draft/enforcement_boundaries/1",
"created_at":"2021-09-21T21:48:40.228Z",
"updated_at":"2021-09-21T21:48:40.241Z",
"deleted_at":null,
"created_by":{
"href":"/users/1"
},
"updated_by":{
```

```
                "href":"/users/1"
                                },
        "deleted_by":null,
        "update_type":"create",
        "name":"Dev to Prod separation",
        "providers":[
                {
                        "label":{
                                "href":"/orgs/1/labels/7",
                                "key":"env",
                                "value":"Production"
                        }
                }
        ],
        "consumers":[
                {
                        "label":{
                                "href":"/orgs/1/labels/9",
                                "key":"env",
                                "value":"Development"
                        }
                }
        ],
        "ingress_services":[
                {
                        "href":"/orgs/1/sec_policy/draft/services/1",
                        "created_at":"2021-09-21T16:31:16.266Z",
                        "updated_at":"2021-09-21T16:31:16.292Z",
                        "deleted_at":null,
                        "created_by":{
                                "href":"/users/0"
                        },
                        "updated_by":{
                                "href":"/users/0"
                        },
                        "deleted_by":null,
                        "update_type":null,
                        "name":"All Services",
```

```
                        "service_ports":[
                                {
                                        "proto":-1
                                }
                        ]
                }
        ],
        "caps":[
                "write",
                "provision"
        ],
        "workload_counts":{

        }
}
```

Create Enforcement Boundaries

```
            curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/1/sec_
policy/draft/enforcement_boundaries -H "Content-Type: application/json" -u
$KEY:$TOKEN -d '{"name": "eb1", "providers": [{"actors": "ams"}], "consumers":
[{"actors": "ams"}], "ingress_services": [{"port": 1, "proto": 6}]}'
```

Edit Enforcement Boundaries

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/1/sec_
policy/draft/enforcement_boundaries/1 -H "Content-Type: application/json" -u
$KEY:$TOKEN -d '{"name": "a4"}'
```

```
{
        "name": "a name here",
        "providers": [
                {"label": "/orgs/1/labels/13"},
                {"label": "/orgs/1/labels/15"},
                {"ip_list": "/orgs/1/sec_policy/draft/ip_lists/22"}
        ],
        "consumers": [
```

```
                {"actors": "ams"}
                ],
        "ingress_services": [
                {"href": "/orgs/1/sec_policy/draft/services/20"},
                {"port": 22, "proto": 6},
                {"port": 8080, "to_port": 8088, "proto": 6}
        ]
}
```

Chapter 7

RBAC for PCE Users

This chapter contains the following topics:

As an Illumio administrator, use the Role-based Access Control (RBAC) API to assign privileges and responsibilities to users as follows:

- Establish the least required privileges to perform a job.
- Limit access to the smallest operation-set to perform a job.
- Separate users' duties, such as give the responsibility or delegate authority to a specific team.
- Allow access based on roles and scopes. Scopes in the Illumio Core specify the domain boundaries granted to a user.
- Manage user authentication and authorization.

RBAC Overview

The Role-based Access Control (RBAC) is an API that gets, creates, updates, or deletes permissions for users and groups. These users and groups are managed locally by the PCE or externally by a single sign-on (SSO) identity provider (IdP).

Before you begin using the RBAC feature with the REST API, learn about the Illumio Core permissions model and its terms and concepts.

RBAC Terms and Concepts

You should be familiar with the following RBAC terms before using this API:

User

A user is a PCE account that provides login or API access to the PCE. A user can be managed locally by the PCE or externally through an IdP.

Permission

A permission represents a combination of a user's account, an RBAC role, and an optional scope. You can grant multiple permissions to a user, depending on your requirements. A permission is a three tuple consisting of a role, a scope, and an authorization security principal:

- **Role**: User personas that are associated with a set of allowed operations, such as creating new labels or provisioning policy changes. Roles can be one of two general types: unscoped and scoped.
-
  - **Unscoped roles** (or roles with “global scopes”) do not have restrictions on the types of resources on which a user can operate. This means that the role is not affected by any label scopes.
  - **Scoped roles** use one or more unique application, environment, and location labels (each with a label HREF, key, and value), to restrict user or group permissions to only those objects that share the same labels. Specifically, scoped roles allow certain users to create rules and rulesets and provision them.
- **Scope**: A set of three labels (one of each type for Application, Environment, and Location) that restricts operations to those workloads sharing the same labels as the scope label set.
  - GET, POST, and PUT permissions methods for the Ruleset Manager (limited or full) or Ruleset Provisioner roles have a required scope parameter. When granting permissions, choose a scope that restricts which resources these users can use in a ruleset, or which resources they can provision.
  - A scope contains zero or more applications, environment, and location labels. Each label in the scope is identified by its HREF. A scope can also contain zero or more label groups.
  - If the scope is an empty array (`[]`), it includes all applications, environments, and locations.

- If one of the label types is not specified, all instances of that type are permitted. For example, if application labels are omitted but environment and location labels are present, all applications are within the scope.

- **Authorization Security Principal**: The binding that connects a user account with its permissions (a role, and depending on the role, scopes).

> NOTE:
> If you are using an external identity provider to manage user access to the PCE, make sure that your identity provider is configured and those external users have been added to the PCE *before* you use this API to assign user permissions.

Grant Permissions Workflow

Granting user permissions with the REST API follows this general workflow:

1. **Create a local user (optional)**

   This step creates a new local PCE user with no permissions and sends an e-mail invitation to the user's e-mail address. (If you use an external identity provider to manage user access to the PCE, skip this step.)

2. **Create an authorization security principal**

   An authorization security principal serves as the binding between a user or a group and an RBAC role and optional scopes.

3. **Grant permissions by assigning a role and scopes to the authorization security principal**

   Once a user account has been associated with an authorization security principal, you can assign an RBAC role to the account and add custom scopes if the user role requires them.

List User Roles and Role Names

The APIs GET roles and GET `role_name` have been promoted from Internal to Public Experimental.

They allow the users to list user roles and role names.

Functionality	HTTP	URI
Get the roles in the organization	`GET`	`[api_version] /orgs/:xorg_id/roles`
Get information for this role name	`GET`	`api_version]/orgs/:xorg_id/roles/:role_name`

RBAC User Operations

This Public Stable API creates, updates, re-invites local users, and converts user status (a local user to an external user or an external user to a local user). This API is intended only for local users managed by the PCE, not users managed by an external identity provider (IdP).

API Methods

Functionality	HTTP	URI
Get a collection of users	`GET`	`[api_version]/users`
GET an individual user	`GET`	`[user_href]`
Get all the orgs the user has accessed after logging in *(this endpoint is Public Experimental)*	`GET`	`[api_version][user_href]/orgs`
Create a local user and send an e-mail invitation	`POST`	`[api_version]/users`
Convert an external user to a local user	`POST`	`[user_href]local_profile`
Delete a local user and convert to an external user	`DELETE`	`[user_href]local_profile`
Re-invite a local user	`PUT`	`[user_href]local_profile/reinvite`
For authenticated users: change your password by sending a request to the agent service.	`PUT`	`[user_href]local_profile/password`

Parameters for RBAC Users

Property	Description	Type	Required
`type`	Indicates that the user created is a `local` user managed by the PCE.	String	No
`id`	User ID	Integer	Yes

Properties for RBAC Users

Property	Description	Type	Required
`href`	User URI	String	Yes
`username`	Identify a local user by an e-mail address, which must meet these requirements: • Be unique • Use the format xxxx@yyyy.zzz • Be 255 characters or less	String (email)	Yes

Property	Description	Type	Required
last_login_on	This is populated automatically after a login	String	Yes
last_login_ip_address	This is populated automatically after a login	String	Yes
login_count	Number of times this user logged in	Integer	Yes
full_name	User's full name	String	Yes
time_zone	Time Zone IANA Region Name	String	Yes
type	User's type, i.e. user authenticated local or remotely via SAML	String	Yes
updated_at	Timestamp when this user was last updated	String	Yes
created_at	Timestamp when this user was first created	String	Yes
current_password	Current password that you want to change	String	Yes
new_password	New password to set	String	Yes

RBAC Users

Get RBAC Users

These methods get a collection of users or an individual user in the organization.

By default, the maximum number of users returned from a GET collection is 500. If you want to get more than 500 users, use an Asynchronous GET Collection.

URI to Get a Collection of Local Users

```
GET [api_version]/users
```

URI to Get an Individual User

```
GET [user_href]
```

Curl Command Get Collection of Local Users

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/users?type=local -H "Accept: application/json" -u $KEY:$TOKEN
```

Response


```
[
    {
        "href": "/users/99",
        "type": "local",
        "effective_groups": [],
        "id": 99,
        "username": "joe.user@example.com",
        "full_name": "Joe User",
        "time_zone": "America/Los_Angeles",
        "locked": false,
        "login_count": 1,
        "last_login_ip_address": "192.x.x.x",
        "last_login_on": "2016-03-11T08:19:17.587Z",
        "local_profile": { "pending_invitation": false },
        "created_at": "2016-03-08T20:58:05.882Z",
        "updated_at": "2016-03-11T08:19:17.588Z"
    }
.....................................................................
    {
        "href": "/users/56",
        "type": "local",
        "effective_groups": [],
        "id": 56,
        "username": "jeff.user@example.com",
        "full_name": "Jeff User",
        "time_zone": "America/New_York",
        "locked": false,
        "login_count": 21,
        "last_login_ip_address": "192.x.x.x",
        "last_login_on": "2017-05-26T14:22:37.643Z",
        "local_profile": { "pending_invitation": true },
        "created_at": "2016-05-02T07:16:21.725Z",
        "updated_at": "2017-05-26T14:23:04.625Z"
    }
]
```

Pending Invitation

Users with `"pending_invitation": "true"` in the response have not yet accepted the invitation to log in and create an account.

```
{
    "href": "/users/56",
    "type": "local",
    "effecve_groups": [],
    "id": 56,
    "username": "jeff.user@example.com",
    "full_name": "Jeff User",
    "time_zone": "America/New_York",
    "locked": false,
    "login_count": 21,
    "last_login_ip_address": "192.x.x.x",
    "last_login_on": "2017-05-26T14:22:37.643Z",
    "local_profile": { "pending_invitation": true },
    "created_at": "2016-05-02T07:16:21.725Z",
    "updated_at": "2017-05-26T14:23:04.625Z"
}
```

Create a Local User

This method creates local users who are managed by the PCE.

URI to Create a Local User

```
POST [api_version]/users
```

Request Body

```
{
  "username": "joe_user@mycompany.com",
  "display_name": "Joe User ",
  "type": "local"
}
```

Curl Command to Create a Local User

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/users -H "Content-Type:
application/json" -u $KEY:$TOKEN -d '{"username": "joe_
user@mycompany.com","display_name": "Joe User","type": "user"}'
```

User Profiles

Change the status of a user by converting a local user to an external user or an external user to a local user.

Convert Local to External User

This method converts a local user to an external user by *deleting* the local user account profile.

Use the user HREF, which is obtained from the response when a user logs into the PCE using the Login API or from the GET collection response.

For example: `/users/14`

URI to Convert a Local User to an External User

```
DELETE [user_href]/local_profile
```

Example

```
DELETE https://pce.my-company.com:8443/api/v2/users/14/local_profile
```

Convert Local User to External User

```
curl -i -X >DELETE https://pce.my-company.com:8443/api/v2/users/14/local_profile -
H "Accept: application/json" -u $KEY:$TOKEN
```

Convert External User to Local User

This method converts externally managed users to local users who are managed by the PCE.

URI to Convert an External User a Local User

```
POST [user_href]/local_profile
```

Example

```
POST https://pce.my-company.com:8443/api/v2/users/14/local_profile
```

Curl Command Convert External User to Local User

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/users/14/local_profile -H
"Content-Type: application/json" -u $KEY:$TOKEN
```

Re-invite a Local User

If you have already created a local user, but that user has not logged in yet for the first time, you can use this method to resend the email invitation. Once they receive the invitation, they can log into the PCE and complete their PCE user account registration.

URI to Re-invite a Local User

```
PUT [user_href]/local_profile/reinvite
```

Example

```
PUT https://pce.my-company.com:8443/api/v2/users/14/local_profile/reinvite
```

Curl Command to Re-invite a Local User

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/users/14/local_
profile/reinvite -H "Content-Type: application/json" -u $KEY:$TOKEN
```

RBAC Permissions

This Public Experimental API grants permissions to PCE users and groups. It also returns a collection of permissions in the organization, gets individual user permissions, and updates and deletes permissions.

> NOTE:In addition to labels, label groups have been added as part of the response and parameters because they are now supported in user scopes.

API Methods

Functionality	HTTP	URI
Get a list of all RBAC permissions for the organization (schema and query parameter format change)	GET	[api_version]orgs/{org_id}/permissions
Get an individual permission (schema change)	GET	[api_version]{org_id}/permissions/{permission_id}
Grant a permission (schema change)	POST	[api_version]orgs/{org_id}/permissions
Update a permission (schema change)	PUT	[api_version]orgs/{org_id}/permissions/{permission_id}

New Schema and Query Parameter

For the above endpoints, the `org_scope.schema.json` is now used instead of `labels_summary.schema.json` and `labels.schema.json`.

For the endpoint `GET /api/v2/orgs/1/permissions`, the query parameter is changed from

```
scope: ["/orgs/1/labels/5", "/orgs/1/labels/3"]
```

to

```
scope: [{"label":{"href":"/orgs/1/labels/5"}},{"label":
{"href":"/orgs/1/labels/3"}}]
```

Parameters for Roles

Unscoped Roles

API Role Name	UI Role Name	Granted Access
owner	Global Organization Owner	Perform all actions: Add, edit, or delete any resource, security settings, or user accounts.
admin	Global Administrator	Perform all actions except cannot change security settings and cannot perform user management tasks.
read_only	Global read-only	View any resource or security settings. Cannot perform any operations.

API Role Name	UI Role Name	Granted Access
`global_object_provisioner`	Global Policy Object Provisioner	Provision rules containing IP lists, services, and label groups, and manage security settings. Cannot provision rulesets, virtual services, or virtual servers, or add, modify, or delete policy items.

Scoped Roles

API Role Name	UI Role Name	Granted Access
`ruleset_manager`	Full Ruleset Manager	Add, edit, and delete all rulesets within a specified scope. Add, edit, and delete rules when the provider matches a specified scope. The rule consumer can match any scope.
`limited_ruleset_manager`	Limited Ruleset Manager	Add, edit, and delete all rulesets within a specified scope. Add, edit, and delete rules when the provider and consumer match the specified scope. Ruleset managers with limited privileges cannot manage rules that use IP lists, user groups, label groups, iptables rules as consumers, or rules that allow internet connectivity.
`ruleset_provisioner`	Ruleset Provisioner	Provision rulesets within a specified scope. Cannot provision virtual servers, virtual services, SecureConnect gateways, security settings, IP lists, services, or label groups.
`scope`		See New Schema and Query Parameter.

Ruleset Manager and Ruleset Provisioner

If you are granting a user or group the Ruleset Manager or the Ruleset Provisioner role, you can also associate a scope to the role so you can control which rulesets they can add and provision.

There is a default read-only user permission that is organization-wide and inherited by all users in the organization. This global permission allows users who have no permissions explicitly granted to them to access the PCE.

> NOTE:
> For information, see Organization-wide Default User Permissions.

Role HREF Syntax

An RBAC role is identified in the REST API by its HREF, the exact syntax of which is based on the PCE organization HREF `[org_href]`.

```
[org_href]/roles/[role_name]
```

For example, if you wanted to grant a user permission with the Global Object Provisioner role, and your PCE organization HREF is `/org/6`, the role HREF would look like:

```
/orgs/6/roles/global_object_provisioner
```

Parameters for RBAC Permissions

Parameter	Description	Type	Required
`org_id`	Organization	Integer	Yes
`auth_security_principal`	The authorization security principal associated with the permission. It is not needed to get an individual permission or to delete a permission. The HREF of the authorization security principal (`auth_security_principal`) associated with the user or group being granted a permission. The HREF of an authorization security principal is returned when you create a new one, or you can GET a collection of authorization security principals in your PCE.	String	POST:Yes PUT:No
`role`	The RBAC role associated with the permissions. An RBAC role is identified in the REST API by its HREF, the exact syntax of which is different for every user and is based on the PCE organization HREF `[org_href]`. For example: `[org_href]/roles/[role_name]` For example, to grant a user permission with the Global Object	String	Yes

Parameter	Description	Type	Required
	Provisioner role, with a PCE organization HREF of `/org/6`, the role HREF would be: `/orgs/6/roles/global_object_provisioner` (For additional information about these roles and their associated capabilities, see the *PCE Administration Guide*.) Unscoped roles: • `owner` • `admin` • `read_only` • `global_object_provisioner` Scoped roles: • `ruleset_manager` • `limited_ruleset_manager` • `ruleset_provisioner`		
`scope`	Scope to filter on, where scope is in the format defined in `org_scope.schema.json`	String	No
`permission_id`	UUID of the permission	String	Yes

RBAC Properties

Parameter	Description	Type	Required
`org_id`	Organization	Integer	Yes
`permission_id`	UUID of the permission. Used to get, update, and delete an individual permission	String	Yes
`auth_security_principal`	The authorization security principal associated with the permission. It is not needed to get an individual permission or to delete a permission. Reference to `auth_security_principal_uri.schema.json`		
`role`	The RBAC role associated with the permissions. Reference to `common/orgs_roles.schema.json`		Yes

Parameter	Description	Type	Required
scope	Scope to filter on, where scope is in the format defined in org_scope.schema.json		Yes

Get RBAC Permissions

These methods get an individual user permission or a collection of permissions in the organization.

By default, the maximum number of permissions returned on a GET collection is 500. If you want to get more than 500, use an Asynchronous GET Collection.

URI to Get All Permissions in Your Organization

```
GET [api_version][org_href]/permissions
```

URI to Get an Individual Permission

```
GET [api_version][permissions_href]
```

Curl Command Get Permissions with a Specific Role

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/7/permissions?role=ruleset_provisioner -H "Accept: application/json" -u $KEY:$TOKEN
```

Grant RBAC Permissions

When an RBAC permission is granted to a user in the PCE, the user account (identified by its authorization security principal) is associated with a role. Depending on the role, scopes can be applied that restrict the permission to operating on specified labeled resources.

URI to Create a New Permission

```
POST [api_version][org_href]/permissions
```

Scoped Permissions

The permission for this scoped role consists of the following elements:

- A scope for the role (application, environment, and location labels)
- The role

- An authorization security principal associated with a user account

> NOTE:See the scope parameter change explained in New Schema and Query Parameter.

Example Request Body with `orgs_permission.schema.json`

```
{
    "scope": [
    {
    "label_group": {
        "href": "/orgs/1/sec_policy/active/label_groups/7d480df0-f5e1-4d1e-b088-
d8105150a883"
        }
    },
    {
    "label": {
         "href": "/orgs/1/labels/12"
        }
    }
    ],
    "role": {
        "href": "/orgs/1/roles/limited_ruleset_manager"
    },
    "auth_security_principal": {
    "href": "/orgs/1/auth_security_principals/177027ca-c3fe-4610-ac14-
fe5cba173af5"
    }
}
```

Example Response for a Scoped Permission

The response shows the new permission (at the top) that has been created identified by its HREF:

```
{
    "href": "/orgs/2",
        "display_name": "Luke",
            "permissions": [
            {
```

```
                "href": "/orgs/2/permissions/23dde367-41ea-4752-bfe5-16c173aad1a5",
                "role": {
                    "href": "/orgs/2/roles/limited_ruleset_manager"
                         },
                 "scope": [
                     {
                "label": {
                      "href": "/orgs/2/labels/452",
                       "key": "app",
                        "value": "App1"
                    }
           }
           {
                label: {
                       "href": "/orgs/2/labels/454",
                        "key": "loc",
                        "value": "Loc1"
                    }
                }
                ],
              "auth_security_principal": {
              "href": "/orgs/2/auth_security_principals/04b63b79-9883-4e84-acc5-
f727f1c67fa1"
                      }
       },
                .........
}
```

Unscoped Permissions

Request - Unscoped Permission

In this request for an unscoped permission, the required `scope` property is defined as an empty JSON array (`[ ]`).

> NOTE:When the `scope` parameter is empty, the change explained in New Schema and Query Parameter does not apply.

```
{
  "scope": [],
  "role": { "href": "/orgs/7/roles/owner" },
  "auth_security_principal":{"href":"/orgs/7/auth_security_principals/xxxxxxxx-
e4bf-4ba5-bd77-ccfc3a8ad999"}
}
```

Response - Unscoped Permission

```
{
  "href": "/orgs/7/permissions/51d9207c-354b-45de-9bf5-d1b613ac3719",
  "role": { "href": "/orgs/7/roles/owner" },
  "scope": [],
  "auth_security_principal":{"href":"/orgs/7/auth_security_principals/xxxxxxxx-
e4bf-4ba5-bd77-ccfc3a8ad999"}
}
```

Update an RBAC Permission

This method updates a permission, for example changing the permission role, authorization security principal, user, or group.

URI to Update a Permission

```
PUT [api_version][permissions_href]
```

Curl Command to Update the Role Permission

```
curl -i -X PUT https://pce.mycompany.com:8443/api/v2/orgs/7/permissions/xxxxxxxx-
354b-45de-9bf5-d1b613ac3719 -H "Content-Type: application/json" -u $KEY:$TOKEN -d
'{"scope": [{"href": "/orgs/7/labels/91", "key": "app", "value": "db"},{"href":
"/orgs/7/labels/92", "key": "loc", "value": "nyc"},{"href": "/orgs/7/labels/100",
"key": "env", "value": "prod"}],"role": {"href": "/orgs/7/roles/global_object_
provisioner"}, "auth_security_principal":{"href":"/orgs/7/auth_security_
principals/xxxxxxxx-e4bf-4ba5-bd77-ccfc3a8ad999"}}'
```

Delete an RBAC Permission

Curl Command to Delete a Permission

```
curl -i -X DELETE
https://pce.mycompany.com:8443/api/v2/orgs/7/.permissions/xxxxxxxx-354b-45de-9bf5-
d1b613ac3719 -H "Accept: application/json-u $KEY:$TOKEN
```

Authorization Security Principals

This Public Experimental API gets, creates, updates, and deletes authorization security principals.

An authorization security principal connects a user account with its permissions, which consists of a role and optional scopes.

API Methods

Functionality	HTTP	URI
Get a collection of authorization security principals in an organization	`GET`	`[api_version][org_href]/auth_security_principals`
Get an individual authorization security principal	`GET`	`[api_version][auth_security_principal_href]`
Create an individual authorization security principal	`POST`	`[api_version][org_href]/auth_security_principals`
Update an authorization security principal	`PUT`	`[api_version][auth_security_principal_href]`
Delete an authorization security principal	`DELETE`	`[api_version][auth_security_principal_href]`

Auth Principals Parameters

Parameters used for Auth Security Principals are:

Parameter	Description	Type	Required
`org_id`	Organization Id	Integer	Yes
`name`	Name of the authorization security principal. • If the user is local (managed by the PCE), the `name`must be an e-mail address of the local user. • If the user or group are managed by an external	String	GET, PUT: No POST: Yes

Parameter	Description	Type	Required
	IdP, use the name that identifies the external user or group in the external system.		
`type`	One of two types of users, either `user` or `group`.	String	GET, PUT: No POST: Yes
`auth_security_principal_id`	UUID of the auth_security_principal. Required for `[api_version][auth_security_principal_href]`	String	Yes
`display_name`	An optional display name for the authorization security principal.	String	No
`access_restriction`	Access restriction assigned to this user	String NULL	No

Auth Principals Properties

Property	Description	Type	Required
`href`	URI of auth_security_principal	String	Yes
`name`	Name of the authorization security principal. • If the user is local (managed by the PCE), the `name`must be an e-mail address of the local user. • If the user or group are managed by an external IdP, use the name that identifies the external user or group in the external system.	String	GET, PUT: No POST: Yes
`type`	One of two types of users, either `user` or `group`.	String	GET, PUT: No POST: Yes
`auth_security_principal_id`	UUID of the auth_security_principal. Required for `[api_version][auth_security_principal_href]`	String	Yes
`display_name`	An optional display name for the authorization security principal.	String	No
`access_restriction`	Access restriction assigned to this user	String NULL	No

Get Authorization Security Principals

This method gets an individual or a collection of authorization security principals in your organization.

By default, the maximum number returned from a GET collection of authorization security principals is 500. If you want to get more than 500, use an Asynchronous GET Collection.

URI to Get a Collection of Authorization Security Principals

```
GET [api_version][org_href]/auth_security_principals
```

URI to Get an Individual Authorization Security Principal

Use the `auth_security_principal_id` in a GET collection response (the last set of numbers in an HREF field).

```
GET [api_version][org_href]/auth_security_principals/{auth_security_principal_id}
```

Curl Command to Get Authorization Security Principals

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/auth_security_
principals -H "Accept: application/json" -u $KEY:$TOKEN
```

Response

Each individual authorization security principal returned is identified by its HREF. You can use the HREF to GET, PUT, or DELETE an authorization security principal.

```
 {
      "href": "/orgs/7/auth_security_principals/97cb9898-027b-474e-9807-
19e04460dfb0",
      "name": "jimmyjo@illum.io",
      "display_name": "Jimmy Joe Meeker",
      "type": "user"
   },
.............................
   {
      "href": "/orgs/7/auth_security_principals/db7a2657-dcb8-4237-a6e7-
7269cdbaea5d",
      "name": "foxy.brown@illumio.com",
```

```
      "display_name": "Foxy Brown",
      "type": "user"
   }
]
```

Curl Command to Get an Authorization Security Principal

```
curl -i -X GET -H "Accept: application/json -u $KEY:'TOKEN' https://pce.my-company.com:8443/api/v2/orgs/2/auth_security_principals/db7a2657-dcb8-4237-a6e7-7269cdbaea5d
```

Create an Authorization Security Principal

This method creates an individual authorization security principal.

URI to Create an Authorization Security Principal

```
POST [api_version][org_href]/auth_security_principals
```

Request Body - Local User Authorization Security Principal

```
{
   "type": "user",
   "name": "joe_user@illumio.com",
   "display_name": "Joe User"
}
```

Response Body - Local User Authorization Security Principal

```
{
   "href": "/orgs/7/auth_security_principals/e8c232d2-e4bf-4ba5-bd77-ccfc3a8ad999",
   "name": "joe_user@illumio.com",
   "display_name": "Joe User",
   "type": "user"
}
```

Request Body - External Group User Authorization Security Principal

```
{
   "type": "group",
   "name": "jCQN=Bank-Admin,OU=EU,DC=Acme,DC=com",
   "display_name": "Provisioners for Bank Accounts"
}
```

Response Body - External Group Authorization Security Principal

```
{
   "href": "/orgs/7/auth_security_principals/e8c232d2-e4bf-4ba5-bd77-
ccfc3a8ad777",
   "name": "jCQN=Bank-Admin,OU=EU,DC=Acme,DC=com",
   "display_name": "Acme Bank Admins",
   "type": "group"
}
```

Curl Command Create an Authorization Security Principal

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/2/auth_security_
principals -u $KEY:$TOKEN -H "Content-Type:application/json" -d '{"type":
"user","name":"joe_user@illumio.com", "display_name": "Joe User"}'
```

Update an Authorization Security Principal

In order to update an individual authorization security principal, use its HREF, which is obtained from the response from a GET collection.

URI to Update an Individual Authorization Security Principal

```
PUT [api_version][auth_security_principal_href]
```

Request Body

```
{
   "type": "user",
   "name": "joe_user2@illumio.com",
```

```
  "display_name": "Joe User"
}
```

Curl Command Create an Authorization Security Principle

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/2/sec_
policy/draft/services/79 -H "Content-Type:application/json" -u $KEY:$TOKEN -d '
{"type": "user", "name": "joe_user2@illumio.com", "display_name": "Joe User"}'
```

Delete an Authorization Security Principal

To delete an authorization security principal, use its HREF, which is returned in the response from a GET collection.

URI to Delete an Individual Authorization Security Principal

```
DELETE [api_version][auth_security_pincipal_href]
```

Curl Command Delete the Authorization Security Principal

```
curl -i -X DELETE -H "Accept: application/json" -u $KEY:$TOKEN https://pce.my-
company.com:8443/api/v2/orgs/2/auth_security_principals/e8c232d2-e4bf-4ba5-bd77-
ccfc3a8ad777
```

Organization-wide Default User Permissions

This Public Experimental API supplies an organization-wide default user permission and allows users to log into the PCE and view resources. These resources don't have to be explicitly assigned to any RBAC roles or scopes.

About Default User Permissions

If you use an external identity provider for user management, you might want to block some of those users from the PCE without removing them from your identity provider. *Deleting* the organization-wide read-only permission allows you to achieve this.

When the read-only user permission is disabled for your organization, users who are not explicitly assigned this permission cannot log into the PCE and access Illumio resources. If users without permissions attempt to log into the PCE, their external identity provider authenticates them but the PCE immediately logs them out.

To disable organization-wide read-only permissions:

1. Get a collection of all authorization security principals in your organization, and search the response for the one named `null`. Once you find this authorization security principal, make a note of its full HREF.
2. Get the HREF of the permissions object associated with the `null`authorization security principal. Keep a record of the JSON object for this permission in the event you want to re-enable the permission at a later date.
3. Delete the permission associated with the `null` authorization security principal.

Get a Collection of Authorization Security Principals

The first step in disabling the organization-wide read-only permission is to get a collection of all authorization security principals in your organization.

Curl Command Get Auth Security Principals Collection

```
curl -i -X GET https://pce.mycompany.com:8443/api/v2/orgs/7/auth_security_
principals -H "Accept: application/json" -u $KEY:$TOKEN
```

Example Response Body

The `null` authorization security principal in the following example is highlighted in blue:

```
[
.....................................................
  {
      "href": "/orgs/7/auth_security_principals/a23ea011-4191-49e6-a22a-
d3dba4fb8058",
      "name": null,
      "display_name": null,
      "type": "group"
   },
.....................................................
]
```

Get Permission for Null Auth Security Principal

To get the permission object associated with the `null` authorization security principal, call the GET Permissions API with the query parameter value set to the HREF for the `null` authorization security principal similar to curl command:

```
curl -i -X GET -H "Accept: application/json" -u $KEY:$TOKEN
https://pce.mycompany.com:8443/api/v2/orgs/7/permissions?auth_security_
principal=/orgs/7/auth_security_principals/a23ea011-4191-49e6-a22a-d3dba4fb8058
```

Response

The response returns the HREF of the permission associated with the organization-wide read-only permission.

```
{
      "href": "/orgs/7/permissions/14c92849-e88e-4930-8804-3245565619e5",
      "role": {
      "href": "/orgs/7/roles/read_only"
   },
      "scope": [],
      "auth_security_principal": {
      "href": "/orgs/7/auth_security_principals/a23ea011-4191-49e6-a22a-
d3dba4fb8058"
}
```

Delete Null Authorization Security Principal Permission

Keep a record of the permission object returned in case you want to re-enable the permission in the future.

Delete the read-only permission HREF to disable it.

Curl Command to Delete Null Authorization Security Principal Permission

```
curl -i -X DELETE -H "Accept: application/json" -u $KEY:$TOKEN
https://pce.mycompany.com:8443/api/v2/orgs/7/permissions?auth_security_
principal=/orgs/7/auth_security_principals//orgs/7/permissions/14c92849-e88e-4930-
8804-3245565619e5
```

Response

An HTTP 200 response is returned on the successful deletion of the organization-wide read-only permission.

Re-Enable Organization Read-Only Permission

If the organization-wide read-only permission was disabled, you can re-enable it by recreating the permission object. This object must be constructed exactly as the object that was returned to you

when you got the permission. The request body below illustrates the JSON structure of this permission object.

URI to Enable the Organization-Wide Read-Only Permission

```
POST [api_version][permission_href]
```

Request Body

```
{
    "role": {
        "href": "/orgs/7/roles/read_only"
    },
    "auth_security_principal": {
        "href": "/orgs/7/auth_security_principals/a23ea011-4191-49e6-a22a-
d3dba4fb8058"
    },
    "scope": []
}
```

Curl Command to Enable Organization Read-Only Permission

```
curl -i -X POST https://pce.mycompany.com:8443/api/v2/orgs/7/permissions -H
"Content-Type: application/json" -u $KEY:$TOKEN -d '{"role": {"href":
"/orgs/7/roles/read_only"}, "auth_security_principal":{"href":"/orgs/auth_
security_principals/a23ea011-4191-49e6-a22a-d3dba4fb8058"}, "scope": []}'
```

Response

An HTTP 201 response is returned on successfully recreating the organization-wide read-only permission.

App Owner RBAC Role

The App Owner RBAC (Role-Based Access Control) role hides information in the PCE that is not relevant to the user with that role. At the same time, the App Owners can write effective rules to secure their apps, as well as restrict visibility within the PCE to the permitted scopes for users.

RBAC was previously restricting only the write permission for users while the read permission was unrestricted, and every user had visibility into PCE. The App Owner RBAC role also restricts

the read permission to correspond to the user roles. It accelerates enterprise-wide expansion so that the customers who acquired Illumio for a single application can expand faster

Introduction of the App Owner role solves these problems because it does the following:

- Accelerates micro-segmentation deployment by allowing for scaling after an organization implements micro-segmentation with a smaller set of applications.
- Assures compliance with good security practices so that users cannot view the sensitive information they are not allowed to see.
- Eliminates the complexity of building a custom portal. The App Owners can use Illumio REST APIs instead of the custom UIs created by customers.

App Owners are responsible for managing vulnerabilities in the applications they own and for which the PCE owners can assign scoped roles.

App Owner Roles

Roles of Ruleset Managers, Ruleset Provisioners, and Workload Managers are assigned to users and user groups. They can be expanded with additional to provide the users with additional read/write permissions. All permissions are additive.

Ruleset Manager with Scoped Reads

This RBAC role has the write permission that allows its owner to make changes to the policy. Users with this role can see in the PCE only the content related to their location instead of having full read-only access to the entire PCE content as before.

The role now also supports scoped reads.

Ruleset Provisioner with Scoped Reads

This RBAC role can provision policy changes to workloads. Users with this role can see in the PCE only the content related to their location instead of having full read-only access to the entire PCE content as before.

The role now also supports scoped reads.

Ruleset Viewer

This RBAC role has access to the PCE to manage one or multiple applications. Users with this role can get a view of their application and its dependencies, but they cannot see information about other applications.

Workload Manager with Scoped Reads

This RBAC role provides a control for managing workloads. Users with this role can see in the PCE only the content related to their scope instead of having full read-only access to the entire PCE content as before.

The role now also supports scoped reads.

Chapter 8

Security Policy Objects

This chapter contains the following topics:

The security policy in Illumio represents a configurable set of rules that protects network assets from threats and disruptions and secures communications between workloads.

The PCE contains security objects, such as IP lists, labels, label groups, and services to help you write your security policy. These objects define version, modifications, dependencies, changes, and whether a policy can be reverted.

In the Illumio's label-based system, the rules you write don't require the use of an IP address or subnet, and you can control the range of your policy by using labels. Use label groups to write rules more efficiently if the same labels are used repeatedly in rulesets.

Security Policy Objects

Security policy objects contain information about policy versions, modifications, whether it is still pending, and can be reverted, policy dependencies, and policy changes.

Active vs. Draft

This Public Stable API operates on provisionable objects, which exist in either a `draft` (not provisioned) state or an `active` (provisioned) state.

Provisionable items include label groups, services, rulesets, IP lists, virtual services, firewall settings, SecureConnect gateways, and virtual servers. For these objects, the URL of the API call must include the element called `:pversion`, which can be set to either `draft` or `active`.

Depending on the method, the API follows these rules:

- For GET operations – `:pversion` can be `draft`, `active`, or the ID of the security policy.
- For POST, PUT, DELETE – `:pversion` can be `draft` (you cannot operate on active items) or the ID if the security policy.

Security Principals

Security principals are typically unique identifiers for Windows Advanced Directory groups, but they can also be unique identifiers for individuals. This Public Stable API allows you to get (one or many), create (one or bulk), update, and delete security principals.

An array of security principals HREFs can be passed into rules and rulesets in the `consuming_security_principals` array.

> NOTE:The common schema `consuming_security_principals` has been replaced by two other APIs: `consuming_security_principals_get` and `consuming_security_principals_put`

Security Principals API Methods

Security Principals Methods	HTTP	URI
Get Security Principals	`GET`	`[api_version][org_href]/security_principals/`
Get a Security Principal	`GET`	`[api_version][org_href]/security_principals/sid`
Create a Security Principal	`POST`	`[api_version][org_href]/security_principals/`
Bulk create Security Principals	`PUT`	`[api_version][org_href]/security_principals/bulk_create`

Security Principals Methods	HTTP	URI
Update a Security Principal	PUT	[api_version][org_href]/security_principals/sid
Delete a Security Principal	DELETE	[api_version][org_href]/security_principals/sid

Query Parameters

The only required parameter for all API methods is org_id.

Parameter	Description	Type	Required
org_id	Organization ID	Integer	Yes
max_results	Maximum number of entries to return	Integer	No
name	Name of security principal to filter by	String	GET, PUT: No POST: Yes
sid	SID of security principal to filter by	String	GET: No POST, PUT, DELETE: Yes

Response Properties

Parameter	Description	Type
href	URI of security principal	String
sid	Active Directory SID	String
name	Name of security principal	String
used_by_ruleset	Flag to indicate if this security principal is being used by a ruleset	Boolean
deleted	Flag to indicate if security principal has been deleted	Boolean

Get Security Principals

This GET command, by default, returns information for 100 security principals if max_results is not specified.

A maximum value of up to 500 can be specified for max_results. To return more than 500 security principals, see Async Job Operations.

Curl Command to Get Security Principals

```
curl -X GET https://pce.my-company.com:8443/api/v2/security_principals -u
$KEY:$TOKEN -H 'Accept: application/json'
```

Example JSON Response Body

```
{
 "sid": "string",
 "name": "string",
 "description": "string"
}
```

Get a specified Security Principal

This GET command returns information about one specific security principal indicated by its `sid`.

Curl Command to Get a Security Principal

```
curl -X GET https://pce.my-company.com:8443/api/v2/security_principals/{sid} -u
$KEY:$TOKEN -H 'Accept: application/json'
```

Example JSON Response Body

```
{
  "sid": "string",
  "name": "string",
  "description": "string"
}
```

Create a Security Principal

This POST command on success returns the HREF of the created security principal.

Curl Command to Create a Security Principal

```
curl -X POST https://pce.my-company.com:8443/api/v2/security_principals -u
$KEY:$TOKEN -H 'Content-Type: application/json'
```

Example JSON Request Body

```
{
  "sid": "string",
  "name": "string",
```

```
  "description": "string"
}
```

Bulk Create Security Principals

This PUT command creates multiple security principals.

A maximum of 2,000 security principals can be added in a call to this API. On success, this API returns an array containing the HREFs of the created security principals.

Curl Command to Bulk Create Security Principals

```
curl -X PUT https://pce.my-company.com:8443/api/v2/security_principals/bulk_create
-u $KEY:$TOKEN -H 'Content-Type: application/json'
```

Example JSON Request Body

```
[
  {
    "sid": "string",
    "name": "string",
    "description": "string"
  },
  {
    "sid": "string_2",
    "name": "string_2",
    "description": "string_2"
  }
]
```

Update a Security Principal

This PUT command updates a security principal.

Curl Command to Update a Security Principal

```
curl -X PUT https://pce.my-company.com:8443/api/v2/security_principals/{sid} -u
$KEY:$TOKEN -H 'Content-Type: application/json'
```

Example JSON Request Body

```
{
  "name": "string",
  "description": "string"
}
```

Delete a Security Principal

This command deletes a security principal.

Curl Command to Delete a Security Principal

```
curl -X DELETE https://pce.my-company.com:8443/api/v2/security_principals/{sid} -u
$KEY:$TOKEN
```

This command returns `204 No Content` for success.

Labels

This Public Stable API gets, creates, updates, and deletes labels.

Labels API Methods

Functionality	HTTP	URI
Get a collection of labels	`GET`	`[api_version][org_href]/labels`
Get an individual label	`GET`	`[api_version][label_href]`
Create a label	`POST`	`[api_version][org_href]/labels`
Update a label	`PUT`	`[api_version][label_href]`
Delete a label	`DELETE`	`[api_version][label_href]`

Get Labels

This API returns all labels in an organization or a single label. When you get labels, they are returned in the form of an HREF path property, for example: `"/orgs/2/labels/1662"`

By default, the maximum number returned on a GET collection of labels is 500. To return more than 500 labels, use an Asynchronous GET Collection.

NOTE:
GET returns any label that contains a match, as opposed to an exact match. For example, a GET request for labels with value=APP could return APP, WEB-APP, WEBAPP.

URI to Get Collection of Labels

```
GET [api_version][org_href]/labels
```

URI to Get an Individual Label

```
GET [api_version][label_href]
```

Query Parameters

Parameter	Description	Type	Required
`org_id`	Organization ID	Integer	Yes
`external_data_reference`	A unique identifier within the external data source	String	No
`external_data_set`	The data source from which a resource originates	String	No
`include_deleted`	Include deleted labels	Boolean	No
`key`	Key by which to filter	String	GET: No POST: Yes
`max_results`	Maximum number of labels to return.	Integer	No
`usage`	Indicate label usage, including if the label is currently used in an RBAC scope for user permissions, if the label is applied to a workload, virtual service, Pairing Profile, selective enforcement, virtual server, or ruleset, and if the label belongs to a label group.	Boolean	No
`value`	Value on which to filter. Supports partial matches.	String	GET, PUT: No POST: Yes
`label_id`	Label ID, for `[api_version][label_href]`	Integer	Yes

Response Properties

Property	Description	Type
key	Key in key-value pair	String
value	Value in key-value pair",	String
href	Label URI	
updated_at	Timestamp when this label was last updated	String
created_at	Timestamp when this label was first created	String
external_data_reference	A unique identifier within the external data source	String, Null
external_data_set	The data source from which a resource originates	String, Null

Curl Command to Get Collection of Labels

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/labels -H "Accept: application/json" -u $KEY:$TOKEN
```

Response Body

In the response body, each label returned is identified as an HREF, for example: `"/orgs/2/labels/1662"`

For example:

```
{
        href: "/orgs/2/labels/1662"
        key: "env"
        value: "Prod"
        created_at: "2020-01-22T18:24:33Z"
        updated_at: "2020-01-22T18:24:40Z"
        created_by: {
           href: "/users/9"
        }
        updated_by: {
           href: "/users/9"
        }
    }
    {
        href: "/orgs/2/labels/1128"
        key: "role"
        value: "DB"
        created_at: "2020-01-22T18:24:53Z"
```

```
          updated_at: "2020-01-22T18:24:59Z"
          created_by: {
              href: "/users/9"
          }
          updated_by: {
              href: "/users/9"
          }
     }
```

Curl Command to Get a Label

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/labels/8 -H "Accept:
application/json" -u $KEY:$TOKEN
```

Response Body

```
{
     href: "/orgs/2/labels/8"
     key: "env"
     value: "Prod"
     created_at: "2020-01-22T18:24:33Z"
     updated_at: "2020-01-22T18:24:40Z"
     created_by: {
          href: "/users/9"
     }
     updated_by: {
          href: "/users/9"
     }
}
```

Create a Label

This API creates a new label inside an organization for one of the following label types, for which you can provide your own string value:

- **Application** (“app”): The type of application the workload is supporting. For example, HRM, SAP, Finance, Storefront.
- **Role** (“role”): The function of a workload. In a simple two-tier application consisting of a web server and a database server, there are two roles: Web and Database.

- **Environment** (“env”): The stage in the development of the application. For example, production, QA, development, staging.
- **Location** (“loc”): The location of the workload. For example, Germany, US, Europe, Asia; or Rack #3, Rack #4, Rack #5; or data center, AWS-east1, AWS-east2, and so on.

System Default “All” for Labels

The PCE provides built-in environment, application, and location labels that are defined as "All" that create broad policies to cover all applications, all environments, and all locations.

For this reason, you cannot create labels of these types defined as "All Applications," "All Environments," or "All Locations" (exactly as written in quotes) in order to prevent confusion for policy writers.

If you attempt to create labels of these types with the exact name as the system defaults (for example, "All Applications"), you receive an HTTP "406 Not Acceptable" error.

Illumio recommends not creating labels with names similar to these default system labels to avoid confusion.

URI to Create a Label

```
POST [api_version][org_href]/labels
```

Example Request Body

```
{
   "key":"role",
   "value":"web"
}
```

Curl Command to Create a Label

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/2/labels -H "Content-Type: application/json" -u $KEY:$TOKEN -d '{"key":"role","value":"web"}'
```

Response Body

The created label is in the form of an HREF path property. For example, in the response below, the label is identified as `"/orgs/2/labels/1677"`.

```
{
     href: "/orgs/2/labels/1677"
     key: "role"
     value: "my_web_app"
     created_at: "2014-04-18T19:39:27Z"
     updated_at: "2014-04-18T19:39:27Z"
     created_by: {
         href: "/users/76"
     }
     updated_by: {
         href: "/users/76"
     }
}
```

Update a Label

This API allows you to update a label applied to a workload, given that you have the label HREF, which is returned when you get all labels in an organization. For example: `"/orgs/2/labels/1662"`

URI to Update a Label

```
PUT [api_version][label_href]
```

Example Request Body

To update a label definition, the JSON request body can be constructed as follows:

```
{ "value":"db" }
```

Curl Command to Update a Label

```
curl -X PUT https://pce.my-company.com:8443/api/v2/orgs/2/labels/1662 -H "Accept: application/json" -u $KEY:$TOKEN -d '{"value":"db"}
```

Delete a Label

This API deletes a label from an organization using the label HREF, which is returned when you get a collection of labels in an organization. For example: `"/orgs/2/labels/1662"`

URI to Delete a Label

```
DELETE [api_version][label_href]
```

Curl Command to Delete a Label

```
curl -i -X DELETE https://pce.my-company.com:8443/api/v2/orgs/2/labels/1662 -H
"Accept: application/json" -u $KEY:$TOKEN
```

Label Groups

This Public Stable API helps you write rules more efficiently if the same labels are used repeatedly in rulesets. When you add labels to a label group, the label group can be used in a rule or ruleset scope to represent multiple labels. A label group can also be a member (child) of other label groups.

Label Groups API Methods

Functionality	HTTP	URI
Get a collection of label groups	`GET`	`[api_version][org_href]/sec_policy/draft/label_groups`
Get an individual label group	`GET`	`[api_version][label_group_href]`
Get an individual label group to see if it is a member of other label groups	`GET`	`[api_version][label_group_href]/member_of`
Create a new label group	`POST`	`[api_version][org_href]/sec_policy/draft/label_groups`
Update an individual label group	`PUT`	`[api_version][label_group_href]`
Delete an individual label group	`DELETE`	`[api_version][label_group_href]`

Active vs. Draft

This API operates on provisionable objects, which exist in either a `draft` (not provisioned) state or an `active` (provisioned) state.

Provisionable items include label groups, services, rulesets, IP lists, virtual services, firewall settings, enforcement boundaries, and virtual servers. For these objects, the URL of the API call must include the element called `:pversion`, which can be set to either `draft` or `active`.

Depending on the method, the API follows these rules:

- For GET operations – `:pversion` can be draft, active, or the ID of the security policy.
- For POST, PUT, DELETE – `:pversion` can be draft (you cannot operate on active items) or the ID if the security policy.

Get Collection of Label Groups

This method gets all label groups in your organization. Use this to discover the `label_group_id` to GET a specific label group or for POST, PUT, and DELETE operations.

By default, the maximum number returned on a GET collection of label groups is 500. If you want to get more than 500 label groups, use an Asynchronous GET Collection.

URI to Get a Collection of Label Groups

```
GET [org_href]/sec_policy/draft/label_groups
```

URI to Get an Individual Label

```
GET [label_group_href]
```

Query Parameters

Parameter	Description	Type	Required
`org_id`	Organization	Integer	Yes
`pversion`	Security Policy Version	String	Yes
`label_group_id`	Label Group UUID, for `[api_version][label_group_href]` and `[api_version][label_group_href]/member_of`	String	Yes
`usage`	Include label usage flags	Boolean	No

Response Properties

Property	Description	Type
`href`	URI of this label group	String
`name`	The specific name of a label group to return. Supports partial matches	String
`key`	Key by which to filter	String
`created_at`	Timestamp when this label group was first created	String date/time
`updated_at`	Timestamp when this label group was last updated	String

Property	Description	Type
		date/time
deleted_at	Timestamp when this label group was deleted	String, null
created_by	User who originally created this label group "$ref": "../common/href_object.schema.json"	
updated_by	User who last updated this label group "$ref": "../common/href_object.schema.json"	
deleted_by	User who deleted this label group "$ref": "../common/href_object.schema.json"	Null
blocked_connection_reject_scopes	Label Group is referenced by Blocked Connection Reject Scopes. Replaces the property blocked_connection_reject_scope	Boolean
loopback_interfaces_in_policy_scopes	Label Group is referenced by Loopback Interfaces in Policy Scopes	Boolean
ip_forwarding_enabled_scopes	Label Group is referenced by IP Forwarding Enabled Scopes	Boolean

Curl Command to Get Label Groups

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/3/sec_
policy/draft/label_groups -H "Accept: application/json" -u $KEY:$TOKEN
```

Response

When you get a collection of label groups, each label group is identified by an HREF. You need the HREF to update or delete an individual label group using the API.

```
{
    "href": "/orgs/2/sec_policy/draft/label_groups/4c8e3325-c6dd-4dc2-aadc-
971e9de270e4",
    "created_at": "2020-07-25T00:46:52.552Z",
    "updated_at": "2020-07-25T00:59:00.177Z",
    "deleted_at": null,
    "created_by": {
      "href": "/users/3"
    },
```

```
    "updated_by": {
      "href": "/users/3"
    },
    "deleted_by": null,
    "name": "AppGroup1",
    "description": null,
    "key": "app",
    "labels": [],
    "sub_groups": [
      {
        "href": "/orgs/2/sec_policy/draft/label_groups/9b30081e-e105-44d8-9945-
4c8a30dbe849",
        "name": "AppGroup3"
      }
    ]
  },
```

Label Group Belonging to Other Groups

This method determines if an individual label group is a member of other label groups. For example, if one label group is also a “child” of three other label groups, the response to this call returns the three “parent” label groups to which the specified label group belongs.

URI to Check if a Label Group Belongs to Other Label Groups

```
GET [api_version][label_group_href]/member_of
```

Response

If the specified label group does not belong to any other label groups, the call returns an HTTP 200 message. If the specified label group does belong to other label groups, the response lists the parent label groups. For example:

```
[
    {
        "href": "/orgs/7/sec_policy/draft/label_groups/b51c986b-db35-47d4-ab77-
aae570d1f164",
        "name": "MyLablesUS"
```

```
    }
]
```

Update a Label Group

To update an individual label group, use the HREF of the label group, which is obtained from an API call to get a collection of label groups.

URI to Update a Label Group

```
PUT [label_group_href]
```

Request Body

This example request body updates the labels contained within a label group.

```
{
  "labels": [
    { "href": "/orgs/28/labels/1100" },
    { "href": "/orgs/28/labels/1098" },
    { "href": "/orgs/28/labels/1099" },
    { "href": "/orgs/28/labels/1101" }
  ],
  "sub_groups": []
}
```

Curl Command to Update Label Groups

In this example, the label group being updated with the request body from the code example above is identified by the its label group HREF.

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/2/sec_
policy/draft/label_groups/3307b3d8-2ca2-48f5-877a-03ada95cd6de -H "Content-
Type:application/json" -u $KEY:$TOKEN -d '{"labels":
[{"href":"/orgs/28/labels/1100"},{"href":"/orgs/28/labels/1098"},
{"href":"/orgs/28/labels/1099"},{"href":"/orgs/28/labels/1101"}],"sub_groups":[]}'
```

Delete a label Group

To delete an individual label group, specify the HREF of the label group you want to delete, which is obtained from an API call to get a collection of label groups.

URI to Delete a Label Group

```
DELETE [api_version][label_group_href]
```

Curl Command to Delete a Label Group

```
curl -i -X DELETE https://pce.my-company.com:8443/api/v2/orgs/2/sec_
policy/draft/label_groups/3307b3d8-2ca2-48f5-877a-03ada95cd6de -u $KEY:$TOKEN
```

Services

This Public Stable API gets, creates, updates, or deletes services. To write services they must be in the “draft” state, which means they have not been provisioned. To provision changes made to services, use the Security Policy API.

Services API Methods

Functionality	HTTP	URI
Get a collection of services	GET	[api_version][org_href]/sec_policy/{pversion}/services
Get an individual service	GET	[api_version][org_href]/sec_policy/{pversion}/services/service_id
Create a new service	POST	[api_version][org_href]/sec_policy/draft/services
Update an individual service	PUT	[api_version][org_href]/sec_policy/draft/services/service_id
Delete an individual service	DELETE	[api_version][org_href]/sec_policy/draft/services/service_id

Active vs. Draft

This API operates on provisionable objects, which exist in either a `draft` (not provisioned) state or an `active` (provisioned) state.

Provisionable items include label groups, services, rulesets, IP lists, virtual services, firewall settings, enforcement boundaries, and virtual servers. For these objects, the URL of the API call must include the element called `:pversion`, which can be set to either `draft` or `active`.

Depending on the method, the API follows these rules:

- For GET operations – `:pversion` can be draft, active, or the ID of the security policy.
- For POST, PUT, DELETE – `:pversion` can be draft (you cannot operate on active items) or the ID if the security policy.

Request Parameters

Parameter	Description	Type	Required
`org_id`	Organization	Integer	Yes
`name`	Name of service on which to filter. This parameter supports partial matches.	String	GET: No POST: Yes
`description`	Description of the service on which to filter. This parameter supports partial matches.	String	No
`pversion`	Security Policy Version	String	Yes
`external_data_set`	The data source from which the resource originates. For example, if service information is stored in an external database.	String	No
`external_data_reference`	A unique identifier within the external data source. For example, if service information is stored in an external database.	String	No
`is_ransomware`	Services associated with ransomware.	Boolean	No
`max_results`	The maximum number of results to return using GET. The maximum limit for returned services is 500. **NOTE**: If this parameter is not specified, or a value greater than 500 is specified, a maximum of 500 results are returned. To get more than 500 services, use an Asynchronous GET Collection.	Integer	No
`name`	Name of service on which to filter. This parameter supports partial matches.	String	GET: No POST: Yes
`port`	Specify port or port range to filter results. The range is from -1 to 65535 (0 is not supported).	String	No
`proto`	Protocol to filter on	Integer	GET: No PUT, POST: Yes

Properties

Properties	Description	Type
`href`	URI	String
`name`	Name of service.	String
`description`	Description of the service.	String
risk_details	This property contains the object `ransomware`, which is reqired to define the Ransomware Dashboard. It contains the following properties: • `category`: Categorization based on Admin or Legacy port used in the service • `severity`: Severity of this service • `os_platforms`: Operating system for this ransomware service, an array with "minItems": 1,	Object, NULL
`description_url`	Description URL Read-only to prevent XSS attacks	String
`process_name`	Name of the process.	String
`service_ports`	Reference to `service_ports.schema.json`	
`windows_services`	Reference to `windows_services.schema.json`	
`external_data_set`	External data set identifier	String NULL
`external_data_referenc`	External data reference identifier	String NULL

sec_policy_post

This schema section shows how the property `risk_details` was added to define categorization of services based on the ransomware threat:

```
{
        "$schema": "http://json-schema.org/draft-04/schema#",
        "type": "object",
        "additionalProperties": false,
        "required": [
                "name"
                ],
```

```
        "properties": {
                "name": {
                "description": "Name (does not need to be unique)",
                        "type": "string"
                },
                "description": {
                        "description": "Description",
                        "type": "string"
                },
                "risk_details": {
                "type": "object",
                "properties": {
                        "ransomware": {
                        "type": "object",
                        "properties": {
                        "category": {
                                "description": "Categorization based on Admin or Legacy port used i
service",
                                "type": "string",
                                "enum": [
                                "admin",
                                "legacy"
                        ]
                },
                "severity": {
                        "description": "Severity of this service",
                        "type": "string",
                        "enum": [
                                "low",
                                "medium",
                                "high",
                                "critical"
                        ]
                },
                "os_platforms": {
                        "description": "Operating system for this ransomware service",
                        "type": "array",
                        "minItems": 1,
```

```
                    "items": {
                    "type": "string",
                    "enum": [
                            "windows",
                            "linux"
                            ]
                    }
                }
            }
        }
    ================================================================================
```

Get Services

This API gets all the services in your organization that are in the “draft” policy state (not yet provisioned).

By default, the maximum number returned on a GET collection of services is 500. To get more than 500 services, use an Asynchronous GET Collection.

URI to Get a Collection of Services

```
GET [api_version][org_href]/sec_policy/draft/services
```

URI to Get an Individual Service

```
GET [api_version][service_href]
```

Curl Command to Get All Services

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/sec_
policy/draft/services -H "Accept: application/json" -u $KEY:$TOKEN
```

Curl Example to Get a Service

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/sec_
policy/draft/services/91 -H "Accept: application/json" -u $KEY:$TOKEN
```

Response

Each individual service returned is identified by a service HREF. To GET, PUT, or DELETE an individual service, identify the service using its HREF in the API call.


```
{
    "href": "/orgs/7/sec_policy/active/services/878",
    "created_at": "2017-02-10T18:10:50.324Z",
    "updated_at": "2017-02-10T18:10:50.324Z",
    "deleted_at": null,
    "updated_by": null,
    "deleted_by": null,
    "name": "ICMP ECHO",
    "description": null,
    "description_url": null,
    "process_name": null,
    "service_ports": [
      {
        "icmp_type": 8,
        "icmp_code": null,
        "proto": 1
      },
      {
        "icmp_type": 128,
        "icmp_code": null,
        "proto": 58
      }
    ]
  }
```


Create a Service

This method creates an individual service. Once a service is created, it can be used to write rules for a security policy.

URI to Create a Service

```
POST [api_version][org_href]/sec_policy/draft/services
```

Example Payload

```
{
  "name": "RDP",
  "description": "Windows Remote Desktop",
  "service_ports": [
    {
      "port": 3389,
      "proto": 6
    }
  ]
}
```

Curl Command to Create Windows Service

This example shows how to create a Windows Remote Desktop (RDP) service.

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/2/sec_
policy/active/services -H "Content-Type:application/json" -u $KEY:$TOKEN -d '
{"name":"RDP", "description":"Windows Remote Desktop","service_ports":
[{"port":3389,"proto":6}]}'
```

Update a Service

In order to update (PUT) an individual service, you need to know the HREF of the service you want to update. A service's HREF is returned when you get a collection of services from the PCE.

URI to Update an Individual Service

```
PUT [api_version][service_href]
```

Request Body

This example illustrates the request body you can pass to update a service, for example, to change the port used by the Nginx service from its current port number to 8080:

```
{
   "name": "nginx",
   "service_ports": [
     {
       "port": 8080,
```

```
      "proto": 6
    }
  ]
}
```

Curl Command to Update Service

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/2/sec_
policy/active/services/79 -H "Content-Type:application/json" -u $KEY:$TOKEN -d '
{"name":"nginx","service_ports":[{"port":8080,"proto":6}]}'
```

Delete a Service

To delete an individual service, use the HREF of the service you want to delete, which is returned when you get a collection of services.

URI to Delete an Individual Service

```
DELETE [api_version][service_href]
```

Curl Command to Delete Service

```
curl -i -X DELETE https://pce.my-company.com:8443/api/v2/orgs/2/sec_
policy/active/services/79 -u $KEY:$TOKEN
```

Core Services Detection

This Public Experimental API helps you identify core services and suggests an appropriate label for them. There are 51 services that can be detected.

Core services (such as DNS, Domain Controller, NTP, and LDP) are essential to your computing environment and run on one or on multiple workloads. Identifying and labeling these workloads is important because they are centrally connected, and other applications depend on them.

When you use the core service detection to label and write policies for core services, you can save time on application policies and introduce enforcement faster.

Users have the ability to change port numbers on which a specific core service is running so that they can adjust them to their environment. Users cannot change ports using the UI, only the APIs.

The user authorized to manage core services is the Organization Administrator.

Common schemas for managing core services:

- `core_services_labels.schema.json`
- `core_services_type_ports_def.schema.json`
- `core_services_type_ports.schema.json`

Services API Methods

Functionality	HTTP	URI
Get all detected core services for this organization	`GET`	`[api_version][org_href]/detected_core_services`
Get a detected core service by UUID	`GET`	`[api_version][org_href]/detected_core_services/<uuid>`
Get detected core service summary details	`GET`	`[api_version][org_href]/detected_core_services_summary`
Get all core service types for this organization	`GET`	`[api_version][org_href]/core_service_types`
Get core service type by UUID	`GET`	`[api_version][org_href]/core_service_types/<uuid>`
Accept, reject or skip the core service recommendation.	`PUT`	`[api_version][org_href]/detected_core_services/:uuid`
Edit suggested labels of a core service type for the organization.	`PUT`	`[api_version][org_href]/core_service_types/:uuid`

Query Parameters

Parameter	Description	Type	Required
`org_id`	Organization ID	Integer	Yes
`action`	The action taken on the detected core services	String	No
`core_service_type`	get all detected core services for a particular core service type.	String	No
`max_results`	The maximum results to be returned	Integer	No
`detected_core_service_id`	UUID of the detected core service	String	Yes

Properties

Property	Description	Type
`href`	The href of this detected core service	String

Property	Description	Type
ip_address	The ip address which is detected as core service	String
core_service_type	Get all detected core services of a particular type, such as Splunk/NFS. The href will be given in the query parameter.	String
method_name	The method by which this core service was detected	String
created_at	Date at which core service was detected	date/time
updated_at	Date core service was updated with action information	date/time
confidence	Confidence of the detected core service. "minimum": 50, "maximum": 100"	Integer
feedback	Feedback provided for this core service recommendation, if any. "maxLength": 500	String
action	User can accept, skip or reject the core service determination.	String
labels_applied	Indicates if the end user applied labels for this workload	Boolean
last_detected_at	Date core service was last recommended by core service detection algorithm	date/time
workload	Reference to traffic_flows_workload.schema.json	Object

Parameters for detected_core_services_summary

Parameter	Description	Type	Required
core_service_type	The unique identifier for the core service type. A core service type is defined by a name, port information and PCE recommended labels	String	Yes
recommended	Total number of detected core services which are skipped or no decision has been made yet	Integer	No
accepted	Number of accepted recommendations	Integer	No
rejected	Number of recommendations rejected by the user	Integer	No

Parameters for core_services_types

Parameter	Description	Type	Required
href	The href of this core service type	URI	Yes
name	The name of the core service type	String	Yes
labels	Reference to core_services_labels.schema.json		
created_at	Timestamp at which this core service type was created	String	Yes
updated_at	Timestamp at which this core service type was updated	String	Yes

Parameter	Description	Type	Required
required_ports	Required ports for this core service type, if any Reference to core_services_type_ports.s-chema.json		
optional_ports	Optional ports for this core service type, if any "$ref": "core_services_type_ports.schema.json"		
priority	Each IP/workload is identified for 1 core service type and they are ordered by priority. For PUT: "minimum": 1	Integer	No
num_optional_ports_required	Number of optional ports required For PUT: "maximum": 65535	Integer	No
provider	Indicates whether the provider is a core service. Default value is true, which means provider is a core service	Booolean	No

Sample URLs and Payloads

GET /api/v2/orgs/1/detected_core_services/ ddfe5204-ad29-4bcd-9821-fcb62353a985.

```
{
    "href" :
        "/orgs/1/detected_core_services/ddfe5204-ad29-4bcd-9821-fcb62353a985" ,
       "ip_address" :
       "103.10.11.44" ,
       "workload" : {
            "hostname" :
               "SE555Q5" ,
           "href" :
               "/orgs/2/workloads/e62d71b3-36c4-4c27-926b-411b93ba6d6f" ,
           "labels" : []
       },
       "core_service_type" : {
            "href" :
             "/orgs/1/core_service_type/3555d1e4-fcb2-49c2-9a4a-215c4d5e86dc"
       },
       "confidence" :
               100  ,
```

```
        "method_name" :
                "process_based" ,
        "created_at" :
                "2020-08-04T05:02:46.648Z" ,
        "updated_at" :
                "2020-08-04T05:02:46.648Z" ,
        "last_detected_at" :
                "2020-09-05T05:02:46.648Z"
}
```

`PUT /api/v2/orgs/1/detected_core_services/3ddd5204-ad29-4bcd-9821-fcb62353a98f`

Take the appropriate action for the identified core services, such as accept the recommendation to apply the suggested labels to the workload.

```
Example
        1  :
{ "action" : "accept" }

Example
        2  :
{ "action" : "accept" ,
        "workload" :{ "href" :
        "/orgs/2/workloads/e62d71b3-36c4-4c27-926b-411b93ba6d6f" }} # for the
        case when an IP is converted to UMWL and accepted as core service

Example
        3  :
{ "action" : "reject" }

Example
        4  :
{ "action" : "reject" ,
        "feedback" : "Not a core service." }

Example
        5  :
```

```
{ "action" : "skip" ,
       "feedback" : "Check with Ops if this is a core service." }

Example
       6  :
{ "labels_applied" : true }
```

GET /api/v2/orgs/ :xorg_id /core_service_types/44dd5204-ad29-4bcd-9821-fcb62353a98f

```
{
       "href" : "/orgs/2/core_service_type/44dd5204-ad29-4bcd-9821-fcb62353a98f" ,
       "core_service" : "splunk" ,
       "required_ports" :[{ "port" : 9997 ,
              "to_port" : 10000 }],
       "optional_ports" :[{ "port" : 112 }, { "port" : 455 }],
       "labels" : [
       {
              "value" : "app-splunk" ,
              "key" :
                     "app"
              "href" : "/orgs/1/labels/2"
       },
       {
              "value" : "role-splunk" ,
              "key" :
                     "role" ,
              "href" : "/orgs/1/labels/12"
       } ],
       "created_at" :
              "2020-08-04T05:02:46.648Z" ,
       "updated_at" :
              "2020-08-05T05:02:46.648Z"
}
```

```
PUT /api/v2/orgs/ :xorg_id /core_service_types/44dd5204-ad29-4bcd-9821-fcb62353a98f
```

```
{
"labels" : [
        {
                "href" : "/orgs/1/labels/3"
        },
        {
                "href" : "/orgs/1/labels/10"
        }]
}
```

Non-corporate Public IP Addresses

The API `sec_policy/rule_coverage` supports non-domain interfaces.

Security Policy Rule Coverage

Security Principals Methods	HTTP	URI
Get Security Principals	`POST`	`[api_version][org_href]/sec_policy/rule_coverage`

Query Parameters

The property `network` accepts `network_href` to correctly determine if the rule applies to a flow.

Parameter	Description
`source`	Source entity Specify labels,such as "href": "/orgs/14/labels/42" "href": "/orgs/14/labels/43"
`destination`	Destination entity Specify an IP list, such as "href": "/orgs/14/sec_policy/active/ip_lists/14"
`network`	The network that the source and destination are on
`services`	Port and protocol, and optional process or windows service names, of matching rules

Response Properties

In the release 23.5, in `sec_policy_rule_coverage_post_response` a new array `rule_edges` was added, which provides a list with a placeholder for each requested source and destination pair.

The previous object `rules` is replaced with a reference to `"$ref": "#/definitions/rule_href_mapping"`, and the previous array `edges` is replaced with a reference to `"$ref": "#/definitions/rule_edges"`.

```
"rule_edges": {
       "type": "array",
       "description": "A list with a placeholder for each requested source and
destination pair",
       "items": {
          "type": "array",
          "description": "A list with with a placeholder for each requested service
                          (per source and destination pair)",
       "items": {
          "type": "array",
          "description": "A list of indexes of matching rules
                          (for each service per source and destination pair)",
       "items": {
          "type": "string",
          "pattern": "^[0-9]+$"
          }
     }
}
```

Before the release 23.5, the response was as follows:

Parameter	Description
`rules`	The rules returned in the API response are rules with network_type (such as `non-brn`) that apply to the given input, such as: "0": { "href": "/orgs/14/sec_policy/draft/rule_sets/21/sec_rules/220" }, "1": { "href": "/orgs/14/sec_policy/draft/rule_sets/21/sec_rules/223" }, "2": { "href": "/orgs/14/sec_policy/draft/rule_sets/21/sec_rules/237" }
`edges`	[[["0", "1", "2"]]]

tba

Virtual Services and Service Bindings

This Public Stable API gives you the ability to write rules on a per-service basis instead of having to write rules that apply to all the services running on a workload. By binding a workload to individual services, you can isolate one or more services running on a workload and create policies specific to those services. By binding services, you have the flexibility to create a finely-grained, highly-segmented security policy.

Once you have created, provisioned, and bound a virtual service to a specific workload, you can use the virtual service in rules. See Create an Individual Virtual Service and Create a Service Binding for information.

Virtual Services

Virtual services can consist of either a single service or a collection of explicitly enumerated port/-port range and protocol tuples. They can be used directly in a rule as a single entity, or labels that represent multiple virtual services can be used to write rules.

Virtual services are dynamically bound to workloads using service bindings. Create a virtual service, and then use a service binding to bind the specific virtual service to a workload. Rules written using a virtual service only apply to the workload to which the service is bound.

Use virtual services in the following scenarios:

- **Apply Rules to a Single Service**

  This scenario represents a service or process on a workload using a name or label. You can write a policy that allows other entities to communicate only with that single service. The policy does not need to change if the service is moved to a different workload or a new set of workloads. Only the workload bindings on the virtual service need to be changed. The PCE dynamically calculates the required rules on the updated workloads to allow this virtual service.

- **Applying Rules to one of the many Virtual Services Running on a Workload**

  In this case, multiple virtual services are running on the workload, with different labels, and the rule targets a subset of those services. You can write a rule to allow other entities to communicate only with that specific service. The policy does not need to change if this service is moved to a different workload or a new set of workloads. Only the workload bindings on the virtual service need to be changed. The PCE dynamically calculates the required rules on the updated workloads to allow the virtual service.

Virtual Services API Methods

Functionality	HTTP	URI
Get a collection of virtual services	GET	[api_version][org_href]/sec_policy/:pversion/virtual_services
Get an individual virtual service	GET	[api_version][org_href]/sec_policy/:pversion/virtual_services/virtual_service_id
Create a new virtual service	POST	[api_version][org_href]/sec_policy/draft/virtual_services
Create a collection of virtual services	PUT	[api_version][org_href]/sec_policy/draft/virtual_services/bulk_create
Update a virtual service	PUT	[api_version][org_href]/sec_policy/draft/virtual_services/virtual_service_id
Update a collection of virtual services	PUT	[api_version][org_href]/sec_policy/draft/virtual_services/bulk_update
Delete a virtual service	DELETE	[api_version][org_href]/sec_policy/draft/virtual_services/virtual_service_id

Active vs. Draft Policy Items

Because virtual services are policy items, changes made to them must be provisioned before they can take effect on your policy. Policy items always exist in either a `draft` (not provisioned) or `active` (provisioned) state.

Security policy items that must be provisioned to take effect include IP lists, rulesets, rules, services, virtual services, label groups, user groups, virtual servers, and PCE security settings.

For these items, the URL of the API call must include the URI element called `:pversion`, which can be set to either `draft` or `active` when you make the API call.

Depending on the method, the API follows these rules:

- For GET operations – `:pversion` can be `draft` or `active`
- For POST, PUT, DELETE – `:pversion` can only be `draft` (you cannot operate on provisioned items)

Query Parameters

Property	Description	Type	Required
org_id	Organization ID	Integer	Yes
pversion	Security Policy Version	String	Yes
external_data_	A unique identifier within the external data source.	String,	

Property	Description	Type	Required
reference	For example, if this virtual service information is stored in an external database.	NULL for PUT only	
external_data_set	The data source from which the resource originates. For example, if this virtual service information is stored in an external database.	String	
name	Name on which to filter. Supports partial matches	String	No
labels	List of lists of label URIs, encoded as a JSON string	String	No
virtual_service_id	Virtual Service ID	String	Yes
service	Service URI	String	No
service_ports.port	Specify port or port range to filter results. The range is from -1 to 65535.	String	No
service_ports.proto	Protocol to filter on.	Integer	No
service_address.fqdn	FQDN configured under service_address property, supports partial matches	String	No
service_address.ip	IP address configured under service_address property, supports partial matches	String	No
usage	Include Virtual Service usage flags	Boolean	No

Virtual Services Properties

Property	Description	Type
href	URI of the virtual service	String
created_at	Timestamp when this virtual service was first created	String date/time
updated_at	Timestamp when this virtual service was last updated	String date/time
deleted_at	Timestamp when this virtual service was deleted	String/NULL date/time
name	Name (does not need to be unique)	String
labels	Virtual service labels References common/label_optional_key_value.schema.json	
update_type	Update type for the virtual service	

Property	Description	Type
	Reference to `common/sec_policy_update_type.s-chema.json`	
`external_data_set`	The data source from which the resource originates. For example, if this virtual service information is stored in an external database.	String, NULL for PUT only
`external_data_ref-erence`	A unique identifier within the external data source. For example, if this virtual service information is stored in an external database.	String, NULL for PUT only
`service_addresses`	Reference to `virtual_service_service_addresses.s-chema.json`	
`ip_overrides`	Array of IPs or CIDRs as IP overrides	

Get Collection of Virtual Services

Use this method to get a collection of Virtual Services.

URI to Get a Collection of Virtual Services

```
GET [api_version][org_href]/sec_policy/:pversion/virtual_services
```

Curl Command

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/7/sec_
policy/active/virtual_services -H "Accept: application/json" -u $KEY:$TOKEN
```

Response

Each individual virtual service returned is identified by the virtual service HREF. To GET, PUT, or DELETE an individual virtual service, identify the service by its HREF in the API call.

```
[
  {
    "href": "/orgs/7/sec_policy/draft/virtual_services/1828d8ff-aeb7-4735-9975-
db692813d193",
    "created_at": "2017-10-29T19:41:15.648Z",
    "updated_at": "2017-10-29T19:41:15.648Z",
    "deleted_at": null,
    "created_by": {"href": "/users/14"},
    "updated_by": {"href": "/users/14"},
```

```
    "deleted_by": null,
    "update_type": null,
    "name": "Jawoo",
    "description": null,
    "service": { "href": "/orgs/7/sec_policy/draft/services/99" },
    "labels": [
      { "href": "/orgs/7/labels/88" },
      { "href": "/orgs/7/labels/82" },
      { "href": "/orgs/7/labels/92" },
      { "href": "/orgs/7/labels/101" }
    ],
    "ip_overrides": [
      "192.0.1.0",
      "192.168.100.0/24"
    ],
    "apply_to": "host_only"
  }
]
```

Get an Individual Virtual Service

Use this method to get an individual virtual service. In the call, you identify the virtual service by its HREF, which can be obtained when you get a collection of virtual services.

Use the following query parameters to restrict the results of the query:

URI to Get an Individual Virtual Service

```
GET [api_version][virtual_service_href]
```

> NOTE:
> For this method, you can get specify either `draft` or `active` for `:pversion`.

Curl Command

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/sec_
policy/draft/virtual_services/89 -H "Accept: application/json" -u $KEY:$TOKEN
```

Response

```
{
    "href": "/orgs/2/sec_policy/draft/virtual_services/6005a35a-1598-4c7b-a827-
be4390f46773",
    "created_at": "2017-12-11T20:56:28.629Z",
    "updated_at": "2017-12-11T21:07:10.407Z",
    "deleted_at": null,
    "created_by": { "href": "/users/9" },
    "updated_by": { "href": "/users/9" },
    "deleted_by": null,
    "update_type": "create",
    "name": "Docker1",
    "description": null,
    "service": { "href": "/orgs/2/sec_policy/draft/services/5" },
    "labels": [
      { "href": "/orgs/2/labels/18" },
      { "href": "/orgs/2/labels/26" },
      { "href": "/orgs/2/labels/126" }
    ],
    "ip_overrides": [
      "192.0.1.0",
      "192.168.100.0/24"
    ],
    "apply_to": "internal_bridge_network"
  }
```

Create an Individual Virtual Service

Use this method to create an individual virtual service. Because a virtual service is a policy item, you must create it in the draft state, and then provision the change using the Security Policy API.

Once the virtual service is provisioned, you can use the service binding method to bind the virtual service to a workload.

URI to Create an Individual Virtual Service

```
POST [api_version][org_href]/sec_policy/draft/virtual_services
```

Request Body

To create a virtual service, you need the HREF of the service you want to “bind” to a workload. You can obtain a service HREF by calling a GET collection with the service binding API.

Additionally, if you want to add labels to the virtual service, you need the HREF of each label you want to add. Label HREFs can be obtained by calling a GET collection with the labels API. Labels are represented in the JSON request body as an array, opened and closed by square brackets ([ ]).

```
{
  "name": "MyVirtualService",
  "description": "Test",
  "service": { "href": "/orgs/7/sec_policy/draft/services/218" },
  "labels": [
    { "href": "/orgs/7/labels/88" },
    { "href": "/orgs/7/labels/82" },
    { "href": "/orgs/7/labels/92" },
    { "href": "/orgs/7/labels/95" }
  ]
}
```

Curl Command

To create a new virtual service:

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/2/virtual_services -H
"Content-Type: application/json" -u $KEY:$TOKEN.-d '{ "name": "MyVirtualService",
"description": "Test", "service": {"href": "/orgs/7/sec_
policy/draft/services/218"}, "labels": [{"href": "/orgs/7/labels/88"}, {"href":
"/orgs/7/labels/82"}, {"href": "/orgs/7/labels/92"}, {"href": "/orgs/7/labels/95"
}]}'
```

Create or Update Virtual Services Collection

> NOTE:
> Bulk operations are rate limited to 1,000 items per operation.

This method enables you to create a collection of virtual services in your organization using a single API call instead of creating individual services one at a time.

This capability is useful if you want to keep a set of PCE resources in sync with your internal representation of the resources, such as a configuration management database (CMDB) that holds the “source of truth” for your PCE resources.

After virtual services are created and the identifiers added to the service properties, you can get a collection of virtual services using query parameters that include the external data reference. You can also run an asynchronous query to get all virtual services through an offline job, which includes the external data references in the response.

The two properties you can use when creating virtual services, `external_data_set` and `external_data_reference` are UTF-8 strings with a maximum length of 255 characters each. The contents must form a unique composite key, meaning that both values of these properties are treated as a unique key. These two properties together are recognized as a unique key, even if one of them is left blank or set to zero.

URI to Create a Collection of Virtual Services

```
PUT [api_version][org_href]/sec_policy/draft/virtual_services/bulk_create
```

URI to Update a Collection of Virtual Services

```
PUT [api_version][org_href]/sec_policy/draft/virtual_services/bulk_update
```

Request Body

To create a collection of virtual services, pass a JSON object that describes the virtual service details. The request body and curl command for this method follow the same structure used to create an individual virtual service, only you add multiple virtual service JSON objects instead of just one.

Additionally, the `href` field must be present in the body for each virtual service that you are updating in the `bulk_update`.

Update a Individual Virtual Service

To update (PUT) an individual virtual service, you need to know the HREF of the virtual service you want to update. Virtual service HREFs are returned when you get a collection of virtual services.

URI to Update an Individual Virtual Service

```
PUT [api_version][org_href]/sec_policy/draft/virtual_services/virtual_service_id
```

Request Properties

The request properties for updating a virtual service are the same as those for creating a virtual service.

Request Body

This example request body can be passed to update a virtual service to include a workload binding:

```
{
  "service": { "href": "/orgs/2/sec_policy/draft/services/91" },
  "labels": [
    { "href": "/orgs/2/labels/316" },
    { "href": "/orgs/2/labels/101" },
    { "href": "/orgs/2/labels/102" },
    { "href": "/orgs/2/labels/103" }
  ]
}
```

Curl Command

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/2/sec_
policy/draft/virtual_services/256525b6-e7c5-4ad7-b7af-e70586aa1078 -H "Content-
Type: application/json" -u $KEY:$TOKEN -d '
{"name":"test","description":null,"service":
{"href":"/orgs/2/labels/316"},"labels": [{"href":"/orgs/2/labels/101"},
{"href":"/orgs/2/labels/102"}, {"href":"/orgs/2/labels/103"}]}'
```

Virtual Service Bindings

After you create a virtual service and provision it, use the service binding API to bind the virtual service to a workload. When you apply your policy to a virtual service, the virtual service must be bound to a workload where that service is running. You can only specify one workload and one virtual service per service binding.

When you bind a virtual service to a workload with a service binding, you must specify the workload to which you want to bind the service. You can also optionally specify any port overrides if you want the virtual service to communicate over a different port than the default.

Unlike virtual services, the service binding API does not require provisioning to take effect.

NOTE:
Updating service bindings doesn't use a PUT method. To update it, delete it, and then POST a new service binding to replace it.

Service Binding API Methods

Functionality	HTTP	URI
Get a collection of service bindings	GET	[api_version][org_href]/service_bindings
Get an individual service binding	GET	[api_version][service_binding_href]
Create a service binding	POST	[api_version][org_href]/service_bindings
Delete an individual service binding	DELETE	[api_version][service_binding_href]

Service Bindings Query Parameters

Parameter	Description	Type	Required
org_id	Organization ID	Integer	Yes
virtual_service	Virtual service href	String	No
service_binding_id	Service Binding ID	String	Yes
workload	The complete HREF of the workload referenced in the service binding.	String	No
external_data_reference	A unique identifier within the external data source. For example, if this virtual service information is stored in an external database.	String	No
external_data_set	The data source from which the resource originates. For example, if this virtual service information is stored in an external database.	String	No

Service Bindings Properties

Property	Description	Type
virtual_service	Virtual service href, required for POST	Object
bound_seviuce	Bound service href, required for GET	Object
workload	The complete HREF of the workload referenced in the service binding.	String

Property	Description	Type
port_overrides	Reference to port_overrides.schema.json	
external_data_reference	A unique identifier within the external data source. For example, if this virtual service information is stored in an external database.	String, NULL
external_data_set	The data source from which the resource originates. For example, if this virtual service information is stored in an external database.	String, NULL
workload	HREF of the workload	String

Create a Service Binding

This method creates one or more service bindings, which associate (or "bind") a virtual service to a workload. When you call this method, you specify the virtual service and workload you want to bind, plus you can optionally specify port overrides to use a different port for the service.

The JSON request body for creating a service binding is an array, which allows you to create multiple service bindings with a single POST.

Before you create a service binding, make sure that the virtual service you want to bind to a workload has been published and is in the active policy state.

URI to Create a Service Binding

```
POST [api_version][org_href]/service_bindings
```

Request Parameters

The request body for creating a service binding is an array of service binding objects. Because this JSON request body is an array, you can create multiple service bindings in a single POST.

> NOTE:
> Make sure that the virtual service you are binding to a workload has been provisioned.

This is an example JSON representation of a single service binding:

```
[{"workload": {"href": "/orgs/1/workloads/45c69cf3-4cbb-4c96-81ee-70e94baea1b8"},
"virtual_service": {"href": "/orgs/1/sec_policy/draft/virtual_services/a735332e-
```

```
5d31-4899-a3a5-fac7055e05c0"}, "port_overrides": [{"port": 14000, "protocol": 6,
"new_port": 26000 }]}]
```

Curl Command

To create a single service binding:

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/2/service_bindings -H
"Content-Type:application/json" -u $KEY:$TOKEN -d '[{"workload":
{"href":"/orgs/1/workloads/45c69cf3-4cbb-4c96-81ee-70e94baea1b8"}, "virtual_
service":{"href":"/orgs/1/sec_policy/draft/virtual_services/a735332e-5d31-4899-
a3a5-fac7055e05c0"}, "port_overrides":[{"port":14000,"protocol":6,"new_
port":26000}]}]'
```

Request Body to Create Multiple Service Bindings

An example JSON request body for creating multiple service bindings with a different port number:

```
[{"workload": {"href": "/orgs/1/workloads/820efcdc-c906-46b9-9729-26bab7a53223"},
"virtual_service": {"href": "/orgs/1/sec_policy/draft/virtual_services/e38ce044-
d2ac-4d7f-aeec-16ef8fbd0b15"}, "port_overrides": [ {"port": 10000, "protocol": 6,
"new_port": 26000 } ]}, {"workload": {"href": "/orgs/1/workloads/820efcdc-c906-
46b9-9729-26bab7a53223"}, "virtual_service": {"href": "/orgs/1/sec_
policy/draft/virtual_services/e38ce044-d2ac-4d7f-aeec-16ef8fbd0b15"}, "port_
overrides": [ {"port": 11000, "protocol": 6, "new_port": 25000} ]}]
```

Service Binding Request Body

If you create more than one service binding with a single POST, all of the service bindings must be constructed properly or the POST will fail and no service bindings will be created.

> NOTE:
> The response of “failure” indicates the error, but it does not confirm that no service bindings have been created.

For example, if you use POST to create 10 service bindings, and one of the workloads referenced in the JSON payload uses an incorrect URI (HREF), the POST fails with an error message similar to the following message:

```
[ { "token": "invalid_uri", "message": "Invalid URI:
{/orgs/1/workloadzzz/820efcdc-c906-46b9-9729-26bab7a53223}" } ]
```

Get Individual or Collection of Service Bindings

You can use these methods to get one or more service bindings.

URI to Get a Collection of Service Bindings

```
GET [api_version][org_href]/service_bindings
```

URI to Get an Individual Service Binding

```
GET [api_version][service_binding_href]
```

Response Body

```
[
   {
     "href": "/orgs/7/service_bindings/287568ad-4a1f-4000-a9fb-e67d1dabce15",
     "virtual_service": {"href": "/orgs/7/sec_policy/active/virtual_
services/256525b6-e7c5-4ad7-b7af-e70586aa1078"},
     "workload": {"href": "/orgs/7/workloads/baef2547-2036-4e00-b6f7-
3f4be1f7669a",
     "name": null,
     "hostname": "AssetMgt-proc2",
     "deleted": false },
     "port_overrides": [{"new_port": 8080,"protocol": 6,"port": 3306}]
     },
        {
           "href": "/orgs/7/service_bindings/faebe7bf-0bb7-49a5-868e-
8297e038fa9e",
           "virtual_service": {"href": "/orgs/7/sec_policy/active/virtual_
services/7b46fce0-4933-4e29-b86c-7a2a71e686ed"},
           "workload": {"href": "/orgs/7/workloads/aee4381b-9836-45b6-b7ab-
aee246bf482f",
           "name": null,
           "hostname": "onlinestore-web2",
           "deleted": false },
```

```
            "port_overrides": []
          },
          {
            "href": "/orgs/7/service_bindings/924ad8c2-94bf-40f5-bc4c-
13474982bd00",
            "virtual_service": {"href": "/orgs/7/sec_policy/active/virtual_
services/256525b6-e7c5-4ad7-b7af-e70586aa1078"},
            "workload": {"href": "/orgs/7/workloads/69fd736b-cd21-4a4c-bdb9-
132207c760ce",
            "name": null,
            "hostname": "test-us",
            ": false },
            "port_overrides": []
          }
]
```

Curl Command to Get an Individual Service Binding

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/service_
bindings/xxxxxxxx-4a86-4dd4-b303-23f699d0ebbf -H "Accept: application/json" -u
$KEY:$TOKEN
```

Curl Command to Get Service Binding Collection

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/service_bindings -H
"Accept: application/json" -u $KEY:$TOKEN
```

Delete an Individual Service Binding

To delete both the service bindings and virtual services, delete the service bindings first, then delete the virtual services.

URI to Delete an Individual Service Binding

```
DELETE [api_version][service_binding_href]
```

Curl Command to Delete a Service Binding

Use this curl command to delete the service binding:

```
curl -i -X DELETE https://pce.my-company.com:8443/api/v2/orgs/2/service_
bindings/xxxxxxxx-4a86-4dd4-b303-23f699d0ebbf -u $KEY:$TOKEN
```

Virtual Servers

A virtual server is similar to a workload. It can be assigned labels and has IP addresses, but does not report traffic to the Illumio Core. Each virtual server has only one VIP. The local IP addresses are used as a source IP address for connections to the pool members (backend servers) when the virtual server is operating in SNAT mode or Auto mode. These IP addresses are likely to be shared by multiple virtual servers on the server load balancer.

A discovered virtual server is a server load balancer (SLB) virtual server (IP address and port(s)) that the NEN has discovered when interrogating SLBs managed by the PCE.

For the topic overview and more details see the Security Policy Guide, Load Balancers and Virtual Servers.

Virtual Server Methods

There are two groups of methods used to manage virtual servers:

- Methods for virtual servers
- Methods for discovered virtual servers

Virtual Servers

Virtual Server Methods

Functionality	HTTP	URI
Get a list of Virtual Servers	GET	[api_version][org_href]/sec_policy/:version/virtual_servers
Get a specified Virtual Server	GET	[api_version][org_href]/sec_policy/:version/virtual_servers/:uuid
Create a Virtual Server object	POST	[api_version][org_href]/sec_policy/:version/virtual_servers
Modify the enforcement mode, labels, and backend/provider labels of a specified Virtual Server	PUT	[api_version][org_href]/sec_policy/:version/virtual_servers/:uuid

Query Parameters for Virtual Servers

Parameter	Description	Type	Required
org_id	Organization ID	Integer	Yes
pversion	Security Policy Version	String	Yes
discovered_virtual_server	URI of discovered virtual server to filter byl	String	No
active_pce_fqdn	FQDN of the PCE	String	No
external_data_reference	A unique identifier within the external data source	String	No
external_data_set	The data source from which a resource originates	String	NO
labels	2D array of label URIs, encoded as a JSON string. Filter by virtual server labels.	String	No
max_results	Maximum number of discovered virtual servers to return	Integer	No
mode	Mode of the virtual server(s) to return	String	No
name	Name of virtual server(s) to return. Supports partial matches	String	No
network_enforcement_node.hostname	Hostname of NEN object to filter virtual server(s)v	String	No
slb	URI of SLB object to filter virtual server(s)	String	No
vip	Frontend (VIP) address of the virtual server(s). Supports suffix-wildcard matches	String	No
vip_port	Port of frontend VIP of the virtual server(s)	Integer	No
vip_proto	Protocol of frontend VIP of the virtual server(s)	Integer	No

Properties for Virtual Servers :

Property	Description	Type	Required
href	href of virtual server	String	Yes
name	The short friendly name of the virtual server	String	Yes
labels	2D array of label URIs, encoded as a JSON string. Filter by virtual server labels. "$ref": "../common/label_optional_key_value.schema.json"	Array	Yes

Property	Description	Type	Required
service	URI of associated service "$ref": "../common/href_object.schema.json"		Yes
providers	minItems: 0, label: "$ref": "../common/label_optional_key_value.schema.json" workload: "$ref": "../common/href_object.schema.json"	Array	Yes
mode	Management mode of the virtual server	String	Yes
discovered_virtual_server	Corresponding discovered virtual server, server URI	String/Null	Yes
update_type	Reference to common/sec_policy_update_type.schema.json		Yes
created_at	The time (rfc3339 timestamp) at which this virtual server was created	String	Yes
updated_at	The time (rfc3339 timestamp) at which this virtual server was last updated	String	Yes
deleted_at	The time (rfc3339 timestamp) at which this virtual server was deleted	String/Null	Yes
created_by	The URI of the user who created this virtual server Reference to common/href_object.schema.json		Yes
updated_by	The URI of the user who last updated this virtual server Reference to common/href_object.schema.json		Yes
deleted_by	The URI of the user who deleted this virtual server Reference to common/nullable_href_object.schema.json		Yes

Discovered Virtual Servers

Discovered Virtual Servers Methods

You can use only three GET methods for discovered virtual servers

Functionality	HTTP	URI
Get a list of Discovered Virtual Servers	GET	[api_version][org_href]/discovered_virtual_servers
Get a specified Discovered Vir-	GET	[api_version][org_href]/discovered_virtual_

Functionality	HTTP	URI
tual Server		`servers/:uuid`
Discovery on-demand: list the discovered virtual servers	`GET`	`[api_version][org_href]/network_enforcement_nodes/virtual_server_discovery_jobs/:uuid`

Discovered Virtual Servers Parameters

Parameter	Description	Type	Required
`org_id`	Organization ID	Integer	Yes
`active_pce_fqdn`	FQDN of the PCE	String	No
`has_virtual_server`	Filter discovered virtual server(s) by whether they are managed by a virtual server object	Boolean	No
`max_results`	Maximum number of discovered virtual servers to return	Integer	No
`name`	Name of discovered virtual server(s) to return. Supports partial matches	String	No
`network_enforcement_node`	Hostname of NEN object to filter discovered virtual server(s)	String	No
`slb`	URI of SLB object to filter discovered virtual server(s)	String	No
`vip`	Frontend (VIP) address of the discovered virtual server (s). Supports suffix-wildcard matches	String	No
`vip_port`	Port of frontend VIP of the discovered virtual server(s)	Integer	No
`vip_proto`	Protocol of frontend VIP of the discovered virtual server (s)	Integer	No
`virtual_server`	URI of virtual server to filter discovered virtual server(s)	String	No
`virtual_server_labels`	2D array of label URIs, encoded as a JSON string. Filter by virtual server labels. Requires usage of has_virtual_server: true	String	No
`virtual_server_mode`	Filter discovered virtual server(s) by virtual server mode. Requires usage of has_virtual_server: true	String	No

Discovered Virtual Server Properties

Property	Description	Type	Required
`href`	href of discovered virtual server	String	Yes
`dvs_iden-tifier`	NFC-generated unique identifier for discovered virtual server	String	Yes
`name`	Configured name of virtual server	String	Yes
`vip_port`	VIP including protocol and port for discovered virtual server. Reference to `common/dvs_vip_port.schema.json`		Yes
`local_ips`	Local IPs of virtual server Format: ipv4	Array	Yes
`mode`	Virtual server mode of operation	String	Yes
`slb`	URI of Service Load Balancer (SLB) object to filter discovered virtual server(s) Reference to `common/href_object.schema.json`		Yes
`service_ checks`	Service checks, which has these prperies: • `protocol` • `port`	Object	Yes
`nfc`	DEPRECATED AND REPLACED (USE 'network_ enforcement_node' INSTEAD) URI of the NFC for this discovered virtual server Reference to `common/href_object.schema.json`		Yes
`created_at`	The time (rfc3339 timestamp) at which this server load balancer was created	date/time	Yes
`updated_at`	The time (rfc3339 timestamp) at which this server load balancer was last updated	date/time	Yes
`created_by`	Reference to `common/href_object.schema.json`		Yes
`updated_by`	Reference to `common/href_object.schema.json`		Yes
`network_ enforcement_ node`	URI of the Network Enforcement Node for this discovered virtual server Reference to `common/href_object.schema.json`		Yes

IP Lists

This Public Stable API can get, create, update, and delete IP lists.

IP lists can be used in rules to define sets of trusted IP address, IP address ranges, or CIDR blocks allowed into your datacenter that are allowed to access workloads in your network.

IP Lists API

Functionality	HTTP	URI
Get a collection of IP lists	`GET`	`[api_version][org_href]/sec_policy/draft/ip_lists`
Get an individual IP list	`GET`	`[api_version][ip_list_href]`
Create an IP list	`POST`	`[api_version][org_href]/sec_policy/draft/ip_lists`
Update an IP list	`PUT`	`[api_version][ip_list_href]`
Delete an IP list	`DELETE`	`[api_version][ip_list_href]`

Active vs Draft

This API operates on provisionable objects, which exist in either a `draft` (not provisioned) state or an `active` (provisioned) state.

Provisionable items include label groups, services, rulesets, IP lists, virtual services, firewall settings, enforcement boundaries, and virtual servers. For these objects, the URL of the API call must include the element called `:pversion`, which can be set to either `draft` or `active`.

Depending on the method, the API follows these rules:

- For GET operations – `:pversion` can be draft, active, or the ID of the security policy.
- For POST, PUT, DELETE – `:pversion` can be draft (you cannot operate on active items) or the ID if the security policy.

Get IP Lists

This API allows you to get a collection of IP lists or a single IP list from an organization.

By default, the maximum number returned on a GET collection of IP lists is 500. If you want to get more than 500 IP lists, use an Asynchronous GET Collection.

URI to Get Collection of IP Lists

```
GET [api_version][org_href]/sec_policy/draft/ip_lists
```

URI to Get an Individual IP List

```
GET [api_version][ip_list_href]
```

Query Parameters

Parameter	Description	Type	Required
`org_id`	Organization	Integer	Yes
`pversion`	Security Policy Version	String	Yes
`description`	Description of IP list(s) to return. Supports partial matches	String	No
`external_data_set`	The data source from which the resource originates. For example, if this workload information is stored in an external database.	String	No
`external_data_reference`	A unique identifier within the external data source. For example, if this workload information is stored in an external database.	String	No
`ip_address`	IP address matching the IP lists to return. Supports partial matches.	String	No
`fqdn`	IP lists matching FQDN. Supports partial matches	String	No
`max_results`	The maximum number of results you want to return when using the GET method. The maximum limit for returned IP lists is 500.	Integer	No
`name`	Name of IP list(s) to return. Supports partial matches	String	No
`ip_list_id`	IP list ID (for `[api_version][ip_list_href]`	String	Yes

Properties

Property	Description	Type	Required
`href`	URI of the ip list	String	Yes
`name`	Name of the IP lists to return, which has to be unique..	String	Yes
`description`	Description of IP list(s) to return. Supports partial matches	String	No
`external_data_set`	The data source from which the resource originates.	String, Null	No

Property	Description	Type	Required
	For example, if this workload information is stored in an external database.		
`external_data_reference`	A unique identifier within the external data source. For example, if this workload information is stored in an external database.	String, Null	No
`fqdns`	Collection of FQDNs.	Array. Required: `fqdn`	No
`ip_list_id`	IP list ID (for `[api_version][ip_list_href]`	String	Yes

Curl Command to Get Collection of IP Lists

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/3/sec_policy/draft/ip_lists -H "Accept: application/json" -u $KEY:$TOKEN
```

Response Body

```
{
     {
        href: "/orgs/2/sec_policy/draft/ip_lists/309"
        id: 309
        created_at: "2020-04-17T21:59:44Z"
        updated_at: "2020-04-17T21:59:44Z"
        deleted_at: null
        created_by: {
            href: "/users/76"
     }
        updated_by: {
            href: "/users/76"
     }
        deleted_by: null
        name: "Good IPs 2"
        description: null
        ip_ranges: [
            {
```

```
            description: "My good IPs for web app"
            from_ip: "192.0.2.0"
            to_ip: null
        }
    ]
}
```

Curl Command to Get an IP List

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/3/sec_policy/draft/ip_
lists/312 -H "Accept: application/json" -u $KEY:$TOKEN
```

Create an IP List

This API allows you to create IP lists (allowlists) so they can be used for creating rules in rulesets. An IP list can contain a single IP address or an IP address range.

> NOTE:
> Denylist IP lists are not supported in this release.

> WARNING:
> Please be aware of the following:
> 0.0.0.0/0 means 0-255 . 0-255 . 0-255 . 0-255 or all possible IP addresses.
> 0.0.0.0 without the trailing "/0", means a single IP (not ANY IP). This is a very rare but sometimes needed object, specifically for things like DHCP Discovery.
> 0.0.0.0 when used improperly might trigger an error, prevent the list from being accepted, and consequently block traffic.
> Use the correct syntax for the intended purpose.

URI to Create an IP List

```
POST [api_version][org_href]/sec_policy/draft/ip_lists
```

Request Properties

Example JSON request body for a single IP list:

```
{
     "name": "Good IPs",
```

```
    "ip_ranges": [
          {
              "description": "Good IPs allowed to access app server",
              "from_ip": "192.0.2.0"
          }
    ]
}
```

Curl Command to Create IP List

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/3/sec_policy/draft/ip_lists -H "Accept: application/json" -u $KEY:$TOKEN -d '{"name": "Good IPs", "ip_ranges":[{"description": "Good IPs allowed to access app server", "from_ip": "192.0.2.0"}]}'
```

Response Body

```
{
     href: "/orgs/2/sec_policy/draft/ip_lists/316"
     created_at: "2020-04-18T00:19:55Z"
     updated_at: "2020-04-18T00:19:55Z"
     deleted_at: null
     created_by: {
          href: "/users/11"
     }
     updated_by: {
          href: "/users/11"
     }
     deleted_by: null
     name: "Good  IPs"
     description: null
     ip_ranges: [
         {
             description: "Good IPs"
             from_ip: "192.0.2.0"
             to_ip: null
         }
     ]
}
```

Update an IP List

This API updates a specific IP list identified by its HREF. Get a collection of IP lists to find IP list HREFs .

Example IP list HREF:

```
/orgs/2/sec_policy/draft/ip_lists/316
```

URI to Update an IP List

```
PUT [api_version][ip_list_href]
```

Example Request Body to Update an IP List

```
{
     "name": "Better IPs",
     "list_type": "allow",
     "ip_ranges": [
         {
              "description": "More allowed IPs for web app",
              "from_ip" : "192.0.2.0"
              "to_ip" : "24"
         }
     ]
}
```

Curl Command to Update IP List

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/3/sec_policy/draft/ip_
lists/312 -H "Content-Type: application/json" -u $KEY:$TOKEN -d '{ "name": "Better
IPs", "list_type": "allow", "ip_ranges": [{"description": "Better IPs for web
app", "from_ip": "192.0.2.0", "to_ip": "24"}]}'
```

Delete an IP List

This API removes an IP list from a organization:

URI to Delete an API List

```
DELETE [api_version][ip_list_href]
```

Curl Command to Delete IP List

```
curl -i -X DELETE https://pce.my-company.com:8443/api/v2/orgs/2/sec_
policy/draft/ip_lists/316 -u $KEY:$TOKEN
```

Chapter 9

Visualization

This chapter contains the following topics:

In addition to reviewing workloads and traffic with the PCE web console, you can analyze the traffic flows and get insight into the exposure to vulnerabilities using the Visualization API.

The Explorer API is used to search and analyze PCE traffic flows. It queries the PCE's traffic database and analyzes these flows for auditing, reporting, and troubleshooting. The VEN adds the DNS names to the flow summary logs and sends them to the PCE, while the Explorer API appends the DNS names to allow auditors and analysts to view them without performing reverse look-ups on random IP addresses.

Vulnerability Maps combine Illumio's Application Dependency Map with vulnerability data from Qualys Cloud Platform to provide insights into the exposure of vulnerabilities and attack paths across your applications.

Explorer

The Public Experimental Explorer APIs search and analyze PCE traffic flows for auditing, reporting, and troubleshooting. You can search for traffic flows between workloads or hosts, labeled workloads, or IP addresses, and you can restrict the search by specific port numbers and protocols.

There are three APIs for the traffic flows search:

- Traffic Analysis Queries
- Asynchronous Queries for Traffic Flows
- Filter for Managed Services
- Database Metrics

Traffic Analysis Queries

This was the basic traffic analyzer for queries that is now deprecated.

Functionality	HTTP	URI
Search the PCE traffic data database to discover traffic patterns and write policy.	`POST`	`[api_version][org_href]/traffic_flows/traffic_analysis_queries`

The maximum of returned results when using `POST [api_version][org_href]/traffic_flows/traffic_analysis_queries` was 100,000, which is a reasonable number a user can view in the UI. However, when Explorer is used for capturing all traffic flows into a CSV file to build rules offline, the queries take longer to return, traffic data contains more than 100,000 rows, and so on. Explorer queries are required to support both the single-node and multi-node Explorer in the SuperCluster environment. Therefore, limitation of 100,000 results was raised to 200,000 to better support SuperCluster environments in Explorer.

> NOTE:This API is now DEPRECATED and replaced with `Asynchronous Queries for Traffic Flows`, where the `max-results` limit is raised from 100,000 to 200,000.

Asynchronous Queries for Traffic Flows

Async Queries API Methods

Functionality	HTTP	URI
Create a new async traffic query	`POST`	`[api_version][org_href]traffic_flows/async_queries`
Get a collection of async traffic queries	`GET`	`[api_version][org_href]traffic_flows/async_queries`
Download the completed async traffic query results	`GET`	`[api_version][org_href]traffic_flows_async/queries/:uuid/dowload`
Update an async traffic query (request cancellation of the queued async query)	`PUT`	`[api_version][org_href]traffic_flows/async_queries/:uuid`
Delete the completed async traffic query	`DELETE`	`[api_version][org_href]traffic_flows/async_queries/:uuid`

Create New Async Traffic Queries

Parameters for POST [api_version][org_href] traffic_flows/async_queries:

Property	Description	Type	Req
`query_name`	Name of the query	String	Yes
`start_date`	Starting date for the query. If left empty, the default interpretation is “today,” which is “now” minus 24 hours.	Date-time string (`YYYY-MM-DDTHH:MM:SS`)	No
`end_date`	Ending date for the query. If left empty, the default interpretation is “today,” which is “now” plus 24 hours.	Data-time string (`YYYY-MM-DDTHH:MM:SS`)	No
`sources`	Source labels, workloads, or IP addresses to include or exclude in the search. The response can contain up to five matching IP addresses. **NOTE:** The response returns `sources` as `consumers`. Sources are treated as consumers for the purposes of the request; the response returns the source of an individual flow as src. Sub-properties: • `include`: Targets that can be included are workloads, labels, or IP addresses identified by their HREF and structured as an array of JSON objects. If this property is left empty, then `include` means consider “ALL” or “ANY” of the object type. • `exclude`: Targets that can be excluded are workloads, labels, or IP addresses identified by their HREF and structured as a JSON array. ◦ When IP List is present in the consumer part of a traffic query, traffic from workloads that belong to that IP List will not be returned by default. If users want to see that traffic, they need to	Object	Yes Yes Yes

Property	Description	Type	Req
	set `exclude_workloads_from_ip_list_query: false` ◦ When IP List is present in the provider part of traffic query, traffic to workloads that belong to that IP List will not be returned by default. If the user wishes to see that traffic, they need to set `exclude_workloads_from_ip_list_query: false`		
`destinations`	Target labels, workloads, or IP addresses to include or exclude in the search. The response returns `targets` as `providers`. Required sub-properties: • `include`: Targets that can be *included* are workloads, labels, or IP addresses identified by their HREF and structured as an array of JSON objects. If this property is left empty, then `include` means consider “ALL” or “ANY” of the object type. • `exclude`: Targets that can be *excluded* are workloads, labels, or IP addresses identified by their HREF and structured as a JSON array. If this property is left empty, then `exclude` means exclude “NONE” of the object types.	Object	Yes Yes
`services`	Services (5-tuple of port/to_port/-proto/process/service) to include or exclude. Not all properties of the service subobjects are required. Required properties: • `include`: List of included services (5-tuple of port/to_port/proto/process/service) • `exclude`: List of excluded services (5-tuple of port/to_port/proto/process/service)", Properties of the `include`and `exclude` subobjects:		Yes

Property	Description	Type	Req
	• `port`: Port Number (integer 0-65535). Also the starting port when specifying a range. • `to_port`: High end of port range inclusive if specifying a range. If not specifying a range then don't send this: • `proto`: Protocol number. For the expected proto values see IANA Protocol Numbers. • `process_name`: name of the process • `windows_service_name`: name of the Sindows ser-vie		
`policy_decisions`	List of policy decisions. Allows you to filter the query based on policy decision: • `allowed`: Allowed traffic. • `potentially_blocked`: Allowed but potentially blocked traffic. This type of traffic occurs when a workload VEN is in the test policy state. • `blocked`: Blocked traffic. • `unknown`	Array of strings	Yes
`boundary_decisions`	List of boundary decisions • `blocked`: blocked due to boundary • `override_deny_rule`: overridden deny rule • `blocked_non_illumio_rule`: Deny rule not written by Illumio	Array	
`max_results`	Maximum number of flows to return. Limit is 200,000	Integer	Yes
`exclude_work-loads_from_ip_list_query`	Exclude workload traffic when IP List is provided either in consumer or provider part of the traffic query	Boolean	Default is: `true`

Download Completed async traffic Query Results

Properties for GET [api_version][org_href] traffic_flows/async_queries_download

Poperty	Description	Type	Req
`src`	Reference to `traffic_flows_endpoint.schema.json`		Yes
`dst`	Reference to `traffic_flows_endpoint.schema.json`		Yes
`service`	Reference to `traffic_flows_service.schema.json`		Yes
`num_connections`	The number of times this flow was seen	Integer	Yes
`policy_decision`	Policy decision made	String	Yes
`draft_policy_decision`	Draft policy decision of the flow (added in release 23.2.10)	String	No

`timestamp_range`	Timestamp ranges for the flow detected. Required properties are:
	`first_detected`
	`last_detected`
		Object	Yes

`caps`	Reference to `rbac_permission_types.schema.json`		Yes
`client_type`	Type of client which reported this flow	String	No

Introducing Illumination Plus

The API `GET [api_version][org_href]traffic_flows_async/queries/:uuid/dowload` has a new property: `caps`.

```
------------------------------------------------------------
},
       "caps": {
              "description": "Array of permissions for the flow for the current user",
              "type": "array",
              "items": {
              "$ref": "rbac_permission_types.schema.json"
       }
```

The `caps` property references the common schema `rbac_permission_types.schema.json`, which indicates the RBAC permission that is used: `write` .

rbac_permission_types.schema.json

```
            {
       "$schema":"http://json-schema.org/draft-04/schema#",
       "type": "string",
              "description": "RBAC Permission types",
              "enum": ["write", "provision"]
}
```

In Illumination Plus, the type `provision` is not used to avoid additional delays when checking the permissions of each flow. Therefore, only permission `write` is used and further verification is handled on the UI side.

Examples

Example Async Explorer Queries

Curl command for POST traffic_flows_async_queries

```
             curl -i -u api_
1195cf055cf8a834c:148afd87ecc980900eaf10d6c54e6c0f607b22e0dbf768dd007e51e731096282
https://devtest0.ilabs.io:8443/api/v2/orgs/1/traffic_flows/async_queries -H
"Content-Type: application/json" -X POST -d '{"sources":{"include":[[{"workload":
{"href":"/orgs/1/workloads/a3ffb374-f6c6-4cce-ac57-642c66f1498f"}}]],"exclude":
[]},"destinations":{"include":[[]],"exclude":[]},"services":{"include":
[],"exclude":[]},"sources_destinations_query_op":"and","start_date":"2016-01-
29T17:04:03.149Z","end_date":"2021-01-29T17:06:03.151Z","policy_decisions":
[],"max_results":1000,"query_name":"worklaod test"}'
```

Response:

```
       HTTP/1.1 202 Accepted
content-location: 7734501b-74a2-47a4-9ded-77bf4ceea938
content-type: application/json
content-length: 615
x-request-id: 00c8fa00-dbd8-4a28-a5c7-354fb5ae3886
cache-control: no-store
```

```
       x-frame-options: DENY
       x-xss-protection: 1; mode=block
       x-content-type-options: nosniff
{"status":"queued","href":"/orgs/1/traffic_flows/async_queries/7734501b-74a2-47a4-
9ded-77bf4ceea938","created_by":{"href":"/users/1"},"query_parameters":{"sources":
{"include":[[{"workload":{"href":"/orgs/1/workloads/a3ffb374-f6c6-4cce-ac57-
642c66f1498f"}}]],"exclude":[]},"destinations":{"include":[[]],"exclude":
[]},"services":{"include":[],"exclude":[]},"sources_destinations_query_
op":"and","start_date":"2016-01-29T17:04:03.149Z","end_date":"2021-01-
29T17:06:03.151Z","policy_decisions":[],"max_results":1000,"query_name":"worklaod
test"},"created_at":"2021-04-09T20:50:30Z","updated_at":"2021-04-09T20:50:30Z"}
```

Curl command for GET traffic_flows/async_queries

This query gets the collection of all async jobs for the current user, including anything that was already submitted.

```
curl -i -u api_
1195cf055cf8a834c:148afd87ecc980900eaf10d6c54e6c0f607b22e0dbf768dd007e51e731096282
https://devtest0.ilabs.io:8443/api/v2/orgs/1/traffic_flows/async_queries
```

Response

```
HTTP/1.1 200 OK
       content-type: application/json
       content-length: 1510
       x-request-id: fcf065e5-e465-4161-ba98-542182734c38
       cache-control: no-store
       x-frame-options: DENY
       x-xss-protection: 1; mode=block
       x-content-type-options: nosniff
[{"matches_count":1984,"flows_
count":1000,"status":"completed","href":"/orgs/1/traffic_flows/async_
queries/88675fbd-a88e-44bd-b358-2d6f2fc4f95a","result":"/orgs/1/traffic_
flows/async_queries/88675fbd-a88e-44bd-b358-2d6f2fc4f95a/download","created_by":
{"href":"/users/1"},"query_parameters":{"sources":{"include":[[{"workload":
{"href":"/orgs/1/workloads/a3ffb374-f6c6-4cce-ac57-642c66f1498f"}}]],"exclude":
[]},"destinations":{"include":[[]],"exclude":[]},"services":{"include":
```

```
[],"exclude":[]},"sources_destinations_query_op":"and","start_date":"2016-01-
29T17:04:03.149Z","end_date":"2021-01-29T17:06:03.151Z","policy_decisions":
[],"max_results":1000,"query_name":"worklaod tesrrrrrt"},"created_at":"2021-04-
09T20:50:19Z","updated_at":"2021-04-09T20:50:27Z"},{"matches_count":1984,"flows_
count":1000,"status":"completed","href":"/orgs/1/traffic_flows/async_
queries/7734501b-74a2-47a4-9ded-77bf4ceea938","result":"/orgs/1/traffic_
flows/async_queries/7734501b-74a2-47a4-9ded-77bf4ceea938/download","created_by":
{"href":"/users/1"},"query_parameters":{"sources":{"include":[[{"workload":
{"href":"/orgs/1/workloads/a3ffb374-f6c6-4cce-ac57-642c66f1498f"}}]],"exclude":
[]},"destinations":{"include":[[]],"exclude":[]},"services":{"include":
[],"exclude":[]},"sources_destinations_query_op":"and","start_date":"2016-01-
29T17:04:03.149Z","end_date":"2021-01-29T17:06:03.151Z","policy_decisions":
[],"max_results":1000,"query_name":"worklaod test"},"created_at":"2021-04-
09T20:50:30Z","updated_at":"2021-04-09T20:50:32Z"
```


Curl command for GET traffic_flows/async_queries/:uuid

This query gets a specific job included in the collection.

```
curl -i -u $KEY:$TOKEN https://devtest0.ilabs.io:8443/api/v2/orgs/1/traffic_
flows/async_queries/88675fbd-a88e-44bd-b358-2d6f2fc4f95a
```

Response


```
             HTTP/1.1 200 OK
       content-type: application/json
       content-length: 756
       x-request-id: f328b845-8542-4b96-a128-43aefdf7ba5a
       cache-control: no-store
       x-frame-options: DENY
       x-xss-protection: 1; mode=block
       x-content-type-options: nosniff
{"matches_count":1984,"flows_count":1000,"status":"completed",
"href":"/orgs/1/hanges for22.4.0 from the Wj/async_queries/88675fbd-a88e-44bd-
b358-2d6f2fc4f95a",
"result":"/orgs/1/traffic_flows/async_queries/88675fbd-a88e-44bd-b358-
2d6f2fc4f95a/download",
"created_by":{"href":"/users/1"},"query_parameters":{"sources":{"include":
[[{"workload":{"href":"/orgs/1/workloads/a3ffb374-f6c6-4cce-ac57-
```

642c66f1498f"}}]],"exclude":[]},"destinations":{"include":[[]],"exclude":[]},"services":{"include":[],"exclude":[]},"sources_destinations_query_op":"and","start_date":"2016-01-29T17:04:03.149Z","end_date":"2021-01-29T17:06:03.151Z","policy_decisions":[],"max_results":1000,"query_name":"worklaodtesrrrrrt"},"created_at":"2021-04-09T20:50:19Z","updated_at":"2021-04-09T20:50:27Z"}

Response for GET traffic_flows/async_queries/:uuid_download


```
          {
  "dst": {
      "ip": "10.244.0.1",
      "workload": {
              "href": "/orgs/1/workloads/35d8efea-f230-4027-a8ee-5f20626c4d21",
              "name": "wl3",
              "labels": [
                      {
                      "key": "env"reserpine for
                      "href": "/orgs/1/labels/7",
                      "value": "Production"
                      },
                      {
                      "key": "loc",
                      "href": "/orgs/1/labels/11",
                      "value": "Amazon"
                      },
                      {
                      "key": "role",
                      "href": "/orgs/1/labels/3",
                      "value": "API"
                      },
                      {
                      "key": "B-label",
                      "href": "/orgs/1/labels/15",
                      "value": "b_label_2"
                      }
              ],
              "managed": false,
              "os_type": "linux",
```

```
            "endpoint": false,
            "hostname": "",
            "enforcement_mode": "visibility_only"
            }
      },
   "src": {
      "ip": "10.0.2.15",
      "workload": {
            "href": "/orgs/1/workloads/fc3801b8-05ec-4954-a957-7f5673123389",
            "name": "wl2",
            "labels": [
            {
                  "key": "env",
                  "href": "/orgs/1/labels/7",
                  "value": "Production"
                  },
                  {
                  "key": "loc",
                  "href": "/orgs/1/labels/11",
                  "value": "Amazon"
                  },
                  {
                  "key": "role",
                  "href": "/orgs/1/labels/3",
                  "value": "API"
                  }
            ],
            "managed": false,
            "os_type": "linux",
            "endpoint": false,
            "hostname": "",
            "enforcement_mode": "visibility_only"
            }
      },
   "caps": [],
   "state": "snapshot",
   "dst_bi": 0,
   "dst_bo": 0,
```

```
    "seq_id": 2,
    "network": {
       "href": "/orgs/1/networks/fbeeb98d-4ed6-428d-9f71-69f542bfd8fd",
       "name": "Corporate"
       },
    "service": {
       "port": 3306,
       "proto": 6
       },
    "flow_direction": "outbound",
    "num_connections": 1,
    "policy_decision": "unknown",
    "timestamp_range": {
       "last_detected": "2022-09-01T20:35:22Z",
       "first_detected": "2022-09-01T20:35:22Z"
       }
}
```


Filter for Managed Services

This API allows you to filter all managed services, such as workloads, virtual services, and so on.

Functionality	HTTP	URI
Get a list of Virtual Servers	`GET`	`[api_version][org_href]/sec_policy/:version/virtual_servers`
Get a specified Virtual Server	`GET`	`[api_version][org_href]/sec_policy/:version/virtual_servers/:uuid`

Database Metrics

The API Database Metrics provides the organization-specific insight into the current traffic database. It gives you ability to monitor how big the traffic database is and how much data you can store. It also gives information about how many days of data is available.

The API `database_metrics` was expanded to include additional optional endpoints metrics: `server`, `backlog` and `endpoint`.

These objects define the server's or endpoint's total flow data per organization for the total number

of days, limit on the total number of days, oldest days, size in gigabytes, and so on.

Database Metrics API Methods

Functionality	HTTP	URI
Returns the organization database usage metrics. Provides to customers organization-specific insight into current traffic database size (#days, #GB).	`GET`	`[api_version][org_href]traffic_flows/database_metrics`

Parameters for Database Usage Metrics

The organization flow Database Usage Metrics has the following required parameters:

Parameters	Description	Type	Required
`flows_days`	Organization's total number of days of flow data	Integer	Yes
`flows_days_limit`	Organization's limit on the total number of days of flow data Limit was increased from 90 to 97	Integer	Yes
`flows_oldest_day`	Organization's oldest day of flow data (yyyy-mm-dd)	String	No
`flows_size_gb`	Organization's limit on the total number of gigabytes of flow data	Number	Yes
`flows_size_gb_limit`	Organization's limit on the total number of gigabytes of flow data	Number	Yes
`server`	Define the server's total flow data per organization for the total numberof days, limit on the total number of days, oldest days, size in gigabytes, and so on.	Object	No
`endpoint`	Organization's total number of days of endpoint flow data.	Object	No
`backlog`	Total gigabytes used to store flow data input files	Object	No
`updated_at`	Timestamp in UTC when these flow metrics were generated	String, date	No

Parameters for `server`

Parameters	Description	Type
`num_flows_days`	Organization's total number of days of the server flow data	Integer
`num_flows_days_limit`	Organization's limit on the total number of days of server flow data	Integer
`flows_oldest_day`	Organization's oldest day of server flow data (yyyy-mm-dd)	String, date
`flows_size_gb`	Organization's limit on the total number of giga-bytes of server flow data	Number
`flows_size_gb_limit`	Organization's limit on the total number of giga-bytes of server flow data	Number
`num_daily_tables`	The number of server daily tables, includ-ing FlowLink and CloudSe-cure, counted once for each unique day	Number
`num_weekly_tables`	The number of server weekly	Number

Parameters	Description	Type
	tables, including FlowLink and CloudSecure, counted once for each unique week	

Parameters for `endpoint`

Parameters	Description	Type
`num_flows_days`	Organization's total number of days of the endpoint flow data	Integer
`num_flows_days_limit`	Organization's limit on the total number of days of endpoint flow data	Integer
`flows_oldest_day`	Organization's oldest day of endpoint flow data (yyyy-mm-dd)	String, date
`flows_size_gb`	Organization's limit on the total number of gigabytes of endpoint flow data	Number
`flows_size_gb_limit`	Organization's limit on the total number of gigabytes of endpoint flow data	Number
`num_daily_tables`	The number of endpoint daily tables, counted once for each unique day	Number
`num_weekly_tables`	The number of endpoint weekly tables, counted once for each unique week	Number

Parameters for `backlog`

Parameters	Description	Type
`total_disk_used_gb`	Total gigabytes used to store flow data input files	Number
`total_file_count`	Total number of flow data input files	Integer

An example response looks such as the following:


```
{
        "org_id":1,
        "server":{
                "flows_size_gb":2.53228759765625,
                "num_flows_days":95,
                "flows_oldest_day":"2023-02-06",
                "num_daily_tables":7,
                "num_weekly_tables":13,
                "flows_size_gb_limit":26,
                "num_flows_days_limit":90
        },
        "endpoint":{
                "flows_size_gb":0.34337615966796875,
                "num_flows_days":6,
                "flows_oldest_day":"2023-05-11",
                "num_daily_tables":6,
                "num_weekly_tables":0,
                "flows_size_gb_limit":26,
                "num_flows_days_limit":14
        },
        "flows_days":95,
        "flows_size_gb":2.8644485473632812,
        "flows_days_limit":90,
        "flows_oldest_day":"2023-02-06",
        "flows_per_second":0.0,
        "flows_size_gb_limit":26,
        "updated_at":"2023-05-16T22:36:25Z"
}
```


Reporting APIs

Reporting APIs allow users to generate application reports. Instead of first exporting generated data, such as traffic flows, and then using other tools to create reports, users can now use built-in

reports.

Users can request one-time or recurring reports, specify time ranges, as well as report types.

Reporting APIs belong to several groups based on their use:

- Reporting Schedules
- Report Templates
- On-Demand Reports

There is also a Dashboard to help you visualize VEN statiscs:

- Reporting APIs

Reporting API Types

Reporting Schedules

These APIs provide a way for the Global Organization Administrator (`this_global_org_user`) to manage report schedules.

Each report can be generated once or recurring, where the recurrence is specified at the time of report configuration.

The default time-range is 30 days, and other possible values are: 1 day, 7 days, 14 days, 30 days, 60 days, and 90 days.

Functionality	HTTP	URI
Returns a collection of report schedules.	`GET`	`[api_version][org_href]/report_schedule`
Returns a scheduled report for the provided UUID.	`GET`	`[api_version][org_href]/report_schedule/:report_schedule_id`
Updates a report schedule for the provided UUID.	`PUT`	`[api_version][org_href]/report_schedule/:report_schedule_id`
Create a new report schedule.	`POST`	`[api_version][org_href]/report_schedule`
Deletes a report schedule for the provided UUID.	`GET`	`[api_version][org_href]/report_schedule/:report_schedule_id`

Request Parameters

Parameter	Description	Parameter Type	Format
`org_id`	Organization	path	Integer
`report_`	UUID of the report schedule	Sring	date/time

Parameter	Description	Parameter Type	Format
schedule_id			

Response Properties

Parameter	Description	Parameter Type
href	Report Schedule URI	URI, required
report_template	Template for the report	Object, required
report_generation_frequency	How often to generate a report: in addition to daily, weekly, monthly, and quarterly reports, you can schedule to receive the report only once.	String
report_parameters	Any specific parameters required for this report template • executive_summary_report_params.schema.json • traffic_flow_report_params.schema.json • explorer_report_params.schema.json • ves_report_params.schema.json	Object, required
created_at	Timestamp (rfc3339 timestamp) in UTC when this report schedule was created	String
created_by	The URI of the user who created this report schedule	URI
updated_at	Timestamp (rfc3339 timestamp) when this report schedule was last updated.",	String
updated_by	The URI of the user who updated this report schedule	URI

Defining Report Schedule Query

To define the query for report schedules, reference the required schemas (explained in Schemas to Define a Report).

- `executive_summary_report_params.schema.json`
- `traffic_flow_report_params.schema.json`
- `report_app_groups.schema.json`
- `custom_date_range.schema.json`
- `ves_report_params.schema.json`

Report Templates

These API's provide a way for the Global Organization Administrator (`this_global_org_user`) to manage report templates. In each report-template, they can specify type, time-range, recurrence and suitable parameters for the report type.

Functionality	HTTP	URI
Lists the collection of all available report templates for this user and organization.	`GET`	`[api_version][org_href]/report_templates`
This API is used to enable/disable a specific report type, which can be done only by the organization administrator.	`PUT`	`[api_version][org_href]/report_templates`

Properties for Report Templates

Property	Description	Parameter Type	Required
`href`	Report Template URI	String	Yes
`name`	Display name for this report template maxLength: 255	String	Yes
`report_parameters`	Any specific parameters required for this report template to define one of the report types, using on of the listed schemas: • `executive_summary_report_params.schema.json`	Object	Yes

Property	Description	Parameter Type	Required
	• traffic_flow_report_params.schema.json • explorer_report_params.schema.json • ves_report_params.schema.json		

Defining Report Templates Query

To define the query for report templates, reference the required schemas (explained in Schemas to Define a Report).

On-Demand Reports

The user authorized as the Global Organization Administrator (`this_global_org_user`) can download various kinds of reports, as well as create them on-demand or add the property `report_format` to determine the format in which the report will be generated.

Functionality	HTTP	URI
Returns a collection of reports.	GET	[api_version][org_href]/reports
Returns a report for the provided UUID.	GET	[api_version][org_href]/reports/:report_id
Downloads a specific report.	GET	[api_version][org_href]/reports/:report_id/download
Creates a new on-demand report.	POST	[api_version][org_href]/reports
Cancels a report if it's not yet completed/failed	PUT	[api_version][org_href]/reports/:report_id
Added a new property `report_format` which determines the format in which the report should be generated	POST	[api_version][org_href]/reports_schedules
Added a new property `report_format` which determines the format in which the report should be generated	PUT	[api_version][org_href]/reports_schedules

Parameters for On-Demand Reports

Parameter	Description	Parameter Type	Required
href		Integer	Yes
report_tem-	Template for the report	Object	Yes

Parameter	Description	Parameter Type	Required
plate			
status	Current status of this report	String	Yes
report_parameters	Any specific parameters required for this report template to define one of the report types, using on of the listed schemas: • executive_summary_report_params.schema.json • traffic_flow_report_params.schema.json • explorer_report_params.schema.json • ves_report_params.schema.json	Object	Yes
send_by_email	Flag for whether to send user report by email	Boolean	True/false
progress_percentage	Progress percentage for this report.	Integer	"minimum": 0, "maximum": 100

Report Settings

Report Settings define for how many days a report will be stored or persisted.

The user authorized as the Global Organization Administrator (this_global_org_user) can manage the report settings, list them or update.

Functionality	HTTP	URI
Get the report settings for an organization	GET	[api_version][org_href]/report_settings
Updates the report settings for an org	PUT	[api_version][org_href]/report_settings

Schemas to Define a Report

These schemas are referenced and used to define the content of a report:

- `executive_summary_report_params.schema.json`

  Reports parameters for the executive summary report, such as `report_time_range` (Time range the report is built across) and references to `report_time_range_definitions.schema.json#/definitions/custom_date_range` or `report_time_range_definitions.schema.json#/definitions/last_num_days`.

- `traffic_flow_report_params.schema.json`

  Reports parameters for traffic flow query report.

- `report_app_groups.schema.json`

  This is the App Group Schema for reports.

- `custom_date_range.schema.json`

  Provides the time range the report is built across.

- `common legacy_workload_modes.schema.json`

  Provides the assigned labels summary with the label URI, as well as the key and value in the key-value pair.

- `report_time_range_definitions.schema.json`

  Provides the report parameters for the executive summary report, such as Start date for the range, End date for the range, and Last x number of days the report is built across.

- `labels_summary.schema.json`

  Provides the assigned labels summary with properties such as label URI, Key in key-value pair, and Value in key-value pair.

- `ves_report_params.schema.json`

  Provides report parameters for the new ves (vulnerability-exposure score) report type.

Examples

Report Templates

GET /orgs/:xorg_id/report_templates

List the report templates for this user and organization.

```
[
        {
                "href": "/orgs/1/report_templates/executive_summary_report",
                "name": "Executive Summary",
                "report_parameters": {}
        },
        {
                "href": "/orgs/1/report_templates/traffic_flow_report",
                "name": "Traffic Flow Query",
                "report_parameters": {
                        "app_groups": []
                }
        }
]
```

This request references the following two schemas (see Schemas to Define a Report).

- `executive_summary_report_params.schema.json`
- `traffic_flow_report_params.schema.json`

Report Schedules

POST /api/v2/orgs/1/report_schedules

Request to create a new report schedule:

```
{
        "report_template": {
                "href": "/orgs/1/report_templates/traffic_flow_report"
        },
        "name": "John's Traffic Flow Report - Quarterly",
        "report_generation_frequency": "quarterly",
```

```
        "report_parameters": {
                "report_time_range": {
                        "last_num_days": 90
                },
                "app_groups": [

                ]
        }
}
```

Response (201 created)

```
{
        "href": "/orgs/1/report_schedules/8a08b381-c8fe-4837-b9c6-071c70861369",
        "report_template": {
                "href": "/orgs/1/report_templates/traffic_flow_report"
        },
        "name": "John's Traffic Flow Report - Quarterly",
        "report_generation_frequency": "quarterly",
        "report_parameters": {
                "app_groups": [],
                "report_time_range": {
                        "last_num_days": 90
                }
        }
}
```

On-demand Reports

POST /api/v2/orgs/1/reports

Request to create an on-demand report in the PDF format (`report_format`):

```
{
        "report_template": {
                "href": "/orgs/1/report_templates/executive_summary_report"
        },
        "description": "John's Executive Summary Report",
        "report_parameters": {
```

```
                "report_time_range": {
                "last_num_days": 30
                }
        },
        "report_format": "pdf"
}
```

Response

```
{
        "href": "/orgs/1/reports/be9b68ec-c35a-49bb-9400-f78c9950e321",
        "report_template": {
                "href": "/orgs/1/report_templates/executive_summary_report",
                "name": "Executive Summary Report"
        },
        "description": "John's Executive Summary Report",
        "created_at": "2021-01-15T05:45:27.130Z",
        "updated_at": "2021-01-15T05:45:27.130Z",
        "progress_percentage": 0,
        "generated_at": null,
        "status": "queued",
        "report_parameters": {
                "report_time_range": {
                        "last_num_days": 30
                }
        },
        "created_by": {
                "href": "/users/1"
        },
        updated_by": {
                "href": "/users/1"
        }
}
```

Report Settings

GET /orgs/:xorg_id/settings/reports

Request to list report settings:

```
{
        "href": "/orgs/1/report_settings",
        "report_retention_days": 1,
        "enabled": true,
        "max_queued_reports": 25
}
```

Ransomware Protection Dashboard APIs

The Ransomware Dashboard is powered by the two main APIs: `time_series` and `risk_summary`.

For more details, see Ransomware Dashboard.

Risk SummaryAPIs

Ransomware Dashboard APIs that evaluate the risk from ransomware attacks are :

- reports_risk_summary_ransomware_timeseries_statistics_post
- reports/risk_summary_get
- num_protected_unprotected_ports

Time Series APIs

New APIs for the Ransomware Dashboard that are dedicated to reporting about ransomware events are:

- reports_time_series_statistics_post
- reports_time_series_statistics_post_response

Workloads APIs Changed for Ransomware

- workloads_get

  The object `risk_summary`
- workloads_risk_details_get
- The object `risk_details`

  Reference for `workloads_risk_details_get`

Settings APIs Changed for Ransomware

- settings_get

  Poperty `num_assets_requiring_ransomware_protection`

- settings_put

  Poperty `num_assets_requiring_ransomware_protection`

Security Policy Changed for Ransomware

- sec_policy_services_post
- sec_policy_services_put
- sec_policy_services_get

These Security Policy APIs are explained in the topic Services.Services

List of APIs

reports_risk_summary_ransomware_timeseries_statistics_post

This API is used to show the time series data:

- Number of managed workloads
- Percent of the ransomware protection coverage
- Number of workloads by exposure

Data is presented with the granularity of `day`, `week`, `month`, and `quarter`, where the default is `day`.

reports/risk_summary_get

Security administrators use this API to view how many workloads are ransomware protection ready and then assess the degree of protection in their whole system. This schema supplies the required information to run the Ransomware Dashboard:

- Number of total workloads
- Number of protected workloads
- Number of risky ports by the severity of their risk exposure (low, medium, high, and critical)
- Workload protection by the port type (admin and legacy)
- Ransomware protection coverage percent
- Date when the status was last updated

Sample Response for reports/risk_summary_get


```
{
   "ransomware":{
       "num_total_workloads":98,
       "num_protected_workloads":22,
       "workload_protection_by_severity":{
         "low":{
              "protected_workload_count":2,
              "unprotected_workload_count":8
          },
          "medium":{
              "protected_workload_count":3,
              "unprotected_workload_count":6
          },
         "high":{
              "protected_workload_count":2,
              "unprotected_workload_count":8
          },
          "critical":{
              "protected_workload_count":3,
              "unprotected_workload_count":6
          }
                       },
   "workload_protection_by_port_type":{
       "admin":{
          "protected_workload_count":2,
          "unprotected_workload_count":8
       },
       "legacy":{
          "protected_workload_count":3,
          "unprotected_workload_count":6
       }
   },
   "ransomware_protection_coverage_percent":56,
   "last_updated_at":"2023-01-21 23:32:42.679673"
   }
}
```

In release 23.5, this API was changed so that the property `risky_ports_by_category` was added to support the widget "Risky ports by type" in the UI.

```
      "risky_ports_by_category": {
         "description": "Risky ports by Port type",
         "type": "object",
          "properties": {
               "admin": {
                       "$ref": "num_protected_unprotected_ports.schema.json"
               },
               "legacy": {
                       "$ref": "num_protected_unprotected_ports.schema.json"
               }
       }
}
```

num_protected_unprotected_ports

This schema is referenced from `reports_risk_summary_get.schema.json` to supply the number of protected and unprotected ports for a specified risk level:

```
{
   "$schema": "http://json-schema.org/draft-04/schema#",
   "type": "object",
   "required": [
       "num_protected_ports",
       "num_unprotected_ports"
  ],
   "properties": {
       "num_protected_ports": {
          "description": "Number of protected ports for this risk level, across all
protection ready workloads",
          "type": "integer"
          },
       "num_unprotected_ports": {
          "description": "Number of unprotected ports for this risk level, across all
protection ready workloads",
          "type": "integer"
```

```
        }
    }
}
```

reports_time_series_statistics_post

This schema supplies the granularity of the time series data.

The API `reports_time_series_statistics_post` includes these properties:

- `num_managed_workloads`, which is requested by the payload. The resolution might be `day`, `week`, `month`, and `quarter`, which defines what the UI will show. The default value is "day".
- `ransomware_protection_coverage_percent`: Percent of the ransomware protection coverage (added in release 23.5)
- `num_workloads_by_exposure`: Number of workloads by exposure (added in release 23.5)

Data is presented with the granularity of `day`, `week`, `month`, and `quarter`, where the default is `day`.

```
{
        "$schema": "http://json-schema.org/draft-04/schema#",
        "type": "array",
        "items": {
           "type": "object",
           "required": [
                "property"
                ],
           "properties": {
                "property": {
                   "description": "The property for which time series data is requested.",
                   "type": "string",
                   "enum": [
                   "num_managed_workloads",
                   "ransomware_protection_coverage_percent",
                   "num_workloads_by_exposure"
                   ]
                },
```

reports_time_series_statistics_post_response

This API specifies the time series data about the protected workloads.

Previously, the schema contained the integer count on the end date of the counted period. This item was removed:

```
"count": {
          "description": "The integer count on the end date of this period.",
          "type": "integer"
          },
       "unit": {
          "description": "The unit of the value returned.",
          "type": "string"
          },
```

Starting from the release 23.5, this API gives the percentage of the end date of the counted period.

It is referencing the schema `num_workloads_by_exposure_time_series`.

```
 "data": {
   "oneOf": [
   {
       "$ref": "../../../agent/schema/v2/num_workloads_by_exposure_time_
series.schema.json"
   },
   {
       "count": {
              "description": "The integer count on the end date of this period.",
              "type": "integer"
              }
       },
       {
       "percentage": {
              "description": "The percentage on the end date of this perioc.",
              "type": "number",
              "minumum": 0,
              "maximum": 100
              }
       }
]
```

workloads_get

This Public Stable API was changed to support the Ransomware Dashboard in the following way:

One new object was added: `risk_summary`, which explains the risk summary for the workload. This object includes a required object `ransomware`, which supplies these properties:

- `workload_exposure_severity`
- `ransomware_protection_percent`
- `last_updated_at`

```
{
 "properties": {
   "risk_summary": {
       "description": "Risk Summary for this workload",
       "type": "object",
       "required": [
          "ransomware"
       ],
       "properties": {
          "ransomware": {
               "type": [
                  "object",
                  "null"
               ],
               "required": [
                  "workload_exposure_severity",
                  "ransomware_protection_percent",
                  "last_updated_at"
               ],
               "properties": {
                  "workload_exposure_severity": {
                       "description": "Exposure severity of the workload",
                       "type": "string"
                       },
                  "ransomware_protection_percent": {
                       "description": "Ransomware protection percentage for this workload",
                       "type": "number"
                       },
                  "last_updated_at": {
```

```
                    "description": "The time at which the ransomware stats are last computed at
                    "type": "string",
                    "format": "date-time"
                    }
                }
        }
}
```

workloads_risk_details_get

This API, which supplies the risk details, you can see in action on the Workloads page, tab Ransomware Protection.

In addition to the organization admin, the users who have access to the workload can view the ransomware protection details for that workload, or how many risky ports are protected and how many risky ports are not protected.

```
{
   "$schema": "http://json-schema.org/draft-04/schema#",
   "type": "object",
   "properties": {
       "risk_details": {
       "type": "object",
       "required": [
          "ransomware"
       ],
       "ransomware": {
          "type": [
               "object",
               "null"
          ],
          "properties": {
             "details": {
               "type": "array",
               "items": {
                  "$ref": "workload_ransomware_services.schema.json"
               }
          },
          "last_updated_at": {
```

```
                "description": "The time at which the protection stats were last computed at",
                "type": "string",
                "format": "date-time"
            }
        }
}
```

Sample Response for `workloads_risk_details_get`

```
{
   "risk_details":{
       "ransomware":{
          "services":[
             {
                "href":"/api/v2/orgs/8/workloads/23131cf5-1d70-42de-9242-39055338d0ef",
                "name":"SSH",
                "port":22,
                "protocol":17,
                "severity":"low",
                "port_status":"listening",
                "protection_state":"unprotected",
                "active_policy":"allowed",
                "draft_policy":"blocked",
                "recommendation":"add_boundary"
          },
          {
                "href":"/api/v2/orgs/8/workloads/23131cf5-1d70-42de-9242-39055338d0ef",
                "name":"SSH",
                "port":22,
                "protocol":6,
                "severity":"high",
                "port_status":"listening",
                "protection_state":"protected",
                "active_policy":"allowed",
                "draft_policy":"blocked",
                "recommendation":"has_draft_policy_needs_provisioning"
          }
       ],
       "last_updated_at":"2023-01-21 23:32:42.679673"
```

```
        }
    }
}
```

Sample Responses for `workloads_risk_details_get` when the evaluation concludes there is no risk for the workload.

When the results are not yet computed

```
{
    "risk_details":{
        "ransomware": null
    }
}
```

The full response looks as follows:

```
[
    {
        "property":"num_managed_workloads",
        "time_series":[
            {
                "start_date":"2022-10-31",
                "end_date":"2022-11-2",
                "count":120
            },
            {
                "start_date":"2022-10-24",
                "end_date":"2022-10-30",
                "count":115
            },
            {
                "start_date":"2022-10-17",
                "end_date":"2022-10-23",
                "count":110
            },
            {
                "start_date":"2022-10-10",
                "end_date":"2022-10-16",
```

```
                "count":100
            }
        ]
    }
]
```

workload_ransomware_services

This schema is referenced from `workloads_risk_details_get` to supply the required service data:

- Service location and name
- Service Port and Protocol
- Severity and Protection state of this service
- Status of the port on the workload
- Active and Draft policy that applies to the Port

```
{
   "$schema": "http://json-schema.org/draft-04/schema#",
   "type": "object",
   "required": [
        "href",
         "port",
         "protocol",
         "severity",
         "port_status",
         "protection_state",
         "active_policy",
         "draft_policy"
       ],
   "properties": {
      "href": {
          "description": "Reference of the service",
          "type": "string"
       },
      "name": {
          "description": "Name of the service",
          "type": "string"
```

```
      },
      "port": {
         "description": "Port Number",
         "type": "integer",
         "minimum": 0,
         "maximum": 65535
      },
      "proto": {
         "description": "Protocol Number",
         "type": "integer"
      },
      "severity": {
         "description": "Severity of this service",
         "type": "string",
         "enum": [
            "low",
            "medium",
            "high",
            "critical"
         ]
      },
      "category": {
         "description": "Category of this service",
         "type": "string",
         "enum": [
            "admin",
            "legacy"
         ]
     },
      "port_status": {
         "description": "Status of the port on the workload",
         "type": "string",
         "enum":
            "listening",
            "inactive"
            ]
         },
      "protection_state": {
```

```
          "description": "Protection state of this service",
          "type": "string",
          "enum": [
              "unprotected",
              "protected_open",
              "protected_closed"
             ]
       },
       "active_policy": {
             "description": "Active Policy that applies to this port",
             "type": "string",
             "enum": [
                 "allowed",
                 "allowed_across_boundary",
                 "blocked_by_boundary",
                "blocked_no_rule"
             ]
         },
        "draft_policy": {
             "description": "Draft Policy that applies to this port",
             "type": "string",
             "enum": [
                  "allowed",
                  "allowed_across_boundary",
                  "blocked_by_boundary",
                  "blocked_no_rule"
               ]
         },
          "recommendation": {
               "description": "Recommendation for this port based on enforcement state, allow
and deny rules and active/draft rule",
               "type": "string",
               "enum": [
                  "add_boundary",
                  "has_draft_policy_needs_provisioning"
               ]
           }
      }
```

```
}
```

In release 23.5, additional information about the operating systems has been added for the ransomware service: Windows and Linux.

```
{
       "properties": {
           "os_platforms": {
               "description": "Operating system for this ransomware service",
               "type": "array",
               "minItems": 1,
               "items": {
                   "type": "string",
                   "enum": [
                   "windows",
                   "linux"
               ]
           }
       }
    }
}
```

settings_get

This Public Stable API was changed to include a new property `num_assets_requiring_ransomware_protection`.

```
{
   "$schema": "http://json-schema.org/draft-04/schema#",
   "type": "object",
   "properties": {
       "href": {
          "description": "Org Setting URI",
          "type": "string",
          "format": "uri"
       },
       "num_assets_requiring_ransomware_protection": {
          "description": "number of assets that need ransomware protection for this
```

```
org",
          "type": [
                "integer",
                 "null"
          ]
       },
              ====================================
```

settings_put

This Public Stable API was changed to include a new property `num_assets_requiring_ransomware_protection`, which provides a number of assets that need ransomware protection in a specific organization (1 - 9999999). Number of assets is between one and 9999999.

```
{
   "$schema": "http://json-schema.org/draft-04/schema#",
   "type": "object",
   "additionalProperties": false,
   "properties": {
         "num_assets_requiring_ransomware_protection": {
               "description": "number of assets that need ransomware protection for this org",
               "type": "integer",
               "minimum": 1,
               "maximum": 9999999
               ==================================
```

VEN Dashboard APIs

The Dashboard uses the following API to aggregate various data from the system and help you focus on the data you are interested in:

```
POST api/v2/orgs/:xorg_id/vens/statistics
```

You can obtain summary statistics for VENs by specifying which statistics you are interested in from a set of options. The API also supports obtaining a count for a specific value of a property (such as a count of VENs from a specific product version).

For more details about the VEN Dashboard, see VEN Dashboard.

POST vens/statistics

Sample Request

```
{
   “property_counts”: [
       {
          "property": "version",
          "values": [
              "19.3",
              "18.3"]
          "filters": [
               {
                  "filter_property": "status",
                  "values": [
                      "active",
                      ""
                  ]
               },
               {
                  "filter_property": "containerized",
                  "values": [
                     "true"
                  ]
               }
           ]
   },
   {
         "property": "version",
          "filters": [
               {
               "filter_property": "status",
               "values": [
                   "active"]
               }
          ]
      },
      {
          "property": "health"
      }
```

```
   ]
}
```

Sample Response

```
{
       “property_counts”: [
          {
             "property": "version",
             “counts”:[
               {
                  "value": "19.1",
                  "count": 1
               },
               {
                  "value": "18.3",
                  "count": 2
               }
          ]
       },
       {
             "property": "version",
             “counts”:[
               {
                  "value": "18.1",
                  "count": 1
               },
               {
                  "value": "18.2",
                  "count": 1
               },
               {
                  "value": "18.3",
                  "count": 2
               },
               {
                  "value": "19.1",
                  "count": 2
               }
```

```
        ]
},
{
          "property": "health",
          “counts”:[
               {
                  "value": "healthy",
                  "count": 3
               },
               {
                  "value": "warning",
                  "count": 3
               },
               {
                  "value": "err",
                  "count": 2
               }
           ]
        }
   ]
}
```

Vulnerabilities

Vulnerabilities are defined as entries based on the possible risk of allowing traffic on a port/-protocol combination, and a vulnerability instance is the existence of a vulnerability.

This Public Experimental API lists, creates, updates, and deletes vulnerabilities.

> NOTE:
> The Illumio Core Vulnerability Maps license is required to import Qualys report data into the Illumio PCE. For information about obtaining the Illumio Core Vulnerability Maps license, contact Illumio Support. When you obtain your license, you also receive information about how to install it.

Delete the Vulnerability License

To delete the vulnerability license, use the following CURL command from your CLI environment:

```
export API_KEY=api_key_username:api_key_secret
```

```
curl -i -H "Content-Type: application/json" https://pce_fqdn:8443/api/v2/orgs/org_id/licenses/9df01357-93cf-4f33-b720-e47bba783c55 -X DELETE -u $API_KEY
```

Replace the variables, which are entered in **blue bold**.

Vulnerability API Methods

Functionality	HTTP	URI
Get vulnerabilities	GET	[api_version][org_href]vulnerabilities
Get an individual vulnerability	GET	[api_version][org_href][vulnerabilities_href]
Create an individual vulnerability	POST	[api_version][org_href][vulnerabilities_href]
Modify an individual vulnerability	PUT	[api_version][org_href][vulnerabilities_href]
Delete an individual vulnerability	DELETE	[api_version][org_href][vulnerabilities_href]

Get Collection of all Vulnerabilities

In this example, the maximum number of vulnerability reports is set to 2. Not using this query parameter in this GET method would return all the vulnerability reports up to a maximum of 500. For more than 500 vulnerabilities, use an Asynchronous GET Collection.

Parameter	Description	Data Type
org_id		Integer
max_results	The maximum number of vulnerabilities returned by a call to GET /vulnerabilities. (Optional. If not specified, all vulnerabilities are returned up to a maximum of 500.)	Integer

Curl Command to Get Collection of Vulnerabilities

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/7/vulnerabilities -H 'Accept: application/json' -u $KEY:$TOKEN
```

Response Body

```
[
  {
    "href": "/orgs/2/vulnerabilities/qualys-xxxxxebe7e17",
    "name": "Host Scan Time",
    "score": 37,
```

```
    "description": "{\"severity\":\"1\"}",
    "cve_ids": [],
    "created_at": "2017-12-21T19:15:48.000Z",
    "updated_at": "2017-12-21T19:17:26.000Z",
    "created_by": null,
    "updated_by": null
  },
.....................................................
]
```

Get an Individual Vulnerability

Parameters

Parameter	Description	Parameter Type
`org_id`	Organization	Integer
`reference_id`	The ID of the vulnerability to return by `GET /vulnerabilities/{reference_id}`.	String

Curl Command to Get an Individual Vulnerability

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/7/vulnerabilities/qualys-xxxxxebe7e18 -H 'Accept: application/json' -u $KEY:$TOKEN
```

Response Body

```
{
  "href": "/orgs/2/vulnerabilities/qualys-xxxxxebe7e18",
  "name": "Host Scan Time",
  "score": 37,
  "description": "{\"severity\":\"1\"}",
  "cve_ids": [],
  "created_at": "2017-12-21T19:15:48.000Z",
  "updated_at": "2017-12-21T19:17:26.000Z",
  "created_by": null,
```

```
  "updated_by": null
}
```

Create or Update a Vulnerability

Parameters

Parameter	Description	Parameter Type	Data Type
`reference_id`	The ID of the vulnerability. The `reference_id` is the last element of the `href` property returned by a call to `GET /vulnerabilities`.	Path	String
`score`	The normalized score of the vulnerability in the range of 0 to 100 inclusive. CVSS Score can be used here with a 10x multiplier.	Body	Integer
`name`	The title/name of the vulnerability.	Body	String
`cve-ids`	The `cve_ids` for the vulnerability.	Body	[String]
`description`	An arbitrary field to store details about the vulnerability class.	Body	String

Curl Command to Create or Update Vulnerability

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/7/vulnerabilities/qualys-xxxxxebe7e18 -H 'Content-Type: application/json' -u $KEY:$TOKEN -d '{"score": 50, "cve_ids": ["CVE-2012-xxxx", "CVE-2017-xxxx"], "description": "My vulnerability test."}'
```

Example Request Body

```
{
  "score": 50,
  "cve_ids": ["CVE-2012-xxxx", "CVE-2017-xxxx"],
  "description": "My vulnerability test."
}
```

Response

On success, the system displays `HTTP/1.1 204 No Content`.

Delete a Vulnerability

To delete an individual vulnerability, specify its HREF, which can be obtained from the response from `GET /vulnerabilities`.

Request Parameter

Parameter	Description	Parameter Type	Data Type
`reference_id`	The reference ID of the vulnerability. The last element of the `href` property of a vulnerability returned by a call to `GET /vulnerabilities`.	Path	String

Curl Command to Delete Vulnerability

```
curl -i -X DELETE https://pce.my-company.com:8443/api/v2/orgs/7/vulnerabilities/qualys-xxxxxebe7e18 -u $KEY:$TOKEN
```

Vulnerability Reports

This Public Experimental API creates, updates, and deletes vulnerability reports.

> NOTE:
> An Illumio Core Vulnerability Maps license is required to import Qualys report data into the Illumio PCE. For information about obtaining the Illumio Core Vulnerability Maps license, contact Illumio Support. When you obtain your license, you also receive information about how to install it.

Vulnerability Reports API Methods

HTTP	Functionality	URI
`GET`	Get a collection of vulnerability reports	`[api_version][org_href]/vulnerability_reports`
`GET`	Get an individual vulnerability report	`[api_version][vulnerability_reports_href]`
`POST`	Create an individual vulnerability report	`[api_version][vulnerability_reports_href]`
`PUT`	Update an individual vulnerability report	`[api_version][vulnerability_reports_href]`
`DELETE`	Delete an individual vulnerability report	`[api_version][vulnerability_reports_href]`

Get a Collection of Vulnerability Reports

This method gets a collection of all vulnerability reports in your organization.

By default, the maximum number returned by a GET collection of vulnerability reports is 500. For more than 500 vulnerability reports, use an Asynchronous GET Collection.

Curl Command to Get Collection of Vulnerability Reports

In this example, the maximum number of vulnerability reports is set to 2. Not using this query parameter in this GET method would return all the vulnerability reports up to a maximum of 500.

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/7/vulnerability_reports
-H 'Accept: application/json' -u $KEY:$TOKEN
```

Query Parameter to Get a Collection of Vulnerability Reports

Parameter	Description	Parameter Type	Data Type
`max_results`	The maximum number of vulnerability reports returned by a call to `GET /vulnerability_reports`. Optional. If not specified, by default, all vulnerability reports are returned up to a maximum of 500.	Query	Integer

Response Body

```
[
  {
    "href": "/orgs/2/vulnerability_reports/qualys-report-12345",
    "report_type": "qualys",
    "name": "my-report-2017-12-21-19-15-47",
    "created_at": "2017-12-21T19:15:48.000Z",
    "updated_at": "2017-12-21T19:15:48.000Z",
    "num_vulnerabilities": 4887,
    "created_by": null,
    "updated_by": null
  },
  {
    "href": "/orgs/2/vulnerability_reports/qualys-report-12346",
    "report_type": "qualys",
    "name": "my-report-2017-12-21-19-17-15",
    "created_at": "2017-12-21T19:17:15.000Z",
```

```
    "updated_at": "2017-12-21T19:17:15.000Z",
    "num_vulnerabilities": 1776,
    "created_by": null,
    "updated_by": null
  }
]
```

Get a Vulnerability Report

Curl Command to Get Vulnerability Report

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/7/vulnerability_
reports/qualys-report-123456 -H 'Accept: application/json' -u $KEY:$TOKEN
```

Request Parameter to Get an Individual Vulnerability Report

The following required path parameter restricts the results of the GET command to the specified vulnerability report.

Parameter	Description	Parameter Type	Data Type
`reference_id`	The ID of the vulnerability report (this is the last element in the vulnerability report HREF returned by a call to `GET /vulnerability_reports`).	Path	String

Response Body

```
{
  "href": "/orgs/2/vulnerability_reports/qualys-report-123456",
  "report_type": "qualys",
  "name": "my-report-2017-12-21-19-17-15",
  "created_at": "2017-12-21T19:17:15.000Z",
  "updated_at": "2017-12-21T19:17:15.000Z",
  "num_vulnerabilities": 1776,
  "created_by": null,
  "updated_by": null
}
```

Create or Update a Vulnerability Report

Curl Command to Update a Vulnerability Report

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/7/vulnerability_
reports/qualys-report-123456 -H 'Content-Type: application/json' -u $KEY:$TOKEN -d
'{"name": "My vulnerability report", "report_type": "qualys"}'
```

Response Properties

Property	Description	Data Type
`name`	User generated the name of the vulnerability report.	Integer
`report_type`	A string representing the type of the report.	String
`authoritative`	Boolean value specifies whether a report is authoritative or not.	[String]
`scanned_ips`	The ips on which the scan was performed. Enforced 100K maxitem limit.	String
`detected_vulnerabilities`	An array of parameters, of which `ip_address`, `workload`, and `vulnerability` are required. Enforced 100K maxitem limit. `ip_address`: (Required) The IP address of the host where the vulnerability is found (string) `port`: The port associated with the vulnerability (integer) `proto`: The protocol that is associated with the vulnerability (integer) `workload`: (Required) The URI of the workload associated with this vulnerability (string) `vulnerability`: (Required) The URI of the vulnerability class associated with this vulnerability (string)	
`external_data_reference`	(`PUT` only) This parameter supports third-party reference data	
`state`	(`PUT` only) Enables deletion, addition, or updating of vulnerabilities	
`exported_at`	(`PUT` only) Saves the timestamp for the next delta pull.	

Example Request Body

```
{
  "name":"My vulnerability report",
  "report_type": "qualys",
  "authoritative": true
}
```

Response

On success, the system displays `HTTP/1.1 204 No Content`.

Delete a Vulnerability Report

To delete an individual vulnerability report, specify the last element of its HREF, which can be obtained from the response from `GET /vulnerabilities`.

Curl Command to Delete Vulnerability Report

```
curl -i -X DELETE https://pce.my-company.com:8443/api/v2/orgs/7/vulnerability_
reports/qualys-report-2017-12-21-19-17-15 -u $KEY:$TOKEN
```

Request Parameter

Parameter	Description	Parameter Type	Data Type
`reference_id`	The ID of the vulnerability report (this is the last element in the vulnerability report HREF returned by a call to `GET /vulnerability_reports`).	Path	String

Chapter 10

Workloads

This chapter contains the following topics:

The Workloads APIs allow you to get information about workloads and network interfaces and to identify unauthorized traffic to or from workloads. Use the Workloads APIs to perform workload-related operations, such as pair workloads, configure pairing profiles, and obtain pairing keys.

Configure pairing profiles to apply properties to workloads as they pair with the PCE, such as what labels to apply. By configuring a pairing profile, you obtain a unique pairing key that identifies the VEN. Pair workloads to install VENs on them. The VEN reports detailed workload information to the PCE, such as which services are running on the workload.

Workload Operations

This Public Stable API allows you to perform workload operations, such as create an unmanaged workload, update workload information, unpair a workload, and delete a workload.

Workload Methods

Functionality	HTTP	URI
Get a collection of all workloads	GET	[api_version][org_href]/workloads
Get a specified workload	GET	api_version][org_href]/-workloads/workload_id
Create an unmanaged workload	POST	[api_version][org_href]/workloads
Update a workload or mark as suspended	PUT	[api_version]/workloads/workload_id
Unpair a workload	PUT	[api_version][org_href]/workloads/unpair
Delete an unpaired workload	DELETE	[api_version][org_href]/workloads

Query Parameters

Parameter	Description	Type	Required
org_id	Organization	Integer	Yes
agent.active_pce_fqdn	FQDN of the PCE	String	No
container_clusters	List of container cluster URIs, encoded as a JSON string	String	No
enforcement_mode	Enforcement mode of workload(s) to return.	String	No
external_data_set	The data source from which a resource originates		
external_data_reference	A unique identifier within the external data source	String	No
hostname	Hostname of workload(s) to return. Supports partial matches	String	No
include_deleted	Include deleted workloads	Boolean	No
ip_address	IP address of workload(s) to return. Supports partial matches	String	No
labels	List of lists of label URIs, encoded as a JSON string. From release 22.3.0, this API is not referencing labels.schema.json and it lists labels associated.with this workload. Required properties are: href, key, and value.	String	No
last_heartbeat_on[gte]	Greater than or equal to value for last heartbeat on timestamp	Integer	No
last_heartbeat_on	Less than or equal to value for last heartbeat	Integer	No

Parameter	Description	Type	Required
`[lte]`	on timestamp		
`log_traffic`	Whether we want to log traffic events from this workload	Boolean	No
`managed`	Return managed or unmanaged workloads using this filter. `True` if the workload is managed, else `false`	Boolean	No
`max_results`	Maximum number of workloads to return.	Integer	No
`mode`	Management mode of workload(s) to return. DEPRECATED AND REPLACED (Use enforcement_mode)	String	No
`name`	Name of workload(s) to return. Supports partial matches	String	No
`online`	Return online/offline workloads using this filter	Boolean	No
`os_id`	Operating System of workload(s) to return. Supports partial matches	String	No
`policy_health`	Policy of health of workload(s) to return. Valid values: active, warning, error, suspended	String	No
`security_policy_sync_state`	Advanced search option for workload based on policy sync state	String	No
`security_policy_update_mode`	Advanced search option for workload based on security policy update mode	String	No
`soft_deleted`	DEPRECATED WITH NO REPLACEMENT: Only soft-deleted workloads	Boolean	No
`ven`	URI of the VEN to filter by. From release 22.3.0,in addition to providing the VENs HREF, it is required to give its `hostname`, `name`, `ven_type`, and `status`. The VEN properties are now clearly displayed, without a need to use expanded representations. The `ven_type` property is introduced through the reference to a common schema `ven_type.schema.json`: `{ "properties": { "ven_type": {`	String	No

Parameter	Description	Type	Required

	<pre>"$ref": "../common/ven_type.schema.json"
	}</pre>		

visibility_level	Filter by visibility level		No
vulnerability_summary.vulnerability_exposure_score[gte]	Greater than or equal to value for vulnerability_exposure_score	Integer	No
vulnerability_summary.vulnerability_exposure_score[lte]	Less than or equal to value for vulnerability_exposure_score	Integer	No

Properties for GET

Property	Description	Type	Required
created_at	The time (rfc3339 timestamp) at which this workload was created	String date/time	Yes
data_center	The name of the data center where the workload is being hosted.		
data_center_zone	The zone inside the data center hosting the workload.	String	No
deleted_at	This workload has been deleted	date/time	
deleted_by	HREF	String	
labels	Labels that are attached to the workload: href, key, and value	Array.	No
name	Short, friendly name of the workload.	String	Yes
os_detail	Additional descriptive information about the workload OS	String	No
os_id	Unique OS identifier given by Illumio to the workload.	String	No
online	Indicates whether the workload is online and can communicate with the PCE.	Boolean.	No
public_ip	The public IP address of the workload.	String Null	No

Property	Description	Type	Required
services	This field contains the following data: • uptime_seconds • created_at • open_service_ports: with the following data: protocol, address, port, process_name, user, package, win_service_name		
service_provider	Name of the service provider that is hosting the workload.	String	No
updated_at	The time (rfc3339 timestamp) at which this workload was last updated	String date/time	Yes
vulnerabilities_summary	Reference to common/vulnerability_summary.schema.json		
detected_vulnerabilities	Reference to common/workloads_detected_vulnerabilities.schema.json		
agent	DEPRECATED AND REPLACED (USE 'ven' INSTEAD). Information about the agent that manages this workload.		
ven	This section of the response returns the following data: • href • hostname • name • status		
container_cluster	Reference to common/compact_container_cluster.schema.json		
ike_authentication_certificate	IKE authentication certificate for certificate-based Secure Connect and Machine Auth connections		

Vulnerability Computation State

The new field `vulnerability_computation_state` is added to `vulnerability_summary` for all APIs that return the namespace. It defines three computation states:

- `not_applicable` (N/A) indicates that the vulnerability exposure score cannot be calculated and happens in the following cases:
  - Unmanaged workloads
  - Idle workloads
  - Vulnerabilities that have no port associated with them.
- `syncing`: For managed workloads, when the vulnerability exposure score hasn't been calculated yet and the value is not available.
- `in_sync`: For managed workloads, when the workload with the VES value is calculated and available.

The following APIs have been updated to return `vulnerability_computation_state`:

- `workloads`(get collection) API
- `workloads/detailed_vulnerability`
- `workloads` (get instance)
- `workloads/:uuid/detected_vulnerabilities`
- `aggregated_detected_vulnerabilities`

Example of Computation States:

syncing: Workload is in syncing state (VES is calculable but hasn't been calculated yet):

```
"vulnerability_summary": {
       "num_vulnerabilities": 30,
       "max_vulnerability_score": 88,
       "vulnerability_score": 1248,
       "vulnerable_port_exposure": null,
       "vulnerable_port_wide_exposure": {
              "any": null,
              "ip_list": null
       },
       "vulnerability_exposure_score": null,
       "vulnerability_computation_state": "syncing"
},
```

Vulnerability Exposure Score (VES) Filters

The workloads GET collection API include query parameters to filter returned workloads based on their Vulnerability Exposure Score .

These vulnerability filters are considered to be experimental and might be changed in the future.

Specify these parameters to get all the workloads that have a specific score.

> NOTE:
> To use these new query parameters, you must also include the query parameter `representation=workload_labels_vulnerabilities`; otherwise, the PCE won't perform any vulnerability calculations.

Some examples for using the filters are:

```
GET api/v1/orgs/:xorg_id/workloads?representation=workload_labels_
vulnerabilities&vulnerability_summary.vulnerability_exposure_score%5Blte%5D=50
```

```
GET api/v1/orgs/:xorg_id/workloads?representation=workload_labels_
vulnerabilities&vulnerability_summary.vulnerability_exposure_
score%5Bgte%5D=50&vulnerability_summary.vulnerability_exposure_score%5Blte%5D=999
```

Update Workload Information

This API allows you to update information about a workload. To make this call, you need the URI of the workload you want to update, which is returned in the form of an HREF path when you get either a single or a collection of workloads in an organization.

URI to Update an Individual Workload's Information

```
PUT [api_version][workload_href]
```

Example Payload

This example shows what the JSON payload looks like for changing the policy state (called `mode` in the API) of a workload from its current state to enforced.

```
{"agent":{"config":{"mode":"enforced","log_traffic":true}}}
```

Curl Command to Update a Workload

A workload state can be build, test, or enforced. This example shows how to use curl to update a workload policy state from its current state to enforced.

This example assumes that you want to update the state of a single workload in an organization. You can obtain an organization ID when you use the Users API to log in a user to Illumio.

```
curl -i -X PUT https://pce.my-company.com/api/v2/orgs/3/workloads/043902c883d133fa
-H "Content-Type:application/json" -u $KEY:$TOKEN -d '{"agent":{"config":
{"mode":"enforced","log_traffic":true}}}'
```

Mark Workload as Suspended

You can use this API to mark a workload VEN as either suspended or unsuspended.

URI to Mark a Workload VEN as Suspended or Unsuspended

```
PUT [api_version][workload_href]
```

Example Payload

This example shows what the JSON payload looks like for marking a workload VEN as suspended, with the status property for the `agent` (the VEN) set to `suspended`.

To mark a workload VEN as unsuspended, use the same JSON body but replace `suspend` with `unsuspend`.

```
{
  "agent": {
    "status": {
      "status": "suspended"
    }
  }
}
```

Curl Command to Mark Workload as Suspended

This example shows you how to use curl to mark a workload VEN as suspended.

This example assumes that you want to mark a single workload VEN as suspended. You can obtain an organization ID when you use the Users API to log in a user to Illumio.

```
curl -i -X PUT https://pce.my-company.com/api/v2/orgs/3/workloads/043902c883d133 -
H "Content-Type:application/json" -u $KEY:$TOKEN -d '{"agent":{"status":
{"status":"suspended"}}}'
```

Create an Unmanaged Workload

The Unmanaged Workload API enables you to create a workload without installing the VEN on it. This API is commonly used if you are using Kerberos authentication between the VEN and the PCE.

URI to Create an Unmanaged Workload

```
POST [api_version][org_href]/workloads
```

Example Payload

For example, to create an unmanaged workload by providing a name, hostname, public IP address, and its Kerberos Service Principal Name, construct the JSON payload as follows:

```
{
     "name":"web_tier1",
     "hostname":"web_workload1.example.com",
     "public_ip":"10.10.10.10",
     "service_principal_name":"my_company-device-auth/web_workload1.example.com",
}
```

Curl Command to Create an Unmanaged Workload

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/4/workloads -H
"Content-Type: application/json" -u $KEY:$TOKEN -d '{"name":"web_tier1",
"hostname":"web_workload1.example.com","public_ip": "10.10.10.10","service_
principal_name":"my_company-device-auth/web_workload1.example.com"}'
```

Delete a Workload Record

If you have unpaired a workload, you can use this API to delete the workload's record from the PCE.

URI to Delete a Workload Record

```
DELETE [api_version][workload_href]
```

Unpair Workloads

This API allows you to unpair workloads from the PCE by uninstalling the Illumio VEN from each workload. You can unpair up to 1,000 workloads at a time.

Pairing a workload installs the Illumio VEN on a workload. Unpairing a workload uninstalls the VEN from the workload so that the workload no longer reports any information to the PCE, and the workload can no longer receive any policy information.

When you unpair workloads with this API, you can set the state for the workload's iptables (Linux) or WFP (Windows) configuration.

URI to Unpair a Workload

```
PUT [api_version][org_href]/workloads/unpair
```

> IMPORTANT:
> The endpoint `workloads/unpair` is DEPRECATED. Use /vens/unpair instead. See Unpairing and Suspending VENs for more details.

Request Parameters

Parameter	Description	Type	Required
`org_id`	Organization	Integer	Yes
`workloads`	Defines the list of workloads you want to unpair. You must specify at least one workload to unpair by defining the workload HREF. You can define up to 1,000 workloads to unpair with this API. Required property: `href`:URI of the workload to unpair.	Array	Yes
`ip_table_restore`	IMPORTANT: Use `/vens/unpair` and the parameter `firewall_restore` instead. This property allows you to determine the state of the workload iptables (Linux) or WFP (Windows) configuration after the workload is unpaired. Options include: • `saved`: Revert the iptables on the workload to the configuration before the VEN was installed. However, depending on how old the iptables or WFP configuration was on the workload, VEN removal could adversely	String	Yes

Parameter	Description	Type	Required
	impact security. • `default`: Apply the recommended policy, which is to uninstall the VEN and allow only port 22 SSH connections to the workload. Safest from a security viewpoint, but if this workload is running a production application, it could break because this workload will no longer allow any connections to it. • `disable`: Uninstall the VEN and leave all port connections on the workload open. This is the least safe from a security viewpoint. If iptables or WFP configuration or Illumio were the only security being used for this workload, the workload would be opened up to anyone and become vulnerable to attack.		

Example Payload for Unpairing Workloads

```
{
    "workloads": [
      {"href":"/orgs/7/workloads/XXXXXXXx-9611-44aa-ae06-fXXX8903db65"},
      {"href":"/orgs/7/workloads/xxxxxxxx-9611-xxxx-ae06-f7bXXX03db71"}
    ],
    "firewall_restore":"saved"
 }
```

Curl Command for Unpairing Workload

```
curl -i -X PUT https://pce.my-company.com/api/v2/orgs/3/workloads/unpair -H
"Content-Type:application/json" -u $KEY:$TOKEN -d '{"workloads": [{"href":
"/orgs/7/workloads/xxxxxxxx-9611-44aa-ae06-fXXX8903db65", "href":
"/orgs/7/workloads/xxxxxxxx-9611-xxxx-ae06-f7bXXX03db71"}], "firewall_restore":
"default"}'
```

Workloads Going Offline

Three new properties are now available to describe `LOG_INFO` level notification, `LOG_WARNING` level notification, and `LOG_ERR` level notification for workloads going offline.

When a VEN does not contact the PCE within a set time interval, it is marked as being offline. Previously, before that happened, a notification was created when the VEN was AWOL (missing) for 25% of the offline time.

These three new optional settings generate different levels of notifications at different intervals so the user can customize the timing and levels of notification.

They are described in the schema `resource_canonical_representations`:

Properties for Workloads Disconnection

Property	Description	Type
`workload_disconnected_timeout_second`	Disconnected timeout in seconds	Integer
`workload_goodbye_timeout_seconds`	Goodbye timeout in seconds	Integer
`workload_disconnect_notification_info`	Threshold in seconds for `LOG_INFO` level notification of a workload going offline	Integer
`workload_disconnect_notification_warning`	Threshold in seconds for `LOG_WARNING` level notification of a workload going offline	Integer
`workload_disconnect_notification_error`	Threshold in seconds for `LOG_ERR` level notification of a workload going offline	Integer

Workload Settings

This Public Stable API allows you to get network interface information from a workload, for either all interfaces on a workload or an individual interface. You can also configure (create) or delete an individual network interface configuration.

Workload Settings Methods

Functionality	HTTP	URI
Get agent timeout notifications	`GET`	`[api_version][org_href]/settings/workloads`
Update agent timeout notifications	`PUT`	`[api_version][org_href]/settings/workloads`

Endpoint Offline Timer

The Endpoint Offline Timer was introduced to overcome the 24-hour limitation that was hard-coded for endpoints heart beating.

If the endpoints did not heartbeat for 24 hours, they were marked as being offline and the endpoint timer was hard coded to 24 hours. However, the 24-hour limit was found to be limiting and was now adjusted to allow for endpoint mobility and usability.

The existing two APIs have been changed:

- `GET /api/v2/orgs/:xorg_id/settings/workloads`: Added properties to reflect the endpoint timeout values: disconnect.
- `PUT /api/v2/orgs/:xorg_id/settings/workloads`: Updated the endpoint offline, heartbeat, and disconnect and quarantine warning timeout values.

The three workload timeout setting fields have been updated:

Workload Timeout Setting Fields

Field	Description	Requiured
`workload_disconnected_timeout_seconds`	Timer setting triggered if the server or endpoint has not heart beaten to the PCE. Referencing the schema `settings_workload_detailed.schema.json`	Yes
`workload_goodbye_timeout_seconds`	Timer setting triggered if the server or endpoint operation is performed (stop, disable, ...). Referencing the schema `settings_workload_detailed.schema.json`	Yes
`workload_disconnected_notification_seconds`	Time period to wait with no heartbeat before a warning is emitted. Referencing the schema `settings_workload_notifications.schema.json`	Yes
`ven_uninstall_timeout_hours`	Defines the period (in hours) to wait before uninstalling a VEN. Referencing the schema `settings_workload.schema.json`	Yes

Schemas that Support the Endpoint Offline Timer

settings_workload_notifications

This schema file was updated and now has an additional property `ven_type` to support the ven type by the referenced timeout fields.

```
{
        "$schema": "http://json-schema.org/draft-04/schema#",
        "type": "array",
        "items": {
                "type": "object",
                "additionalPropertes": false,
                "required": [
                        "scope",
                        "warning"
                ],
                "properties": {
                        "scope": {
                        "$ref": "labels.schema.json"
                },
                "warning": {
                "description": "Workload disconnect warning timeout",
                "type": "integer",
                        "minimum": -1,
                        "maximum": 2147483647
                },
                "ven_type": {
                "description": "The ven type that this property is applicable to",
                "type": [
                        "string",
                        "null"
                ],
                "enum": [
                        "server",
                        "endpoint"
                ]
           }
        }
```

```
    },
        "uniqueitems": true
}
```

settings_workload_detailed

The new schema `settings_workload_detailed` is expanded from the previous schema `settings_workload` so that additional information about the `ven_type` was added.

```
{
        "$schema": "http://json-schema.org/draft-04/schema#",
        "type": "array",
        "items": {
                "type": "object",
                "additionalProperties": false,
                "required": [
                        "scope",
                        "value"
                ],
                "properties": {
                        "scope": {
                        "$ref": "labels.schema.json"
                },
                "value": {
                "description": "Property value associated with the scope",
                "type": "integer",
                        "minimum": -1,
                        "maximum": 2147483647
                },
                "ven_type": {
                "description": "The ven type that this property is applicable to",
                "type": [
                        "string",
                        "null"
                        ],
                "enum": [
                        "server",
                        "endpoint",
```

```
                    null
                ]
            }
        }
    },
  "uniqueItems": true
}
```

To ensure backend compatibility, the new field ven_typeis specified as optional. If it is missing in the request, the parameter is considered as being of a server type.

Examples

The example below represents the complete JSON string returned by the `GET /api/v2/orgs/:xorg_id/settings/workloads` request:

```
{
        "href": "/orgs/1/settings/workloads",
        "workload_disconnected_timeout_seconds": [
        {
                "scope": [],
                "value": 10800,
                "ven_type": "server"
        },
        {
                "scope": [],
                "value": 3600,
                "ven_type": "endpoint"
        },
        ],
        "workload_goodbye_timeout_seconds": [
        {
                "scope": [],
                "value": 12000,
                "ven_type": "server"
        },
        {
                "scope": [],
                "value": 7200,
```

```
                "ven_type": "endpoint"
        }
        ],
        "workload_disconnected_notification_seconds": [
        {
        {
                "scope": [],
                "info": 1800,
                "warning": 3600,
                "error": 5400,
                "ven_type": "server
        },
        {
                "scope": [],
                "info": 1801,
                "warning": 3602,
                "error": 5403,
                "ven_type": "server
        }
        }
        ],
        "ven_uninstall_timeout_hours": [
        {
                "scope": [],
                "value"=>300
                }
        ]
}
```

The following example shows how to set all four workload timeout setting properties via the `PUT /api/v2/orgs/:xorg_id/settings/workloads` request:

```
{
        "workload_disconnected_timeout_seconds": [
        {
                "scope": [],
                "value": 10800,
                "ven_type": "server"
        },
```

```
        {
                "scope": [],
                "value": 3600,
                "ven_type": "endpoint"
        },
        ],
        "workload_goodbye_timeout_seconds": [
        {
                "scope": [],
                "value": 12000,
                "ven_type": "server"
        },
        {
                "scope": [],
                "value": 7200,
                "ven_type": "endpoint"
        }
        ],
        "workload_disconnected_notification_seconds": [
        {
        {
                "scope": [],
                "info": 1800,
                "warning": 3600,
                "error": 5400,
                "ven_type": "server"
        },
        {
                "scope": [],
                "info": 1801,
                "warning": 3602,
                "error": 5403,
                "ven_type": "endpoint"
                }
             }
        ],
        "ven_uninstall_timeout_hours": [
        {
```

```
                "scope": [],
                "value"=>300
                }
        ]
}
```

Workload Interfaces

This Public Stable API allows you to get network interface information from a workload, either for all interfaces on a workload or an individual interface. You can also configure (create) or delete an individual network interface configuration.

API Methods

Functionality	HTTP	URI
Request the list of the workload_interfaces (outside of the workloads or vens scope). The `href` property in the API response is deprecated.	`GET`	`[api_version][workload_href]/interfaces`
Get an instance for the workload interface with the name. (DEPRECATED)	`GET`	`[api_version][workload_href]/interfaces/:name`
Directly creates a workload interface. The request payload was not changed, however the `href` field in the API response is deprecated.	`POST`	`[api_version][workload_href]/interfaces`
Delete the workload interface with the name. (DEPRECATED)	`DELETE`	`[api_version][workload_href]/interfaces/:name`
Set the network manually, update the automatic network detection. (DEPRECATED)	`PUT`	`[api_version][workload_href]/interfaces/:name/network`

Get Workload Network Interface

This API allows you to get information about one or all of the interfaces on a workload. You can retrieve workload interface information such as its connectivity (up, down, unknown), interface IP address, number of bits in the subnet, the IP address of the default gateway, and the associated network.

URI to Get a Collection of a Workload's Network Interfaces

```
GET [api_version][workload_href]/interfaces
```

Properties for GET

Property	Description	Type	Required
`name`	Interface name.	String	Yes
`address`	The IP address assigned to the interface.	String	Yes
`cidr_block`	The number of bits in the subnet (for example, /24 is 255.255.255.0).	Integer, Null	Yes
`default_gateway_address`	The default IP address of the default gateway.	String, Null	Yes
`link_state`	State of the interface connection, which is one of three values: • `up`: Interface is communicating. • `down`: Interface is not communicating. • `unknown`: State of the interface is unknown.	String, Null	Yes
`network_detection_mode`	Network Detection Mode	String, Null	Yes
`friendly_name`	User-friendly name given to the interface.	String, Null	Yes
`network`	Network that the interface belongs to	Object, Null	Yes
`href`	DEPRECATED WITH NO REPLACEMENT	String	No

Create Workload Network Interface

Directly creates a workload interface. The request payload was not changed, however the `href` field in the API response is deprecated.

URI to Create a Workload Network Interface Configuration

```
POST [api_version][workload_href]/interfaces
```

Properties for POST

Properties	Description	Type	Required
`name`	The short friendly name of the workload	String	Yes
`link_state`	State of the interface connection, which is one of three values: • `up`: Interface is communicating. • `down`: Interface is not communicating.	String	Yes

Properties	Description	Type	Required
	• `unknown`: State of the interface is unknown.		
`address`	The IP address assigned to the interface. Reference to common schema `ip_address_format_validation.schema.json`	String	No
`cidr_block`	The number of bits in the subnet (for example, /24 is 255.255.255.0).	Integer	No
`default_gateway_address`	The default IP address of the default gateway. Reference to common schema `ip_address_format_validation.schema.json`	String	No
`friendly_name`	User-friendly name given to the interface.	String	No
`href`	DEPRECATED WITH NO REPLACEMENT	String	No

Request Body

```
{
   "name": "eth0.public",
   "address": "192.0.2.0",
   "cidr_block": 32,
   "default_gateway_address": 255.255.255.0,
   "link_state": "up",
}
```

Curl Command Create Network Interface

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/2/workloads/xxxxxxxx-c4e9-44e7-8a31-e86acf6b276c/interfaces -H "Content-Type: application/json" -u $KEY:$TOKEN -d '{"name": "eth0.public", "address": "192.0.2.0","cidr_block": "32", "default_gateway_address": "255.255.255.0", "link_state": "up"}'
```

Examples

Request for all workload interfaces with a specific name

Request: `GET /api/v2/orgs/:org_id/workloads/:workload_id/interfaces?name=eth0.public`

Response includes the deprecated href field in the response:

```
[
   {
       "href": "/orgs/1/workloads/561bd65e-136c-4005-8aa2-
bdc8af1b3600/interfaces/eth0.public"
       "name": "eth0.public",
       "cidr_block": null,
       link_state": null,
       "network_detection_mode": null,
       "friendly_name": null,
       "network": {
              "href": "/orgs/1/networks/366ff4c1-ec60-49be-a05f-3a5ccab09c2f"
       },
       "loopback": false,
       "address": "1.1.1.1",
       "default_gateway_address": null
   },
   {
       "href": "/orgs/1/workloads/561bd65e-136c-4005-8aa2-
bdc8af1b3600/interfaces/eth0.public"
       "name": "eth0.public",
       "cidr_block": null,
       "link_state": null,
       "network_detection_mode": null,
       "friendly_name": null,
       "network": {
              "href": "/orgs/1/networks/366ff4c1-ec60-49be-a05f-3a5ccab09c2f"
       },
       "loopback": false,
       "address": "2.2.2.2",
       "default_gateway_address": null
}
```


API request/response creating new workload interface

Request: `POST /api/v2/orgs/:org_id/workloads/:workload_id/interfaces`


```
{
       "name": "eth1.private",
       "cidr_block": 32,
```

```
        "link_state": "up",
        "address": "99.99.99.7"
}
```

Response body (with the `href` deprecated):

```
{
        "href": "/orgs/1/workloads/561bd65e-136c-4005-8aa2/interfaces/eth1.private"
        "name": "eth1.private",
        "cidr_block": 32,
        "link_state": "up",
        "network_detection_mode": "single_private_brn",
        "friendly_name": null,
        "network": {
                "href": "/orgs/1/networks/5b25c11d-4e95-42d3-abd2-488506e48b02"
        },
        "loopback": false,
        "address": "99.99.99.7",
        "default_gateway_address": null
}
```

API request deleting multiple workload interfaces (bundle delete)

Request: `PUT /api/v2/orgs/:org_id/workloads/:workload_id/interfaces/delete`

Successful delete

Payload - all interfaces with the name `eth0.public` and only one interface with the name `eth-1.private` are deleted.

Response code - 200

```
{
        "name": "eth0.public"
},
{
        "name": "eth1.private",
        "address": "10.10.10.1"
}
```

Workload Bulk Operations

This Public Stable API supports “bulk” operations on collections of workloads. These operations create, update, or delete a collection of workloads using a single API call, instead of requiring separate API calls on individual workloads one at a time.

> IMPORTANT:
> Any tooling that parses the HTTP headers should be changed to allow case-insensitive header name matching in order to retain compatibility with future PCE releases. Refer to RFC 7230, section 3.2, "Header Fields," which states that field names should be case insensitive.

About Bulk Operations

When you use a bulk operations API to create a collection of workloads, the workload record is created in the PCE in the “unmanaged” state, which means that a VEN has not been installed on the workload and no policy can be applied to the workload. To apply a policy to unmanaged workloads, you can do one of the following:

- Pair the workloads using the pairing script located in the PCE web console
- Install and activate the VEN on the workload using the VEN control interface.

When you use this API to *update* a collection of workloads, those workloads can be either **managed** or **unmanaged**.

When you use this API to *delete* a collection of workloads, those workloads can only be **unmanaged**.

Workload Bulk Operations Methods

Functionality	HTTP	URI
Create a collection of workloads	`PUT`	`[api_version][org_href]/workloads/bulk_create`
Update a collection of workloads	`PUT`	`[api_version][org_href]/workloads/bulk_update`
Delete a collection of workloads	`PUT`	`[api_version][org_href]/workloads/bulk_delete`

Caveats for Workload Bulk Operations

> NOTE:
> Bulk operations are rate limited to 1,000 items per operation.

Bulk operations are rate limited to 1,000 items per operation. When you create, update, or delete a collection of workloads (also referred to as “bulk” operations), you can only execute one such

bulk operation at a time. This means you must wait for the first bulk operation to finish before executing a new one. If you execute a new bulk operation before the currently running operation has completed, the second operation will fail with an `HTTP 429` error.

Additionally, when you perform a bulk workload operation, any policy changes caused by the operation are applied to the affected VENs after the next VEN heartbeat to the PCE.

After a bulk operation completes, *all* of your PCE VEN connectivity states show as `Syncing` until the next VEN heartbeat. Only affected VENs receive a policy update. VENs that are not affected by policy changes transition from `Syncing` to `In Sync` after the VENs heartbeat. This process can take up to 5 minutes.

External Data Reference IDs for Workloads

External data references are a way to add your own internal identifiers to new workloads, independent of Illumio PCE-created workload HREFs. You can add these entities when you create a collection of workloads using the `bulk_create` method, or when you create an individual workload using the public API.

External data references are useful if you want to keep a set of PCE resources in sync with your internal representation of the resources, such as a configuration management database (CMDB) that holds the “source of truth” for your workloads. Once workloads are created with these identifiers added to their properties, you can run an asynchronous query to get all workloads through an offline job, which includes the external data references in the response.

The schema for creating and updating External data references includes two properties:

- `external_data_set`: Identifies the original data source of the resource.
- `external_data_reference`: A unique identifier within that data source.

These properties are UTF-8 strings with a maximum length of 255 characters each. The contents must form a unique composite key, meaning that both values of these properties are treated as a unique key. These two properties together are recognized as a unique key even if one of them is left blank or set to zero.

Create a Collection of Workloads

URI to Create a Collection of Workloads

```
PUT [api_version][org_href]/workloads/bulk_create
```

Request Body

When you create a collection of workloads, you need to pass a JSON object request body that describes the workload details.

Although this example illustrates the request body for a single workload, you can add as many workloads as you want.

For example:

```
{
  "name": "workload 0",
  "description": "workload desc 0",
  "service_principal_name": "spn 0",
  "hostname": "workload-0.example.com",
  "public_ip": "24.1.1.1",
  "external_data_set": "cmdb",
  "external_data_reference": "0",
  "interfaces": [
    {
      "name": "eth0",
      "link_state": "up",
      "address": "10.0.0.2",
      "cidr_block": 24,
      "ip_version": 4,
      "default_gateway_address": "10.0.0.1",
      "friendly_name": "wan"
    }
  ],
  "labels": [
    {
      "href": "/orgs/2/labels/1"
    },
    {
      "href": "/orgs/2/labels/9"
    }
  ],
  "service_provider": "service provider",
  "data_center": "central data center",
  "os_id": "os id 0",
  "os_detail": "os detail 0",
  "online": true,
  "agent": {
    "config": {
      "enforcement_mode": "full",
```

```
        "visibility_level": "flow_summary"
      }
    }
  }
```

Curl Command to Use Bulk Create

This curl example illustrates how to create two workloads using the `bulk_create` API.

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/2/workloads/bulk_create
-H "Content-Type:application/json" -u $KEY:$TOKEN -d '[{"name":"web_
app1","description":"workload desc 0","service_principal_name":"spn 0",
"hostname":"workload-0.example.com","public_ip":"24.1.0.1","external_data_
set":"cmdb", "external_data_reference":"0","interfaces":[{"name":"eth0","link_
state":"up","address":"10.0.0.2", "cidr_block":24,"ip_version":4,"default_gateway_
address":"10.0.0.1","friendly_name":"wan"}], "labels":
[{"href":"/orgs/2/labels/1"},{"href":"/orgs/2/labels/9"}],"service_provider":
"service provider","data_center":"central data center","os_id":"os id 0","os_
detail":"os detail 0", "online":true,"agent":{"config":{"enforcement_
mode":"visibility_only", "visibility_level":"flow_summary"}}}]
```

Update Collection of Workloads

This method allows you to update information about a collection of workloads. To update workload information, construct the JSON object for each workload exactly as you would for modifying one workload, but modify the properties as needed.

URI to Update a Collection of Workloads

```
PUT [api_version][org_href]/workloads/bulk_update
```

Curl Command to Bulk Update Workloads

This example shows how to update two workloads with the Bulk Update API.

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/2/workloads/bulk_update
-H "Content-Type:application/json" -u $KEY:$TOKEN -d '[{"name":"web_
app1","description":"workload desc 0","service_principal_name":”spn
0","hostname":"workload-0.example.com","public_ip":"24.1.2.1","external_data_
set":"cmdb","external_data_reference":"0",”interfaces":[{"name":"eth0","link_
```

```
state":"up","address":"10.0.0.2","cidr_block":24,“ip_version":4,"default_gateway_
address":"10.0.0.1","friendly_name":"wan”}],”labels":[{"href":"/orgs/2/labels/1"},
{"href":"/orgs/2/labels/9”}],”service_provider":"service provider","data_
center":"central data center","os_id":"os id 0",”os_detail":"os detail
0","online":true,"agent":{"config":{"enforcement_mode":"visibility_
only","visibility_level":"flow_summary"}}},{“name":"web_app2
0","description":"workload desc 0","service_principal_name":”spn
0","hostname":"workload-0.example.com","public_ip":"24.1.3.1",”external_data_
set":"cmdb","external_data_reference":"0",”interfaces":[{"name":"eth0","link_
state":"up","address":"10.0.0.2","cidr_block":24,”ip_version":4,"default_gateway_
address":"10.0.0.1","friendly_name":"wan"}],”labels":[{"href":"/orgs/2/labels/1"},
{"href":"/orgs/2/labels/9"}],”service_provider":"service provider","data_
center":"central data center","os_id":"os id 0",”os_detail":"os detail
0","online":true,"agent":{"config":{"enforcement_mode":"full", ”visibility_
level":"flow_summary"}}}]'
```

Delete a Collection of Workloads

You can use this Public Experimental API to delete a collection of unmanaged workloads.

When you call this method, you identify each unmanaged workload to delete by its HREF. For example:

```
/orgs/7/workloads/XXXXXXX-9611-44aa-ae06-fXXX8903db65
```

If an unmanaged workload has the following two properties:

- `external_data_set=cmdb`
- `external_data_reference=25`

you can construct the HREF as a query parameter that matches the values of these two properties as they are defined on the target workload. For example:

```
/orgs/1/workloads?external_data_set=cmdb&external_data_reference=25
```

> NOTE:
> Both query parameters must match for the exact same parameters on the workload for the delete operation to succeed.

URI to Delete a Collection of Workloads

```
PUT [api_version][org_href]/workloads/bulk_delete
```

Request Properties

Property	Description	Type	Required
`href`	The HREF of a specific workload or unmanaged workload using the `external_data_set` and `external_data_reference` query parameters.	String	Yes

Request Body

```
[
 {"href": "/orgs/1/workloads/92f4a252-68d1-40ef-8cf0-b46e4ec551r"},
 {"href": "/orgs/1/workloads/92f4a252-68d1-40ef-8cf0-b46e4ecd642ix"},
 {"href": "/orgs/1/workloads?external_data_set=cmdb&external_data_reference=25"},
 {href": "/orgs/1/workloads/92f4a252-74d1-40ef-5af0-b46a4acd333dt"}
]
```

Curl Command to Delete Collection of Workloads

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/3/workloads/bulk_delete
-H "Accept: application/json" -u $KEY:$TOKEN '[{ "href":
"/orgs/1/workloads/92f4a252-68d1-40ef-8cf0-b46e4ec551rse" },{"href":
"/orgs/1/workloads/92f4a252-68d1-40ef-8cf0-b46e4ecd642ix" }, {"href":
"/orgs/1/workloads/92f4a252-68d1-40ef-8cf0-b46e4ecd5421q" }, {"href":
"/orgs/1/workloads/92f4a252-68d1-40ef-8cf0-b46e4ecd214dt" }, {"href":
"/orgs/1/workloads/c20efa40-c615-4fa7-b8a1-badbadbadbad" },]'
```

Response

A successful response returns an HTTP 200 message and an array of status objects, one for each workload and each workload that failed to be deleted as requested. If all unmanaged workloads are successfully deleted, an empty array is returned.

For example, two errors are shown–one where the operation was not allowed (due to lack of permissions) and one where the workload did not exist.

```
[
  {
```

```
    "href": "/orgs/1/workloads/c20efa40-c615-4fa7-b8a1-c3af4d34c5f5",
    "errors": [{"token": "method_not_allowed_error", "message": "Not allowed"}]
  },
  {
    "href": "/orgs/1/workloads/c20efa40-c615-4fa7-b8a1-badbadbadbad",
    "errors": [{"token": "not_found_error", "message": "Not found"}]
  }
]
```

Bulk Import using a CSV File

workloads/bulk_import

This new API is used to update workloads using a CSV file, and the only allowed input type is 'text/csv'.

We recommend users to export a CSV file from the workloads page before they use this import function, so that they can just modify the CSV file they exported with the labels they would like to assign to the workloads.

- `PUT /api/v2/orgs/:xorg_id/workloads/bulk_import?delete_token`
  If the value in the CSVfile for the `label_dimension` is the same as the delete token passed in the request, the label in that label dimension will be deleted for the workload. When users use CSV to update workload labels, they can pass in the delete token in the request to specify the labels to be deleted.
- `PUT /api/v2/orgs/:xorg_id/workloads/bulk_import?create_labels=true/false` (default is false)
  Provides an option in the CSV labels update to create new labels if they don't exist. If the option is `false`, rows with non-existent labels will be skipped entirely.
- `PUT /api/v2/orgs/:xorg_id/workloads/bulk_import?dry_run=true/false` (default is false)
  If users set this parameter to be `true`, the API will only return the potential changes and error tokens without making actual changes to the workloads.

Agents on Workloads

This Public Experimental API gets and updates an agent on a workload.

> WARNING:
> The term Agent has been deprecated and replaced by VEN. It will be removed in future releases.
>
> Instead of this deprecated API, see the information in VEN Operations.

Agents API Methods

Functionality	HTTP	URI
Get an individual agent	GET	[api_version][agent_href]
Update an agent	PUT	[api_version][agent_href]/update
Get all agents	GET	[api_version]/agents

Get an Individual Agent

This API returns an agent record.

Curl Command to Get an Agent

To obtain the agent ID, make a call to a managed workload (a workload associated with a VEN) `GET /workloads/workload_id`. To get all managed workloads, make a call to `GET /workloads?managed=true`.

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/7/agents/12345 -H
"Accept: application/json" -u $KEY:$TOKEN
```

Path Parameters

Parameter	Description	Type
org_id	The ID of the organization.	String
agent_id	The agent ID. To obtain the agent ID, make a call to a managed workload (a workload associated with a VEN) `GET /workloads/workload_id`. To get all managed workloads, make a call to `GET /workloads?managed=true`. NOTE: The `agent_id` is the integer at the end of the agent href. **Example:** "href":"/orgs/7/agents/12345"	Integer

Example Response Body


```
{
  "href": "/orgs/7/agents/12345",
  "org_id": 1,
  "name": null,
  "description": null,
  "uid": "xxxxx-xxxxx--8ef6-c0fb3fea3cf5",
  "last_heartbeat_on": "2018-09-08T01:12:00.999Z",
  "uptime_seconds": 9944537,
  "hostname": "my-hostname",
  "agent_version": "16.9.0",
  "public_ip": "xx.xxx.xxx.xxx",
  "status": "active",
  "online": true,
  "fw_rules_generation_actual": 1,
  "service_provider": "my-datacenter-provider.com",
  "data_center": "123.my-datacenter.com",
  "data_center_zone": "3",
  "instance_id": "pi@xxxxxxxx93816",
  "fw_config_current?": true,
  "managed_since": "2018-06-13T03:23:00.000Z",
  "os_id": "ubuntu-x86_64-xenial",
  "os_detail": "4.4.0-97-generic #120-Ubuntu SMP Tue Sep 19 17:28:18 UTC 2017
(Ubuntu 16.04.1 LTS)",
  "visibility_level": "flow_summary",
  "created_at": "2018-06-13T03:23:00.000Z",
  "updated_at": "2018-09-08T01:02:00.000Z",
  "created_by": {
    "href": "/orgs/7/agents/12345"
  },
  "updated_by": null,
  "service_report": {
    "uptime_seconds": 9887714,
    "created_at": "2018-09-07T21:05:44.825Z",
    "open_service_ports": [
      {
        "protocol": 17,
        "address": "0.0.0.0",
```

```
        "port": 67,
        "process_name": "dhcpd",
        "user": "root",
        "package": null,
        "win_service_name": null
      },
      {
        "protocol": 6,
        "address": "0.0.0.0",
        "port": 53,
        "process_name": "bind",
        "user": "root",
        "package": null,
        "win_service_name": null
      },
      {
        "protocol": 17,
        "address": "0.0.0.0",
        "port": 123,
        "process_name": "ntpd",
        "user": "root",
        "package": null,
        "win_service_name": null
      }
    ]
  },
  "labels": [
    {
      "href": "/orgs/7/labels/2010"
    },
    {
      "href": "/orgs/7/labels/300"
    },
    {
      "href": "/orgs/7/labels/2000"
    },
    {
      "href": "/orgs/7/labels/260"
```

```
    }
  ],
  "mode": "illuminated",
  "target_pce_fqdn": null,
  "active_pce_fqdn": null
}
```

Get all Agents

This API fetches all agents. This API was DEPRECATED and replaced (use `/orgs/:xorg_id/vens/:ven_uuid` instead).

The agents API is deprecated, and the VEN `href` should be used to identify and manipulate this resource.

Update an Agent

This API updates the agent `target_pce_fqdn` parameter to point to the FQDN of a Supercluster or a PCE that is a member of a Supercluster.

URI to Update an Agent "target_pce_fqdn" Parameter

```
PUT [api_version][agent_href]/update
```

Curl Command to Update an Agent

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/7/agents/12345 -H
"Content-Type: application/json" -u $KEY:$TOKEN -d '{"target_pce_fqdn":
"my.supercluster.pce.my-company.com"}'
```

Request Body

Property	Description	Type
`target_pce_fqdn`	FQDN of the target Supercluster or the target PCE that is a member of a Supercluster.	String

Example Request Body

```
{
      "target_pce_fqdn": "my.supercluster.pce.my-company.com"
}
```

Blocked Traffic to and from Workloads

This Public Experimental API was used to identify unauthorized traffic to or from workloads. It would get a list of blocked or potentially blocked traffic between workloads and other entities managed by the PCE.

> WARNING:
> In the 19.1.0 release, blocked traffic was marked for deprecation and is now turned off by default.
> The functionality previously provided by blocked traffic API is available in Explorer.
> The Blocked Traffic page continues to work, and when you configure the Explorer feature, this page uses the Explorer API to get the data from PCE.

Pairing Profiles and Pairing Keys

The Public Stable API for pairing profiles gets, creates, updates, and deletes pairing profiles.

The Public Stable API for pairing keys creates a pairing key to use for pairing workloads.

About Pairing Profiles and Keys

Pairing Profiles apply specific properties to workloads as they pair with the PCE, such as labels and the workload policy state.

When you configure a pairing profile, the pairing script contains a unique pairing key at the end of the script (activation-code) that identifies the VEN securely so it can authenticate with the PCE. You can configure a pairing key for one-time use or more, and you can also set time and use limits.

The Pairing Key API can generate a new pairing key from a specified pairing profile.

Pairing Profile Methods

Functionality	HTTP	URI
Get a collection of pairing profiles	`GET`	`[api_version][org_href]/pairing_profiles`
Get the specified pairing profile	`GET`	`[api_version][org_href]/pairing_profile_href`
Create an individual pairing profile	`POST`	`[api_version][org_href]/pairing_profiles`
Update an individual pairing profile	`PUT`	`[api_version][pairing_profile_href]`
Delete an individual pairing profile	`DELETE`	`[api_version][pairing_profile_href]`

Get Pairing Profiles

This method allows you to get a collection of all pairing profiles in your organization or just an individual pairing profile.

By default, the maximum number returned on a GET collection of pairing profiles is 500. For more than 500 pairing profiles, use an Asynchronous GET Collection.

URI to Get a Collection of Pairing Profiles

```
GET [api_version][org_href]/pairing_profiles
```

Curl Command to Get Collection of Pairing Profiles

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/1/pairing_
profiles -H 'Accept: application/json' -u $KEY:'TOKEN'
```

Parameters for Pairing Profiles

Parameter	Description	Type
`org_id`	Organization ID	
`agent_software_release`	The agent software release for pairing profiles.	
`description`	The long description of the pairing profile. Supports partial matches.	String
`external_data_set`	The data source from which the resource originates. For example, if the pairing profile information is stored in an external database.	String NULL
`external_data_reference`	External data reference identifier	String Null
`name`	The short friendly name of the pairing profile. Supports partial matches.	String
`labels[]`	Return only pairing profiles that have all of these labels specified as part of the pairing profile. labels are structured in JSON as a list of lists of label HREFs.	Array
`ven_type`	Specifies the pairing profile by the VEN type: server, endpoint, or specified_during_activation	String

Properties for Pairing Profiles

All properties are required

Property	Description	Type
`name`	The short friendly name of the Pairing Profile	String
`description`	The long description of the pairing profile. Supports partial matches.	String
`mode`	Reference to the common schema `legacy_workload_modes.schema.json`	String
`enabled`	The enabled flag of the pairing profile	Boolean
`total_use_count`	The number of times the Pairing Profile has been used	Integer
`allowed_uses_per_key`	The number of times the pairing profile can be used. Minimum: 1	String
`key_lifespan`	Number of seconds pairing profile keys will be valid for. Minimum: 1	Integer (min 1)
`last_pairing_at`	Timestamp when this pairing profile was last used for pairing a workload	String NULL
`created_at`	Timestamp when this pairing profile was first created	String date-time
`updated_at`	Timestamp when this pairing profile was last updated	String date-time
`created_by`	User who originally created this pairing profile Reference by common schema `href_object.s-chema.json`	Object String
`updated_by`	User who last updated this pairing_profile Reference by common schema `href_object.s-chema.json`	
`last_pairing_key_gen-erated_at`	Timestamp of when the last pairing key was generated • Null • Refenced by the common shema `href_object.s-chema.json`	String, Null date/time
`last_pairing_key_gen-erated_by`	User who generated the last pairing key	
`is_default`	Flag indicating this is default auto-created Pairing Pro-file	Boolean
`labels`	Reference to `labels.schema.json`	

Property	Description	Type
env_label_lock	Flag that controls whether env label can be overridden from pairing script	Boolean
loc_label_lock	Flag that controls whether loc label can be overridden from pairing script	Boolean
app_label_lock	Flag that controls whether app label can be overridden from pairing script	Boolean
enforcement_mode_lock	Flag that controls whether enforcement mode can be overridden from pairing script	Boolean
mode_lock	DEPRECATED AND REPLACED | (USE /enforcement_mode_lock INSTEAD) Flag that controls whether mode can be overridden from the pairing script.	Boolean
log_traffic	DEPRECATED AND REMOVED. Alerting status	Boolean
log_traffic_lock	DEPRECATED AND REMOVED. Flag that controls whether log_traffic can be overridden from pairing script	Boolean
visibility_level_lock	Flag that controls whether visibility_level can be overridden from pairing script	Boolean
status_lock	Flag that controls whether status can be overridden from pairing script	Boolean
external_data_set	External data set identifier	String, Null
external_data_reference	External data reference identifier	String, Null
agent_software_release	Agent software release associated with this paring profile	String, Null
ven_type	Referenced to common/pairing_profile_ven_type.schema.json	

Examples of query parameters for filtering pairing profiles:

Filter by Name:

```
/api/v2/orgs/1/pairing_profiles?name=prod_app
```

Filter by Description:

```
/api/v2/orgs/1/pairing_profiles?description=”some description string”
```

Filter by software release:

```
/api/v2/orgs/1/pairing_profiles?agent_software_release=”xx.x.x”
```

Response Body

Response includes generated pairing keys


```
{
     "href": "/orgs/4002/pairing_profiles/4101",
     "name": "org 3 pp 1",
     "description": "org 3 pp 1",
     "total_use_count": 0,
     "enabled": true,
     "is_default": false,
     "created_at": "2022-01-21T00:44:16.863Z",
     "updated_at": "2022-01-21T00:44:16.863Z",
     "created_by": {"href"=>"/users/0"},
     "updated_by": {"href"=>"/users/0"},
     "mode": "illuminated",
     "enforcement_mode": "visibility_only",
     "key_lifespan": "unlimited",
     "allowed_uses_per_key": "unlimited",
     "last_pairing_at": nil,
     "last_pairing_key_generated_at": "2022-01-21T00:49:13.841Z",
     "last_pairing_key_generated_by": {"href"=>"/users/6"},
     "labels": [{"href"=>"/orgs/4002/labels/4104"}],
     "env_label_lock": true,
     "loc_label_lock": true,
     "role_label_lock": true,
     "app_label_lock": true,
     "mode_lock": true,
     "enforcement_mode_lock": true,
     "log_traffic": false,
     "log_traffic_lock": true,
     "visibility_level": "flow_summary",
      "visibility_level_lock": true,
```

```
      "agent_software_release": "Default (19.3.0)",
      "caps": ["write", "generate_pairing_key"]
}
```

Create a Pairing Profile

This method creates an individual pairing profile. The only required parameter for POST method is `enabled`, others are optional.

URI to Create a Pairing Profile

```
POST [api_version][org_href]/pairing_profiles
```

Example Request Body

```
{
  "href": "/orgs/2/pairing_profiles/12375",
  "name": "Limited Pairing",
  "description": "",
  "total_use_count": 0,
  "enabled": true,
  "is_default": false,
  "created_at": "2015-11-01T01:20:06.135Z",
  "updated_at": "2015-11-01T01:20:06.135Z",
  "created_by": {
    "href": "/users/18"
  },
  "updated_by": {
    "href": "/users/18"
  },
  "enforcement_mode": "visibility_only",
  "key_lifespan": "unlimited",
  "allowed_uses_per_key": "unlimited",
  "last_pairing_at": null,
  "labels": [
    {
      "href": "/orgs/2/labels/6"
    },
```

```
    {
      "href": "/orgs/2/labels/14"
    },
    {
      "href": "/orgs/2/labels/8"
    },
    {
      "href": "/orgs/2/labels/12"
    }
  ],
  "env_label_lock": false,
  "loc_label_lock": false,
  "role_label_lock": false,
  "app_label_lock": false,
  "mode_lock": true,
  "visibility_level": "flow_summary",
  "visibility_level_lock": true
}
```


Curl Command to Create Pairing Profile


```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/2/pairing_profiles -H
"Content-Type:application/json" -u $KEY:'TOKEN'-d '{"href":"/orgs/2/pairing_
profiles/12375","name":"Limited Pairing","description":"","total_use_
count":0,"enabled":true,"is_default":false,"created_at":"2015-11-
01T01:20:06.135Z","updated_at":"2015-11-01T01:20:06.135Z","created_by":
{"href":"/users/18"},"updated_by":{"href":"/users/18"},"enforcement_
mode":"visibility_only","key_lifespan":"unlimited","allowed_uses_per_
key":"unlimited","last_pairing_at":null,"labels":[{"href":"/orgs/2/labels/6"},
"href":"/orgs/2/labels/14"},"href":"/orgs/2/labels/8"},"href":"/orgs/2/labels/12"}
],"env_label_lock":false,"loc_label_lock":false,"role_label_lock":false,"app_
label_lock":false,"visibility_level":"flow_summary","visibility_level_lock":true}'
```


Update a Pairing Profile

To update a pairing profile, specify its HREF, which can be obtained from getting a collection of pairing profiles.

URI to Update a Pairing Profile

```
PUT [api_version][pairing_profile_href]
```

Curl Command to Update Pairing Profile

```
curl -i -X PUT https://pce.my-company.com:8443/api/v2/orgs/2/pairing_profiles -H
"Accept: application/json" -u $KEY:'TOKEN'-d '{"href":"/orgs/2/pairing_
profiles/12375","name":"Limited Pairing","description":"","total_use_
count":0,"enabled":true,"is_default":false,"created_at":"2015-11-
01T01:20:06.135Z","updated_at":"2015-11-01T01:20:06.135Z","created_by":
{"href":"/users/18"},"updated_by":{"href":"/users/18"},"enforcement_
mode":"visibility_only","key_lifespan":"unlimited","allowed_uses_per_key":"one_
use","last_pairing_at":null,"labels":[{"href":"/orgs/2/labels/6"},
{"href":"/orgs/2/labels/14"},{"href":"/orgs/2/labels/8"},
{"href":"/orgs/2/labels/12"}],"env_label_lock":false,"loc_label_lock":false,"role_
label_lock":false,"app_label_lock":false,"visibility_level":"flow_
summary","visibility_level_lock":true}'
```

Delete a Pairing Profile

To delete an individual pairing profile, specify its HREF that you can obtain from a collection of pairing profiles.

URI to Delete a Pairing Profile

```
DELETE [api_version][pairing_profile_href]
```

Curl Command to Delete Pairing Profile

```
curl -i -X DELETE https://pce.my-company.com:8443/api/v2/orgs/2/pairing_
profiles/12375 -H "Accept: application/json" -u $KEY:'TOKEN'
```

Pairing Key API Method

Functionality	HTTP	URI
Create a pairing key	POST	[api_version][org_href]/pairing_profiles[pairing_profile_href]/pairing_key

Create a Pairing Key

To create a pairing key, you need a pairing profile HREF to pass as a parameter. You can obtain the pairing profile HREF from the pairing profile page in the PCE web console.

A pairing key is governed by the parameters configured in the pairing profile.

URI to Create a Pairing Key

Obtain the pairing key HREF from the response body returned by an API call to get a collection of pairing keys.

```
POST [api_version][pairing_key_href]/pairing_key
```

Request Body

The request body is an empty JSON object.

```
{}
```

Curl Command to Create Pairing Key

```
curl -i -X POST https://pce.my-company.com:8443/api/v2/orgs/3/pairing_
profiles/34/pairing_key -H 'Content-Type: application/json' -u $KEY:'TOKEN' -d "
{}"
```

VEN Operations

Overview of VEN Suspension

The VEN Update API (`PUT [api-version][ven-href]`) allows you to mark a VEN as either suspended or unsuspended in the PCE. It does not, however, actually suspend or unsuspend the VEN.

To suspend a VEN, use the `illumio-ven-ctl` command-line tool, which isolates a VEN on a workload so that you can troubleshoot issues and determine if the VEN is the cause of any anomalous behavior.

When you suspend a VEN, the VEN informs the PCE that it is in suspended mode.

However, if the PCE does not receive this notification, you must mark the VEN as "Suspended" in the PCE web console (select the VEN and click **Edit**), or you can use this API to mark the VEN as suspended.

When you don't mark the VEN as suspended in the PCE, after one hour, the PCE assumes that the workload is offline and removes it from the policy. When you mark the VEN as suspended, that VEN's workload is still included in the policy of other workloads.

When a VEN is suspended:

- The VEN still appears in the PCE in the VEN list page.
- The VEN's host cannot be unpaired.
- An organization event (`server_suspended`) is logged. This event is exportable to CEF/LEEF and has the severity of WARNING.
- Heartbeats or other communications are not expected, but when received, all communications are logged by the PCE.
- If the PCE is rebooted, the VEN remains suspended.
- Any custom iptables rules are removed, and you need to reconfigure them manually.

When a VEN is unsuspended:

- The PCE is informed that the VEN is no longer suspended and is able to receive policy from the PCE. If existing rules affect the unsuspended workload, the PCE reprograms those rules.
- An organization event (`server_unsuspended`) is logged. This event is exportable to CEF/LEEF and has the severity of WARNING.
- The workload reverts to the policy state it had before it was suspended.
- Custom iptables rules are configured back into the iptables.

VEN upgrade APIs allow you to specify an array of VEN hrefs to upgrade to a specific version instead of a list of agent ID's.

Rules when validating with the VEN upgrade APIs are as follows:

- No downgrades are allowed
- Users cannot upgrade to a VEN version higher than the PCE version
- No AIX, Solaris, or C-VEN are allowed
- Users can only upgrade VENs paired to the same region
- Only workload managers can upgrade VENs, and they can only upgrade VENs within their scope.

VEN API Methods

In addition to the page in the PCE web console that lists all VENs and shows the details of a single VEN, there is a Public Experimental API for getting VEN collections and VEN instances.

Other new APIs support VEN filtering in the PCE web console, and update and unpair VENs.

VEN Methods	HTTP	URI
Get the collection of all VENs (The `href` property in each interface in ven interfaces array is dropped from the response.)	`GET`	`[api_version][org_href]/vens/`
Get details on a VEN instance (The `href` property in each interface in ven interfaces array is dropped from the response)	`GET`	`[api_version][org_href]/vens/ven_id`
Use to get the default release without iterating through the whole collection.	`GET`	`[api_version[org_href]/software/vens/default`
Support VEN filtering in the PCE web console	`GET`	`[api_version][org_href]/vens/autocomplete` `[api_version][org_href]/vens/-facets`
To set the `target_pce_fqdn` on a VEN	`PUT`	`[api_version][org_href]/vens/ven_id`
Update a VEN	`PUT`	`[api_version][org_href]/vens/update`
Upgrade a VEN. This API accepts a list of `hrefs` instead of `agent_ids`. The upgrade endpoint falls under `/vens/resource` instead of the `/software/resource`.	`PUT`	`[api_version][org_href]vens/upgrade`
Lists the VEN releases available to the org, one per VEN version, along with some metadata such as whether it is the default version, whether that release supports servers and/or endpoints, and so on. The list of images is longer than the list of releases, and multiple images belong to the same release version.	`GET`	`[api_version]/software/ven/releases`
Shows the full list of VEN images. There is one image for each Linux distribution we support (such as RHEL, Ubuntu), plus images for Windows and macOS.	`GET`	`[api_version]/software/ven/releases-images`
Unpair a VEN: trigger the unpairing of one or more VENs. **NOTE**: This endpoint replaces `/workloads/unpair`, which is deprecated.	`PUT`	`[api_version][org_href]/vens/unpair`

VEN Methods	HTTP	URI
Provided so that users can set the default version for VEN pairing.	PUT	[api_version][org_href]/software/vens/default

VEN Parameters

Parameter	Description	Type	Required
org_id	Organization ID	Integer	Yes
activation_type	The method in which the VEN was activated	String	No
active_pce_fqdn	FQDN of the PCE	String	No
activation_recovery	Return VENs in or not in authentication recovery	Boolean	No
condition	A specific error condition to filter by	String	No
container_clusters	Array of container cluster URIs, encoded as a JSON string	Object	No
disconnected_before	Return VENs that have been disconnected since the given time	date/time	No
health	The overall health (condition) of the VEN	String	No
hostname	Hostname of VEN(s) to return. Supports partial matches.	String	No
ip_addressl	IP address of VEN(s) to return. Supports partial matches	Sring	No
last_goodbye_at	The time (rfc3339 timestamp) of the last goodbye from the VEN.	String, Null	
os_platform	OS platform of the host managed by the VEN	String, Null	
version	Software version of the VEN.	String	
status	The current status of the VEN. Options are: "active", "suspended", "uninstalled"	String	
activation_type	The method in which the VEN was activated. Options are: "pairing_key", "kerberos", "certificate"	String, Null	No
active_pce_fqdn	The FQDN of the PCE that the VEN last connected to	String, Null	No
target_pce_fqdn	cluster FQDN for target PCE	String, Null	
labels	Labels assigned to the host managed by the	Array	

Parameter	Description	Type	Required
	VEN.		
`interfaces`	Network interfaces of the host managed by the VEN.	Array	
`workloads`	The only required property is `HREF`, the others are optional: `name`, `managed`, `hostname`, `os_id`, `os_detail`, `labels`, `interfaces`, etc.	Array	
`description`	Description of VEN(s) to return. Supports partial matches	String, Null	
`last_heartbeat_at`	The last time (rfc3339 timestamp) a heartbeat was received from this VEN.	String, Null	
`status`	VEN Status: • "active" • "suspended"	String	
`ven_type`	The `ven_type` property is introduced through the reference to a common schema `ven_type.schema.json`:	String	No

Properties

Parameter	Description	Type	Required
`ven_type`	The type of the release marked as default: "server", "endpoint"	String	No
`default_release_ven_types`	The type of the release marked as default	String	
`name`	Friendly name for the VEN	String, Null	
`hostname`	The hostname of the host managed by the VEN	String, Null	Yes
`uid`	The unique ID of the host managed by the VEN	String, Null	
`os_id`	OS identifier of the host managed by the VEN	String, Null	
`os_detail`	Additional OS details from the host managed by the VEN	Sring, Null	

Parameter	Description	Type	Required
`os_platform`	OS platform of the host managed by the VEN	String, Null	
`version`	Software version of the VEN.	String	
`status`	The current status of the VEN. Options are: "active", "suspended", "uninstalled"	String	
`activation_type`	The method in which the VEN was activated. Options are: "pairing_key", "kerberos", "certificate"	String, Null	No
`active_pce_fqdn`	The FQDN of the PCE that the VEN last connected to	String, Null	No
`target_pce_fqdn`	cluster FQDN for target PCE	String, Null	
`labels`	Labels assigned to the host managed by the VEN.	Array	
`interfaces`	Network interfaces of the host managed by the VEN.	Array	
`workloads`	The only required property is `HREF`, the others are optional: `name`, `managed`, `hostname`, `os_id`, `os_detail`, `labels`, `interfaces`, etc. `managed`: `True` if the workload is managed, else `false`.	Array	
`container_clusters`	Array of container cluster URIs, encoded as a JSON string	Object	No
`secure_connect`	Issuer name match criteria for certificate used during establishing secure connections	Object, Null	
`last_heart-beat_at`	The last time (rfc3339 timestamp) a heartbeat was received from this VEN.	String, Null	
`last_goodbye_at`	The time (rfc3339 timestamp) of the last goodbye from the VEN.	String, Null	
`status`	VEN Status: • "active" • "suspended"	String	
`disconnected_before`	Return VENs that have been disconnected since the given time	date/time	
`health`	The overall health (condition) of the VEN	String	
`ip_address`	IP address of VEN(s) to return. Supports partial	String	

Parameter	Description	Type	Required
	matches		
`firewall_restore`	The strategy to use to restore the firewall state after the VEN is uninstalled. The strategy to use to restore the firewall state after the VEN is uninstalled: Options are: `saved`, `default`, and `disable`. The default is: `default`. Works with `vens_unpair_put`.	String	
`ven_id`	VEN ID (works with `GET /api/v2/orgs/{org_id}/vens/{ven_id}`)	String	
`vens`	VENs to unpair (works with `PUT /api/v2/orgs/{org_id}/vens/unpair`) Required property: `href`	Array	Yes
`secure_connect`	Property: `matching_issuer_name`. Issuer name match criteria for certificate used during establishing secure connections. `matching_issuer_name`: Issuer name match criteria for certificate used while establishing secure connections.	Object String	
`security_policy_applied_at`	Last reported time when policy was applied to the workload (UTC), only present in expanded representations.	`date-time`	
`security_policy_received_at`	Last reported time when policy was received by the workload (UTC), only present in expanded representations.	`date-time` Null	
`enforcement_mode`	Policy enforcement mode, only present in expanded representations. Options are: "idle", "visibility_only", "full", "selective"	String	
`visibility_level`	The amount of data the VEN collects and reports to the PCE from a workload in the enforced mode (policy state), so you can control resource demands on workloads. The higher levels of detail are useful for visualizing traffic flows in	String	

Parameter	Description	Type	Required
	the Illumination map inside the PCE web console. If this parameter is not set, then VEN visibility level is set to `flow_summary`. • `flow_summary`: ("High Detail" in the PCE web console) The VEN collects traffic connection details (source IP, destination IP, protocol, and source and destination port) for both allowed and blocked connections. This option creates traffic links in the Illumination map and is typically used during the building and testing phase of your security policy. • `flow_drops`: ("Less Detail" in the PCE web console.) The VEN only collects traffic connection details (source IP, destination IP, protocol, and source and destination port) for blocked connections. This option provides less detail for Illumination but demands fewer system resources from a workload and is typically used for policy enforcement. • `flow_off`: ("No Detail" in the PCE web console.) The VEN does not collect any details about traffic connections. This option provides no Illumination detail and demands the least amount of resources from workloads. This mode is useful when you are satisfied with the rules that have been created and do not need additional overhead from observing workload communication.		
`upgrade_pending`	Only return VENs with/without a pending upgrade	Boolean	No
`ven_type`	The `ven_type` property is introduced through the ref-	String	No

Parameter	Description	Type	Required
	erence to a common schema `ven_type.schema.json`:		
`upgrade_expires_at`	The time (rfc3339 timestamp) at which the PCE stops attempting VEN upgrade	String. Null	No
`upgrade_target_version`	The software release to upgrade to	String, Null	No
`upgrade_timeout_seconds`	Number of seconds during which the PCE tries to trigger the agent upgrade: "minimum": 900, "maximum": 15552000	Integer	

Software VEN Releases

release_ven_types

```
{
        "$schema": "hp://json-schema.org/draft-04/schema#",
        "description": "Supported ven types in this release",
        "type": "array",
        "items": {
                "type": "string",
                        "enum": ["server", "endpoint"]
                }
}
```

The new common schema `release_ven_types` is introduced to show `ven_types` for each release and to filter releases by `ven_type`.

Previously, the `ven_type` was not stored for the release, and database records looked as follows:

Release	Distributiion
22.5.1	CentOS
22.5.1	MacOS
22.5.1	Windows

With the property `ven_type` added, the database records are expanded with an additional `ven_types` column:

Release	Distributiion	ven_types
22.5.1	CentOS	server + endpoint
22.5.1	MacOS	server + endpoint
22.5.1	Windows	server + endpoint

Note that in release 22.5.1 the code supports the type "server+endpont". However, Centos (Linux) supports a server-only VEN image, MasOS supports endpoint-only image, and Windows supports both server and endpoint:

Release	Distributiion	ven_types
22.5.1	CentOS	server
22.5.1	MacOS	endpoint
22.5.1	Windows	server + endpoint

When a user opens the list of release images via UI and looks for the type `server + endpoint`, only the Windows image will show up as a complete match.

To fix this issue, the `ven_type` is now based on release and distribution:

- All releases before 21.2.2 were just `server` (there was no endpoint)
- Any release with 22.3.x was `endpoint` (there was no server)
- Any other releases were `server + endpoint` , but instead of setting it to all the images (database records), the `ven_types` are set in a way that is specific for the Os.

GET VENs

To get a collection of VENs that have a specific label applied to them, take a label string that was returned when you got a collection of VENs, and then add a query parameter to GET all VENs with that specific label.

Curl Command to Get VENs with a Specific Label

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/vens?labels="
[[/orgs/2/labels/1642]]" -H "Accept: application/json" -u $KEY:$TOKEN
```

To restrict the type of VENs you want returned and set a limit on how many results you want returned, use the relevant query parameters. For example, you might want to get a collection of no more than 50 VENs running CentOS 6.3 that have an active status.

Curl Command to Get VENs using other Query Parameters

```
curl -i -X GET https://pce.my-company.com:8443/api/v2/orgs/2/vens?os_id=centos-x86_64-6.3&max_results=50&status=active -H "Accept: application/json"-u $KEY:$TOKEN
```

Unpairing and Suspending VENs

Instead of unpairing and suspending workloads, use the new VEN APIs to unpair and suspend VENs.

> NOTE:
> The endpoint `workloads/unpair` is DEPRECATED. Use `/vens/unpair` instead.

Curl Command for Unpairing VENs

```
curl -i -X PUT https://pce.my-company.com/api/v2/orgs/3/vens/unpair -H "Content -Type:application/json" -u $KEY:$TOKEN -d '{"vens": [{"href": "/orgs/7/vens/xxxxxxxx-9611-44aa-ae06-fXXX8903db65"}, {"href": "/orgs/7/vens/xxxxxxxx-9611-xxxx-ae06-f7bXXX03db71"}], "firewall_restore": "default"}'
```

Curl Command to Mark VEN as Suspended

```
curl -i -X PUT https://pce.my-company.com/api/v2/orgs/3/vens/xxxxxxxx-9611-xxxx-ae06-f7bXXX03db71 -H "Content-Type:application/json" -u $KEY:$TOKEN -d' {"status":"suspended"}'
```

Network Enforcement Nodes (NEN) APIs

Network Enforcement Node Reassignment

`network_enforcement_nodes_put`

This API is used to change the target PCE FQDN of an agent.

It updates the `target_pce_fqdn` property so that the NEN can be managed by a different PCE in a Supercluster.

Change Target PCE

When you have the NEN HREF, you can update the target PCE with the PCE FQDN the NEN will use. In your JSON request body, pass the following data:

```
       "target_pce_fqdn": "new-pce-fqdn.example.com"
}
```

The URI for this operation is:

```
PUT [api_version][nen_href]/update
```

This curl example shows how you can pass the `target_pce_fqdn` property containing the `FQDN` of the new PCE:

```
curl -u api_
xxxxxxx64fcee809:'xxxxxxx5048a6a85ce846a706e134ef1d4bf2ac1f253b84c1bf8df6b83c70d9
5' -H "Accept: application/json" -H "Content-Type:application/json" -X PUT -d '
{"target_pce_fqdn":"new-pce-fqdn.example.com"}'
https://my.pce.supercluster:443/api/v1/orgs/3/network_enforcement_nodes/f67d35d5-
ea71-42da-b40d-8dcc3b1420c2/update
```

Authorization and Exposure Changes

Some of the existing Experimental APIs have been changed in release 23.5.0 to facilitate creation of fully scripted integrations of endpoint management systems with the PCE using the Network Enforcement Nodes (NEN) Switch integration capabilities.

Exposure Changes

Exposure of the listed NEN APIs was changed in release 23.5.0 from Public Experimental to Public Stable.

Authorization Changes

Authorization of some NEN APIs was changed in release 23.5.0 from the default ("Global Administrator" and "Global Organization Owner") to authorize additional users as listed in the table.

API	Exposure Change	New Authorization Change
network_device_config	YES	NO
network_device_get	YES	NO
network_device_network_endpoint_get	YES	NO
network_devices_enforcement_instructions_applied_post	YES	"Global Policy Object Provisioner" and " Ruleset Provisioner"
network_devices_enforcement_instructions_request_post	YES	"Global Policy Object Provisioner" and " Ruleset Provisioner"
network_devices_get	YES	"Global Policy Object Provisioner", "Global Read Only", "Limited Ruleset Manager", "Ruleset Provisioner", "Ruleset Viewer", "Workload Manager"
network_devices_multi_enforcement_instructions_applied_post	YES	"Global Policy Object Provisioner" and " Ruleset Provisioner"
network_devices_multi_enforcement_instructions_request_post	YES	"Global Policy Object Provisioner" and " Ruleset Provisioner"
network_devices_network_endpoints_get	YES	NO
network_devices_network_endpoints_post	YES	"Workload Manager"
network_devices_network_endpoints_put	YES	"Workload Manager"
network_devices_put	YES	"Workload Manager"
network_endpoint_config	YES	NO
network_enforcement_node_get	YES	NO
network_enforcement_nodes_get	YES	"Full Ruleset Manager", "Global Policy Object Provisioner", "Global Read Only", "Limited Ruleset Manager", "Ruleset Provisioner", "Ruleset Viewer", "Workload Manager"

API	Exposure Change	New Authorization Change
network_enforcement_nodes_network_devices_post	YES	"Workload Manager"
network_enforcement_nodes_put	YES	NO

Filtering and Aggregating Traffic

This Public Stable API method allows you to handle broadcast and multicast traffic better, save storage in the traffic database, and reduce the stress of the whole data pipeline.

Windows-heavy environments can have a large amount of broadcast or multicast traffic, which can be as much as 50% in syslog data and 30% in traffic data. Because some broadcast and multicast data might not be useful for writing policies, this API provides a function to filter out or aggregate the broadcast and multicast traffic that is not useful.

> NOTE:
> This API is implemented in Supercluster.

> NOTE:
> Only administrators and users with appropriate privileges can make filtering changes.

Traffic Collector API Methods

Use these methods to get, create, update, or delete a traffic collector.

Functionality	HTTP	URI
Get a traffic collector collection	GET	[api_version][org_href]/settings/traffic_collector
Get a specific collector instance	GET	[api_version][org_href]/settings/traffic_collector/:uuid
Create a traffic collector	POST	[api_version][org_href]/settings/traffic_collector
Update a specific traffic collector instance	PUT	[api_version][org_href]/settings/traffic_collector/:uuid
Delete a specific traffic collector instance	DELETE	[api_version][org_href]/settings/traffic_collector/:uuid

Parameters

Parameters	Description	Type
`org_id`	Org ID	Integer
`traffic_collector_setting_id`	traffic_collector setting UUID	String

These are the properties for Traffic Collector Methods

Property	Description	Type
`href`	URI of the destination	String
`transmission`	(For the transmission type, choose `broadcast`, `multicast` or `unicast`	String
`action`	Drop or aggregate the target traffic: • If you select "drop," the PCE drops all the traffic that matches the filters you supply. The data will be lost forever. • If you select "aggregate," the PCE performs aggregation on broadcast traffic and multicast traffic . If one broadcast or multicast traffic flow is received by multiple workloads, all reported flows on the same traffic are aggregated into one record in the traffic database, and the destination workload information will be lost. • PUT method will fail if you change the aggregator from “broadcast” to “multicast” because the default port and protocol will not pass the validation step.	String
`target`	(PUT, POST) The target object has the following properties: • `dst_port`: Single destination ip address or CIDR . Can be an Integer or NULL • `proto`: Port is required for POST • `dst_ip`: Single destination ip address or CIDR • `src_port`: Single source ip address or CIDR. Allows users to filter traffic based on the source port. • `src_ip`: Single source ip address or CIDR If `dst_port` and `dst_ip` are not specified for the target session, traffic is dropped on "all ips" and "all ports" by default. PUT method will fail If the traffic filter you want to modify has “ANY”	Object Integer Integer String String

Property	Description	Type
	in port or protocol field, and you want to modify other fields in this filter. The change will fail because the default port and protocol will not pass the validation step. Oracle flows are currently filtered via a runtime `src_ip/dst_ip` (CIDR) setting and this feature is not available in SaaS. Runtime changes also require a PCE restart, while API settings do not. The collector filters now support `src_ip` (CIDR) so that various filters can be created per organization without restarting the PCE.	
`data_source`	Flow summary data source to support more granular filters, for endpoints in particular.	String
`network`	Flow summary network to support more granular filters, for endpoints in particular.	String

Examples for Traffic Collector

CURL Command for settings_traffic_collector_post

```
curl -i -u api_
10415cd5bcc0e14cc:'2ac31cbee8cd3e8fa7ca79d32d39a0249636624ada675965dd2ec239e3ea8af
0' --request POST --data '{"action":"drop","transmission":"unicast","target":
{"proto":6,"src_ip":"10.1.2.3"}}'
https://2x2testvc360.ilabs.io:8443/api/v2/orgs/2/settings/traffic_collector --
header "Content-Type: application/json"
```

Broadcast Transmission and Drop Action

```
curl 'https://pce.my-company.com:8443/api/v2/orgs/1/settings/traffic_collector' -H
'Origin: https://pce.my-company.com:8443' -H 'Accept-Encoding: gzip,deflate, br' -
H 'content-type: application/json' -H 'accept: application/json' -H 'Referer:
https://pce.my-company.com:8443/' -i -u api_
1dfe2432a7b314ee6:'21c10ea1a4ad38d76ef22977e8ac45bc10839c5cc6ebffd650eae4f95dc5b36
4'--data-binary '{"transmission": "broadcast","action": "drop","target":{"proto":
17,"dst_port": 20, "dst_ip":"10.255.255.255"}}' --compressed
```

Multicast Transmission and Aggregate Action

```
curl 'https://pce.my-company.com:8443/api/v2/orgs/1/settings/traffic_collector' -H
'Origin: https://pce.my-company.com:8443' -H 'Accept-Encoding: gzip, deflate, br'
-H 'content-type: application/json' -H 'accept: application/json' -H 'Referer:
https://pce.my-company.com:8443/' -i -u api_
1dfe2432a7b314ee6:'21c10ea1a4ad38d76ef22977e8ac45bc10839c5cc6ebffd650eae4f95dc5b36
4'--data-binary '{"transmission": "multicast","action": "aggregate"} ' --
compressed
```

Example Response

```
{
	"$schema": "http://json-schema.org/draft-04/schema#",
	"type": "object",
	"required": ["href", "transmission", "action"],
	"properties":{
		"href": {
			"description": "URI of the destination",
			"type": "string"
		},
		"transmission":{
			"description":"transmission type: broadcast/multicast",
			"type":"string",
			"enum":[
			"broadcast",
			"multicast"
			]
		},
		"target":{
			"type":"object",
			"required":[
			"proto"
			],
			"properties":{
			"dst_port":{
			"type":"integer"
			},
			"proto":{
```

```
                    "type":"integer"
                    },
                    "dst_ip":{
                    "type":"string",
                    "description": "single ip address or CIDR"
                    }
                    }
            },
            "action":{
                    "description":"drop or aggregate the target traffic",
                    "type":"string",
                    "enum":[
                    "drop",
                    "aggregate"
                    ]
                    }
            }
    }
```