TECHNICAL DOCUMENTATION

Illumio Core 23.2 Install, Configure, Upgrade

Table of Contents

In this Library	Description
PCE Installation and Upgrade Guide	Describes how to install and configure the Illumio Core Policy Compute Engine (PCE).
PCE Supercluster Deployment Guide	Describes how to deploy and administer a PCE Supercluster, a single administrative domain that spans two or more replicating PCEs.
VEN Installation Guide	Explains the two methods for installation: using the PCE web console to pair VENs with your hosts or to download the VEN software from the Illumio Support portal and install the software by using the VEN command line interface.
Illumio LW-VEN Installation Guide	Describes how to install and use the Legacy Windows VEN (LW-VEN) with the Illumio Core PCE to enforce security policies.
NEN Installation Guide	Describes how to install the Illumio Network Enforcement Node (NEN), to configure Server Load Balancers (SLBs) and switches to work with it, and to use the NEN to secure workloads attached to those network devices.
Illumio Core for Kubernetes and OpenShift Guide	Provides information about how to use Illumio Core with containerized applications running in clusters orchestrated by Kubernetes, and/or by other similar operators like OpenShift.
Flowlink Configuration and Usage Guide	Describes how to install, configure, and use Flowlink, an Illumio standalone application that can be used with the Illumio Core to collect network flow data from different types of network sources, such as switches, routers, F5 load balancers, cloud monitoring tools, and syslog exporters.

Upgrade Summary

Upgrading Illumio Core: Why and How

IMPORTANT
In addition to VEN upgrade, this guide discusses reasons to upgrade your PCE platform. Illumio Core Cloud customers rely on Illumio Operations to upgrade and manage the PCE platform. However, reading this guide is useful to learn about the key updates to Illumio Core by release.

Have You Been Missing Out?

Illumio adds new features constantly! We're always busy innovating and improving. If you've had your Illumio software up and running for a while without upgrading, we'd like to encourage you to consider moving to a newer version.

We understand the appeal of stability and the trustworthiness of your current deployment. Your applications are vital to your organization, and stability is crucial. It can seem like a big step to update software that's already working so well. But why not take a few moments to imagine how life could be if you had access to some of Illumio's recent features (not to mention the fixes to known issues and limitations). And if you're thinking the upgrade itself will be complicated, we've included a short summary below of how it's done, so you can see the path ahead.

Best Practice: Pick a Standard or LTS Release

Pick the release type that's right for your organization and maintain N-1 version status.

Standard Release

Standard releases of the PCE and VEN receive limited maintenance for 1 year. For customers who want to take advantage of the latest features as soon as they are available, and can commit to upgrading to the next release for bug fixes and security updates. Remember, you don't have to upgrade your VENs to use most new PCE features. Check Versions and Compatibility on the Illumio Support site.

Long Term Support (LTS) Release

Long Term Support releases of the PCE and VEN receive active maintenance for 1 year and limited maintenance for another 2 years. For customers who wish to upgrade less frequently and stay on a version of the PCE or VEN with active maintenance support.

Features You've Been Missing

Here are just *some* of the things our tireless teams have engineered, tested, and sent out into the world, maybe while you weren't looking, to increase the capabilities of Illumio Core and keep your essential systems even safer.

23.5.0

An Illumio Core Cloud customer release only.

November 2023. Works with VEN 18.2.3 - 23.2

- New UI replaces classic UI to maximize user productivity and enable intuitive platform administration.
- Policy templates: predefined policy definitions for some of the most popular security practices
- New widgets in Ransomware Protection Dashboard.
- Bulk import and export of workload labels.
- Enhancements to visualization tools: Vulnerabilities tab, mapping members with no traffic, new group tabs.
- New object type: Windows outbound process.
- Traffic data for unpaired VENs.
- Splunk TA and app version upgraded to 4.0.0, including support for MT4L, multiple PCEs, multiple organizations, and faster search.

23.4.0

An Illumio Core Cloud customer release only.

July 2023. Works with VEN 18.2.3 - 22.5.20

- New PCE user interface (UI) is default for all new and upgrading customers.
- Quick links to workload detail pages from the Dashboard.
- Display more related information by clicking in pie charts on the Dashboard.
- Policy Check is aware of network type, which makes rule writing easier.
- New lost agent warning event indicates when a VEN has been unpaired from the PCE for longer than the uninstall timeout, then returns.

23.2.0

May 2023. Works with VEN 18.2.3 - 22.5.20

- New PCE user interface (UI) to maximize user productivity and enable intuitive platform administration.
- Ransomware Protection information is provided for each workload, aggregated in the main Dashboard.
- Ringfencing rules shrink the security perimeter from a subnet or VLAN to a single application.
- Set the VEN type, server or endpoint, in the pairing profile.
- Configurable VEN upgrade timeout.
- Configurable second FQDN for southbound traffic from VEN to PCE.

- Extended support for RHEL 5 on VENs.
- Explore features are enhanced in several ways, including daily auto-generation of a Default Graph report.

22.5.20

April 2023. Works with VEN 18.2.3 - 22.5.20

- Regular maintenance release that solved software and security issues to refine the software and improve its reliability and performance. See "Resolved Issue in 22.5.20" in the *Illumio Core Release Notes 22.5*.
- Deployment of Core VENs on Kubernetes nodes is deprecated. Instead, use Illumio Core for Kubernetes.

22.5.10+UI2

February 2023.

Returned the Explorer feature to the PCE web console for customers who still want to use the functionality in that area of the GUI. (In Illumio Core 22.5.0 and 22.5.10, Illumio removed the Explorer feature from the PCE web console main menu.)

22.5.10 (LTS Candidate)

January 2023. Works with VEN 18.2.3 - 22.5.10

- VEN dashboard with broad, visualized information about VEN statistics.
- VEN tampering protection: require a token with commands to avoid accidental or malicious actions.
- Context menus in Illumination Plus.
- See the amount of data transferred into and out of workloads and applications in a data-center.
- Mac OS support.

22.4.0 (Standard)

An Illumio Core Cloud customer release only.

October 2022. Works with VEN 18.2.3 - 22.4.0

- ML/AI-based scanner detection, so that discovered scanners can become workloads.
- Usability enhancements in the VEN details page.
- Illumio Core REST API enhancements.

22.3.0 (Standard)

An Illumio Core Cloud customer release only.

August 2022. Works with VEN 18.2.3 - 22.3.0

- Prevent the creation of spurious labels in the PCE.
- REST API: SLB and Virtual Server API consistency and filtering.

- Support for shared SNAT out of public clouds.
- Support for non-domain joined interfaces for the Illumio Endpoint - Windows.

22.2.10 (LTS Candidate)

July 2022. Works with VEN 18.2.3 - 22.2.0

- Replication and failover, with a PCE on "warm standby" for increased reliability.
- Solved software and security issues to refine the software and improve its reliability and performance. See the *Illumio Core Release Notes 22.2.10*.

22.2.1 (LTS Candidate)

May 2022. Works with VEN 18.2.3 - 22.2.0

Regular maintenance release that solved software and security issues to refine the software and improve its reliability and performance. See the *Illumio Core Release Notes 22.2.1*.

22.2.0 (Standard)

May 2022. Works with VEN 18.2.3 - 22.2.0

- Policy exclusions in ruleset scopes and rules. Define security policy by excluding unwanted labels.
- Scopeless rules. Choose whether or not to include scopes when creating new rulesets.
- Simplified rule writing UI. Show advanced rule writing features only when you need them.
- Label type (Role, Application, etc.) is indicated by an icon, and you can filter by type when entering label names.
- Illumination shows whether traffic is blocked by full enforcement or an enforcement boundary.
- Disable and enable enforcement boundaries.
- Generate reports from Explorer traffic flow queries.

22.1.3 PCE (Standard)

June 2022. Works with VEN 21.2.0 - 22.1.0

IMPORTANT

Release available for Illumio Core Cloud customers only.

Regular maintenance release that solved software and security issues to refine the software and improve its reliability and performance. See the Illumio Core Release Notes 22.1.3.

22.1.2 PCE (Standard)

April 2022. Works with VEN 21.2.0 - 22.1.0

> **IMPORTANT**
> Release available for Illumio Core Cloud customers only.

Regular maintenance release that solved software and security issues to refine the software and improve its reliability and performance. See the Illumio Core Release Notes 22.1.2.

22.1.1 PCE (Standard)

March 2022. Works with VEN 21.2.0 - 22.1.0

> **IMPORTANT**
> Release available for Illumio Core Cloud customers only.

Regular maintenance release that solved software and security issues to refine the software and improve its reliability and performance. See the Illumio Core Release Notes 22.1.1.

22.1.0 PCE (Standard — unreleased PCE version)

2022. Works with VEN 21.2.0 - 22.1.0

- Rule writing available from Explorer
- Enforcement Boundary information in Reports
- Display of traffic blocked by Enforcement Boundaries in Explorer Reported view
- Numerous Explorer usability enhancements
- Numerous policy and workload incremental usability improvements
- Single Pane of Glass (SPOG) enhancements
- Certificate validation for SSO configuration
- Signing for SAML requests

21.5.20 PCE (LTS)

March 2022. Works with VEN 18.2.3 - 21.5.20

Regular maintenance release that solved software and security issues to refine the software and improve its reliability and performance. See the Illumio Core Release Notes 21.5.20.

21.5.12 PCE (LTS)

F**February 2022.** Works with VEN 18.2.3 - 21.5.20

The official LTS release for the Illumio Core 21.5.x release series. This regular maintenance release solved software and security issues to refine the software and improve its reliability and performance. See the Illumio Core Release Notes 21.5.12.

21.5.10 PCE (LTS Candidate)

January 2022. Works with VEN 18.2.3 - 21.5.20

- VEN compatibility report updates for IPv6 support
- Performance enhancements: policy provisioning, container workloads, Kubelink service updates, and convergence times
- VEN support on IBM Z systems running RHEL 7 and RHEL 8
- Label-based Security setting for IP Forwarding
- Rotate database passwords and other internal secrets

21.5.1 PCE (Standard)

December 2021.

Regular maintenance release that solved software and security issues to refine the software and improve its reliability and performance. See the Illumio Core Release Notes 21.5.1.

21.5.0 PCE (Standard)

November 2021.

- PCE Support Bundles: generate support reports in web console instead of command line.
- Core services detector
- Alerts displayed to tell whether the hardware provisioned for each node meets requirements.
- Use FQDN instead of hostname to more easily distinguish messages from different hosts.
- Enhanced security for keys and secrets.
- 8-region Supercluster.

21.4.1 PCE (Illumio Core Cloud only)

October 2021. Works with VEN 18.2.x - 21.2.3 (Standard release)

- Service Account management in the PCE web console
- SSO support for service providers
- Enhancements for the HAProxy TLS version limit

21.3.0 PCE (Illumio Core Cloud only)

August 2021. Works with VEN 18.2.x - 21.2.3 (Standard release)

- Service Accounts: Instead of being associated with a user, an API key can be associated with a service.
- Parallel Coordinates: In Explorer, search results using the Parallel Coordinates format are improved.

21.2.3 PCE (LTS)

October 2021. Works with VEN 18.2.x - 21.2.3 (LTS)

Regular maintenance release that solved software and security issues to refine the software and improve its reliability and performance. See the Illumio Core Release Notes 21.2.3.

21.2.2 PCE (LTS)

August 2021. Works with VEN 18.2.x - 21.2.2 (LTS)

- Illumio Core Maintenance Release
- Cloud available through the PCE web console

21.2.1 PCE (LTS candidate)

June 2021. Works with VEN 18.2.x - 21.2.1 (LTS candidate)

- RHEL 8 supported for PCE

NOTE

21.2.1-PCE and 21.2.1-VEN are candidates for LTS. For information on Illumio software support for Standard and LTS releases and to verify the LTS status of the Core 21.2.1 release, see Versions and Releases on the Illumio Support portal.

21.2.0 PCE (Standard)

May 2021. Works with VEN 18.2.x - 21.2.0 (Standard)

- Enforcement Boundaries: Build deny-list type rules without rule conflict, and maintaining Zero Trust.
- Global Explorer: Unified results for members from every region in a supercluster.
- Asynchronous Explorer queries: No need to wait for query to return results. Results can be retrieved for up to 24 hours.
- Label groups for RBAC: Use label groups to define user permissions.
- Supercluster replication enhancements.
- Supercluster rolling upgrade when upgrading to a hotfix or a maintenance release.
- LDAP improvements to support better user lookup.
- Enhanced security for PCE TLS configuration.
- Reports (preview): generate executive summary and app group summary reports.
- RHEL 8 supported for PCE (preview)

21.1.0 PCE (Standard)

March 2021. Works with VEN 18.2.x - 21.1.0 (Standard)

- **LDAP authentication:** PCE supports LDAP authentication for users with OpenLDAP and Active Directory.
- Label restrictions for Kubernetes namespaces: Illumio administrators have a way to control which labels can be assigned by the developers managing their Kubernetes environments.
- PCE hardening: PCE now takes additional steps to ensure its own security.
- VEN proxy support on Linux, AIX, and Solaris: Extends VEN proxy support from Windows systems to Linux, AIX, and Solaris devices.
- Aggressive tampering protection for nftables: Provides support for nftables that runs on RedHat. Detection of external firewall configurations are called aggressive tampering and the VEN will log such errors. Now, the VEN can detect any configuration that is not explicitly configured on the VEN.

20.2.0 PCE (Standard)

October 2020. Works with VEN 18.2.x - 20.2.0 (Standard)

- **Selective enforcement**: Set enforcement policy to protect only a subset of applications or processes on a workload.
- Access restrictions: Restrict the usage of API keys and the PCE web interface by IP address and block API requests coming in from non-allowed IP addresses.
- Multi-node traffic database: Scale traffic data by sharding it across multiple PCE data nodes to store more data and improve read and write performance.
- Centralized, automated VEN upgrade.

20.1.0 PCE (Standard)

April 2020. Works with VEN 18.2.x - 20.1.0 (Standard)

- **RBAC for app owners**: Delegate policy writing to downstream teams; read-only access; per-app read/write. Think of this as zero-trust views into Illumination.
- Reject connections: Configure workloads to send reject messages if there is a policy violation.
- Flow collection: Windows broadcast traffic can clog up the system. Configure the PCE to drop or aggregate it.

19.3.3 PCE – 19.3.6 PCE (LTS)

September 2020 - January 2021. Works with VEN 18.2.x - 19.3.6

Regular maintenance releases that solved software and security issues to refine the software and improve its reliability and performance.

19.3.2 PCE (LTS)

July 2020. Works with VEN 18.2.x - 19.3.2

- **Health Metrics:** Application metrics with configurable thresholds have been added to PCE health monitoring.
- Workload clone alerts: Filter workloads by whether a clone has been detected.
- Oracle Exadata support.
- Containers now available in Supercluster member regions.

19.3.1 PCE (Standard)

March 2020. Works with VEN 18.2.x - 19.3.1

- **Containers** are now supported with visibility and enforcement for **Kubernetes and OpenShift**.
- UI: The App Group map is enhanced; in Explorer, the Connection State and draft view are improved.

19.1.0 PCE (Standard)

June 2019. Works with VEN 18.2.x - VEN 19.1.0 (Standard)

- **Switch visibility and enforcement with Network Enforcement Node**: Secure workloads that are attached to network switches.

- Use Rule Search to search and analyze rules, so you can fine-tune them and optimize enforcement.
- Internal PCE log file rotation: To increase the amount of log data, customize the rotation of PCE log files.
- **Export reports** in JSON and CSV to share the information with anyone who doesn't have access to the PCE.
- **Workload Manager**: A new user role to manage workloads, pairing profiles, and API keys.

18.3.1 PCE (Standard)

March 2019. Works with VEN 18.2.x - VEN 18.3.1 (Standard)

- **Internal syslog**: You no longer need to manage the syslog and log rotation.
- **Policy revert**: Return to previous version if a policy rollout is unsuccessful.
- Session timeout can be set by organization owner to control timeout on user sessions.
- Firewall coexistence between Illumio firewall and existing firewalls that use WFP or IPTables (requires upgrade to VEN 18.3.1)

18.2.5 PCE (LTS)

September 2019. Works with VEN 18.2.x - VEN 18.2.5 (Standard)

- **Common Criteria** certified.
- Risk & Compliance capability.
- **Vulnerability Maps**: tune your segmentation to reduce the exposure of unpatched workloads
- Events storage: Improved use of capacity in the events database.

How to Upgrade the Illumio PCE

> **IMPORTANT**
>
> This information is provided for reference only. Illumio Core Cloud customers do not upgrade the PCE. Illumio Operations performs this function for Illumio Core Cloud customer.

This is a quick summary. For full instructions, see the PCE Installation Guide for the version you are upgrading to.

Starting with Illumio Core 19.3.0, the PCE is shipped in two parts:

- PCE – Policy Compute Engine. This is the main processing engine.
- UI – Browser-based console.

Quick Summary

Before you begin, choose the PCE version you want to upgrade to. Check the Upgrade Path page to see whether you can upgrade directly from your current version in one step, or whether you need to upgrade to another intermediate version first. Then start the upgrade:

- Download the software, check upgrade prerequisites, and do a few additional preparation steps.
- Back up the PCE and stop it.
- Install the software.
- You might need to update the PCE's runtime environment file, depending on your currently installed version.
- Migrate the PCE database.

  To migrate the PCE from one datacenter to another, refer to the Knowledge Base article Migrate the PCE from one Datacenter to Another.

The upgrade is done! Set the PCE's runlevel to 5 to resume normal operations.

Steps

1. Download the two software packages (PCE and UI) from the Illumio Support portal to the `/tmp` folder on your PCE nodes. The UI package is only needed on core nodes, not data nodes.
2. Back up the PCE with these commands:
   a. Find the node that is running `agent_traffic_redis_server`:

      ```
      sudo -u ilo-pce illumio-pce-ctl cluster-status
      ```
   b. On the node that is running `agent_traffic_redis_server`:

      ```
      sudo -u ilo-pce illumio-pce-db-management dump --file db-dump-file
      ```
   c. Store a copy of each node's `runtime_env.yml` file.
3. Stop the PCE on all nodes:

   ```
   sudo -u ilo-pce illumio-pce-ctl stop --wait
   ```
4. On core nodes, install the UI and PCE. Specify both RPM files on the command line:

   ```
   rpm -Uvh illumio_pce_rpm illumio_ui_rpm
   ```
5. On data nodes, install the PCE only:

   ```
   rpm -Uvh illumio_pce_rpm
   ```
6. Check the Release Notes for all versions after the one you are upgrading from to see whether any of the changes require updates to `runtime_env.yml`. If so, update the file on all nodes, then check it:

   ```
   sudo -u ilo-pce illumio-pce-ctl check-env
   ```
7. Start the PCE on all nodes:

   ```
   sudo -u ilo-pce illumio-pce-ctl start --runlevel 1
   ```
8. Migrate the database (you can run this on any node):

   ```
   sudo -u ilo-pce illumio-pce-db-management migrate
   ```
9. After database migration is complete, bring the PCE back up to its normal operational level:

   ```
   sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
   ```
10. VEN software can also be upgraded now. Go on to the next section to find out how.

How to Upgrade the Illumio VEN

This is a quick summary. For full instructions, see the VEN Installation Guide for the version you are upgrading to.

> **NOTE**
> See also the Knowledge Base article named "Can You Install a VEN on the Workloads Running the Illumio PCE?" for more information.

After you upgrade the PCE, earlier versions of the VEN will still work, but some of the newer PCE features only work with newer VENs (consult the Release Notes to find out exactly which ones). And, of course, each VEN version includes new features and fixed issues. If you want to upgrade your VEN software, follow these steps.

1. Download the software from the Illumio Support portal.
2. Run this command on any PCE core node:

   (If you want to upgrade just some of the VENs, you can list them instead of using the --all flag. See "VEN Installation & Upgrade with VEN CTL" in the VEN Installation and Upgrade Guide.)

   ```
   sudo -u ilo-pce illumio-pce-ctl ven-upgrade venReleaseId --all
   ```

Congratulations – you have upgraded your PCE cluster and VENs to a newer version. Start enjoying the improved UI, better role-based access control, containerization, and other new capabilities of the upgraded PCE and VEN.

If you have questions, don’t hesitate to reach out to your Customer Success Advisor. We will be happy to provide guidance.

PCE Install Upgrade

Overview of PCE Installation

Understanding the concepts in this overview will help you achieve a successful PCE installation.

1. Download and install the software.
2. Configure the PCE including required and optional settings.
3. Start and initialize the PCE.
4. Understand the next steps that are required for a full Illumio Core installation after the PCE is installed and running.

Before you begin

Illumio recommends that you be familiar with the following technology:

- Your organization's security goals
- General knowledge of Illumio Core
- General computer system administration of Linux and Windows operating systems, including startup/shutdown, and common processes or services
- Linux shell (bash), Windows PowerShell, or both
- TCP/IP networks, including protocols, well-known ports, and the Domain Name System (DNS)
- Familiarity with TLS/SSL certificates

Notational Conventions

- Newly introduced terminology is italicized. Example: *activation code* (also known as pairing key)
- Command-line examples are monospace. Example: `illumio-ven-ctl --activate`
- Arguments on command lines are monospace italics. Example: `illumio-ven-ctl --activate activation_code`
- In some examples, the output might be shown across several lines but is actually on one single line.
- Command input or output lines not essential to an example are sometimes omitted, as indicated by three periods in a row. Example:

  ```
  ...
  some command or command output
  ...
  ```

PCE Installation Overview

This overview introduces some essential concepts that you'll need to understand before installing the PCE.

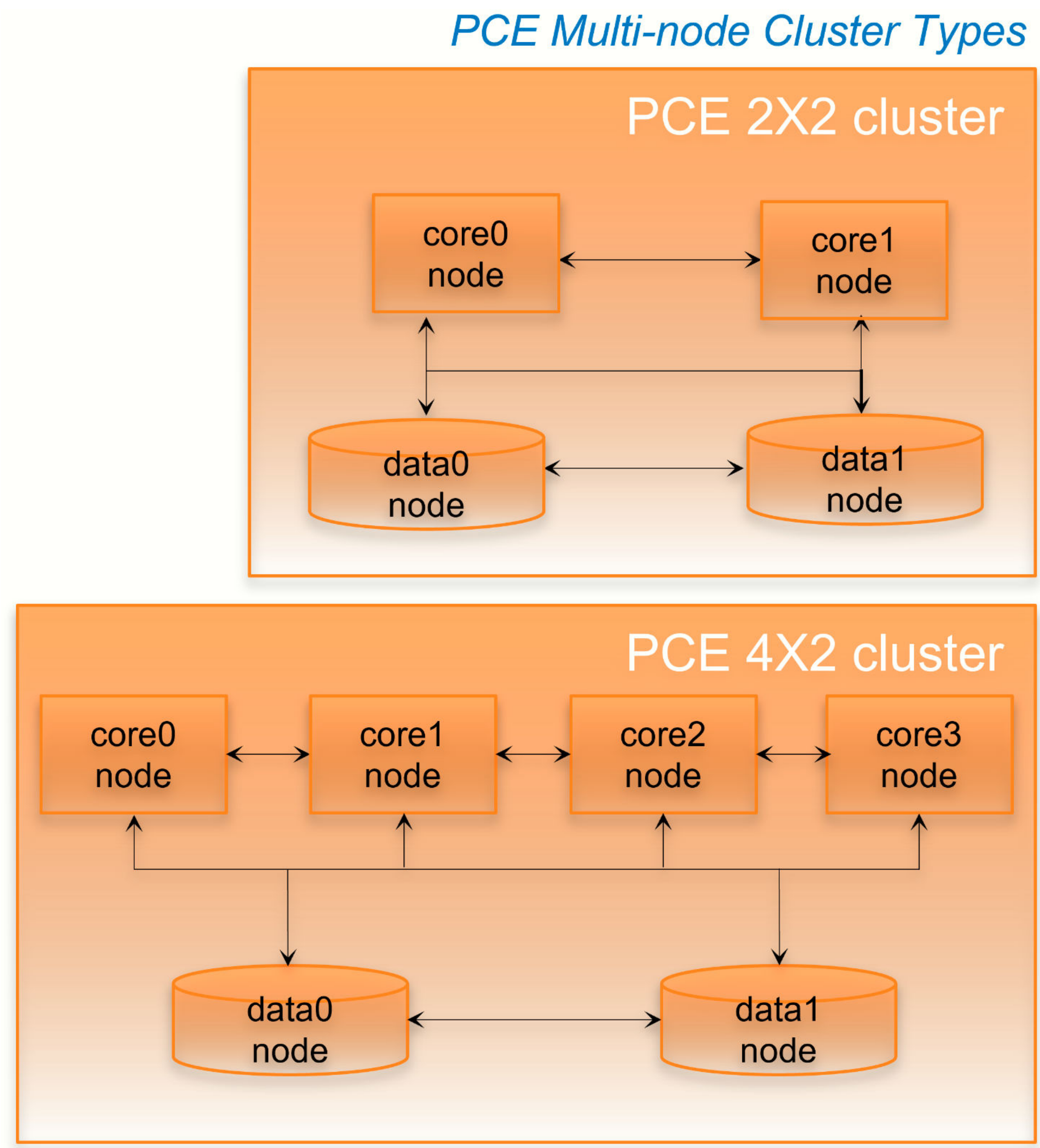
Nodes and Clusters

A *PCE node* is a single host (server or VM) that runs the PCE. Each node in the cluster is configured by its node type, which defines its services:

- Core node, known as core0, core1, core2, and core3
- Data node, known as data0 and data1
- Single node in an *single-node cluster* (SNC), which combines core and data nodes in one

The total collection of nodes is a *PCE cluster*. In production, the PCE is typically deployed as a *multiple-node cluster* (MNC).

- For smaller deployments where high availability is not necessary, you can deploy a PCE SNC.
- In a typical PCE deployment, for redundancy, you deploy two instances of each node type in a *PCE 2x2 cluster*.
- For larger deployments, you can expand the PCE cluster to four core nodes and two data nodes in a *PCE 4x2 cluster*.
- To construct a single administrative domain that spans two or more replicating PCE clusters, deploy a *PCE supercluster*. See PCE Supercluster Deployment Guide.



Single-node Clusters

In an SNC, some special considerations apply.

Because it contains only a single node, an SNC does not provide high availability (HA) features. The SNC is a single point of failure. Therefore, Illumio recommends taking some additional precautionary steps:

- Set up periodic, automated backups.
- Practice restoring from backup to a separate machine (physical or virtual) before putting the SNC into production use.
- Store a copy of the PCE software installation packages, the PCE database backup, and the `runtime_env.yml` file, which stores the PCE's configuration. Store them on a separate physical machine, preferably in a different datacenter, using fault tolerant storage.

- If you are running the SNC as a virtual machine, you can make use of the hypervisor's high availability and disaster recovery (HA/DR) features.

To prepare for PCE installation on an SNC:

- Have a reserved virtual machine or physical machine ready for the backups of the PCE software, database, and `runtime_env.yml`.
- This machine must be able to use the existing IP address of the PCE. Alternatively, you can reserve a new IP address for the backup machine, and configure this IP address in the PCE.

Software Distribution: PCE and UI Packages

Illumio distributes PCE software as two packages: PCE and UI. The PCE package contains the software for the Policy Compute Engine (PCE), and the UI package contains the PCE web console. You can choose to install these packages separately or together:

- **PCE package plus UI package:** This choice is the most common installation scenario. See Installing the PCE and UI [46].
- **PCE package alone:** The PCE still serves responses to API calls, but there is no graphical user interface for display in a browser.
- **UI package alone:** With this separate package, you can upgrade the UI whenever you want more recent UI fixes and features, without having to upgrade the entire PCE. The UI-only installation procedure is much simpler than the full installation. For the UI to work, a compatible version of the PCE must already be installed. See UI-Only Upgrade [70].

Prepare for PCE Installation

Before installing the PCE software, review the prerequisites and planning checklist.

PCE Planning Checklist

Use this checklist to plan your PCE installation. Details for each task are described in following sections.

Prerequisite	See Section
☐ Capacity sizing for CPUs, RAM, and storage device size and IOPS	PCE Capacity Planning [22]
☐ PCE storage device partitions	PCE Storage Device Partitions [27]
☐ Verify PCE reserved port ranges (for MNCs; does not apply in an SNC)	Port Ranges for Cluster Communication [33]
☐ Load balancer setup	Load Balancer Requirements [34]
☐ IP address for the PCE	PCE IP Address [35]
☐ DNS domain name setup	DNS Requirements [35]
☐ Mail software	SMTP Requirements [35]
☐ TLS setup, including SSL certificate types and settings	• TLS (SSL) Requirements [35] • TLS Versions for Communications [38]
☐ Validate and configure the TLS certificate (Optional)	(Optional) Validate and Configure TLS Certificate [51] either before or after configuring the PCE
☐ (Optional) SAML IdP	SAML IdP [39]
☐ OS package dependencies, libraries, NTP, iptables, UTF-8, Trusted CA, syslog, process and file limits, and kernel parameters	OS Setup and Package Dependencies [39]
☐ Your full organization name	About Your Organization Name and ID [45]
☐ VEN installation, including planning and prerequisites	VEN Installation and Upgrade Guide [161]

NOTE

See the Knowledge Base article: Can you install a VEN on the workloads running the Illumio PCE?. You need to log in to view this article.

PCE Capacity Planning

Use these guidelines and requirements to estimate host system capacity based on typical usage patterns.

The exact requirements vary based on a large number of factors, including, but not limited to:

- Whether you are using physical or virtual hardware
- Number of managed workloads
- Number of unmanaged workloads and other labeled objects, such as virtual services
- Policy complexity, which includes the following factors:
  - Number of rules in your rulesets
  - Number of labels, IP lists, and other objects in your rules
  - Number of IP ranges in your IP lists
  - Number of workloads affected by your rules
- The frequency at which your policies change
- Frequency at which workloads are added or deleted, or workload context changes, such as change of IP address
- Volume of traffic flows per second reported to the PCE from all VENs

  See the “Maximum Flow Capacity” table for information about maximum flow capacity of the PCE.
- Total number of unique flows reported to the PCE from all VENs

CPU, Memory, and Storage

The capacity planning tables in this section list the minimum recommended sizes for CPU, memory, and storage. This section provides two tables, one for physical hardware and one for virtual machines. Use these tables to plan your deployment.

> **NOTE**
>
> Based on your actual usage and other factors, your capacity needs might be greater than the recommended sizes. For example, if you have installed additional software along with the PCE, such as application performance management (APM) software or an endpoint protection agent, this consumes additional system resources.

Data nodes are configured with a dedicated storage device for each database on the data nodes. This configuration accommodates growth in traffic data, which is used by Explorer. See Runtime Parameters for Traffic Datastore on Data Nodes [30].

For more than 150 IOPS, locally attached, spinning hard disk drives (HDD) are not sufficient. You will require either mixed-use Solid-State Disk (SSD) or Storage Area Network (SAN).

The PCE does not require that you set up swap memory, but it is permissible to enable swap memory. As long as the PCE nodes are provisioned with the recommended memory (RAM) as shown in the tables below, the use of swap memory should not cause any issues.

Physical Hardware

Use this table if you are installing the PCE on physical hardware. If you are using virtual machines, see the table Virtual Hardware [25].

Cluster Type + Max VENs and Total Workloads	Cores/Clock Speed	RAM per Node	Storage Device Size and IOPS	
			Core Nodes	Data Nodes
SNC • 250 VENs [1] • 1,250 total workloads [1]	• 3 cores [2] • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or equivalent	16GB	A single node including both core and data: • 1 x 50GB4 • 100 IOPS per device [5]	N/A
2x2 Small • 2,500 VENs [1] • 12,500 workloads [1] Cluster type: `4node_v0_small`	• 4 cores per node [2] • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or equivalent	32GB	Minimum: • Disk: 50GB3, [4] • 150 IOPS per device [5]	Minimum: • Disk 1: 250GB [4] • Disk 2: 250GB [4] • 600 IOPS per device [5]
2x2 • 10,000 VENs [1] • 50,000 workloads [1] Cluster type: `4node_v0` or `4node_dx`	• 16 cores per node [2, 6] • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or equivalent	• Recommended: 128GB [6] • Minimum: 64GB	Minimum: • Disk: 50GB [3, 4] • 150 IOPS per device [5]	Minimum: • Disk 1: 1TB4 • Disk 2: 1TB [4] • 1,800 IOPS per device[5]
4x2 • 25,000 VENs [1] • 125,000 workloads [1] Cluster type: `6node_v0` or `6node_dx`	• 16 cores per node [2, 6] • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or equivalent.	128GB [6]	Minimum: • Disk: 50GB [3, 4] • 150 IOPS per device [5]	Minimum: • Disk 1: 1TB [4] • Disk 2: 1TB [4] • 5,000 IOPS per device [5]

Footnotes:

[1] Number of total workloads is the sum of both managed and unmanaged workloads, which cannot exceed the maximum number of VENs (managed workloads). For example, if an SNC contains the maximum number of VENs (250), it can support only 1,000 unmanaged workloads, for a total of 1,250 workloads. IF the SNC contains less than the max number of VENs, it can support more unmanaged workloads.

[2] CPUs: The recommended number of cores is based only on physical cores from allocated CPUs, irrespective of hyper-threading.

[3] This is the absolute minimum needed. In the future, other applications, support reports, or new features may require additional disk.

[4] Additional disk notes: Storage requirements for network traffic data can increase rapidly as the amount of network traffic increases. Network File Systems (NFS) is not supported for Illumio directories specified in runtime -- for example, `data_dir`, `persistent_data_dir`, or `ephemeral_data_dir`.

[5] Input/output operations per second (IOPS) are based on 8K random write operations. IOPS specified for an average of 300 flow summaries (80% unique `src_ip, dest_ip`, `dest_port`, `proto`) per workload every 10 minutes. Different traffic profiles might require higher IOPS.

[6] In the case of fresh installs or upgrades of a 2x2 for 10,000 VENs or a 4x2 for 25,000 VENs, if you deploy a system without sufficient cores, memory, or both, then the PCE will automatically reduce the object limits to 2,500 workloads. Object limit is the number of VENs (agents) per PCE. Adding more than 2,500 workloads will fail and an event is logged indicating that object limits have been exceeded. The workaround is to increase the number of cores, memory, or both to the recommended specifications and then increase the object limits manually. See PCE Default Object Limits in the PCE Administration Guide.

Virtual Hardware

Use this table if you are installing the PCE on virtual machines. If you are using physical hardware, see the table Physical Hardware [23].

Cluster Type + Max VENs and Total Workloads	Virtual Cores/ Clock Speed	RAM per Node	Storage Device Size and IOPS	
			Core Nodes	Data Nodes
SNC • 250 VENs [1] • 1,250 workloads [1]	• 6 virtual cores (vCPU) [2] • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or higher	16GB [7]	Minimum: • Disk: 50GB [3, 4] • 150 IOPS per device [5]	N/A
2x2 Small • 2,500 VENs [1] • 12,500 workloads [1] Cluster type: `4node_v0_small`	• 8 virtual cores (vCPU) per node [2] • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or higher	32GB [7]	Minimum: • Disk: 50GB [3, 4] • 150 IOPS per device [5]	Minimum: • Disk 1: 250GB • Disk 2: 250GB • 600 IOPS per device
2x2 • 10,000 VENs [1] • 50,000 workloads [1] Cluster type: `4node_v0` or `4node_dx`	• 32 virtual cores (vCPU) per node [2, 6] • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or higher	• Recommended: 128GB [6, 7] • Minimum: 64GB	Minimum: • Disk: 50GB [3, 4] • 150 IOPS per device[5]	Minimum: • Disk 1: 1TB4 • Disk 2: 1TB4 • 1,800 IOPS per device5
4x2 • 25,000 VENs [1] • 125,000 workloads [1] Cluster type: `6node_v0` or `6node_dx`	• 32 virtual cores (vCPU) per node [2, 6] • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or higher	128GB [6, 7]	Minimum: • Disk: 50GB [3, 4] • 150 IOPS per device [5]	Minimum: • Disk 1: 1TB4 • Disk 2: 1TB4 • 5,000 IOPS per device [5]

Footnotes:

[1] Number of total workloads is the sum of both managed and unmanaged workloads, which cannot exceed the maximum number of VENs (managed workloads). For example, if an SNC contains the maximum number of VENs (250), it can support only 1,000 unmanaged

workloads, for a total of 1,250 workloads. IF the SNC contains less than the max number of VENs, it can support more unmanaged workloads.

[2] Full reservations for vCPU. No overcommit.

[3] This is the absolute minimum needed. In the future, other applications, support reports, or new features may require additional disk.

[4] Additional disk notes:

- Storage requirements for network traffic data can increase rapidly as the amount of network traffic increases.
- Network File Systems (NFS) is not supported for Illumio directories specified in runtime; for example, `data_dir`, `persistent_data_dir`, or `ephemeral_data_dir`.

[5] Input/output operations per second (IOPS) are based on 8K random write operations. IOPS specified for an average of 300 flow summaries (80% unique `src_ip, dest_ip`, `dest_port`, `proto`) per workload every 10 minutes. Different traffic profiles might require higher IOPS.

[6] In the case of fresh installs or upgrades of a 2x2 for 10,000 VENs or a 4x2 for 25,000 VENs, if you deploy a system without sufficient cores, memory, or both, then the PCE will automatically reduce the object limits to 2,500 workloads. Object limit is the number of VENs (agents) per PCE. Adding more than 2,500 workloads will fail and an event is logged indicating that object limits have been exceeded. The workaround is to increase the number of cores, memory, or both to the recommended specifications and then increase the object limits manually. See PCE Default Object Limits in the PCE Administration Guide.

[7] Full reservations for vRAM. No overcommit.

Maximum Flow Capacity

The following table shows the maximum capacity of the PCE to accept flow data from all VENs.

Cluster Type + VENs and Total Workloads	Flow Rate (flow-summa-ries/second)	Equivalent Flow Rate (flows/second)[2]
SNC • 250 VENs • 1,250 total workloads	100	1,030
2x2 • 2,500 VENs • 12,500 total workloads	1,000	10,300
2x2 • 10,000 VENs • 50,000 total workloads	4,100	422,000
4x2 • 25,000 VENs • 125,000 total workloads	10,400 [1]	1,070,000

Footnotes

[1] The PCE might need to be tuned to achieve this rate. If you need to tune the PCE, please contact Illumio Support for assistance.

[2] Real-world observation shows that 102 flows result in one flow summary on average.

PCE Storage Device Partitions

PCE Storage Device Layout

You should create separate storage device partitions to reserve the amount of space specified below. These recommendations are based on PCE Capacity Planning [22].

The values given in these recommendation tables are guidelines based on testing in Illumio's labs. If you wish to deviate from these recommendations based on your own platform standards, please first contact your Illumio support representative for advice and approval.

PCE Single-Node Cluster for 250 VENs

Storage Device	Partition mount point	Size to Allocate	Node Types	Notes
Device 1, Partition A	`/`	8GB	Core, Data	The size of this partition assumes the system temporary files are stored in `/tmp` and core dump file size is set to zero. The PCE installation occupies approximately 500MB of this space.
Device 1, Partition B	`/var/log`	16GB	Core, Data	The size of this partition assumes that PCE application logs and system logs are both stored in `/var/log`. PCE application logs are stored in the `/var/log/illumio-pce` directory. The recommended size assumes average use by the OS with common packages installed and logging levels set to system defaults. Log size limits are configurable, so your system may require more or less log space. To find the potential maximum disk space required for your logs, use this command: `sudo -u ilo-pce illumio-pce-env logs --diag`
Device 1, Partition C	`/var/lib/illumio-pce`	Balance of Device 1	Core, Data	The size of this partition assumes that Core nodes use local storage for application code in `/var/lib/illumio-pce`, and also assumes that PCE support report files, and other temporary (ephemeral) files, are stored in `/var/lib/illumio-pce/tmp`.

PCE 2x2 Multi-Node Cluster for 2,500 VENs

Storage Device	Partition mount point	Size to Allocate	Node Types	Notes
Device 1, Partition A	`/`	16GB	Core, Data	The size of this partition assumes the system temporary files are stored in `/tmp` and core dump file size is set to zero.
Device 1, Partition B	`/var/log`	32GB	Core, Data	The size of this partition assumes that PCE application logs and system logs are both stored in `/var/log`. PCE application logs are stored in the `/var/log/illumio-pce` directory.
Device 1, Partition C	`/var/lib/illumio-pce`	Balance of Device 1	Core, Data	The size of this partition assumes that Core nodes use local storage for application code in `/var/lib/illumio-pce`, and also assumes that PCE support report files, and other temporary (ephemeral) files, are stored in `/var/lib/illumio-pce/tmp`.
Device 2, Single partition. Applicable in a two-storage-device configuration	`/var/lib/illumio-pce/data/Explorer`	All of Device 2 (250GB)	Data	For network traffic data in a two-storage-device configuration for the data nodes, it should be a separate device that is mounted on this directory. Set the `runtime_emv.yml` to `data_dir: /var/lib/illumio-pce/data/Explorer`, which will automatically create a subdirectory called `/var/lib/illumio-pce/data/Explorer/traffic_datastore` The partition mount point and the runtime setting must match. If you customize the mount point, make sure that you also change the runtime setting accordingly.

PCE 2x2 Multi-Node Cluster for 10,000 VENs and

PCE 4x2 Multi-Node Cluster for 25,000 VENs

Storage Device	Partition mount point	Size to Allocate	Node Types	Notes
Device 1, Partition A	/	16GB	Core, Data	The size of this partition assumes the system temporary files are stored in `/tmp` and core dump file size is set to zero.
Device 1, Partition B	`/var/log`	32GB	Core, Data	The size of this partition assumes that PCE application logs and system logs are both stored in `/var/log`. PCE application logs are stored in the `/var/log/illumio-pce` directory.
Device 1, Partition C	`/var/lib/illumio-pce`	Balance of Device 1	Core, Data	The size of this partition assumes that Core nodes use local storage for application code in `/var/lib/illumio-pce`, and also assumes that PCE support report files, and other temporary (ephemeral) files, are stored in `/var/lib/illumio-pce/tmp`.
Device 2, Single Partition Applicable in a two-storage-device configuration	`/var/lib/illumio-pce/data/traffic`	All of Device 2 (1TB)	Data	For network traffic data in a two-storage-device configuration for the data nodes, it should be a separate device that is mounted on this directory. In `runtime_env.yml`, set the `traffic_datastore : data_dir` parameter to match the value of the partition mount point (see previous column) as follows: `traffic_datastore: data_dir: /var/lib/illumio-pce/data/traffic`. The partition mount point and the runtime setting must match. If you customize the mount point, make sure that you also change the runtime setting accordingly.

Runtime Parameters for Traffic Datastore on Data Nodes

For the traffic datastore, set the following parameters in `runtime_env.yml`:

`traffic_datastore:`

> `data_dir`: `path_to_second_disk (e.g. /var/lib/illumio-pce/data/traffic)`
>
> `max_disk_usage_gb`: Set this parameter according to the table below.
>
> `partition_fraction`: Set this parameter according to the table below.

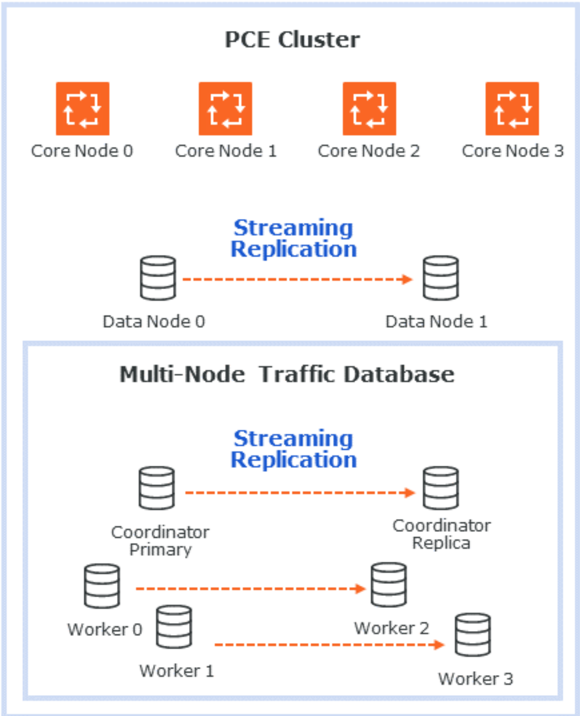
The recommended values for the above parameters, based on PCE node cluster type and estimated number of workloads (VENs), are as follows:

Setting	2x2 | 2,500 VENs	2x2 | 10,000 VENs	4x2 | 25,000 VENs	Note
`traffic_datastore:max_disk_usage_gb`	100 GB	400 GB	400 GB	This size reflects only part of the required total size, as detailed in PCE Capacity Planning [22]. The remaining disk capacity is needed for database internals and data migration during upgrades.
`traffic_datastore:partition_fraction`	0.5	0.5	0.5	

For additional ways to avoid disk capacity issues, see "Manage Data and Disk Capacity" in the PCE Administration Guide.

Scale Traffic Database to Multiple Nodes

When deploying the PCE, you can scale traffic data by sharding it across multiple PCE data nodes. In this way, you can store more data and improve the performance of read and write operations on traffic data. The traffic database is sharded by setting up two coordinator nodes, each of which has at least one pair of worker nodes.



Hardware Requirements for Multi-Node Traffic Database

The following table shows the minimum required resources for a multi-node traffic database.

CPU	RAM	Storage	IOPS
16 vCPU	128GB	1TB	5,000

Cluster Types for Multi-Node Traffic Database

The following PCE cluster types support scaling the traffic database to multiple nodes:

- `4node_dx` - 2x2 PCE with multi-node traffic database. The 2x2 numbers do not include the coordinator and worker nodes.
- `6node_dx` - 4x2 PCE with multi-node traffic database. The 4x2 numbers do not include the coordinator and worker nodes.

Node Types for Multi-Node Traffic Database

The following PCE node types support scaling the traffic database to multiple nodes:

- `citus_coordinator` - The sharding module communicates with the PCE through the coordinator node. There must be two (2) coordinator nodes in the PCE cluster. The two nodes provide high availability. If one node goes down, the other takes over.
- `citus_worker` - The PCE cluster can have any even number of worker nodes, as long as there are at least two (2) pairs. As with the coordinator nodes, the worker node pairs provide high availability.

Runtime Parameters for Multi-Node Traffic Database

The following runtime parameters in `runtime_env.yml` support scaling the traffic database to multiple nodes:

- `traffic_datastore:num_worker_nodes` - Number of traffic database worker node pairs. The worker nodes must be added to the PCE cluster in sets of two. This supports high availability (HA). For example, if there are 4 worker nodes, `num_worker_nodes` is 2.
- `node_type` - This runtime parameter can be assigned one of the values `citus_coordinator` and `citus_worker`. They are used to configure coordinator and worker nodes.
- `datacenter` - In a multi-datacenter deployment, the value of this parameter tells which datacenter the node is in. The value is any desired descriptive name, such as "west" and "east."

Set Up a Multi-Node Database

When setting up a new PCE cluster with a multi-node traffic database, use the same installation steps as usual, with the following additions.

- Install the PCE software on core, data, coordinator, and worker nodes, using the same version of the PCE on all nodes.
- There must be exactly two (2) coordinator nodes. There must be two (2) or more pairs of worker nodes.
- Set up the `runtime_env.yml` configuration on every node as follows. For examples, see Example Configurations for Multi-Node Traffic Database [33].
  - Set the cluster type to `4node_dx` for a 2x2 PCE or `6node_dx` for a 4x2 PCE.
  - In the `traffic_datastore` section, set `num_worker_nodes` to the number of worker node pairs. For example, if the PCE cluster has 4 worker nodes, set this parameter to 2.
  - On each coordinator node, in addition to the settings already desribed, set `node_type` to `citus_coordinator`.

- On each worker node, in addition to the settings already desribed, set `node_type` to `citus_worker`.
- If you are using a split-datacenter deployment, set the `datacenter` parameter on each node to an arbitrary value that indicates what part of the datacenter the node is in.

For installation steps, see Install the PCE and UI [46] for a new PCE, or Upgrade the PCE [65] for an existing PCE.

Example Configurations for Multi-Node Traffic Database

Following is a sample configuration for a coordinator node. This node is in a 4x2 PCE cluster (not counting the coordinator and worker nodes) with two pairs of worker nodes:

```
cluster_type: 6node_dx
node_type: citus_coordinator
traffic_datastore:
  num_worker_nodes: 2
```

Following is a sample configuration for a worker node. This node is in a 4x2 PCE cluster (not counting the coordinator and worker nodes) with two pairs of worker nodes:

```
cluster_type: 6node_dx
node_type: citus_worker
traffic_datastore:
  num_worker_nodes: 2
```

Following is a sample configuration for a split-datacenter configuration.

The following settings are for nodes on the left side of the datacenter:

```
cluster_type: 6node_dx
traffic_datastore:
  num_worker_nodes: 2
datacenter: left
```

The following settings are for nodes on the right side of the datacenter:

```
cluster_type: 6node_dx
traffic_datastore:
  num_worker_nodes: 2
datacenter: right
```

Port Ranges for Cluster Communication

> **NOTE**
> This topic applies to MNCs only and not to SNCs.

The following port ranges are needed for communications between the PCE cluster nodes.

Protocols	Port Range
TCP	3100 to 3600
TCP	5100 to 6300
TCP	8443 **IMPORTANT** When using an SLB for load balancing or your PCE core nodes use DNS load balancing on port 8443 TCP, your PCE cluster must be able to access the PCE VIP. This requirement also applies when you have defined any custom port for `front_end_https_management_port` in the PCE runtime settings.
TCP and UDP	8000 to 8400
TCP	11200 to 11300
TCP and UDP	24200 to 25300

Requirements for PCE Installation

Before installing the PCE, be sure your underlying systems are sufficient to successfully install and run the PCE. Check all the following system requirements.

Load Balancer Requirements

> **NOTE**
> These load balancer requirements apply only to MNCs, and do not apply to SNCs.

A server load balancer or DNS-level load balancer is required to distribute traffic to the PCE core nodes.

Configure the load balancer to use the Illumio REST API to monitor which cluster core nodes are available to receive requests. See the REST API Developer Guide for exact usage.

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

No authentication is required to call this API. An HTTP status code of 200 means the node is able to receive requests. Any other status code or no response means the node is unable accept requests. Unhealthy or unresponsive nodes should be removed from the load balancing pool.

- The PCE Health Check API can experience up to a 30-second delay to return the actual status of the node.
- In the 4x2 configuration, a maximum of two core nodes are available (return a status code of 200) at any time.
- When using a DNS load balancer, it should only serve IP addresses for the cluster FQDN of those nodes that respond with a 200 to the `/node_available` API. For rapid failover in the event of a core node failure, Illumio recommends a DNS TTL of between 30 and 60 seconds.

PCE IP Address

Illumio recommends a statically-assigned IP address. By default, the PCE automatically uses the first available private IP address on the node. The PCE does not automatically bind to a public IP address.

When you use a public IP address or the node has multiple interfaces, you need to configure the PCE with the interface you want to use. To do so, set `internal_service_ip` in the configuration file `runtime_env.yml`. For example:

```
internal_service_ip: 10.2.8.89
```

To configure networking, see your OS vendor's documentation on the `ifcfg-ethN` script.

DNS Requirements

Your Domain Name System (DNS) must resolve the PCE's fully qualified domain name (FQDN). The FQDN must be resolvable on all managed workloads, on all nodes in the PCE cluster, and for all users of the PCE web console and REST API.

If you are using DNS-level load balancing, the PCE FQDN should resolve to the IP addresses of the core nodes. If you are using a server load balancer, the PCE FQDN should resolve to the VIPs of the server load balancer.

SMTP Requirements

An SMTP relay is required to send user invitations and “forgot password” email replies from the PCE.

The SMTP configuration parameter during PCE installation is `smtp_relay_address`. Allowable values are either an IP address with its SMTP port (default 587) or a resolvable FQDN with the SMTP port.

TLS Requirements

PCE communication is secured using the Transport Layer Security (TLS) protocol, the successor to the deprecated Secure Sockets Layer (SSL) protocol. TLS is used for securing the following communication sessions:

- User access to the PCE web console and REST API over the HTTPS protocol.
- Communication between the PCE and VENs.

  VEN-to-PCE communications for the EventService (default is port 8444) are secured by the ECDHE suite of cryptographic ciphers, which use an elliptic curve Diffie-Hellman key exchange. This exchange is signed with RSA signature algorithms.

- Communication between PCE nodes in a multi-node cluster.

If you want to generate a temporary, self-signed certificate, see Understanding Illumio Trial Certificates in the Knowledge Base (log in required).

For an in-depth discussion of deploying the PCE with TLS, see Preparing Certificates for a PCE deployment in the Knowledge Base (log in required).

X.509 Certificate

An X.509 server certificate must be installed on each PCE node during installation. When any client (the VEN) opens a TLS session to the PCE (for example, pairing a workload, accessing the PCE web console, retrieving updated policy), the PCE presents the server certificate to secure the communication. The server certificate is uploaded as part of a certificate bundle that contains the server certificate and the chain of CA certificates (Intermediate or Root) to establish the chain of trust back to a Root CA.

> **CAUTION**
> The client must be able to validate the chain of trust back to the Root CA for this certificate; otherwise, the TLS handshake fails. You might need to add all the certificates in the chain of trust to the keychain of the client.

The certificate package for the Illumio PCE must meet the following basic criteria:

- The file must contain PEM-encoded certificates.
- The subject value and issuer of the certificate must start with a leading slash character (/).
- As a best practice, duplicate the subject in the Subject Alternative Name (`subjectAlt-Name`).
- The certificate's signature algorithm must be SHA256WithRSAEncryption.
- The certificate's signature algorithm must *not* be RSASSA-PSS.
- The file must contain the server certificate and the entire certificate chain necessary to establish the chain of trust back to a Root CA.

  1. The package must include all of the CA certificates (Intermediate and/or Root) needed to establish the chain of trust back to a Root CA.
     - If the certificate is generated by a Private CA, all certificates in the chain of trust back to the Root CA must be included. This includes the Root CA certificate and any applicable Intermediate CA certificates.
     - If the certificate is generated by a major Public CA (such as, VeriSign, GeoTrust, Entrust, or Thawte), any Intermediate CA certificates needed to establish the chain of trust back to the Public Root CA must be included.
  2. Pay careful attention to the order of the certificates in the bundle. The server certificate must be first. If you have an Apache-style bundle generated by a standard certificate request process, you need to open the file in a text editor and reverse the order of the certificates. Apache always expects the root certificate to come first, then any intermediates in order (from the root down), and the server certificate is last. The PCE uses nginx, which expects the opposite order. For additional details, see the Nginx documentation.
     The certificate bundle should look something like this:

```
-----BEGIN CERTIFICATE-----
<server cert goes here>
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
<intermediate CA cert goes here>
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
<root CA cert goes here>
-----END CERTIFICATE-----
```

- All certificates in the bundle must be valid for the current date, which depends on the system time being set correctly.
- A trusted root store must be available for OpenSSL to validate certificates.
- The certificate must match the PCE FQDN, which can be an exact match (for example, pce.mycompany.com) or a wildcard match (for example, *. mycompany.com). If separate northbound and southbound FQDNs are specified in the runtime parameters `pce_fqdn` and `agent_pce_fqdn`, both FQDNs must be present in the certificates.
- SNC does not require client certificate authentication. If a SAN is used, the CN is ignored. The CN only needs to match the FQDN if there is no SAN.

Client authentication is used between nodes in an MNC. Run the following command and verify `TLS Web Server Authentication, TLS Web Client Authentication` appears within the `X509v3 Extended Key Usage` section.

```
$ openssl x509 -text -noout -in pce.mycompany.com.bundle.crt
...
X509v3 Extended Key Usage:
         TLS Web Server Authentication, TLS Web Client Authentication
...
```

RSASSA-PSS Signature Algorithm Not Supported

The certificate signature algorithm RSASSA-PSS, which is based on PKCS 1 version 2.1, is not supported, because it cannot be validated. This limitation is a widely known problem with this signature algorithm.

The PCE certificate requires the SHA256WithRSEencryption signature.

CAUTION

If you use Microsoft Certificate Authority (CA) to sign PCE certificates, make sure to use the SHA256WithRSEencryption. PKCS#1 version 2.1 is enabled by default on Microsoft CAs and produces the unsupported RSASSA-PSS signature algorithm.

Minimum key length with RHEL8+ cryptographic policy set to FUTURE

If the cryptographic policy is set to FUTURE on RHEL8, RHEL9, and related operating systems, the RSA key length must be 3072 bits or greater. See this Knowledge Base article: RHEL8 VEN Offline with error: EE certificate key too weak.

Private Keys

The private key that matches the X.509 certificate must be installed on each PCE node during installation, and the following guidelines must be met:

- The private key must be PEM-encoded.
- The file must not be encoded in other ways, e.g., DER, PKCS7/P7B, PKCS8 or PKCS12/PFX.
- The file must not be password protected.

TLS Versions for Communications

The PCE uses Transport Layer Security (TLS) version 1.2 by default for VEN-to-PCE communications, the PCE's web server for the PCE web console, and the REST API.

- The PCE default minimum version is TLS 1.2.
- For VEN versions 18.1 and later, all VENs use TLS 1.2.
- Older operating systems might not support TLS 1.2. For example, SUSE VEN version 17.1.x, a legacy VEN version that is no longer supported by Illumio, requires minimum version TLS 1.0.
- Windows Server 2008 R2 SP1: The HTTP Client library, WinHttp, does not have the necessary API to limit SSL negotiation only to TLS 1.2. This must be configured through the Registry. See the Microsoft Support article Update to enable TLS 1.1 and TLS 1.2.

Changing the Minimum TLS Version

The default minimum TLS version is TLS 1.2. The minimum TLS version is configurable, but it is recommended that you leave the setting at its default of 1.2. Earlier TLS versions, such as 1.0 and 1.1, are considered less secure than 1.2, so it is recommended you do not use them. In some rare circumstances, you might need to change the minimum TLS version, such as when using older VEN operating systems that do not support TLS 1.2. In addition, you should verify that any browser you use is capable of negotiating the minimum version you set.

If you want to change the minimum TLS version, edit the following parameter in `runtime_env.yml`:

```
min_tls_version
```

The value of `min_tls_version` configures the PCE front end ports in `runtime_env.yml`:

- `front_end_https_port` (default 8443)
- `front_end_https_management_port` (defaults to `front_end_https_port`)
- `front_end_event_service_port` (default 8444)

Allowable values:

- `tls1_0` allows TLS 1.0, 1.1, and 1.2.
- `tls1_1` allows TLS 1.1 and 1.2.
- `tls1_2` allows TLS 1.2.

(Optional) Verify PCE Package Signature

For additional security, verify the identity of the downloaded PCE packages against the Illumio public key. The public key is available in the file `illumio_pce_pub.key`, which ships with the packages.

For information about using a public key to verify package signatures, see Checking a Package's Signature on the Red Hat Customer Portal.

(Optional) Configure SAML IdP for User Login

After installation, you can configure the PCE to rely on an external, third-party SAML identity provider system. See "Single Sign-on Configuration" in the PCE Administration Guide. The guide has step-by-step details for a wide variety of IdPs.

OS Setup and Package Dependencies

See PCE OS Support and Package Dependencies on the Illumio Support portal. You must be logged in to the Illumio Support Portal to view this article.

NTP

Set up a Network Time Protocol (NTP) client for time synchronization. It is recommended that you use chrony, although ntpd can also be used. On RHEL8, chrony is the default.

To install and configure the NTP client, use the procedure in the documentation for the client on your operating system.

After you finish installing the PCE, you can use the following command to verify that the NTP client is installed, running, and synchronized to a time source:

```
sudo -u ilo-pce illumio-pce-env check
```

IPTables

For the initial installation, you should disable iptables.

If iptables is enabled, you must configure it to allow inbound HTTPS connections to the PCE core nodes and service ports.

```
# service iptables stop
# On CentOS 7.x, use the systemctl stop firewalld command.
# chkconfig iptables off
```

Language: UTF-8

Set the system language to a UTF-8 variant of English: either `en_US.UTF-8` or `en_GB.UTF-8`.

Set the variable `LANG="en_US.UTF-8"` or `LANG="en_GB.UTF-8"` in the file `/etc/locale.conf`.

Trusted Public CA Store

A trusted root public Certificate Authority (CA) store must be available for OpenSSL to validate certificates.

If you rely on a certificate signed by a public CA, be sure to install the latest public root CA certificates `ca-certificates` package.

```
yum install ca-certificates
```

When your certificate is signed by a private CA or the signing CAs are already included in each node's trusted root CA store, the `ca-certificates` package is not required.

PCE Internal Syslog

The PCE comes with an internal syslog configuration. The purpose of the PCE internal syslog is to help organizations use syslog without installing it themselves. See Additional Installation Tasks. [55]

Process and File Limits and Kernel Parameters

This section describes how to set the process and file limits and OS kernel parameters that are required for PCE operation. The approach is different depending on whether you are configuring an SNC or MNC, and which operating system you are using, so look for the appropriate sections in the discussion that follows.

Three categories of settings must be configured:

- Process and file limits
- OS kernel parameters
- Kernel module tuning

> **WARNING**
>
> The parameter modifications described in this section are strict requirements and must be followed to ensure proper functionality of the Illumio Core. If an Illumio support case is opened, and analysis finds that these parameters are not met, you will be directed to meet these requirements before any additional troubleshooting can be performed.

Keep the following in mind when managing these parameters:

- Root access is needed for many of these procedures. Before you start, be sure you have login credentials for a user account with root permissions.
- When your settings are already greater than these, you do not need to reduce them to these values.
- Make sure you do not have any automated processes that change these values.

SNC Process and File Limits and OS Kernel Parameters

The following table shows the required process and file limits for single-node clusters. To set these values, see Set and Verify Process and File Limits.

Parameter	Value
`core (hard)`	`0`
`core (soft)`	`0`
`nofile (hard)`[1]	`65535`
`nofile (soft)`[1]	`65535`
`nproc (hard)`	`65535`
`nproc (soft)`	`65535`

Footnote:

[1] When you run additional processes on the PCE, such as monitoring or other operations processes, you might need to increase the value of `nofile`.

The following table shows the required OS kernel parameter values for single-node clusters. To set these values, see Set and Verify OS Kernel Parameters.

Parameter	Value
`fs.file-max`	`2000000`
`net.core.somaxconn`	`16384`
`vm.overcommit_memory`	`1`

The following table shows the required SNC kernel module tuning. To set this value, see Tune the Kernel Module.

Parameter	Value
`nf_conntrack hashsize`	`262144`

MNC Process and File Limits and OS Kernel Parameters

The following table shows the required process and file limits for multi-node clusters. To set these values, see Set and Verify Process and File Limits.

Parameter	Core Nodes	Data Nodes
`core (hard)`	`0`	`0`
`core (soft)`	`0`	`0`
`nofile (hard)`[1]	`65535`	`65535`
`nofile (soft)`[1]	`65535`	`65535`
`nproc (hard)`	`65535`	`65535`
`nproc (soft)`	`65535`	`65535`

Footnote:

[1] When you run additional processes on the PCE, such as monitoring or other operations processes, you might need to increase the value of `nofile`.

The following table shows the required OS kernel parameter values for multi-node clusters. To set these values, see Set and Verify OS Kernel Parameters below.

Parameter	Core Nodes	Data Nodes
`fs.file-max`	`2000000`	`2000000`
`net.core.somaxconn`	`16384`	Use system default setting
`vm.overcommit_memory`	Use system default setting	`1`

The following table shows the required kernel module tuning. To set this value, see Tune the Kernel Module.

Parameter	Core Nodes
`nf_conntrack hashsize`	`262144`

Set and Verify Process and File Limits

Process and file limits are set by editing configuration files and issuing commands. The techniques vary depending on the operating system version and which system management daemon you are using, `systemd` or `init.d`. If you are not sure which system management daemon is being used, run the following command:

```
ps -p1 | egrep "init|upstart|systemd"
```

CentOS 7.x or RHEL 7.x/8.x/9.x with systemd

On **every core and data node**, do the following steps:

1. As root, edit the following Illumio-specific configuration file:
   `/etc/systemd/system/illumio-pce.service.d/override.conf`
2. Place the following lines in the file.:

   ```
   [Service]
   LimitCORE=0
   LimitNOFILE=65535
   LimitNPROC=65535
   ```
3. Reload the daemon configuration and restart the service to apply the change:

   ```
   # systemctl daemon-reload
   # systemctl restart illumio-pce.service
   ```
4. Verify that the correct settings are now in effect. As the PCE runtime user, run the following commands. Verify that the output is as shown:

   ```
   sudo -u ilo-pce /bin/bash

   sudo -u ilo-pce ulimit -n
   65535
   sudo -u ilo-pce ulimit -c
   0
   sudo -u ilo-pce ulimit -u
   65535
   ```

See also the Linux systemd man page, especially "Table 1. Resource limit directives."

CentOS 7.x or RHEL 7.x/8.x/9.x with init.d

On **every core and data node**, do the following steps:

1. As root, edit the following Illumio-specific configuration file:
   `/etc/security/limits.d/99-illumio.conf`
2. Place the following lines in the file. The `ilo-pce` on each line indicates that the limits apply to only the PCE runtime user, which is `ilo-pce` unless this default user name was overridden during PCE installation. If you want the limits to apply to all users, use asterisks (`*`) instead of `ilo-pce`.

   ```
   ilo-pce    soft    core      0
   ilo-pce    hard    core      0
   ilo-pce    soft    nofile    65535
   ilo-pce    hard    nofile    65535
   ilo-pce    soft    nproc     65535
   ilo-pce    hard    nproc     65535
   ```
3. Restart the Illumio services:

   ```
   sudo -u ilo-pce illumio-pce-ctl restart
   ```

   If all nodes have been change already, restart with:

   ```
   sudo -u ilo-pce illumio-pce-ctl cluster-restart
   ```
4. Verify that the correct settings are now in effect. As the PCE runtime user, run the following commands. Verify that the output is as shown:

   ```
   sudo -u ilo-pce ulimit -n
   65535
   ```

```
sudo -u ilo-pce ulimit -u
65535
sudo -u ilo-pce ulimit -c
0
```

Set and Verify OS Kernel Parameters

Kernel parameters are set by editing configuration files and issuing commands. The commands are the same on all PCE-supported versions of CentOS and RHEL, but the techniques vary depending on whether you are configuring an SNC or MNC.

SNC: Set and verify OS kernel parameters

1. As root, edit the following Illumio-specific configuration file:
   `/etc/sysctl.d/99-illumio.conf`
2. Place the following lines in the file:
   ```
   fs.file-max          = 2000000
   vm.overcommit_memory = 1
   net.core.somaxconn   = 16384
   ```
3. Apply the settings:
   ```
   sysctl -p /etc/sysctl.d/99-illumio.conf
   ```
4. Verify that the correct settings are now in effect. As the PCE runtime user, run the following command. Verify that the output is as shown:
   ```
   sudo -u ilo-pce sysctl -a 2>/dev/null | egrep "fs.file-max|
   vm.overcommit_memory|net.core.somaxconn"
   fs.file-max = 2000000
   net.core.somaxconn = 16384
   vm.overcommit_memory = 1
   ```

See Configuring Kernel Parameters at Runtime in the Red Hat documentation.

MNC: Set and verify OS kernel parameters

1. As root, on **each core node**, edit `/etc/sysctl.d/99-illumio.conf` and add the following lines:
   ```
   fs.file-max          = 2000000
   net.core.somaxconn   = 16384
   ```
2. As you go, on **each data node**, apply the settings:
   ```
   sysctl -p /etc/sysctl.d/99-illumio.conf
   ```
3. As root, on **each data node**, edit `/etc/sysctl.d/99-illumio.conf` and add the following lines:
   ```
   fs.file-max          = 2000000
   vm.overcommit_memory = 1
   ```
4. As you go, on **each data node**, apply the settings:
   ```
   sysctl -p /etc/sysctl.d/99-illumio.conf
   ```
5. Verify that the correct settings are now in effect. As the PCE runtime user, run the following command. Verify that the output is as shown:

```
sudo -u ilo-pce sysctl -a 2>/dev/null | egrep "fs.file-max|
vm.overcommit_memory|net.core.somaxconn"
fs.file-max = 2000000
net.core.somaxconn = 16384
vm.overcommit_memory = 1
```

See Configuring Kernel Parameters at Runtime in the Red Hat documentation.

Tune the Kernel Module

Adjust the hash size setting for the kernel conntrack module as follows. For this setting, the commands are the same on all PCE-supported versions of CentOS and RHEL.

On **all core nodes**:

1. As root, run the following commands to tune the kernel conntrack module. The commands take effect immediately.

   ```
   # modprobe nf_conntrack
   # echo 262144 > /sys/module/nf_conntrack/parameters/hashsize
   ```
2. Run the following command to apply the same setting automatically on reboot:

   ```
   echo "options nf_conntrack hashsize=262144" > /etc/modprobe.d/
   illumio.conf
   ```
3. Verify that the correct setting is now in effect. Run the following command to inspect the hash size. Verify that the output is as shown:

   ```
   # cat /sys/module/nf_conntrack/parameters/hashsize
   262144
   ```

About Your Organization Name and ID

An organization is a group of policies and users in the Illumio Core. An organization can contain any number of users, workloads, and policy objects (rulesets, IP lists, services, and security settings). When you sign up with Illumio, you will receive an email invitation to create your company's organization in Illumio Core.

Have ready your full organization name, which you specify at installation.

For on-premise PCE deployments, installation creates an organization identifier (org ID) and assigns the value of `1` to org ID. The value `1` distinguishes your on-premises PCE from the Illumio Core Cloud (SaaS) service, where each customer has a unique org ID.

The org ID is needed with the REST API, where you set org-ID to `1` for the on-premises PCE, and for other purposes.

PCE Installation

This section provides step-by-step instructions for installing PCE software. Before performing these steps, be sure to understand the concepts in the Overview [18], and make sure your system is ready for installation as described in Prepare for PCE Installation [45].

Install the PCE and UI

When installing the PCE and UI packages together, you perform the following high-level steps:

1. Prepare for installation by planning your deployment and reviewing the prerequisites, such as capacity planning and OS setup. See Prepare for PCE Installation [21].
2. Download the software [46].
3. Install the PCE and UI [46] software.
4. Configure the PCE [47].
5. (Optional) Validate TLS certificate and private key [51].
6. Install the TLS certificate and private key [52].
7. Verify the runtime environment [53] was configured correctly.
8. Start the PCE [53].
9. Initialize the PCE [53].
10. Install Virtual Enforcement Nodes (VENs) to enable the PCE to manage your workloads as described in VEN Installation and Upgrade Guide

    At this point, the PCE is up and running, receiving communication about workloads from the VENs.

    After installing the PCE software, perform these additional procedures to complete your PCE deployment.
11. Configure backups [55].
12. (Optional) Configure the internal syslog. See (Optional) Configure PCE Internal syslog [55] for information.

> **NOTE**
> The following tasks describe installing the PCE as an MNC. When you install the PCE as an SNC, you do not repeat the steps on the additional nodes. You can disregard those instructions in the following tasks.

Download the Software

1. Download the software from the Illumio Support portal (login required).
2. On the **core nodes only**, copy the Illumio PCE UI RPM file to the `/tmp` folder. The following steps refer to this file as `illumio_ui_rpm`.
3. On **each node** in the cluster, copy the Illumio PCE software RPM file to the `/tmp` folder. The following steps refer to this file as `illumio_pce_rpm`.

Install the PCE and UI Packages

The packages to install depend on the type of PCE node:

- **Core nodes:** Two packages, the PCE RPM and UI RPM.
- **Data nodes:** One package, the PCE RPM.

1. On **each core node** in the cluster, log in as root and install the PCE RPM:

```
$ rpm -Uvh illumio_pce_rpm
```

For `illumio_pce_rpm`, substitute the path and filename of the software you downloaded from the Illumio Support portal.

2. On **each core node** in the cluster, log in as root and install the UI RPM:

```
$ rpm -Uvh illumio_ui_rpm
```

For `illumio_ui_rpm`, substitute the path and filename of the software you downloaded from the Illumio Support portal.

3. On **each data node** in the cluster, log in as root and install the PCE RPM:

```
$ rpm -Uvh illumio_pce_rpm
```

For `illumio_pce_rpm`, substitute the path and filename of the software you downloaded from the Illumio Support portal.

4. After installing the RPMs, configure the software using the PCE setup wizard. See Configure the PCE [47]for information.

Configure the PCE

Before running the PCE, set up its runtime configuration.

Use the PCE Runtime Environment File (`runtime_env.yml`) to configure the PCE software. By default, the file is located in `/etc/illumio-pce/runtime_env.yml`. You can create the `runtime_env.yml` file manually or use the PCE software setup script to create and modify the file using interactive prompts at the command line.

For detailed descriptions of the runtime parameters, see Reference: PCE Runtime Parameters [74].

IMPORTANT

- The `runtime_env.yml` file contains sensitive information that should be kept secret, such as encryption keys. Take steps to ensure the confidentiality of this file.
- The `runtime_env.yml` file is not included in automatic PCE backups. You must manually back up this file to a secure location.

Set Configuration File Location

By default, `runtime_env.yml` is located in `/etc/illumio-pce/runtime_env.yml`. You can override the default location by setting the `ILLUMIO_RUNTIME_ENV` environment variable. If you do, you must also set `ILLUMIO_RUNTIME_ENV` in in the file `/etc/sysconfig/illumio-pce` to enable the PCE software start-up script to find the file. Log in as root and run the following command (replace location with the actual full path).

```
root> echo "ILLUMIO_RUNTIME_ENV=location/runtime_env.yml" > /etc/sysconfig/
illumio-pce
```

For example, if the location is `/var/lib/illumio/data`, run the following command:

```
root> echo "ILLUMIO_RUNTIME_ENV=/var/lib/illumio/data/runtime_env.yml"
> /etc/sysconfig/illumio-pce
```

Run the PCE Setup Script

From the host command line, *as root*, run the following command to launch the setup script:

```
[root]# illumio-pce-env setup
```

The setup script interactively prompts you to provide values for configuration parameters. For descriptions of all the parameters, see Reference: PCE Runtime Parameters [74]. A few of these values, such as `node_type`, will not be the same on all nodes of the PCE cluster; however, many of the values will be the same on all the nodes.

> **WARNING**
>
> The `service_discovery_encryption_key` value *must* be identical on all the nodes in the cluster or the PCE won't start. Be sure to use the same value for this parameter on all nodes.

> **WARNING**
>
> The `cluster_type` runtime parameter *must* be set on all PCE nodes, except in a single node cluster (SNC).

When you start the setup script, it checks whether the `$ILLUMIO_RUNTIME_ENV` environment variable is set.

```
$ Illumio PCE Runtime Setup (new configuration -> ENV=my_pce.yml):
```

The `ILLUMIO_RUNTIME_ENV` variable controls where the runtime file will be stored. When the `ILLUMIO_RUNTIME_ENV` variable is not set, the setup script alerts you that the configuration is new and displays the default directory for the runtime file: `/etc/illumio-pce/runtime_env.yml`.

```
$ Illumio PCE Runtime Setup (new configuration)
```

General Configuration

The setup script displays descriptive help text followed by a prompt where you can accept the previous value, the default value, or enter a new value. When the field is optional, press Enter to leave the field empty or accept the default value (if one exists). Fields that have default values display `# default` next to the values.

The prompt shows the previous value in brackets:

```
node_type [core]:
```

Press Enter to use the value in brackets.

TIP

To determine whether a value is the previously set, default, or recommended value, enter a question mark (?) to display the default value when one exists:

```
opts => core [ data0 data1 ]
node_type [core]: ?
```

When a field has multiple options, type the first few characters of the option and press Tab to auto-complete the field or suggest choices. When prompted for a directory or filename, using auto-complete can help you quickly populate the field.

Press CTRL+C to escape to a control menu, which provides the following options:

- Quit without saving
- Restart the script (with an optional field value)
- Skip to a future field (with a field value)
- Save (with an optional target file)
- Exit

For example, entering this command saves the configuration to a different file and quits the setup.

```
$ Type (q)uit, (r)estart, (f)ield, (s)ave to file or default resume:
save /tmp/sample.cfg
```

Command-line Batch or List Mode

To operate the setup script from the command line, use the `--batch` option. Instead of prompting for each value, it accepts any previous or default values automatically. When the configuration is missing required fields, the script displays an error and returns a non-zero exit code.

To set a value on the command line:

```
[root]# illumio-pce-env setup front_end_https_port=7443
pce_fqdn="sample.illumio.com" -b
```

This command sets the values instead of prompting for them. You can also pre-set the values in non-batch mode by using `key=value` arguments.

NOTE

Batch mode automatically saves the new configuration in a new configuration file unless there is an error.

To display the currently configured values and replace them with command-line values, use the `--list` option. The `--list` option does not prompt for values or save the configuration to the `runtime_env.yml` file. The `--list` option is useful to validate your TLS certificate [51].

Advanced Runtime Environment Parameters

Your Illumio Support Representative might provide advanced parameters to add to your `runtime_env.yml` file. When you include the name of these parameters on the command line, the setup script prompts for them.

```
[root]# illumio-pce-env setup advanced_parameter_name_1
advanced_parameter_name_2 ...
```

Additional Options

When using the setup script, several additional options are available. You can use `-h` to display these options.

Usage

```
[root]# illumio-pce-env setup [options...]
[field[:field...]=[value[,value...]]...
```

Display Options

Option	Descriptions
`-b, --batch`	Don't prompt for field values.
`-d, --default`	Show default values.
`-e, --empty`	Display empty fields (implies `-d`).
`-f, --field *[:*][,...]`	Specify a field pattern list; only process these items.
`-g, --[no-]guide`	Show descriptive information for each field where available (default).
`-h, --help`	Provide usage statement.
`-m, --macros`	Show list of available shortcut keys.
`-o, --[no-]optional`	Process optional fields (default).
`-q, --quiet`	Don't display help text for each field (same as `--no-guide`)
`-r, --reveal`	Don't mask secret keys in field output.
`-t, --text`	Use regular text instead of colors.

File Options

Option	Description
`-c, --config <file>`	Process a different environment file (`new=-`).
`-s, --save <file>`	Save results to a different file (`stdout=-, system default=!`).
`-z, --zap`	Remove pre-existing default fields.

(Optional) Validate and Configure TLS Certificate

The PCE validates your TLS certificates at start up and displays an error message when the certificate or its chain of trust is invalid.

For information on the contents and formats of your certificates, see Requirements for PCE Installation [34].

You can validate the certificates yourself before or after configuring the PCE as described in Run the PCE Setup Script [48].

To validate your TLS certificate yourself, including the chain of trust and other aspects, run the following command:

```
illumio-pce-env setup --list
```

The specifying `--list` option checks your configuration and certificates, and indicates possible problems; it does not create a new `runtime_env.yml` configuration file.

Validate After Configuring PCE

To validate the certificates you have already configured and saved in the locations defined in the `runtime_env.yml` file, run the following command:

```
illumio-pce-env setup --list --test 5
```

Specify a verbosity level argument—1 (least) to 5 (most)—with the `--test` option. At verbosity level 5, the command displays the results of its certificate validation.

Alternative Syntax for Certificate Validation

After configuring the PCE, you can validate your certificates in the following additional ways:

- `illumio-pce-env setup --list --test 5:`*`some.alternative.hostAndDomainName`*

  This syntax checks the certificate and chain against the specified `some.alternative.hostAndDomainName`, such as the FQDN you plan to use for the PCE in production.
- `illumio-pce-env setup --list --test 5+`

  The `+` syntax creates a loopback OpenSSL server running on port 4433 and attempts to curl to it.

Validate Before Configuring PCE Certificates

If you have not configured your `runtime_env.yml` file yet, and want to validate your certificates before copying them to your planned production location, run the following command.

```
# illumio-pce-env setup --batch --list \
email=required@emailaddress node=value \
cert=/path/to/cert \
pkey=/path/to/private_key \
trust=/path/to/certificate_chain \
--test 5
```

Option	Description
`email=required@emailaddress`	(Required) Your email address.
`node=value`	Topology to check. For allowable values, see the parameter `node_type` in Optional Runtime Parameters [77] and see the discussion in Nodes and Clusters. [19]
`cert=/path/to/cert`	The absolute path to your certificate.
`pkey=/path/to/private_key`	The absolute path to your certificate's private key.
`trust=/path/to/certificate_chain`	The absolute path to your certificate's CA chain of trust.

Messages, Errors, and Warnings

These messages indicate correctly configured certificates:

- `Valid: Certificate chain is verified`
- `Valid: web_service_certificate tests passed.`

These error message indicate possible problems with your certificates:

- `Warning: group xxx can write to web_service_certificate`
- `Error: unable to find trusted_ca_bundle yyy`
- `Warning: trusted_ca_bundle missing or inaccessible.`
- `Missing CA`
- `Error: unable to verify certificate chain`
- `Error: unable to validate web_service_certificate`

Install Certificate

Copy the TLS certificate and private key to each node in your deployment.

You can store the files in any readable location on the node. The PCE RPM installation creates the `/var/lib/illumio-pce/cert` directory where you can store these files.

The certificate and private key must be readable by the PCE runtime user.

Verify the PCE Runtime Environment

1. After configuring the `runtime_env.yml` file, run the environment check command as the PCE runtime user to ensure the node is set up correctly:

   ```
   sudo -u ilo-pce illumio-pce-env check
   Checking PCE runtime environment.
   OK
   ```

   This command checks various aspects of the PCE setup. For example, it verifies that the NTP client is installed, running, and synchronized to a time source.
2. Correct any errors.
3. Proceed to the next task: Start and Initialize the PCE [53].

Start and Initialize the PCE

Starting and initializing the PCE are the final steps in installing it. After completing these steps, you are ready to install VENs on hosts in your environment as described in the VEN Installation and Upgrade Guide.

Start the PCE

As the PCE runtime user, perform the following steps:

1. On *all nodes*, start the PCE at runlevel 1:

   ```
   sudo -u ilo-pce illumio-pce-ctl start --runlevel 1
   ```

   Troubleshooting: If this command fails, verify that you have set `service_discovery_encryption_key` to the same value in `runtime_env.yml` on all PCE nodes.

   Wait while all the nodes process the start command, which can take up to 10 minutes. When a node has finished, its status is `RUNNING`.
2. On *all nodes*, verify that they started:

   ```
   sudo -u ilo-pce illumio-pce-ctl status
   ```

   Expected output:

   ```
   Checking Illumio Runtime            RUNNING 0.38s
   ```

   If any nodes do not start after 10 minutes, check the following issues:
   - Network connectivity between nodes and iptables is configured correctly. See IP Tables [39] for information.
   - The certificates must be configured correctly. See TLS Requirements [35] for information.
   - The system locale must be UTF-8. See Language: UTF-8 [39] for information.
   - The runtime environment is configured correctly. See Verify the PCE Runtime Environment [53] for information.

Initialize the PCE

As the *PCE runtime user*, perform the following steps:

1. On *any node*, initialize the PCE database:

```
sudo -u ilo-pce illumio-pce-db-management setup
```

2. On the *data0 node*, bring the system up to runlevel 5:

```
sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
```

3. On *any core node*, check the status of the cluster:

```
sudo -u ilo-pce illumio-pce-ctl cluster-status
```

   Make sure the cluster status is `RUNNING` before proceeding to the next step.

4. On *any core node*, create the initial PCE user and organization name:

```
sudo -u ilo-pce illumio-pce-db-management create-domain --user-name user-
email-address --full-name user-full-name --org-name organization-name
```

   You are prompted for a password. The password must conform to these restrictions: at least 8 characters, no more than 128 characters, at least 1 upper case character, 1 lower case character and 1 number.
   For example:

```
sudo -u ilo-pce illumio-pce-db-management create-domain --user-name
myuser@mycompany.com --full-name 'Joe User' --org-name 'ACME Inc.'

Reading /var/illumio-pce-data/runtime_env.yml.
INSTALL_ROOT=/var/illumio-pce
RENV=production (defaulted because not set in runtime_env.yml)
Please enter a password with at least 8 characters with one uppercase,
one lowercase and one number.

Enter Password:
Re-enter Password:
------------------------------------------------------------
Running cd /var/illumio-pce/illumio/webservices/people &&
RAILS_ENV=production bundle exec rails
runner script/create_org_owner
--output-file /tmp/illumio/org.yml --user-name myuser@mycompany.com
--create-org
--org-name 'ACME Inc.'
Completed in 5.471846432 sec. Exit Code = 0
------------------------------------------------------------
Running cd /var/illumio-pce/illumio/webservices/agent &&
RAILS_ENV=production bundle
exec rails runner script/create_org_defaults
--input-file /tmp/Illumio/org.yml
Completed in 5.609754678 sec. Exit Code = 0
------------------------------------------------------------
Running cd /var/illumio-pce/illumio/webservices/login &&
RAILS_ENV=production
ILO_*********************bundle exec rails runner
script/setup_initial_config --org-data /tmp/Illumio/org.yml
--user-name myuser@mycompany.com
--full-name 'Joe User'
domain_name=mycompany.com
Completed in 5.303522871 sec. Exit Code = 0
Done.
```

5. (RHEL 7+ only) Check to be sure the expected session limits for `nofile` and `nproc` meet the minimum requirements for the PCE (see the "Process and File Limits" topic). Use the following command:

```
cat /proc/$(pgrep -f config_listener.rb)/limits | grep -e open -e
processes
```

If the limits are too low, correct the issue. See Session Limits Too Low [86].

6. Point a web browser to the PCE FQDN and log in using the account you just created. You should see the PCE web console.

VEN Deployment

In addition to deploying PCE nodes, you must also deploy the Virtual Enforcement Node (VEN) on your distributed, on-premise systems. For more information, see the VEN Installation and Upgrade Guide.

Additional PCE Installation Tasks

After installing the PCE, perform these additional tasks.

Configure PCE Backups

You should maintain and perform regular backups of the PCE database based on your company's backup policy. Additionally, always back up your PCE database before upgrading to a new version of the PCE. See "PCE Database Backup" in the PCE Administration Guide.

Internal Syslog and Events Configuration Required

This section applies to you if you are:

- Performing a fresh installation of Illumio 20.2.0 or later rather than upgrading from a previous version, and
- You want to send events and traffic flow summaries to an external SIEM.

For new installations, you must configure the syslog and set up events forwarding.

In previous PCE versions, a `local` syslog configuration was created by default. This local setting is no longer created. If you want to gather events data, the internal syslog must be configured. This was previously an optional installation step.

You must configure the following:

- Set up the internal syslog. See (Optional) Configure PCE Internal syslog [55].
- Set up events forwarding. See "Events Settings" in the Events Administration Guide.

If you are upgrading from a previous PCE version, you can also do this configuration, if needed. However, it is more likely that you already have an appropriate configuration in place.

(Optional) Configure PCE Internal syslog

Configuring the PCE internal syslog is optional only if you are performing either of these tasks:

- You are upgrading to Illumio 21.2.0 or later from an earlier version where you already have an appropriate configuration in place.

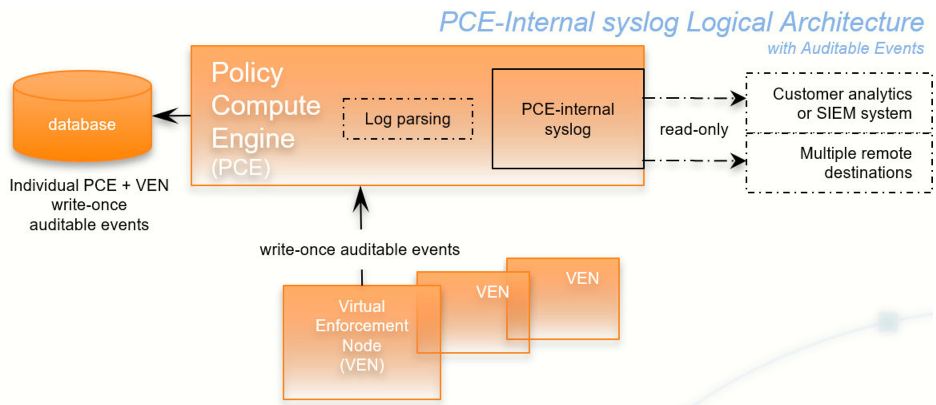
- You are performing a fresh installation of Illumio 21.2.0 or later, but you don't care about gathering events data or sending events and traffic flow summaries to an external SIEM.

In every other case, it is required.

With the PCE internal syslog, you use the PCE web console to control and configure the relaying of syslog messages from the PCE to multiple remote destinations.

This feature eliminates the need to manage syslog on the PCE by yourself.

You can achieve a smooth transition from existing syslog installations by using a default configuration called "Local." Using this default, the PCE internal syslog relays messages to the existing syslog.



Utilizing the internal syslog works well with the PCE's auditable events data. See the Events Administration Guide.

The PCE internal syslog has the following features:

- Syslog message routing to an unlimited number of remote destinations
- Auditable events for syslog service, as required by Common Criteria
- Integration with PCE Support Reports
- Common timestamps defined by RFC 3339, including fractional timestamps, such as milliseconds
- PCE log rotation and disk usage management
- SIEM support by enabling sending events to remote destinations
- Optional data-in-motion encryption

Do Not Write Additional Information to `log_dir`

Though *not recommended*, you can put the PCE internal syslog into operation while still running any syslog implementation you already have. However, keep the following information in mind.

> **CAUTION**
> Do not store auditable events in `log_dir`
>
> If you continue to use a previously configured syslog (prior to Illumio Core version 18.2), Illumio recommends that your own local syslog configuration be changed to *not* store any additional information in `log_dir`. The `log_dir` parameter in `runtime_env.yml` defines where logs are written and by default is `/var/log/illumio-pce`. This recommendation includes avoiding storing your auditable events logs in this directory.
>
> The PCE Support Report includes all data in this directory. Illumio considers the auditable event information as private, confidential data. Storing it in `log_dir` could inadvertently release this information by way of the PCE Support Report to persons other than your organization's auditors.

Configure Events and Syslog

After installing the PCE, configure events and the syslog server using the PCE web console.

For information, including configuring remote syslog destinations, see Events Settings in Events Administration Guide.

(Optional) Customize PCE Log File Rotation

Internal PCE log file rotation is governed by two values: maximum file size (default: 100MB) and maximum retention (default: 10 files). In larger-scale deployments, these values could be an insufficient amount of log data to successfully troubleshoot runtime issues.

To customize the rotation of PCE log files, run the following command:

```
sudo -u ilo-pce illumio-pce-env logs --modify logfile[:size][/rotation]
```

In `logfile`, enter the name of the file. If you do not already know the name of the log file, run this command to list all logs:

```
sudo -u ilo-pce illumio-pce-env logs
```

In `size`, specify a number and append `m` to specify a size in MB or `g` to specify a size in GB. In `rotation`, enter a number to control how many past rotated log files to keep. When this number is exceeded, the oldest file is deleted. To return to the default log rotation values of 100MB and 10 files, run this command with `logfile` alone, without the `size` or `rotation` parameters.

For example:

Argument	Result
`haproxy.log:1g/20`	Rotate the haproxy log when it reaches 1GB, and keep the last 20 rotated files.
`haproxy.log:3m`	Set the haproxy.log to 3MB, indicated by the `m`.
`haproxy.log/5`	Keep the 5 most recent `haproxy.log` files after rotation. Discard older ones.
`nginx.log`	Return the `nginx.log` file to the default settings.

To confirm that the hosts have sufficient disk space to accommodate the log files with these rotation settings, run this command:

```
sudo -u ilo-pce illumio-pce-ctl check-env
```

It issues a warning if the log usage is too great for the partition size.

(Optional) Set Path to Custom TLS Certificate Bundle

When you enable Transport Layer Security (TLS) mutual authentication, the channel to the remote syslog destination can be secured by your own TLS CA certificate bundle. A CA bundle is a file that contains root and intermediate certificates. The end-entity certificate along with a CA bundle constitutes the certificate chain.

The value of the `runtime_env.yml` file optional parameter `trusted_ca_bundle` is the path to your own CA certificate bundle.

- When a custom TLS bundle is provided by the user during configuration, this bundle is used for certificate verification.
- When a custom TLS bundle is not configured for a particular destination, the PCE trust store is used (`runtime_env.yml` parameter `trusted_ca_bundle`).

Remote Destination Setup for Syslog Server

Enabling TLS with the syslog protocol allows you to secure the communication to your syslog service with public CA certificates or with TLS certificates from your own CA.

On the remote syslog server, ensure restricted access to the data by relying on the OS-level user access mechanisms. In addition, limit the number of users allowed access to the syslog storage itself. If possible, rely on an enterprise-class log management system to post-process the event data.

RFC 5424 Message Format Required

Ensure that your remote syslog destination is configured to use the message format defined by RFC 5424, The Syslog Protocol, with the exception.

Traffic flow summary messages include a prefix of an octal number, like the string **611** high-lighted in bold at the beginning of the snippet of a LEEF record below. Ensure that your parsing programs on the remote syslog destination account for this prefix:

```
611 <14>1 2018-08-06T11:47:26.000000+00:00 core1-2x2devtest59 illumio_pce/
collector 22724 - [meta sequenceId="3202"] sec=556046.963 sev=INFO
```

```
pid=22724 tid=30548820 rid=e163020f-32c5-4c59-ab06-dfb93b60ff4e LEEF:2.0|
Illumio|PCE|18.2.0|flow_allowed|cat=flow_summary
...
```

NOTE

Notes on RFC 5424

- You must ensure that your remote syslog uses the `network(flags(syslog-protocol))` form for receiving messages.
- RFC 5424-formatted messages might not be fully functional with rsyslog versions earlier than 5.3.4.

Message Size: 8K

The size of the PCE internal syslog messages is up to 8K bytes. However, many implementations of syslog have a default message size of 4K bytes. Ensure that your remote syslog configuration is set for 8K message size. Configuring the remote destination's syslog message size depends on your implementation of syslog. Consult your vendor documentation for information.

After PCE Installation

This section describes some of the basic things you can see immediately after installing the PCE.

WARNING

Any adverse effects of using security scanners or other mechanisms intended to probe or exercise various parts of the PCE or its environment cannot be anticipated by Illumio and are therefore unsupported. Doing so may interfere with or even prevent the PCE from operating properly.

RPM Installation Directories

The PCE software RPM installs to the following directories:

Location	Contents at Installation	Permissions / Ownership
`/opt/illumio-pce/`	PCE software	`dr-xr-x---. root ilo-pce`
`/etc/illumio-pce`	Empty	`drwxr-x---. root ilo-pce`
`/etc/init.d/illumio-pce`	Service script	`-rwxr-xr-x. root root`
`/var/lib/illumio-pce/`	Empty	`drwxr-x---. root ilo-pce`
`tmp/`		`drwx------. ilo-pce ilo-pce`
`runtime/`		`drwx------. ilo-pce ilo-pce`
`data/`		`drwx------. ilo-pce ilo-pce`
`keys/`		`drwx------. ilo-pce ilo-pce`
`cert/`		`drwxr-x---. root ilo-pce`
`/var/log/illumio-pce`	Log files	`drwx------. ilo-pce ilo-pce`

RPM Runtime User and Group

The PCE installation creates a runtime user and group named `ilo-pce` to run the PCE software. For security, the `ilo-pce` user is configured without a login shell or home directory.

> **CAUTION**
> For better security, do not give the `ilo-pce` user a login shell or home directory.

You should run PCE commands as root or as a user belonging to the `ilo-pce` group. You run the PCE software with `sudo`, as shown throughout this guide:

```
sudo -u ilo-pce somePCEcommand
```

You might put several users into the `ilo-pce` group for shared maintenance or other needs. However, only the `ilo-pce` user is actually used to run the software.

PCE Control Interface and Other Commands

The Illumio PCE control interface `illumio-pce-ctl` is a command-line tool for performing key tasks for operating your PCE cluster, such as starting and stopping nodes, setting cluster runlevels, and checking the cluster status.

> **IMPORTANT**
> In this guide, all command-line examples based on an RPM installation. When you install the PCE using the tarball, you must modify the commands based on your PCE user account and the directory where you installed the software.

The PCE includes other command-line utilities used to set up and operate your PCE:

- `illumio-pce-env`: Verify and collect information about the PCE runtime environment.
- `illumio-pce-db-management`: Manage the PCE database.

The PCE control interface can only be executed by the PCE runtime user (`ilo-pce`), which is created during the PCE RPM installation.

Control Command Access with `usr/bin`

For easier command execution, PCE installation creates softlinks in `/usr/bin` by default for the Illumio PCE control commands. The `/usr/bin` directory is usually included by default in the `PATH` environment variable in most Linux systems. When your `PATH` does not include `/usr/bin`, add it to your `PATH` with the following command. You might want to add this command to your login files (`$HOME/.bashrc` or `$HOME/.cshrc`).

```
export PATH=$PATH:/usr/bin
```

Syntax of `illumio-pce-ctl`

To make it simpler to run the PCE command-line tools, you can run the following Linux softlink commands or add them to your `PATH` environment variable.

```
$ cd /usr/bin
sudo ln -s /opt/illumio-pce/illumio-pce-ctl ./illumio-pce-ctl
sudo ln -s /opt/illumio-pce/illumio-pce-db-management ./illumio-pce-db-
management
sudo ln -s /opt/illumio-pce/illumio-pce-env ./illumio-pce-env
```

After these commands are executed, you can run the PCE command-line tools using the following syntax:

```
sudo -u ilo-pce illumio-pce-ctl sub-command --option
```

Where:

`sub-command` is an argument displayed by `illumio-pce-ctl --help`.

PCE Service Script `illumio-pce` for Boot

The `illumio-pce` service script in `/etc/init.d/illumio-pce` switches to the runtime user (`ilo-pce`) prior to running other PCE programs. The primary purpose of the `init.d` service script is to start the product on boot. The service script can also be run with the `/sbin/service` command:

```
$ service illumio-pce
Usage: illumio-pce {start|stop|restart|[cluster-]status|{set|get}-runlevel|
control|database|environment|setup}
```

PCE Runlevels

PCE runlevels define the system services started for common operations, such as upgrade, downgrade, and restore.

The runlevel is set with the following command:

```
illumio-pce-ctl set-runlevel numeric_runlevel
```

The `numeric_runlevel` varies by type of operation.

Setting the runlevel might take some time to complete, depending on the cluster configuration. Check the progress with the following command:

```
illumio-pce-ctl cluster-status -w
```

Alternative: Install the PCE Tarball

You can use these alternative steps instead of the normal installation procedure described in Install the PCE and UI [46].

> **NOTE**
> The preferred installation mechanism is the RPM distribution, which is easier than the tarball installation.

Process for Installing PCE Tarball

If you are installing the PCE tarball distribution, perform the following tasks on each nodes in your deployment:

1. Create the PCE user account.
2. Resolve OS dependencies.
3. Create the directory structure for the PCE. The PCE tarball supports a configurable directory structure. This feature allows you to choose the directory structure that best meets your needs.

   The following table lists the directories used by the PCE. You need to create these directories and update the listed PCE Runtime Environment File with the proper values.

Directory	Use	Permissions	Example
`install_root`	PCE binaries and scripts	Read/Execute	`/opt/illumio-pce`
`persistent_data_root`	A writable location where the PCE writes its persistent data Must be owned by the user that runs the PCE.	Read/Write	`/var/lib/illumio-pce/data`
`runtime_data_root`	A writable location where the PCE writes runtime data Must be owned by the user that runs the PCE.	Read/Write	`/var/lib/illumio-pce/runtime`
`ephemeral_data_root`	A writable location where the PCE writes temporary files	Read/Write	`/var/lib/illumio-pce/tmp`
`log_dir`	Directory where the PCE writes text file logs You must configure `logrotate` (or similar) to ensure log files do not grow too large.	Read/Write	`/var/log/illumio-pce`

The default location of the PCE Runtime Environment File is `/etc/illumio-pce/runtime_env.yml`, but for the exact location on your systems, check the value of the `log_dir` parameter.

4. Copy the PCE tarball to the `install_root` directory and untar it.
5. Create an `init` script to run `install_root/illumio-pce-ctlstart` at boot.

Upgrade PCE Tarball Installation

The `$ILLUMIO_RUNTIME_ENV` shell environment variable defines the location of the `runtime_env.yml` file.

The following variables used in this section refer to entries in the `runtime_env.yml` file for each node in the cluster:

- `install_root`
- `persistent_data_root`
- <log_dir>

On *all nodes* in the cluster, perform the following steps:

1. Move the old PCE version to a backup directory:

   ```
   $ mv install_rootinstall_root_previous_release
   ```

For example:

```
$ mv /opt/illumio-pce /opt/illumio-pce-previous-release
```

2. Install the new PCE TGZ version:

```
$ mkdir install_root
$ cd install_root
$ tar -xzf illumio_pce_tar_gz
```

Change Tarball to RPM Installation

Perform these steps to install a first-time RPM to replace the previous tarball installation.

1. On *all nodes*, as the *previous PCE runtime user*, stop the PCE:

```
# illumio-pce-ctl stop set-runlevel 1
```

2. Move all files under the `pce_installation_root` directory to a backup directory:

```
# mv pce_installation_root previousinstall-root
```

3. Change the previous PCE runtime user and group to `ilo-pce:ilo-pce`:

```
# usermod --login ilo-pce previous-user
# groupmod --new-name ilo-pce previous-group
```

4. Install the PCE via the RPM:

```
# rpm –ivh --nopre illumio-pce-16.6-0.x86_64
```

NOTE

The `--nopre` option prevents the RPM from creating these two empty directories: `/var/lib/illumio-pce` and `/var/log/illumio-pce`.

5. Move the existing `runtime_env.yml` file to `/etc/illumio-pce`.
6. Update the `ILLUMIO_RUNTIME_ENV` environment variable to `/etc/illumio-pce/runtime_env.yml` or delete this environment variable. The PCE looks for the runtime environment file in this location.
7. If necessary, change the `install_root` parameter in the `runtime_env.yml` file to `/opt/illumio-pce`.
8. On *all nodes*, as the *new PCE runtime user*, start the PCE:

```
# sudo -u ilo-pce illumio-pce-ctl start
```

9. On the *data0 node*, as the *new PCE runtime user*, migrate the database:

```
# sudo -u ilo-pce illumio-pce-db-management migrate
```

10. As the *new PCE runtime user*, bring the PCE to runlevel 5:

```
# sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
```

PCE Upgrade, Downgrade, and Uninstall

This section describes how to change to a newer or previous PCE software version (upgrade or downgrade) or remove PCE software. This section assumes that you have previously installed the PCE as described in PCE Installation [45].

PCE Upgrade Prerequisites

This section provides information on how to prepare to upgrade the PCE.

Upgrade Paths and Planning Tool

For information about upgrade paths for versions of the PCE and VEN, see Versions and Releases on the Illumio Support portal (login required).

For information to help you plan your upgrade, see the Illumio Core Upgrade Path page on the Illumio Support portal (login required).

Upgrade Prerequisites

Consider the following essential requirements before you begin upgrading or downgrading the PCE:

- Do not upgrade your VENs until the PCE version upgrade is successful. After Illumio VENs are upgraded, rolling back the PCE upgrade is not supported.
- Ensure that no asynchronous jobs are submitted before you begin the upgrade. As a best practice, wait until all asynchronous jobs have finished before upgrading the PCE.
- For a multi-version upgrade, the "Back Up PCE Database and Current Software" steps should only be done once at the beginning of the first upgrade sequence. This method allows you to roll back to the starting version if there is an issue with the upgrade.
- If you are upgrading from a PCE version earlier than 21.2, and you used the scripts **`ilo-pipgen`** and **`ilo-vpngen`** to install the earlier version, remove those scripts. They are not needed in version 21.2 and later. If installed in the default location, the scripts can be found at `/var/tmp/illumio-pipgen/ilo-pipgen` and `/var/tmp/illumio-vpngen/ilo-vpngen`.

Upgrade the PCE

This section describes how to upgrade the PCE and its UI together. To upgrade the UI alone, see UI-Only Upgrade [70] for information.

When upgrading the PCE and UI packages together, perform the following high-level tasks:

1. Verify that all Upgrade Prerequisites [65] are met.
2. Perform PCE installation planning and prerequisite steps if they have not already been done on this PCE, as described in Prepare for PCE Installation [21].
3. Back up the PCE [66].
4. Download the software [66].
5. Stop the PCE [67].
6. Install the new PCE and UI [67].
7. Update the runtime environment file [67].
8. Migrate the database [68].
9. Set runlevel to 5 [69].
10. Verify successful upgrade [69].

Back Up the PCE

When you are upgrading from a previous PCE version, the first step is to back up your existing data.

Back Up PCE Data

1. (On an SNC, skip this step.) Before you back up the PCE, determine which data node is running the `agent_traffic_redis_server` service:

   ```
   sudo -u ilo-pce illumio-pce-ctl cluster-status
   ```

   You see the following output:

   ```
   SERVICES (runlevel: 5) NODES (Reachable: 1 of 1)
   ====================== ===========================
   agent_background_worker_service 192.168.33.90
   agent_service NOT RUNNING
   agent_slony_service 192.168.33.90
   agent_traffic_redis_cache 192.168.33.90
   agent_traffic_redis_server 192.168.33.90          <=== run the dump
   command from this node
   agent_traffic_service NOT RUNNING
   ...
   ```
2. On the *data node* that is running the `agent_traffic_redis_server` service, run the following commands:

   ```
   sudo -u ilo-pce illumio-pce-db-management dump --file <location-of-db-
   dump-file>
   sudo -u ilo-pce illumio-pce-db-management traffic dump --file <location-
   of-traffic-dump-file>
   ```

   In
   location-of-db-dump-file
   and
   location-of-traffic-dump-file
   enter a file name for the policy database dump and the traffic database dump files, respectively.

   > **NOTE**
   > On an SNC, run these commands on the single node.

3. After the dump commands finish, copy the backup files to a fault-tolerant storage location.

Back up the PCE Runtime Environment File

Store a copy of each node's `runtime_env.yml` file on a system that is not part of the cluster or Supercluster. The default location of the PCE Runtime Environment File is `/etc/illumio-pce/ runtime_env.yml`.

Download the Software

1. Download the software from the Illumio Support portal (login required).

2. On the *core nodes only*, copy the Illumio PCE UI RPM file to the `/tmp` folder. In the following steps, this file is referred to as `illumio_ui_rpm`.
3. On *all nodes* in the cluster, copy the Illumio PCE software RPM file to the `/tmp` folder. In the following steps, this file is referred to as `illumio_pce_rpm`.

Stop the PCE

1. On *all nodes* in the cluster, stop the PCE:

```
sudo -u ilo-pce illumio-pce-ctl stop --wait
```

2. On *all nodes* in the cluster, verify the PCE status is STOPPED:

```
sudo -u ilo-pce illumio-pce-ctl status -sv --wait
```

Install the New PCE and UI

The packages to install depend on the type of PCE node:

- **Core nodes:** Two packages, the PCE RPM and UI RPM.
- **Data nodes:** One package, the PCE RPM.

1. On **each core node** in the cluster, log in as root and install the PCE RPM and the UI RPM. Be sure to specify both of the RPM file names on the command line:

```
$ rpm -Uvh illumio_pce_rpm illumio_ui_rpm
```

   For `illumio_pce_rpm` and `illumio_ui_rpm`, substitute the paths and filenames of the two RPM files you downloaded from the Illumio Support portal.
2. On **each data node** in the cluster, log in as root and install the PCE RPM:

```
$ rpm -Uvh illumio_pce_rpm
```

   For `illumio_pce_rpm`, substitute the path and filename of the software you downloaded from the Illumio Support portal.
3. If your PCE also includes an installed NEN, run the following commands:

```
$ rpm -e illumio_pce_nen --noscripts
$ rpm -Uvh illumio_pce_rpm --noscripts
$ rpm -Uvh illumio_nen_rpm
```

   For
   illumio_pce_rpm
   and
   illumio_nen_rpm
   , substitute the paths and filenames of the two RPM files you downloaded from the Illumio Support portal.
   Then use the following command to check the environment:

```
sudo -u ilo-pce illumio-pce-ctl check-env
```

Update the Runtime Environment File

See "What's New and Changed in This Release" to determine if any changes to the PCE Runtime Environment File (`runtime_env.yml`) are required to upgrade.

> **WARNING**
> The `cluster_type` runtime parameter *must* be set on all PCE nodes starting in PCE 21.5.0, except on a single node cluster (SNC). If you are upgrading from a version earlier than 21.5.0, be sure to set this parameter. For details about what value to use on each type of node, see Reference: PCE Runtime Parameters [74].

To make changes to the runtime environment configuration:

1. On *all nodes* in the cluster, update the `runtime_env.yml` file.
2. On *all nodes* in the cluster, check the validity of the `runtime_env.yml` file:

   ```
   sudo -u ilo-pce illumio-pce-ctl check-env
   ```

   If any issues are reported by this command, correct them before moving on to the next step.

Migrate the PCE Database

Ensure that you have upgraded all nodes in the cluster to the same version before you perform these steps. Otherwise, none of the nodes in your cluster will start.

1. On *all nodes* in the cluster, start the PCE:

   ```
   sudo -u ilo-pce illumio-pce-ctl start --runlevel 1
   ```
2. For some upgrades, you might be prompted to upgrade the database PostgreSQL software on one of the data nodes.

   At the prompt, enter **yes** to continue the upgrade.

   If you do not see this prompt, go to the next step.

   ```
   Upgrade needed. The agent database is currently running at postgres
   version 9.6 and needs to be upgraded to version 11.4
   Upgrade needed. The traffic database is currently running at postgres
   version 9.6 and needs to be upgraded to version 11.4
   The PCE software will now upgrade to a newer version of the postgres
   software from an older version.
   You need to migrate the PCE database after this process has finished.
   Prior to this upgrade, Illumio recommends you make a backup of the
   relevant data directories:
   /var/illumio_pce_data/persistent
   The upgrade must not be interrupted, otherwise the data might get
   corrupted and prevent the PCE from starting.
   Do you wish to continue with the database upgrade. [yes/no]: yes
   Proceeding with database upgrade.
   Please wait until the system has reached runlevel 1.
   ```
3. On *all nodes* in the cluster, verify the PCE status:

   ```
   sudo -u ilo-pce illumio-pce-ctl status -sv --wait
   ```
4. Verify that the runlevel is 1:

   ```
   sudo -u ilo-pce illumio-pce-ctl get-runlevel
   ```
5. On *any node* in the PCE cluster, migrate the database to the latest schema version:

```
sudo -u ilo-pce illumio-pce-db-management migrate
```

> **NOTE**
> This command can take some time to complete, depending on the amount of data. To check progress, view the Health page in the PCE web console. If the migration is still underway, you will see a message like "Traffic database migration in progress."

Set Runlevel 5

Bring the PCE to runlevel 5, full operation:

```
sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
```

> **IMPORTANT**
> If you did not run the `illumio-pce-db-management migrate` command on the primary database, you *cannot* bring the node up to runlevel 5 and you *cannot* start the other nodes in the cluster. If some of the nodes in the cluster are already running, they will shut down until you successfully migrate the database. If you attempt to start the upgraded PCE cluster without migrating the database, this error is displayed:
>
> ```
> sudo -u ilo-pce illumio-pce-ctl start
> Starting Illumio Runtime STARTING 20.96s
> $
> $ Stopping PCE software: DB migrations mismatch for DB:
> avenger_executor_dev: Missing migrations.
> ```

Verify Success

1. On *all nodes* in the cluster, verify the PCE status:

   ```
   sudo -u ilo-pce illumio-pce-ctl status -s -v -w
   ```
2. On *any node*, verify the cluster status:

   ```
   sudo -u ilo-pce illumio-pce-ctl cluster-status -w
   ```
3. When you have a front end load balancer (F5 or DNS), make sure that the load balancer is sending requests to the two core nodes in the cluster.
4. Log into the PCE web console, pair a VEN, and verify VEN sync status is showing as “Verified” for a few randomly selected workloads. Select a Workload's details page and verify that the Policy Sync section shows “Verified.”
5. Test that the firewall is correctly configured by connecting to restricted services using `telnet`. When the PCE is running, connections on TCP port 8300 should be accepted from other PCE hosts but rejected otherwise. Connections on TCP port 8443 should be accepted from all sources (unless modified using the `-p` option).

Upgrading to RHEL 9

Starting in PCE version 23.2.20, support is added for installing the PCE on RHEL 9. When upgrading from a previous operating system and previous PCE release, follow these recommended additional steps.

1. Upgrade the PCE to 23.2.20, using the steps outlined earlier in this topic.
2. Take a backup of the 23.2.20 PCE. Store the backup in a safe storage.
3. Upgrade the operating system to RHEL 9.
4. Install the PCE version 23.2.20 on RHEL 9.
5. Restore the backup that was made in step 2.

For details on steps 2 through 5, see "Upgrade the OS on a Running PCE" in the PCE Administration Guide.

PCE UI-Only Upgrade

You can upgrade the PCE web console UI alone, as long as the UI version is compatible with the installed PCE version. This upgrade method is useful for taking advantage of bug fixes and other updates that affect the PCE web console only.

> **NOTE**
> You can't install the UI by itself. For the UI to work, a compatible version of the PCE must already be installed.

You do not need to change the PCE runlevel or stop the PCE to upgrade the UI.

To upgrade the UI only:

1. Download the UI software from the Illumio Support portal (login required).
2. On the *core nodes only*, copy the Illumio PCE UI RPM file to the `/tmp` folder. In the next step, this file is referred to as `illumio_ui_rpm`.
3. On *all core nodes* in the cluster, upgrade to the new PCE UI:

   ```
   rpm -Uvh illumio_ui_rpm
   ```

Downgrade PCE to Previous Version

This section describes how to roll back the PCE to a previous version in the event of a PCE upgrade failure or defect. You can downgrade to any currently supported PCE version; see Versions, Compatibility & Support Status on the Illumio Support site.

Downgrade the PCE

To downgrade to a previous PCE version, you will need the following files:

- PCE software installation files for the older version. See Download the Software [46].
- Database backups taken on the previous version. See Database Backup in the PCE Administration Guide.
- Backup of the `runtime_env.yml` file from the previous version. See Back Up the PCE Runtime Environment File in the PCE Administration Guide.

To downgrade the PCE, perform the following steps:

1. On *all nodes* in the cluster, stop the PCE:

   ```
   sudo -u ilo-pce illumio-pce-ctl stop
   ```
2. On *all nodes* in the cluster, downgrade the installation by installing the older version of the PCE and UI.

   RPM Installation:

   ```
   sudo rpm -Uh --force illumio-pce.rpm illumio-pce-ui.rpm
   ```

   For example:

   ```
   sudo rpm -Uh --force illumio-pce-21.2.8-2.c6.x86_64.rpm illumio-pce-ui-21.2.8.UI1-1.x86_64.rpm
   ```

   > **WARNING**
   > Be sure to install the corresponding UI RPM after installing the older PCE version.

   Next, install the PCE package on each data node:

   ```
   sudo rpm -Uh --force illumio-pce.rpm
   ```

   Tarball Installation:

   ```
   mv install_root_previous_release install_root
   ```

   For example:

   ```
   mv /opt/illumio-pce-previous-release /opt/illumio-pce
   ```
3. If you changed the `runtime_env.yml` file, restore the previous version of the file:

   ```
   cp /etc/illumio-pce/runtime_env.yml-backup /etc/illumio-pce/runtime_env.yml
   ```
4. On *all nodes* in the cluster, reset the node:

   ```
   sudo -u ilo-pce illumio-pce-ctl reset
   ```
5. On *all nodes* in the cluster, start the PCE at runlevel 1:

   ```
   sudo -u ilo-pce illumio-pce-ctl start --runlevel 1
   ```
6. On *all nodes* in the cluster, verify the PCE status and runlevel:

   ```
   sudo -u ilo-pce illumio-pce-ctl status -s -v -w
   sudo -u ilo-pce illumio-pce-ctl cluster-status -w
   ```
7. Set up the database. First, determine the primary database (on an SNC, you can skip this step, as there is only one possible node):

   ```
   sudo -u ilo-pce illumio-pce-db-management show-primary
   ```

8. On the *primary data node*, run this command to set up the database:

   ```
   sudo -u ilo-pce illumio-pce-db-management setup
   ```
9. Restore the PCE policy database. On *one of the data nodes* of the cluster (or in an SNC, on the single node), restore a known good backup:

   ```
   sudo -u ilo-pce illumio-pce-db-management restore --file
   location_of_db_dump_file
   ```

   For example, if you are downgrading because of an unsuccessful upgrade attempt, restore the backup you took before doing the upgrade.
10. (On an SNC, you can skip this step.) Copy the restored Illumination data file to the *other data node*. The file is located in the following directory:

    ```
    persistent_data_root/redis/redis_traffic_0_master.rdb
    ```
11. Migrate the PCE database. On *one of the data nodes* in the cluster, migrate the database to the latest schema version:

    ```
    sudo -u ilo-pce illumio-pce-db-management migrate
    ```
12. Restore the traffic database. Run this command on the same node where you took the traffic database backup:

    ```
    sudo -u ilo-pce illumio-pce-db-management traffic restore --file /
    path/to/traffic_db_dump_file
    ```

    This command prompts you to return the PCE to runlevel 5. You can choose to go to runlevel 5 or not.
13. If you did not accept the change to runlevel 5 after restoring the traffic database, you can bring the PCE to runlevel 5, full operation, at any time using the following command:

    ```
    sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
    ```

    IMPORTANT

    If you did not run the `illumio-pce-db-management migrate` command, you *cannot* bring the node up to runlevel 5 and you *cannot* start the other nodes in the cluster. If some of the nodes in the cluster are already running, they will shut down until you successfully migrate the database. If you attempt to start the upgraded PCE cluster without migrating the database, this error is displayed:

    ```
    sudo -u ilo-pce illumio-pce-ctl start
    Starting Illumio Runtime STARTING 20.96s
    $
    $ Stopping PCE software: DB migrations mismatch for DB:
    avenger_executor_dev: Missing migrations.
    ```

> **IMPORTANT**
>
> As part of the downgrade process, you must take the PCE out of Listen Only mode as the last step.
>
> 1. From *one of the data nodes*, disable Listen Only mode:
>
>    ```
>    sudo -u ilo-pce illumio-pce-ctl listen-only-mode disable
>    ```
> 2. Verify that PCE Listen Only mode is disabled:
>
>    ```
>    sudo -u ilo-pce illumio-pce-ctl listen-only-mode status
>    ```

Verify Success of Downgrade

1. On *all nodes* in the cluster, verify the PCE status:

   ```
   sudo -u ilo-pce illumio-pce-ctl status -s -v -w
   ```
2. On *any node*, verify the cluster status:

   ```
   sudo -u ilo-pce illumio-pce-ctl cluster-status -w
   ```
3. When you have a front end load balancer (F5 or DNS), make sure that the load balancer is sending requests to the two core nodes in the cluster.
4. Log into the PCE web console, pair a VEN, and verify VEN sync status is showing as "Verified" for a few randomly selected workloads. Select a Workload's details page and verify that the Policy Sync section shows "Verified."
5. Test that the firewall is correctly configured by connecting to restricted services using `telnet`. When the PCE is running, connections on TCP port 8300 should be accepted from other PCE hosts but rejected otherwise. Connections on TCP port 8443 should be accepted from all sources (unless modified using the `-p` option).

Uninstall the PCE

In order to completely uninstall and remove the PCE for your system, perform the following steps:

1. Run this command to remove the PCE:

   ```
   $ rpm -e illumio-pce
   ```
2. Manually delete these directories:

   ```
   /var/lib/illumio-pce
   /var/log/illumio-pce
   /etc/illumio-pce
   ```

PCE Installation Reference

This section contains reference material related to PCE installation. You might need to refer to this information while you are performing the installation steps in Install the PCE and UI [46], or refer to it for ongoing needs after the PCE is installed.

Reference: PCE Runtime Parameters

This section lists important PCE runtime configuration parameters, their meaning, their purpose, and their exposure levels.

When configuring the PCE with the `illumio-pce-env setup` script, you are prompted for many of these parameters. See Configure the PCE [47] for information.

IMPORTANT

- The `runtime_env.yml` file contains sensitive information that should be kept secret, such as encryption keys. Take steps to ensure the confidentiality of this file.
- The `runtime_env.yml` file is not included in automatic PCE backups. You must manually back up this file to a secure location.

Runtime File Exposure Levels

The Illumio PCE `runtime_env.yml` file provides the following exposure levels for PCE configuration:

- **Public Stable** (`public_stable`): These `runtime_env.yml` parameters can be used by all customers. All changes are backward compatible.
- **Public Experimental** (`public_experimental`): These `runtime_env.yml` parameters can be used by all customers but might change from release to release with no guarantee of backwards compatibility.

Required Runtime Parameters

The following table lists the required `runtime_env.yml` file parameters for each PCE software node you deploy. All required parameters have no default values. All paths configured in this file must be absolute.

Required Parameter	Description	Exposure
enabled_preview_features	Includes sub-parameters to enable identified preview features	
install_root	The full path to the location of the PCE binaries and scripts The software does not write to any files in this directory, so it can be read-only. For example: `install_root: /opt/illumio-pce`	Public Stable
runtime_data_root	The full path to the location where the PCE writes runtime data This data can be deleted on reboot if necessary. This directory should have 700 permissions, but all of its files will have 600 permissions. This directory must be owned by the user that runs the PCE software. For example: `runtime_data_root: /var/lib/illumio-pce/runtime`	Public Stable
persistent_data_root	The full path to the location where the PCE writes persistent data This data must persist across reboots for the software to work properly. This directory should have 700 permissions, but all of its files will have 600 permissions. This directory must be owned by the user that runs the PCE software. For example: `persistent_data_root: /var/lib/illumio-pce/data`	Public Stable
ephemeral_data_root	The full path to the location where the PCE writes temporary files These files must not be deleted while the software is running, but they should be deleted on reboot. This directory should have 700 permissions, but all of its files will have 600 permissions. For example: `ephemeral_data_root: /var/lib/illumio-pce/tmp`	Public Stable
log_dir	The directory where the PCE software writes some text file logs (although most PCE services log to syslog) For example: `log_dir: /var/log/illumio-pce`	Public Stable
pce_fqdn	The fully qualified domain name (FQDN) of the PCE cluster. For example: `pce_fqdn: pce.mycompany.com`	Public Stable

Required Parameter	Description	Exposure
`cluster_public_ips: cluster_fqdn`	The FQDN of your entire cluster **NOTE** If you change the value of `cluster_public_ips`, wait for the paired VENs to receive the new IP addresses and begin heartbeating to them.	Public Stable
`web_service_certificate`	Full path to the X.509 public certificate used by this node for TLS See TLS Requirements [35] for more information on the contents of the certificate files. For example: `web_service_certificate: /etc/pki/tls/certs/my_cert.crt`	Public Stable

`web_service_private_key`	The RSA private key for TLS that matches the public certificate
	
	The private key must be PEM encoded in PKCS#12 format without a password.
	
	For example:
	
	`web_service_private_key: /var/lib/illumio-pce/cert/rsa_private_key.key`
	
	Alternatively, you can specify a script (using `$` notation) that outputs the private key. This approach is useful when you need to store the key in a hardware security module (HSM) or other key store.
	
	For example:
	
	`web_service_private_key: $ /var/lib/illumio-pce/cert/get_rsa_private_key.sh`
	
	This script can be located anywhere on the file system as long as it is executable by the `ilo-pce user`.
	
	Example script output:
	
	`$ /local/scripts/get_rsa_private_key.sh`
	`-----BEGIN RSA PRIVATE KEY-----`
	`MIIE...`
	`many lines trimmed here`
	`-----END RSA PRIVATE KEY-----`
		Public Stable

`email_address`	Email sender address used by the PCE when sending emails from the system; for example, to send invitations and notifications For example: `email_address: noreply@exampleblocked_traffic.com`	Public Stable
`service_discovery_fqdn`	The FQDN or IP address of the first core node	Public Experimental

Required Parameter	Description	Exposure
`service_discovery_encryption_key`	The key used to encrypt Service Discovery node traffic. This value must be the same for all PCE nodes. This key must be 16 bytes that are base64 encoded. For example: `service_discovery_encryption_key: 05TlqH1W0cKcK797DV73yg==`	Public Stable
`node_type`	The type of the PCE software node Allowable values: • `core`: core node • `data0`: data node • `data1`: data node • `snc0`: single-node cluster • `citus_coordinator`: coordinator node for multi-node traffic database • `citus_worker`: worker node for multi-node traffic database For example: `node_type: core`	Public Stable
`login_banner`	A custom message on the PCE login screen typically used to display legal notice or company policy when a user logs in	Public Stable
`cluster_type`	PCE cluster type. Required on every node in a multi-node cluster (MNC). Not required on a single-node cluster (SNC). One of the following: • `4node_v0`: 2x2 PCE cluster • `4node_v0_small`: 2x2 PCE cluster with fewer compute and memory resources • `6node_v0`: 4x2 PCE cluster • `4node_dx`: 2x2 PCE cluster with multi-node traffic database • `6node_dx`: 4x2 PCE cluster with multi-node traffic database **Default:**`4node_v0`	Public Stable

Optional Runtime Parameters

The following table lists common optional `runtime_env.yml` file parameters for each PCE software node you deploy. Your Illumio Professional Services representative might provide additional parameters to configure certain advanced functions.

Optional Parameter	Description	Exposure
`agent_pce_fqdn`	A fully-qualified domain name (FQDN) for VENs to use to send communications to the PCE. This is a second FQDN in addition to the required parameter `pce_fqdn`. When `agent_pce_fqdn` is specified in addition to `pce_fqdn`, the TLS certificates for the PCE must include both domains. The front-end port remains the same; see `front_end_https_port`.	Public Experimental

`ven_repo_url`	The base URL used to fetch the VENs and to enable workload pairing with the PCE
	
	Required format: `https://host[:port]/repo_dir`
	
	You can use alternate ports by specifying the port at the end of hostname. `repo_dir` cannot be empty.
	
	For example:
	
	`https://repo.example.com:8443/`
	`onpremgCBURz8Y4zkGk1u7N9ialjPGlZ`
	
	Default: None
		Public Stable

`ven_repo_ips`	IP addresses of the VEN repository
	
	These IP addresses are injected into iptables to allow outbound access to the `yum/apt get` repositories without having to write an explicit PCE policy.
	
	Setting this parameter allows outbound access on ports 80 and 443 to these IP addresses. You can specify both single IP addresses or IP addresses with CIDR notation.
	
	When you do not specify this parameter, the VEN won't be allowed to access the repository containing VEN software packages.
	
	For example:
	
	`ven_repo_ips:`
	`- 1.2.3.4`
	`- 5.6.7.8/8`
	
	Default: None
		Public Stable

`internal_service_ip`	The IP address of the PCE Set this value manually only when you want to use a public IP address or the PCE node has multiple interfaces. For example: `internal_service_ip: 10.2.8.89` **Default:** The first available private IP address on the node	Public Stable
`front_end_https_port`	The front end HTTPS port	Public Stable

Optional Parameter	Description	Exposure
	When the cluster is front-ended by a server load balancer, such as F5, it must be configured to forward this port. For example: `front_end_https_port: 8443` **Default:** TCP 8443 if not set by `front_end_management_https_port` or `front_end_https_port`	
`front_end_event_service_port`	The front end Event Service port When the cluster is front-ended by a server load balancer, such as F5, it must be configured to forward this port. The idle connection timeout on the server load balancer might need to be configured to maintain the connections on this port. Please contact your Illumio Professional Services representative for information on configuring your server load balancer. For example: `front_end_event_service_port: 8444` **Default:**`8444`	Public Stable
`front_end_management_https_port`	The port for PCE web console and REST API. This key separates different kinds of communication. See also `front_end_https_port`. **Default:** TCP 8443 if not set by `front_end_management_https_port` or `front_end_https_port`	Public Stable
`syslog_event_export_format`	The export format (CEF, LEEF, or JSON) for VEN flow summaries and Organization events. When you specify CEF or LEEF format, you will continue getting traffic flows and Organization events in JSON format. For example: `syslog_event_export_format: cef` **Default:**`json`	Public Stable
`min_tls_version`	The minimum Transport Layer Security (TLS) version used to secure VEN-to-PCE communications, the PCE's web server for the PCE web console, and the REST API. It is recommended that you use the default setting, 1.2. Earlier TLS versions, such as 1.0 and 1.1, are considered less secure, so it is recommended you do not use them. In rare circumstances, such as when using older operating systems, you might need to change the minimum TLS version; see TLS Versions for Communications [38]. Allowable values: `tls1_0`, `tls 1_1`, `tls1_2`. For example: `min_tls_version: tls1_2`	Public Stable

Optional Parameter	Description	Exposure
	Default:`tls1_2`	
`insecure_tls_weak_ciphers_enabled`	Specifies whether to allow the use of weaker TLS ciphers, such as cipher block chaining (CBC) ciphers. Stronger ciphers are recommended. Illumio recommends you keep the default value (`true`) for this setting when using clients or operating systems that can only negotiate TLS using CBC ciphers. If your environment is not impacted by this limitation, Illumio recommends that you change the value to `false` so that you use strong ciphers. For example: `insecure_tls_weak_ciphers_enabled: true` **Default:**`true`	Public Stable
`trusted_ca_bundle`	The path to the trusted root certificate bundle. The PCE uses this parameter to validate that the certificates are trusted and indicates the path to the trusted root certificate bundle file. For example: `trusted_ca_bundle: /etc/ssl/certs/ca-bundle.crt` **Default:**`/etc/ssl/certs/ca-bundle.crt`	Public Stable
`email_display_name`	Email display name to be used when sending email from the system. For example, to send invitations and notifications from the PCE. For example: `email_display_name:'noreply'` **Default:**`noreply`	Public Stable
`smtp_relay_address`	SMTP relay information used by the PCE to send email; for example, to send invitations and notifications. The PCE assumes that an SMTP Relay runs on localhost and listens on `127.0.0.1/587`. When this isn't the case, you must specify the configuration on the *core nodes*. Use *one* of the following formats: • `ip_address (e.g. 127.0.0.1)` • `ip_address:port (e.g. 127.0.0.1:587)` For example: `smtp_relay_address: 127.0.0.1:587` **Default:**`127.0.0.1:587`	Public Stable
`export_flow_summaries_to_fluentd`	The types of traffic flow summaries to export to Fluentd. Values: `accepted` (allowed), `potentially_blocked`, `blocked`	Public Experimental

<table>
<tr><th>Optional Parameter</th><th>Description</th><th>Exposure</th></tr>
<tr><td></td><td>For example:
<pre>export_flow_summaries_to_fluentd:
  - accepted
  - potentially_blocked
  - blocked</pre></td><td></td></tr>
<tr><td><code>export_flow_summaries_to_syslog</code></td><td>Enables traffic flow summaries to syslog.

Values: <code>accepted</code> (allowed), <code>potentially_blocked</code>, <code>blocked</code>

For example:
<pre>export_flow_summaries_to_syslog:
  - accepted
  - potentially_blocked
  - blocked</pre>To export blocked traffic summaries, include only the flow summary type when specifying the parameter; for example:
<pre>export_flow_summaries_to_syslog:
  - blocked</pre></td><td>Public Experimental</td></tr>
<tr><td><code>internal_syslog_fqdn_enabled</code></td><td>Specifies whether to use the PCE's fully-qualified domain name (FQDN) or the hostname in syslog messages. The FQDN can be more helpful if the short hostnames are difficult to distinguish.

Values: <code>true</code> (the <code>host=</code> field uses the FQDN), <code>false</code> (default)

For example:
<pre>internal_syslog_fqdn_enabled: true</pre></td><td>Public Experimental</td></tr>
</table>

FIPS Compliance for PCE and VEN

> **NOTE**
> This release supports FIPS compliance for the PCE and Linux and Windows VENs. It does not support FIPS compliance for the AIX and Solaris VENs.

This section describes the operational requirements for compliance with Federal Information Processing Standard (FIPS) 140-2 for the PCE and VEN.

FIPS Prerequisites

- PCE server hardware requires the Intel Ivy Bridge CPU (2012) or later.
- RedHat v7.4 or later required.
- Customer-provided SSL certificates from a public CA or a customer CA. The certificates must have a minimum key size of 2048 to secure PCE communications.

FIPS-related Government and Vendor Documentation

- Federal Information Processing Standard (FIPS) 140-2, Security Requirements for Cryptographic Modules
- Red Hat Enterprise Linux OpenSSL Cryptographic Module NIST Security Policy
- RHEL v7.1 Red Hat Enterprise Linux Kernel Crypto API Cryptographic Module v4.0
- RHEL v7.4 Red Hat Enterprise Linux Kernel Crypto API Cryptographic Module v5.0
- RHEL v8.x Red Hat Enterprise Linux 8 OpenSSL Cryptographic Module v8.0
- Windows Server 2012 NIST Security Policy
- Windows Server 2016 NIST Security Policy

Non-Government Customers without FIPS Requirement

Compliance to FIPS 140-2 requires additional operational restrictions, such as specific OS versions and server hardware.

Illumio recommends that non-government customers who do not have requirement for FIPS 140-2 do not configure and deploy Illumio Core to support FIPS compliance.

Compliance Affirmation Letters

Third-party FIPS-compliance affirmation letters for the Illumio Core are available at FIPS 140-2 Affirmation Letters (PDF download).

Prerequisites for Linux VEN FIPS Compliance

For SecureConnect (IPsec encryption among workloads), to claim FIPS compliance, the VEN must be installed on RHEL v7.1, RHEL v7.4, or RHEL v8.0 and configured to operate in FIPS mode as described in the following vendor documents:

- RedHat v7.1, Section 9.1 ("Cryptographic Officer Guidance") of the RHEL v7.1 Red Hat Enterprise Linux Kernel Crypto API Cryptographic Module v4.0.
- RedHat v7.4, Section 9.1 ("Cryptographic Officer Guidance") of the RHEL v7.4 Red Hat Enterprise Linux Kernel Crypto API Cryptographic Module v5.0.
- RHEL v8.0, Section 9.1 ("Crypto Officer Guidance") of the RHEL v8.0 Red Hat Enterprise Linux 8 OpenSSL Cryptographic Module v8.0

The Linux VEN versions do not have other special OS requirements or additional configurations to enable FIPS-compliant OpenSSL communications. The Linux VEN's FIPS OpenSSL module is built directly into the VEN and is not supplied by the underlying OS; the Linux VEN operates by default in FIPS mode.

Prerequisites for Windows VEN FIPS Compliance

For FIPS compliance on Windows, either Windows Server 2012 or Windows Server 2016 must be configured according to the following vendor documents:

- Windows 2012 conforming with Section 2 of the Windows Server 2012 NIST Security Policy
- Windows 2016 conforming Section 2 of the Windows Server 2016 NIST Security Policy

Enable PCE FIPS Compliance

1. After installing RHEL7.4, follow the required steps in Section 9.1, Crypto Officer Guidance, Red Hat Enterprise Linux OpenSSL Cryptographic Module NIST Security Policy.

2. Reboot the system.
3. After reboot, verify that the setting `/proc/sys/crypto/fips_enabled` is equal to 1.
4. Install the Illumio PCE RPM. See After PCE Installation [59].
5. During PCE installation, provide the PCE with SSL certificates that have a minimum RSA key size of 2048.

After completing the PCE setup, the PCE is FIPS compliant.

FIPS Compliance for Red Hat/Linux VENs

For all Illumio supported Linux workloads, the standard 18.1 GA VEN release and later support VEN Linux FIPS compliance.

Starting with the Linux VEN 18.1 release, all VEN OpenSSL communications by default operate in a FIPS compliant mode.

- FIPS is supported on the VEN 18.1 release through the 20.2 release.
- FIPS is *not* supported on the VEN 21.1 release through the 21.5 release due to the OpenSSL 1.1 upgrade.
- FIPS is supported on the VEN 22.2 release and later.

FIPS for SecureConnect

To claim FIPS compliance for the VEN SecureConnect feature (IPsec encryption between workloads), the VEN must be installed on RHEL v7.1 or RHEL v7.4 and configured to operate in FIPS mode as documented in either of the following documents:

- Section 9.1 ("Cryptographic Officer Guidance") of the RHEL v7.1 Red Hat Enterprise Linux Kernel Crypto API Cryptographic Module v4.0
- Section 9.1 ("Cryptographic Officer Guidance") of the RHEL v7.4 Red Hat Enterprise Linux Kernel Crypto API Cryptographic Module v5.0

FIPS Compliance for Windows VENs

For Windows workloads, the standard 18.1 GA VEN release and later support VEN Windows FIPS compliance.

Windows VEN is FIPS compliant when installed on Windows Server 2012 or Windows Server 2016.

To operate the FIPS-compliant Windows VEN, the Windows system must be configured to operate in FIPS mode as documented in Section 2 of the Windows Server 2012 NIST Security Policy or Section 2 of the Windows Server 2016 NIST Security Policy.

OpenSSL 3.0 Module and RHEL 8 FIPS 140-2 Certification

OpenSSL 3.0 module and RHEL 8.0 OS are both currently undergoing certification for FIPS 140-2.

For more on the latest certification status for RHEL 7.x and RHEL 8.x, see the following NIST Cryptographic Module Validation Program (CMVP) document: Cryptographic Module Validation Program CMVP Modules In Process List

PCE Installation Troubleshooting Scenarios

This section describes issues that can arise during PCE installation or upgrade and how to resolve them.

Session Limits Too Low

(RHEL7+ only)

Symptom:

The expected session limits, configured in `/etc/security/limits.conf`, might not be in effect for the PCE. This can cause a severe performance impact that may go unnoticed for some time.

Cause:

It typically affects systems which have a PAM authentication configuration utilizing the `loginuid` module. This type of configuration relies on `systemd` rather than the traditional security limits for the application's session limits. The issue can also arise if the documented installation preparation steps are not followed (see "Process and File Limits") or if the values are later altered.

You can verify the session limits by inspecting a running PCE with the following command:

```
cat /proc/$(pgrep -f config_listener.rb)/limits | grep -e open -e processes
```

Solution:

If the output of this command shows values for `nofile` and `nproc` that are lower than required, provide an override file to properly configure these limits.

1. Create the following file on the PCE:
   `/etc/systemd/system/illumio-pce.service.d/override.conf`
2. Add the following lines to the file and save:
   ```
   [Service]
   LimitNOFILE=65535
   LimitNPROC=65535
   ```
3. Reboot or restart the PCE.
4. Run the `grep` command above again and inspect the `config_listener.rb` session limits to ensure that they are now correct.

Database Migrations Mismatch

Symptom:

Error message “Stopping PCE software: DB migrations mismatch for DB: avenger_executor_dev: Missing migrations” when you try to bring the PCE to runlevel 5.

Cause:

Attempted to start the upgraded PCE without migrating the database. If you did not run the `illumio-pce-db-management migrate` command on the primary database node, you will not be able to bring any PCE node up to runlevel 5, and you will not be able to start the other nodes in the cluster. If some of the nodes in the cluster are already running, they will be shut down until you successfully migrate the database.

Solution:

Follow the steps in Database Migrations Mismatch [84].

Database Already Exists

Symptom:

The `illumio-pce-db-management setup` command finishes abnormally with Exit Code = 1 and displays the following type of messages:

```
Database 'traffic_prod' already exists
...
psql:/opt/illumio-pce/illumio/webservices/traffic_query/db/
structure.sql:52: ERROR:  relation "ar_internal_metadata" already exists
rake aborted!
failed to execute:
psql -v ON_ERROR_STOP=1 -q -f /opt/illumio-
pce/illumio/webservices/traffic_query/db/structure.sql traffic_prod
```

Cause:

The database has already been set up.

Solution:

If you are trying to set up a new database, you must first remove the existing database.

> **WARNING**
> All existing data will be lost.

To remove the database and all its data, run the command `illumio-pce-db-management drop`. Then, retry the `illumio-pce-db-management setup` command.

PCE UI Missing

Symptom:

Error message displayed in the browser, such as “PCE UI Missing.”

Cause:

The PCE package has been installed without the UI package.

Solution:

Install the UI package as described in UI-Only Upgrade [70].

PCE Troubleshooting Scenarios

This section describes issues that can arise during PCE installation or upgrade and how to resolve them.

Session Limits Too Low

(RHEL7+ only)

Symptom:

The expected session limits, configured in `/etc/security/limits.conf`, might not be in effect for the PCE. This can cause a severe performance impact that may go unnoticed for some time.

Cause:

It typically affects systems which have a PAM authentication configuration utilizing the `loginuid` module. This type of configuration relies on `systemd` rather than the traditional security limits for the application's session limits. The issue can also arise if the documented installation preparation steps are not followed (see "Process and File Limits") or if the values are later altered.

You can verify the session limits by inspecting a running PCE with the following command:

```
cat /proc/$(pgrep -f config_listener.rb)/limits | grep -e open -e processes
```

Solution:

If the output of this command shows values for `nofile` and `nproc` that are lower than required (see "Process and File Limits"), provide an override file to properly configure these limits.

1. Create the following file on the PCE:
   /etc/systemd/system/illumio-pce.service.d/override.conf
2. Add the following lines to the file and save:
   ```
   [Service]
   LimitNOFILE=65535
   LimitNPROC=65535
   ```
3. Reboot or restart the PCE.
4. Run the `grep` command above again and inspect the `config_listener.rb` session limits to ensure that they are now correct.

Database Migrations Mismatch

Symptom:

Error message “Stopping PCE software: DB migrations mismatch for DB: avenger_executor_dev: Missing migrations” when you try to bring the PCE to runlevel 5.

Cause:

Attempted to start the upgraded PCE without migrating the database. If you did not run the `illumio-pce-db-management migrate` command on the primary database node, you will not be able to bring any PCE node up to runlevel 5, and you will not be able to start the other nodes in the cluster. If some of the nodes in the cluster are already running, they will be shut down until you successfully migrate the database.

Solution:

Follow the steps in Migrate the PCE Database [68].

Database Already Exists

Symptom:

The `illumio-pce-db-management setup` command finishes abnormally with Exit Code = 1 and displays the following type of messages:

```
Database 'traffic_prod' already exists
...
psql:/opt/illumio-pce/illumio/webservices/traffic_query/db/
structure.sql:52: ERROR:  relation "ar_internal_metadata" already exists
rake aborted!
failed to execute:
psql -v ON_ERROR_STOP=1 -q -f /opt/illumio-
pce/illumio/webservices/traffic_query/db/structure.sql traffic_prod
```

Cause:

The database has already been set up.

Solution:

If you are trying to set up a new database, you must first remove the existing database.

> **WARNING**
> All existing data will be lost.

To remove the database and all its data, run the command `illumio-pce-db-management drop`. Then, retry the `illumio-pce-db-management setup` command.

PCE UI Missing

Symptom:

Error message displayed in the browser, such as “PCE UI Missing.”

Cause:

The PCE package has been installed without the UI package.

Solution:

Install the UI package as described in UI-Only Upgrade [70].

PCE Supercluster

Overview of Supercluster Deployment

This section introduces concepts that you need to understand in order to achieve a successful PCE Supercluster deployment.

How to Use This Guide

- Overview of PCE Supercluster possible architectures and components
- General tasks required to deploy, operate, and use a PCE Supercluster: health monitoring, PCE web console and API access, backup and restore, and workload pairing considerations
- Basic theory of PCE Supercluster operations

Use this guide in conjunction with PCE Installation and Upgrade Guide and PCE Administration Guide.

Before You Begin

Become familiar with the following technology:

- Your organization's security goals
- Illumio Core
- General computer system administration of Linux and Windows operating systems, including startup/shutdown, standard processes or services
- Linux shell (bash) and Windows PowerShell
- TCP/IP networks, including protocols and well-known ports
- PKI certificates

Notational Conventions

- Newly introduced terminology is italicized. Example: *activation code* (also known as pairing key)
- Command-line examples are monospace. Example: `illumio-ven-ctl --activate`
- Arguments on command lines are monospace italics. Example: `illumio-ven-ctl --activate activation_code`
- In some examples, the output might be shown across several lines but is actually on one single line.
- Command input or output lines not essential to an example are sometimes omitted, as indicated by three periods in a row. Example:

```
...
some command or command output
...
```

About This Supercluster Guide

The following sections provide helpful information to help you get the most out of this guide.

PCE Supercluster Concepts

A Policy Compute Engine (PCE) Supercluster consists of a single administrative domain that spans two or more replicating PCEs. One PCE in the Supercluster is the Supercluster leader and the other PCEs are Supercluster members.

- A Supercluster deployment has only one leader. Any member can be manually promoted to be the leader.
- The leader has a central PCE web console and REST API endpoint for configuring and provisioning security policy. The web interface on the leader also provides other centralized management functions including an aggregated Illumination map to visualize network traffic and policy coverage for all workloads.
- Members in the Supercluster mostly have a read-only PCE web console and REST API for viewing local data.

To illustrate how a PCE Supercluster works, consider this example three-tier application (web, processing, database) that is deployed across three datacenters in the US, Europe, and Asia. Each datacenter has its own PCE and the US PCE is the leader. The policy for this application is designed to micro-segment the application in each datacenter while allowing the database tier to replicate across datacenters.

Workload Management

All PCEs in the Supercluster can manage workloads. You can deploy a leader without managed workloads to reduce the load on the leader and maintain performance for policy computation and other tasks.

- Pairing profiles must always be created on the leader, from which they are replicated to all members.
- On the members, you can generate pairing keys and pairing scripts tied to the members themselves for activation and not the leader.

Pairing Workloads

Before workloads can be paired, a pairing profile must be created on the leader, which is then replicated to all other PCEs in the Supercluster. Workloads can be paired to a specific PCE FQDN or to the Supercluster FQDN. In the latter case, you must use a Global Server Load Balancing (GSLB) or DNS server that supports persistent routing of workloads to the nearest PCE based on geolocation.

- When a workload is paired with a PCE, a managed workload object is created on the PCE and its labels are assigned based on the settings in the pairing profile. The PCE calculates policy and distributes firewall rules to the newly paired workload and other managed workloads. This allows the workloads to communicate with the newly paired workload.
- The PCE also replicates information about the new workload to the other PCEs. This in turn re-computes and re-distributes firewall rules to their managed workloads that are allowed to communicate with the newly paired workload.

In this example, when you pair a new instance of the database in the US, the following events occurs:

1. The US PCE sends firewall rules to the US database workload.
2. The US PCE sends send new firewall rules to the US web and processing workloads because the policy allows these workloads to communicate.
3. The US PCE replicates information about the new US database workload to the PCEs in Europe and Asia.
4. The PCEs in Europe and Asia re-calculate policy and send new firewall rules to their database workloads because the policy allows these databases to communicate with the US database.

There might be a short time period when one of the database workloads has received rules allowing outbound traffic but the other database workloads have not yet received their corresponding inbound rules to allow the connection. This condition can occur with a single PCE (for example, a non-Supercluster deployment) but can take slightly longer with a PCE Supercluster due to replication delays between PCEs.

Pairing with Specific Members

A pairing profile must always be created on the Supercluster leader. This pairing profile is propagated to all members. On a member, you can generate new pairing keys from the propagated profile. The pairing script generated from a pairing profile pairs the workload to the specific member.

Making Policy Modifications

Changes to your policy are made and provisioned on the leader using the PCE web console or the Illumio Core REST API, which in turn is replicated to all other PCEs in the Supercluster. Whenever a PCE receives updated policy, it re-computes policy for its own managed workloads and sends firewall rules to any other affected managed workloads.

Example: The original policy was written to allow the database workloads to communicate across datacenters using all ports. The organization has decided to tighten this policy and restrict it to just the port needed for database replication.

When the new policy is provisioned on the leader, the following actions occurs:

1. The US PCE recalculates policy and sends new firewall rules to its database workload.
2. The US PCE replicates the policy to the PCEs in Europe and Asia.
3. On receiving the new policy, each of these PCEs re-computes policy and sends new firewall rules to their database workloads.

Adapting to Environmental Changes

Changes to a workload’s assigned labels, IP address changes, or when a workload goes offline, are handled similarly to pairing a new workload. The PCE managing the workload detects the changes and re-calculates and re-distributes new firewall rules for its managed workloads. It also replicates information about the change to the other PCEs, and these PCEs re-calculate policy and send new firewall rules to any of their managed workloads that are affected by the change.

Security Policy Replication

Security policy provisioned on the leader is replicated to all other PCEs in the Supercluster. All Supercluster leader and members replicate copies of each workload's context, such as IP addresses, to all other PCEs in the Supercluster. This behavior ensures the Supercluster can dynamically adapt the policy to changes in the environment, even when the leader is down.

- Policy and workload replication is performed using standard database replication technology of the PCE databases. The replication is trigger-based and only the deltas are transmitted to minimize delays and make efficient use of bandwidth.
- Each member PCE in the Supercluster computes and distributes the firewall rules to its managed workloads based on the replicated policy and workload information. This design leverages the full computing power of the Supercluster to minimize policy convergence times for organization-wide policy changes affecting large numbers of workloads. Distributed policy computation also allows each member PCE to continuously enforce the latest policy, even when the leader is unavailable.

Flow Data and Illumination

Each PCE processes the summarized flow data reported by its managed workloads and stores a computed view of the traffic in memory, as if each were a standalone PCE. The leader periodically queries this data from each PCE to generate an aggregated Illumination map for the entire Supercluster. The raw summarized flow data is not sent to the leader, only the computed view of the flow data. When the raw flow data is needed, it can be streamed from each individual PCE in the Supercluster to one or more log collectors using either syslog or Fluentd.

Supercluster in PCE Web Console

This section describes how to use the PCE web console with a PCE Supercluster.

Overview of Supercluster in Web Console

Each PCE in the Supercluster processes the summarized traffic data reported by its managed workloads and stores a computed view of the traffic in memory, just as on a standalone PCE. The display of this data in the Illumination map, however, looks different depending on whether you are logged into the leader or one of the members:

- The Illumination map on the leader shows an aggregated view of traffic data for the entire Supercluster. The leader periodically queries traffic data from each PCE to generate this map.
- The Illumination map on Supercluster members only shows data from workloads that have been paired with that member PCE.

The following Illumination features are not available in a Supercluster (leader or member):

- Clear traffic for one traffic link
- Increase the VEN reporting rate

These features are only available on a leader (and not available on a member):

- Add a rule from Illumination
- Policy Generator
- App Group configuration

VEN heartbeat and uptime data is not replicated in a Supercluster. It is available only on the leader itself and the individual members themselves:

Leader: Aggregated Illumination Data

The leader of the Supercluster shows a complete picture of all aggregated traffic from all PCEs in your Supercluster. Traffic data from members is refreshed periodically and then cached on the leader.

The refresh interval increases with the number of workloads that you pair with the Supercluster, with a minimum sync interval of 10 minutes and up to 24 hours, depending on how many workloads are paired with your Supercluster. You can force a sync of traffic data from members to the leader at any time, but the sync can take several minutes to complete.

Depending on your network speeds and possible latency, the Illumination map's traffic data can be delayed temporarily while the data is syncing.

Supercluster Illumination Sync with Members

In the lower right of the Illumination map on the leader, a small timer indicates when the Illumination map data was last refreshed.

Click the timer to launch a dialog from which you can refresh the Illumination map data so all traffic from all PCEs in the Supercluster is displayed.

Member: Local Illumination Data

The Illumination map on a member displays traffic information only from those workloads that have been paired with the member PCE. When viewing the Illumination map on a member, you can see a message indicating that you are viewing a local set of traffic data.

Web Console Filtering Problem

In the PCE web console on a Supercluster member, filtering the workload view with **Policy Sync: Active** displays the workloads for the entire Supercluster, instead of workloads for the member on which the report is run. This filter includes workloads marked as "Unavailable."

Workaround: In addition to **Policy Sync: Active**, use the PCE member FQDN filter to exclude all workloads not paired with the desired member. This filter combination is available:

Policy Sync: Active and PCE:Member PCE FQDN

REST API and Supercluster

The types of operations you can perform with the Illumio Core REST API are determined by the permissions granted to your user account by a PCE administrator.

Available Operations on Leader vs. Members

Regardless or your user's permissions, you can only perform read operations on a member, which means you can perform GET operations on members, but not any POST, PUT, or DELETE operations using the REST API.

On the leader, you can perform full CRUD (GET, POST, PUT, DELETE) operations when your user account has the permissions to do so. Other REST API requests that assist in PCE operations, such as checking a node's availability, or determining the Supercluster leader, are available on the leader and members.

REST Operation	Leader	Members
POST, PUT, DELETE	Yes	No
GET	Yes	Yes
DELETE blocked traffic	Yes	Yes
Generate a workload support report	Yes	Yes
Asynchronous GET collections	Yes	Yes
GET product version	Yes	Yes
Check node availability	Yes	Yes
Determine Supercluster leader	Yes	Yes

During a Supercluster rolling upgrade, you can use the REST API on all PCEs except the one that is currently being upgraded. During a Supercluster simple upgrade, you cannot use the REST API until the upgrade has finished on all PCEs. For more information, see Upgrade Supercluster [119].

REST API Login Response

When you have deployed a PCE Supercluster and use the REST API to connect to a PCE in the Supercluster, the response indicates when the PCE is a member of the Supercluster.

For example, when you log into a PCE in a Supercluster:

```
GET https://my.pce.supercluster:443/api/v1/login
```

The response contains a JSON property named `pce_cluster_type` and has a value of either member or leader. For example, you see this response from a leader when you log in:

```
"pce_cluster_type": "leader"
```

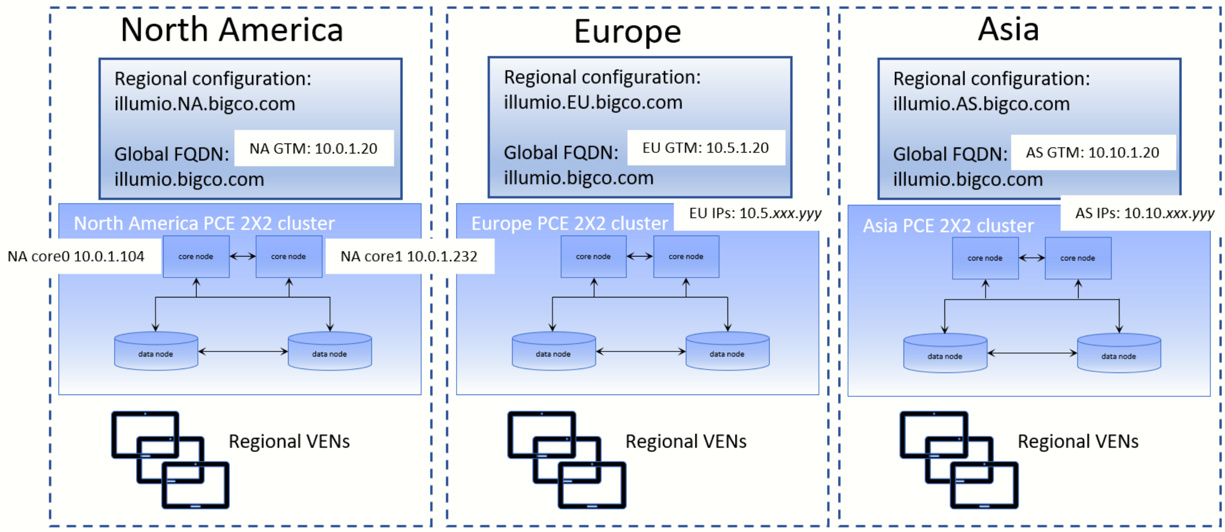
Design Supercluster Deployment

A PCE Supercluster consists of a single administrative domain that spans two or more replicating PCEs. One PCE in the Supercluster is the Supercluster *leader* and the other PCEs are Supercluster *members*. A Supercluster deployment has only one leader. Any member can be manually promoted to be the leader.

The leader has a central PCE web console and REST API endpoint for configuring and provisioning security policy. The PCE web console on the leader provides other centralized management functions, including an aggregated Illumination map to visualize network traffic and policy coverage for all workloads. Members in the Supercluster mostly have a read-only PCE web console and REST API for viewing local data.

Supercluster Logical Architectures

The diagram below shows geographically distributed datacenters for a fictitious company called BigCo.com. This example presents only one of many possible Supercluster configurations.



Supercluster Design Considerations

When planning a PCE Supercluster deployment, consider these important factors:

- **How many total workloads does the PCE Supercluster need to support?** Scale constraints apply to both the number of managed workloads connected to each PCE and the total number of replicated workloads and other policy objects in the PCE's database.
- **How many managed workloads will be connected to each PCE?** Deployments should be sized such that each PCE is able to support the required number of locally-connected workloads and influx of workloads from a different PCE cross-PCE failover is configured.
- **What level of isolation is needed to support PCE outages (failures and maintenance)?** Each PCE in the Supercluster is independent and even a complete failure will not affect other PCEs. Deploying more PCEs in a Supercluster increases the number of failure domains.
- **What should happen to VENs when an extended PCE outage occurs?** By default, VENs continue to enforce the current policy when their PCE is unavailable. When you need to provision policy changes during an extended PCE outage, you can use a GSLB to route orphaned VENs to another PCE in the Supercluster.
- **Which PCE in the PCE Supercluster will be the Leader?** The leader should be in a central location that can be readily accessed by PCE users and REST API clients. The leader should have reliable connectivity to all other PCEs in the Supercluster. Some organizations choose to deploy a leader with no managed workloads to reduce load on this PCE and optimize for REST API data loading.

High Availability and Disaster Recovery

A PCE Supercluster provides multiple levels of redundancy and failover for high availability (HA) and disaster recovery (DR).

Local Recovery

Each PCE in the Supercluster is a multi-node cluster (MNC) that can automatically survive a hardware or software failure affecting any one node. Each half of the PCE can be split across multiple LAN-connected buildings or availability zones, with 10 milliseconds latency between availability zones. Proper operation of Illumination and Explorer is assured when latency is 10ms or less. The PCE can survive a building failure, but manual action (issuing a PCE administrative command) might be necessary, depending on which building is lost.

When a complete failure of a PCE in the Supercluster occurs, its VENs continue to enforce the last known good policy until the PCE is restored or rebuilt from backup. When the leader becomes unavailable, each PCE operates autonomously and continues to distribute the latest provisioned policy to existing and newly paired workloads.

(Optional) Cross-PCE Failover and Recovery

During an extended outage of a PCE, workloads can optionally be failed over to any other PCE to continue to receive policy. Cross-PCE failover requires a GSLB or manual DNS. During failover, a workload's reported traffic flows are streamed via syslog and Fluentd but are not recorded by the PCE.

> **IMPORTANT**
>
> Failover must be carefully managed to ensure the PCE does not exceed its capacity and become overloaded. For this reason, Illumio strongly recommends that failover be done manually and not automated.

PCE Supercluster Deployment

This section describes how to set up a PCE Supercluster, a single administrative domain that encompasses two or more replicating PCEs. Before you start, be sure to understand the concepts explained in Overview of Supercluster Deployment [89].

PCE Supercluster Deployment Planning

Review these requirements before deploying a PCE Supercluster.

Plan Supercluster FQDNs Carefully

Be sure to plan the fully qualified domain names (FQDNs) you want to use with your Supercluster PCEs. Be careful to define these names exactly how you want them before you deploy the Supercluster. Changing FQDNs after deploying a Supercluster is possible but time-consuming. The PCE FQDNs are set in the `pce_fqdn` parameter in `runtime_env.yml`.

For example, you might want to have identifying strings in the FQDNs that indicate the geographic location of the various members of the Supercluster, such as the following examples:

- `illumio-eu.bigco.com: eu` in the hostname indicates Europe.

- `illumio.na.bigco.com`: North America as a separate domain.

You can also configure a global FQDN for the Supercluster. The global FQDN is used by the VENs rather than individual PCE FQDNs. The global Supercluster FQDN is set in the `supercluster_fqdn` parameter in `runtime_env.yml`.

When set, the PCE provides this FQDN instead of its own FQDN to VENs during pairing. This parameter must be set on *all* nodes in *each* PCE of the Supercluster. When you configure this option, each PCE server certificate must include the global FQDN in the SAN field. For example:

- `illumio-supercluster.bigco.com`

Number of Supercluster PCEs

A PCE Supercluster consists of a minimum of two and a maximum of eight (8) PCEs. One of the PCEs is always the Supercluster leader, while the others are Supercluster members.

Capacity Planning for Supercluster PCEs

Use these guidelines and requirements to estimate host system capacity based on typical usage patterns.

The exact requirements vary based on a large number of factors, including, but not limited to:

- Whether you are using physical or virtual hardware
- Number of managed workloads
- Number of unmanaged workloads and other labeled objects, such as virtual services
- Policy complexity, which includes the following factors:
  - Number of rules in your rulesets
  - Number of labels, IP lists, and other objects in your rules
  - Number of IP ranges in your IP lists
  - Number of workloads affected by your rules
- The frequency at which your policies change
- Frequency at which workloads are added or deleted, or workload context changes, such as change of IP address
- Volume of traffic flows per second reported to the PCE from all VENs

  See the “Maximum Flow Capacity” table for information about maximum flow capacity of the PCE.
- Total number of unique flows reported to the PCE from all VENs

Recommended CPU, Memory, and Storage

The capacity planning tables in this section list the minimum recommended sizes for CPU, memory, and storage. This section provides two tables, one for physical hardware and one for virtual machines. Use these tables to plan your deployment.

NOTE

Based on your actual usage and other factors, your capacity needs might be greater than the recommended sizes. For example, if you have installed additional software along with the PCE, such as application performance management (APM) software or an endpoint protection agent, this consumes additional system resources.

Data nodes are configured with a dedicated storage device for each database on the data nodes. This configuration accommodates growth in traffic data, which is used by Explorer. See "Runtime Parameters for Traffic Datastore on Data Nodes" in PCE Installation and Upgrade Guide.

For more than 150 IOPS, locally attached, spinning hard disk drives (HDD) are not sufficient. You will require either mixed-use Solid-State Disk (SSD) or Storage Area Network (SAN).

The PCE does not require that you set up swap memory, but it is permissible to enable swap memory. As long as the PCE nodes are provisioned with the recommended memory (RAM) as shown in the tables below, the use of swap memory should not cause any issues.

Physical Hardware

Use this table if you are installing the PCE on physical hardware. If you are using virtual machines, see the table Virtual Hardware [100].

MNC Type + Workloads/VENs	Cores/Clock Speed	RAM per Node	Storage Device Size and IOPS	
			Core Nodes	Data Nodes
SNC • 250 VENs1 • 2500 workloads	• 3 cores2 • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or equivalent	16GB	A single node including both core and data: • 1 x 50GB4 • 100 IOPS per device5	N/A
2x2 Small • 2,500 VENs1 • 12,500 workloads Cluster type: `4node_v0_small`	• 4 cores per node2 • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or equivalent	32GB	Minimum: • Disk: 50GB3, 4 • 150 IOPS per device5	Minimum: • Disk 1: 250GB4 • Disk 2: 250GB4 • 600 IOPS per device5
2x2 • 10,000 VENs1 • 50,000 workloads Cluster type: `4node_v0` or `4node_dx`	• 16 cores per node2, 6 • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or equivalent	• Recommended: 128GB6 • Minimum: 64GB	Minimum: • Disk: 50GB3, 4 • 150 IOPS per device5	Minimum: • Disk 1: 1TB4 • Disk 2: 1TB4 • 1,800 IOPS per device5
4x2 • 25,000 VENs1 • 125,000 workloads Cluster type: `6node_v0` or `6node_dx`	• 16 cores per node2, 6 • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or equivalent.	128GB6	Minimum: • Disk: 50GB3, 4 • 150 IOPS per device5	• Disk 1: 1TB4 • Disk 2: 1TB4 • 5,000 IOPS per device5

Footnotes:

1 Number of VENs/workloads is the sum of both the number of managed VENs and the number of unmanaged workloads.

2 CPUs:

- The recommended number of cores is based only on physical cores from allocated CPUs, irrespective of hyper-threading.

3 This is the absolute minimum needed. In the future, other applications, support reports, or new features may require additional disk.

4 Additional disk notes:

- Storage requirements for network traffic data can increase rapidly as the amount of network traffic increases.
- Network File Systems (NFS) is not supported for Illumio directories specified in runtime; for example, data_dir, persistent_data_dir, ephemeral_data_dir.

5 Input/output operations per second (IOPS) are based on 8K random write operations. IOPS specified for an average of 300 flow summaries (80% unique `src_ip, dest_ip`, `dest_port`, `proto`) per workload every 10 minutes. Different traffic profiles might require higher IOPS.

6 In the case of fresh installs or upgrades of a 2x2 for 10,000 VENs or a 4x2 for 25,000 VENs, if you deploy a system without sufficient cores, memory, or both, then the PCE will automatically reduce the object limits to 2,500 workloads. Object limit is the number of VENs (agents) per PCE. Adding more than 2,500 workloads will fail and an event is logged indicating that object limits have been exceeded. The workaround is to increase the number of cores, memory, or both to the recommended specifications and then increase the object limits manually. See "PCE Default Object Limits" in the PCE Administration Guide.

Virtual Hardware

Use this table if you are installing the PCE on virtual machines. If you are using physical hardware, see the table Physical Hardware [98].

MNC Type + Workloads/VENs	Virtual Cores/ Clock Speed	RAM per Node	Storage Device Size and IOPS	
			Core Nodes	Data Nodes
SNC • 250 VENs1 • 2500 workloads	• 6 virtual cores (vCPU)2 • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or higher	16GB7	Minimum: • Disk: 50GB3, 4 • 150 IOPS per device5	N/A
2x2 Small • 2,500 VENs1 • 12,500 workloads Cluster type: `4node_v0_small`	• 8 virtual cores (vCPU) per node2 • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or higher	32GB7	Minimum: • Disk: 50GB3, 4 • 150 IOPS per device5	Minimum: • Disk 1: 250GB • Disk 2: 250GB • 600 IOPS per device
2x2 • 10,000 VENs1 • 50,000 workloads Cluster type: `4node_v0` or `4node_dx`	• 32 virtual cores (vCPU) per node2, 6 • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or higher	• Recommended: 128GB6, 7 • Minimum: 64GB	Minimum: • Disk: 50GB3, 4 • 150 IOPS per device5	Minimum: • Disk 1: 1TB4 • Disk 2: 1TB4 • 1,800 IOPS per device5
4x2 • 25,000 VENs1 • 125,000 workloads Cluster type: `6node_v0` or `6node_dx`	• 32 virtual cores (vCPU) per node2, 6 • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or higher	128GB6, 7	Minimum: • Disk: 50GB3, 4 • 150 IOPS per device5	• Disk 1: 1TB4 • Disk 2: 1TB4 • 5,000 IOPS per device5

Footnotes:

1 Number of VENs/workloads is the sum of both the number of managed VENs and the number of unmanaged workloads.

2 Full reservations for vCPU. No overcommit.

3 This is the absolute minimum needed. In the future, other applications, support reports, or new features may require additional disk.

4 Additional disk notes:

- Storage requirements for network traffic data can increase rapidly as the amount of network traffic increases.
- Network File Systems (NFS) is not supported for Illumio directories specified in runtime; for example, data_dir, persistent_data_dir, ephemeral_data_dir.

5 Input/output operations per second (IOPS) are based on 8K random write operations. IOPS specified for an average of 300 flow summaries (80% unique `src_ip, dest_ip`, `dest_port`, `proto`) per workload every 10 minutes. Different traffic profiles might require higher IOPS.

6 In the case of fresh installs or upgrades of a 2x2 for 10,000 VENs or a 4x2 for 25,000 VENs, if you deploy a system without sufficient cores, memory, or both, then the PCE will automatically reduce the object limits to 2,500 workloads. Object limit is the number of VENs (agents) per PCE. Adding more than 2,500 workloads will fail and an event is logged indicating that object limits have been exceeded. The workaround is to increase the number of cores, memory, or both to the recommended specifications and then increase the object limits manually. See "PCE Default Object Limits" in the PCE Administration Guide.

7 Full reservations for vRAM. No overcommit.

Maximum Flow Capacity

The following table shows the maximum capacity of the PCE to accept flow data from all VENs.

Cluster Type + VENs and Total Workloads	Flow Rate (flow-summaries/second)	Equivalent Flow Rate (flows/second)[2]
SNC • 250 VENs • 1,250 total workloads	100	1,030
2x2 • 2,500 VENs • 12,500 total workloads	1,000	10,300
2x2 • 10,000 VENs • 50,000 total workloads	4,100	422,000
4x2 • 25,000 VENs • 125,000 total workloads	10,400 [1]	1,070,000

Footnotes

[1] The PCE might need to be tuned to achieve this rate. If you need to tune the PCE, please contact Illumio Support for assistance.

[2] Real-world observation shows that 102 flows result in one flow summary on average.

Storage Device Layout

You should create separate storage device partitions to reserve the amount of space specified below. See "PCE Capacity Planning" in PCE Installation and Upgrade Guide.

The values given in these recommendation tables are guidelines based on testing in Illumio's labs. If you wish to deviate from these recommendations based on your own platform standards, please first contact your Illumio support representative for advice and approval.

PCE Single-Node Cluster for 250 VENs

Storage Device	Partition mount point	Size to Allocate	Node Types	Notes
Device 1, Partition A	/	8GB	Core, Data	The size of this partition assumes the system temporary files are stored in /tmp and core dump file size is set to zero. The PCE installation occupies approximately 500MB of this space.
Device 1, Partition B	`/var/log`	16GB	Core, Data	The size of this partition assumes that PCE application logs and system logs are both stored in `/var/log`. PCE application logs are stored in the `/var/log/illumio-pce` directory. The recommended size assumes average use by the OS with common packages installed and logging levels set to system defaults. Log size limits are configurable, so your system may require more or less log space. To find the potential maximum disk space required for your logs, use this command: `sudo -u ilo-pce illumio-pce-env logs --diag`
Device 1, Partition C	`/var/lib/illumio-pce`	Balance of Device 1	Core, Data	The size of this partition assumes that Core nodes use local storage for application code in /var/lib/illumio-pce, and also assumes that PCE support report files, and other temporary (ephemeral) files, etc., are stored in /var/lib/illumio-pce/tmp.

PCE 2x2 Multi-Node Cluster for 2,500 VENs

Storage Device	Partition mount point	Size to Allocate	Node Types	Notes
Device 1, Partition A	`/`	16GB	Core, Data	The size of this partition assumes the system temporary files are stored in /tmp and core dump file size is set to zero.
Device 1, Partition B	`/var/log`	32GB	Core, Data	The size of this partition assumes that PCE application logs and system logs are both stored in `/var/log`. PCE application logs are stored in the `/var/log/illumio-pce` directory.
Device 1, Partition C	`/var/lib/illumio-pce`	Balance of Device 1	Core, Data	The size of this partition assumes that Core nodes use local storage for application code in /var/lib/illumio-pce, and also assumes that PCE support report files, and other temporary (ephemeral) files, etc. are stored in /var/lib/illumio-pce/tmp.
Device 2, Single partition. Applicable in a two-storage-device configuration	`/var/lib/illumio-pce/data/Explorer`	All of Device 2 (250GB)	Data	For network traffic data in a two-storage-device configuration for the data nodes, it should be a separate device that is mounted on this directory. Set the `runtime_emv.yml` to `data_dir: /var/lib/illumio-pce/data/Explorer`, which will automatically create a subdirectory called `/var/lib/illumio-pce/data/Explorer/traffic_datastore` The partition mount point and the runtime setting must match. If you customize the mount point, make sure that you also change the runtime setting accordingly.

PCE 2x2 Multi-Node Cluster for 10,000 VENs and PCE 4x2 Multi-Node Cluster for 25,000 VENs

Storage Device	Partition mount point	Size to Allocate	Node Types	Notes
Device 1, Partition A	/	16GB	Core, Data	The size of this partition assumes the system temporary files are stored in /tmp and core dump file size is set to zero.
Device 1, Partition B	`/var/log`	32GB	Core, Data	The size of this partition assumes that PCE application logs and system logs are both stored in `/var/log`. PCE application logs are stored in the `/var/log/illumio-pce` directory.
Device 1, Partition C	`/var/lib/illumio-pce`	Balance of Device 1	Core, Data	The size of this partition assumes that Core nodes use local storage for application code in /var/lib/illumio-pce, and also assumes that PCE support report files, and other temporary (ephemeral) files, etc. are stored in /var/lib/illumio-pce/tmp.
Device 2, Single Partition Applicable in a two-storage-device configuration	`/var/lib/illumio-pce/data/traffic`	All of Device 2 (1TB)	Data	For network traffic data in a two-storage-device configuration for the data nodes, it should be a separate device that is mounted on this directory. In `runtime_env.yml`, set the `traffic_datastore : data_dir` parameter to match the value of the partition mount point (see previous column) as follows: `traffic_datastore: data_dir: /var/lib/illumio-pce/data/traffic`. The partition mount point and the runtime setting must match. If you customize the mount point, make sure that you also change the runtime setting accordingly.

Runtime Parameters for Two-Storage-Device Configuration

In the two-storage-device configuration, to accommodate growth in the traffic data store, set the following parameters in `runtime_env.yml`:

> **NOTE**
> When you are deploying the two-storage-device configuration, you must set these parameters.

`traffic_datastore:`

`data_dir`: `path_to_second_disk`

`max_disk_usage_gb`: Set this parameter according to the table below.

`partition_fraction`: Set this parameter according to the table below.

`time_bucket_type`: Set this parameter according to the table below.

The recommended values for the above parameters based on PCE node cluster type (2x2 or 4x2) and the estimated number of workloads (VENs) are as follows:

Setting	2x2 | 2,500 VENs	2x2 | 10,000 VENs	4x2 | 25,000 VENs	Note
`traffic_datastore:max_disk_usage_gb`	100 GB	400 GB	400 GB	This size reflects only part of the required total size, as detailed in "PCE Capacity Planning" in the PCE Installation and Upgrade Guide.
`traffic_datastore:partition_fraction`	0.5	0.5	0.5	
`traffic_datastore:time_bucket_type`	Day	Day	Day	

Network Traffic Between PCEs

PCEs in the Supercluster communicate via the following ports. Any network firewalls between the PCEs must be configured to allow this traffic.

Ports	Sources	Destinations
The default TCP 8443 or the management port configured for the PCE Web Console and REST API in `runtime_env.yml`. This port must be the same on all PCEs in the Supercluster.	Core nodes of leader PCE	PCE FQDN of all member PCEs
TCP 5432	All nodes of all PCEs	IP addresses of all other PCE data nodes
TCP 5532	Core nodes of leader PCE	IP addresses of all other PCE data nodes
TCP 8302	All nodes of all PCEs	PCE FQDN of all other PCEs and IP address of all nodes of all other PCEs
UDP 8302	All nodes of all PCEs	IP address of all nodes of all other PCEs
TCP 8300	All nodes of all PCEs	IP address of all nodes of all other PCEs

Load Balancers

Similar to a single PCE, all PCEs in the Supercluster must be front-ended with a load balancer (DNS or L4) to distribute requests across the PCEs' core nodes.

GSLB or a manual DNS update can be used to fail over VENs to a different PCE. See GSLB Requirements [107] and High Availability and Disaster Recovery [95].

Traffic Load Balancer Configuration

When you use L4 load balancers in front of the PCEs, the load balancers should already be configured to forward inbound connections on the default TCP 8443 or the management port configured for the PCE web console and REST API in `runtime_env.yml` and 8444 to an available, healthy core node.

In a Supercluster, the L4 load balancer must also be configured to forward additional inbound TCP 8302 connections originating from the other PCEs to an available, healthy core node.

GSLB Requirements

Workloads can be paired to a specific PCE, or you can optionally use a GSLB to route workloads to the required PCE in your Supercluster.

When you are using a GSLB to route workloads, consider the following general guidelines.

For normal operations:

- When all PCEs are available, workloads should be routed to the nearest PCE based on proximity and geolocation.
- GSLB persistence (also known as “stickiness”) must be enabled so workloads are always routed to the same PCE that they are paired with (non-failure case). Balancing workloads across multiple PCEs is not supported.

For failover:

- Recommended: A dedicated failover PCE joined to the Supercluster that has no other VENs.
- Failover to any other PCE in the Supercluster. In this case, take care to prevent overloading the PCE beyond its rated capacity and to avoid cascading failures. One strategy is to configure a “buddy” PCE for each PCE that the GSLB uses for failover.
- Workload failover time depends on the DNS time-to-live (TTL) configured in the GSLB.
- Illumio strongly recommends that you do not automate workload failover using GSLB and instead initiate it manually.

Configure SAML IdP for User Login

After installation, you can configure the PCE to rely on an external, third-party SAML identity provider system IIdP). See "Single Sign-On Configuration" in PCE Administration Guide .The guide provides set up instructions for a wide variety of IdPs.

For the PCE Supercluster, you configure the details in the leader PCE web console exactly as you do for the standalone PCE, with one exception: you are presented an intermediate page that lists all the PCEs in the Supercluster, including the leader and all members. Follow the same processes detailed in PCE Administration Guide to configure all the Supercluster PCEs, both leader and members.

Certificate Requirements

PCE-to-PCE communication is done over TLS v1.2. The root CA certificate that signed each PCEs certificate must be in the root CA bundle on all other PCEs in the Supercluster.

Object Limits and Supercluster

The PCE enforces certain soft and hard limits to restrict the total number of system objects you can create. These limits are based on tested performance and capacity limits of the PCE. Most PCE object limits apply to the entire Supercluster. The limits are enforced by the leader when objects are created.

The object limit for the number of VENs per PCE (`active_agents_per_pce`) is not cluster-wide and applies to each PCE. When the VENs per PCE limit is reached, no more VENs can be paired to that PCE. This limit is enforced by moving VENs from one PCE to another via the REST API.

An exception is made when VENs are failed over by the system itself from one PCE to a different PCE in the cluster. The VENs that failover do not count towards the limit, allowing you to temporarily exceed the limit of VENs per PCE when an extended outage to a PCE in the Supercluster occurs.

Changes to the object limit for the number of VENs per PCE (`active_agents_per_pce`) made on the Supercluster leader are propagated to the members within 30 minutes.

For more information on object limits and how to view your current object limit usage, see PCE Administration Guide, the command **"illumio-pce-ctl obj-limits list**".

RBAC Permissions: Leader or Member

In general, when you are using the Illumio PCE web console or the Illumio REST API, the types of operations you can perform depend on your PCE role-based access control (RBAC) permissions and whether you have logged into the leader or a member, as shown in the table below.

User Role	Operations	Leader	Members
Any Role	View objects	Yes	Yes
Global Administrator & User Manager	Add, delete users	Yes	No
(Organization Owner)	Add, modify, delete, and provision system objects and rulesets (includes creating a pairing script).		
Global Administrator	Add, modify, delete, and provision system objects and rulesets (includes creating a pairing script)	Yes	No
Global read-only	View all objects	Yes	Yes
Global Policy Object Provisioner	Provision system objects	Yes	No
Ruleset Manager	Create, update, and delete rulesets within defined scopes.	Yes	No
Ruleset Provisioner	Provision rulesets within defined scopes.	Yes	No

Process, File Limits, and Kernel Parameters

Even if you are running **systemd**, the file and kernel limits must be set as outlined in our build document for init.d systems.

Servers with `systemd` also need to make the configuration changes outlined for `init.d` systems because some of our supercluster command-line tool commands are hard-coded to reference `init.d` security limits. It is necessary to set file and process limits for both configuration file changes. Please refer to our build documentation for the required settings.

For reference, see "Requirements for PCE Installation" in "PCE Installation and Upgrade".

Configure PCE Internal Syslog on Leader

You can configure the PCE's internal syslog service in the PCE web console on the Supercluster leader, for both the leader and the member PCEs. The internal syslog cannot be configured on a member PCE.

NOTE

When a standalone PCE is installed, a local destination for the PCE internal syslog is created to record events. When the PCE is joined as a member of the Supercluster, this local destination is removed.

After joining a member, you have to log into the Supercluster leader and configure the internal syslog for each member individually.

When the events occurring before joining a PCE as a member are essential to preserve, back up the PCE before you join it to the Supercluster.

See PCE Installation and Upgrade Guide for information about the PCE internal syslog.

Deploy a PCE Supercluster

You can deploy the Illumio Supercluster in several ways:

- **New:** You have never deployed a PCE and want to deploy a new Supercluster. See Deploy New Supercluster [110].
- **Expand:** You have already deployed a standalone PCE and want to expand it to a Supercluster. See Expand Standalone PCE to Supercluster [114].
- **Join:** You already have more than one standalone PCE and you want to join them together into a Supercluster. Contact your Illumio Customer Support for assistance.

Deploy New Supercluster

Deploying a new PCE Supercluster follows this general workflow:

1. Install the leader PCE as a standalone PCE.
2. Install and configure each member PCE as a standalone PCE.
3. Initialize the Supercluster leader.
4. Join members to the Supercluster.
5. Bring the leader and members to a fully operational state.
6. Verify that the Supercluster is ready for use.

> **NOTE**
> The sequence of events for deploying a Supercluster is not bound by any time requirements; for example, there is no time limit between initializing a Supercluster leader and joining individual members.

Before You Begin: Runtime Configuration

Before you deploy your PCE Supercluster, be aware of the following `runtime_env.yml` configurations:

- The value of the parameter `service_discovery_encryption_key` in the `runtime_env.yml` file must be exactly the same on all nodes on all PCEs in your Supercluster.
- You do not need to configure the public IP addresses of other PCEs under the `cluster_public_ips` parameter. Supercluster PCEs automatically exchange their configured public IP addresses with each other, which get programmed by the VEN to allow workloads to migrate between PCEs.

Optional

Depending on your deployment environment, you might need to make the following changes to the `runtime_env.yml` file on each PCE in the Supercluster.

When the nodes of each PCE use multiple IP addresses or they use IP addresses other than the one advertised on the node for communication with other PCEs, such as having a NAT between the PCEs in your Supercluster, configure this optional parameter:

- `supercluster.node_public_ip`: The public IP address of this node is advertised to other PCEs in your Supercluster deployment. This IP address must be reachable from all other Supercluster PCEs that you want to join. This parameter must be set on *all* nodes in *each* PCE. When your PCE is deployed in a public cloud, such as AWS, this must be a public IP address.

When you configure your GSLB for routing VENs to the appropriate PCE, configure this optional parameter on each node in a PCE:

- `supercluster.fqdn`: The PCE responds to this FQDN, instead of its own canonical FQDN to VENs during pairing. This parameter must be set on *all* nodes in *each* PCE of the Supercluster.

For example:

```
supercluster:
   node_public_ip: 192.168.33.10
   fqdn: global-pce.mycompany.com
```

Install Leader

The first step to deploy a new Supercluster is to install and configure the leader PCE, just as you would install a standalone PCE.

For detailed information about installing a PCE, see the PCE Installation and Upgrade Guide.

Install Members

Install each member of your Supercluster by following the exact same procedures you use installing a standalone PCE, except *do not* create a domain during deployment.

For information about installing a PCE, see the PCE Installation and Upgrade Guide.

Initialize Supercluster Leader

After the leader has been installed, configured, and verified, you initialize the leader.

NOTE

You must initialize the leader *before* you start joining any members.

1. On *any node*, bring all nodes to runlevel 2:

   ```
   sudo -u ilo-pce illumio-pce-ctl set-runlevel 2
   ```

   Setting the run level might take some time to complete.

2. Check the progress with `illumio-pce-ctl cluster-status -w` to see when the status is Running:

   ```
   sudo -u ilo-pce illumio-pce-ctl cluster-status -w
   ```

   The nodes must be at runlevel 2 before you run the next command. When all of the nodes have reached runlevel 2, you see the following output:

   ```
   Illumio Runtime System                           RUNNING [2] 34.28s
   ```
3. On *any node*, initialize the leader:

   ```
   sudo -u ilo-pce illumio-pce-ctl supercluster-init-leader
   ```

Join Each Member to Supercluster

> **IMPORTANT**
> You must join only one member one at a time, and complete all steps before joining the next member. Ensure that each member is at runlevel 2 before joining.

In this procedure, you join the new member to the Supercluster.

All nodes must start at runlevel 2. The nodes should already be at runlevel 2 from the previous procedure.

1. If necessary, on *any node*, bring all nodes to runlevel 2:

   ```
   sudo -u ilo-pce illumio-pce-ctl set-runlevel 2
   ```
2. On *any node*, run the following command while you wait for all nodes to reach runlevel 2:

   ```
   sudo -u ilo-pce illumio-pce-ctl status --wait
   ```
3. On *any core node* or the *data0 node of the member cluster*, join the member to the Supercluster (identified by the leader's FQDN):

   ```
   sudo -u ilo-pce illumio-pce-ctl supercluster-join leader_pce_fqdn
   ```

   While this command runs, the PCE temporarily sets the runlevel to 1. When the command is interrupted, you might unexpectedly see runlevel 1 .

   > **IMPORTANT**
   > Running this command can take an hour or more depending on the number of PCEs in your Supercluster and size of the PCE database. When this command fails due to network latency, do not proceed until you can run the command again and it executes successfully.
4. Repeat step 3 for all members you want to join to the Supercluster.
5. On *all PCEs*, restart the PCEs in the Supercluster:

   ```
   sudo -u ilo-pce illumio-pce-ctl cluster-restart
   ```

6. On *all PCEs*, bring the PCEs to runlevel 5:

```
sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
```

Verify Supercluster Readiness

Before you begin using your Supercluster, verify that the leader and members are all joined and all PCEs in the Supercluster have a good health status.

> **NOTE**
> It can take up to 10 minutes for all PCEs in your Supercluster to achieve full healthy status.

To verify that your Supercluster is ready to use:

1. Log into the leader.
2. On *any core node*, show Supercluster membership:

   ```
   sudo -u ilo-pce illumio-pce-ctl supercluster-members
   ```

   The output should show all PCEs in your Supercluster.
3. Log into the PCE web console of the leader.
4. Click the Health status icon at the top of the PCE web console. You should see all PCEs in your Supercluster with **Normal** health status.

If a new PCE being added to the Supercluster has a different value for the parameter `service_discovery_encryption_key` defined in its `runtime_env.yml` file than the value specified in the `runtime_env.yml` files in all the other PCEs in the Supercluster, the new PCE will fail to join the Supercluster.

To remedy this possible problem when a new PCE does not join the Supercluster, follow these steps:

1. On the new PCE, edit its `runtime_env.yml` file so that its value for `service_discovery_encryption_key` is identical to the value set in the `runtime_env.yml` files of all other Supercluster nodes.
2. Reset all nodes:

   ```
   sudo -u ilo-pce illumio-pce-ctl reset
   ```
3. On all nodes, start services at runlevel 1:

   ```
   sudo -u ilo-pce illumio-pce-ctl start --runlevel 1
   ```

   Note: If a node gets stuck in the PARTIAL state, reboot the node.
4. On any node, set up the database:

   ```
   sudo -u ilo-pce illumio-pce-db-management setup
   ```
5. On any node, set runlevel 5:

   ```
   sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
   ```

Expand Standalone PCE to Supercluster

To expand your existing standalone PCE to a Supercluster, the steps are similar to the steps for installing a new Supercluster, with additional checks of the standalone PCE before the expansion.

The general workflow for expanding an existing PCE deployment into a Supercluster follows these steps:

1. Change the `pce_fqdn` on your standalone PCE; then log into the standalone PCE's web console to verify that the standalone PCE is healthy and is working correctly. See Change FQDN and Verify PCE Health [114] for information.
2. Verify network connectivity to the database nodes. See Network Connectivity from PCEs to Data Nodes [114] for information.
3. Initialize your existing PCE as the Supercluster leader. See Initialize Supercluster Leader [111] for information.
4. Install and configure the new PCEs that will become members of the new Supercluster. See the PCE Installation and Upgrade Guide for information.
5. Join members to the Supercluster. See Join Each Member to Supercluster [112] for information.

Illumio recommends that you perform each these operations during different change windows.

After your Supercluster is operational, you can reassign workloads connected to the leader to a different PCE in the Supercluster.

Change FQDN and Verify PCE Health

See How to Rename the PCE in the Illumio Knowledge Base for information. (Login required)

> **WARNING**
> After changing the `pce_fqdn` parameter and before preceding with the expansion, you must log into the standalone PCE's web console to verify that the standalone PCE is healthy and is working correctly.

Network Connectivity from PCEs to Data Nodes

Before expansion of a Supercluster, ensure that every data node in the standalone cluster can connect to the database nodes via the Supercluster FQDN.

To verify the connections, use `telnet` or the `nc` (netcat) utility, which is part of the NMAP set of tools. If not already installed, install NMAP with the following command:

```
# yum install nmap
```

> **IMPORTANT**
>
> Required runlevels
>
> Be sure that the PCEs are set to the following runlevels before checking connectivity:
>
> - On the PCE from which you run the check: Runlevel 2
> - On the PCEs in other regions that you are checking: Runlevel 2 or higher

For example, you have three regions. With the following `nc` commands on data0 and data1 in each region, test the connection to the other regions by connecting to port 5432 for the other regions' data0 and data1 nodes.

- From region 1: Set the PCE from which you are testing to runlevel 2:

  ```
  nc -zv region2_data0_ip 5432
  nc -zv region3_data0_ip 5432
  ```
- From region 2: Set the PCE from which you are testing to runlevel 2:

  ```
  nc -zv region1_data0_ip 5432
  nc -zv region3_data0_ip 5432
  ```
- From region 3: Set the PCE from which you are testing to runlevel 2:

  ```
  nc -zv region1_data0_ip 5432
  nc -zv region2_data0_ip 5432
  ```

Migrate to New Supercluster

When you need to migrate your existing Supercluster to a new set of machines, follow these general steps:

1. On the PCE that is being migrated, pre-configure the IP addresses of the new Supercluster in the `runtime_env.yml` file. See Pre-configure New IP Addresses [115] for information.
2. Back up the current Supercluster. See Back Up Supercluster [147] for information.
3. Restore the old Supercluster configuration and data to the new systems. If you are migrating all PCEs in the Supercluster, see Restore an Entire Supercluster [155]. If you are migrating just one PCE in a Supercluster, see Restore a Single PCE in a Supercluster [152].

Pre-configure New IP Addresses

Before the migration: When you use DNS-based load balancing (such as round-robin DNS) and are using new IP addresses for the restored PCE, be sure to record those new IP addresses in the `runtime_env.yml` file on all Supercluster core nodes. This allows VENs to continue to communicate with the PCEs after migration.

> **NOTE**
> When you use traffic-based load balancing, such as with the F5, you do *not* need to add the new IP addresses to `runtime_env.yml`. The VENs communicate exclusively with the traffic load balancers' virtual IP addresses, and not directly with the PCEs.

To update `runtime_env.yml` with additional IP addresses:

If you are migrating more than one PCE, repeat the following steps for each PCE one at a time.

1. On *all nodes* of the PCE being migrated, edit the `runtime_env.yml` file. Under the `cluster_public_ips.cluster_fqdn` parameter, add the new IP addresses of all of the core nodes in this PCE:

   ```
   cluster_public_ips:
     cluster_fqdn:
       - <old IP address>
       - <old IP address>
       - <new IP address>
       - <new IP address>
     cluster_event_service_fqdn:
       - <old IP address>
       - <old IP address>
       - <new IP address>
       - <new IP address>
   ```
2. Restart the PCE that was migrated to send the configuration update to the other PCEs:

   ```
   sudo -u ilo-pce illumio-pce-ctl restart
   ```
3. Wait for all services to start on the restarted PCE.

   ```
   sudo -u ilo-pce illumio-pce-ctl cluster-status
   ```

   The command output should show RUNNING.
4. Restart the other PCEs in the Supercluster using the same commands in steps 2 and 3.
5. If you are migrating more than one PCE in the Supercluster, repeat these steps for each PCE one at a time.

Supercluster Command-line Reference

The Illumio PCE control interface for Supercluster commands often have restrictions on the type of node they can be run on. For example, setting a cluster's runlevel can be run from any core or data node. Other database specific commands must only be run on specific data nodes. The following tables list the different command line operations you can perform and the specific node (or nodes) the commands must be run on.

Supercluster Commands to Node Reference

This section lists commands you can use to control behavior of PCEs and PCE databases in a Supercluster.

Supercluster PCE Control Commands

The commands have the following general syntax. The `--retry-count` is optional and defaults to 5.

```
sudo -u ilo-pce illumio-pce-ctl sub-command--option [--retry-count]
```

The following table shows commands you can use to control PCE behavior:

Command	Description	Run on Node
`reset`	Revert a PCE to standalone state.	The affected node being repaired to join the Supercluster
`supercluster-assign-leader`	Designate an existing member PCE cluster to be a Supercluster leader.	Any node
`supercluster-drop [failed_pce_fqdn]`	Temporarily drops the failed PCE from the Supercluster, so it is no longer replicated.	Any node
`supercluster-init-leader`	Assign a PCE cluster as the leader of your Supercluster.	Any node
`supercluster-join`	Joins a PCE cluster to a Supercluster. Running this command can take up to 30 minutes depending on the number of PCEs in your Supercluster and size of the PCE database.	On any core node
`supercluster-members`	Displays all current active Supercluster PCEs, members and leader.	On any core node or the data0 node
`supercluster-restore`	Restores a formerly failed leader or member to be restored to the Supercluster. This command can be run for a leader or a member. Executing this command can take up to 1 hour depending on the number of PCEs in the Supercluster and size of the PCE database.	On any core node

Supercluster PCE Database Commands

The commands have the following general syntax. The `--retry-count` is optional and defaults to 5.

```
sudo -u ilo-pce illumio-pce-db-management sub-command --option [--retry-count]
```

The following table shows commands you can use to control PCE database behavior:

Command	Description	Run on Node
`supercluster-data-restore`	Restores a failed PCE's data, using a backup taken from that PCE before the failure.	On one of the data nodes only
`supercluster-quiesce`	Pauses all the pending database replication; for example, during a software upgrade.	Any node

Re-runnable illumio-pce-ctl Arguments

All arguments to `illumio-pce-ctl` are re-runnable in case of a command failure.

Argument on illumio-pce-ctl	Description
`supercluster-init-leader`	Configures this PCE as the Supercluster leader.
`supercluster-join [supercluster_leader_fqdn]`	Joins this PCE into the Supercluster specified by the FQDN of the Supercluster leader. While this command is running, it temporarily sets the runlevel to 1. When the command is interrupted, you might see runlevel 1 unexpectedly.
`supercluster-assign-leader`	Assigns a new Supercluster leader.
`supercluster-restore failed_pce_fqdn [supercluster_leader_fqdn] [--restore-type single_pce|entire_supercluster]`	Restores a failed PCE and rejoins it to the Supercluster. Replace `failed_pce_fqdn` with the FQDN of the PCE you are restoring. The `supercluster_leader_fqdn` is required when you are restoring a member PCE, not the leader PCE. While this command is running, it temporarily sets the runlevel to 1. When the command is interrupted, you might see runlevel 1 unexpectedly.
`supercluster-drop [failed_pce_fqdn]`	Temporarily drops the failed PCE from the Supercluster, so it is no longer replicated.
`supercluster-members`	Shows the members in the Supercluster.
`supercluster-config`	Shows the Supercluster configuration.
`supercluster-replication-check [--detailed] [--show-data-detail]`	Displays the state of replication. The `--detailed` option displays more verbose output. The `--show-data-detail` option displays primary keys for the replication data check.

Re-runnable illumio-pce-db-management Arguments

All arguments to `illumio-pce-db-management` are re-runnable in case of a command failure.

Argument on illumio-pce-db-managment	Description
`supercluster-data-dump --file desired_location_of_backup_file`	Writes the database persistent state in Supercluster to a file.
`supercluster-data-restore --restore-type single_pce|entire_supercluster [skip-db-restore]`	Restores data and persistent state of a Supercluster database.
`show-supercluster-replication-info`	Display Supercluster node replication information.
`supercluster-quiesce [wait_timeout]`	Quiesces all pending replication.
`supercluster-replication-debug [--detailed]`	Show replication-related information for debugging.

Upgrade Supercluster

This topic describes installing a newer software version on PCEs in a Supercluster.

> **IMPORTANT**
> The **supercluster-quiesce** and **set-runlevel** commands can be run on any node.

Before Upgrading

Before you upgrade the Supercluster, perform these steps:

1. Back up the PCE.
   Before the upgrade, back up the leader and all member databases and each PCE's `runtime_env.yml` file.
2. Ensure all PCEs are in a healthy state.
   Before upgrading, make sure all PCEs in the entire Supercluster are in a healthy state.
   In the PCE web console, check the PCE Health page to ensure the PCE health status is **Normal**.

Types of Supercluster Upgrade

You can choose to perform a simple upgrade or a rolling upgrade.

- **Supercluster simple upgrade**: The Supercluster simple upgrade procedure requires you to set all the PCEs in the Supercluster to runlevel 1 for the duration of the upgrade. During a simple upgrade, the Supercluster is not fully operational. See Supercluster Simple Upgrade [120].
- **Supercluster rolling upgrade**: A rolling upgrade keeps the Supercluster operational while individual PCEs are upgraded one at a time. See Supercluster Rolling Upgrade [123].

NOTE

Supercluster rolling upgrades are supported only for a hotfix or maintenance releases. The major and minor release numbers in the installed and upgrade versions must match. For example, you can do a rolling upgrade from 21.2.0 to 21.2.1.

Supercluster Simple Upgrade

A Supercluster simple upgrade follows these general steps:

1. On all PCEs, quiesce the data replication.
2. Synchronize data.
3. Upgrade the software on all nodes of all PCEs.
4. Migrate the database on all PCEs.
5. Bring all PCEs back to runlevel 5.

Steps for Upgrade

1. **Quiesce data replication.**
   a. On *any node* in the PCE cluster, bring all PCEs to runlevel 2:
   b. In *the PCE clusters*, repeat step (a) for all leaders and all members.
      The cluster status should be RUNNING.
   c. On *any node in all PCE clusters*, verify that the `set-runlevel` command finished and the cluster status is `RUNNING`:

      ```
      sudo -u ilo-pce illumio-pce-ctl cluster-status -w
      ```

      Do not proceed to the next step until the `set-runlevel` command finishes.
   d. Quiesce database replication.
      On the *active DB node*, run the following command.

      ```
      sudo -u ilo-pce illumio-pce-db-management supercluster-quiesce
      timeout_in_seconds
      ```

      ```
      sudo -u ilo-pce illumio-pce-db-management supercluster-quiesce 600
      ```

      This command waits for data replication to finish, which can take some time. To set a time limit, use `timeout_in_seconds` (default: 600). If the command doesn't complete within this time, it will stop. You must then run the command again.
      Expected output when database replication is successfully quiesced:

      ```
      Replication is complete.
      ```
2. **Synchronize the data.**
   Ensure replication data is properly synchronized between all PCEs in the Supercluster. Exactly how you ensure this synchronization differs depending on whether the version of the Supercluster PCEs being upgraded is older than 23.5.30 or not.
   If upgrading from Supercluster PCEs running a release older than 23.5.30:
   a. Contact your Illumio representative or Illumio Support to obtain a `supercluster_management.rb` script applicable to your current (pre-upgrade) PCE version.
   b. Copy the provided script to the PCE scripts directory (typically `/opt/illumio-pce/illumio/scripts`).

c. Run the following command to ensure primary keys in replication tables match across all PCEs:

```
/opt/illumio-pce/illumio/scripts/supercluster-management.rb
supercluster-replication-check --detailed-data-check --max-id-
consistency-check
```

d. If this check fails, see Fixing Inconsistency Errors [122].

If upgrading from Supercluster PCEs running release 23.5.30 or newer:

a. Run the following command to ensure primary keys in replication tables match across all PCEs.

```
illumio-pce-ctl supercluster-replication-check --detailed-data-check --
max-id-consistency-check
```

b. If this check fails, see Fixing Inconsistency Errors [122].

3. **Upgrade the software.**

   Because this is a simple upgrade, you upgrade the software on all nodes of all PCEs in parallel.

   a. On *any node*, stop the PCE cluster:

   ```
   sudo -u ilo-pce illumio-pce-ctl cluster-stop
   ```

   b. The packages to install depend on the type of PCE node:
      - Core nodes: Two packages, the PCE RPM and UI RPM.
      - Data nodes: One package, the PCE RPM.

      On *each core node* in the cluster, log in as root and install the PCE RPM and UI RPM. Be sure to specify both of the RPM file names on the command line:

      ```
      rpm -Uvh illumio_pce_rpm illumio_ui_rpm
      ```

      For `illumio_pce_rpm` and `illumio_ui_rpm`, substitute the paths and filenames of the two RPM files you downloaded from the Illumio Support portal.

   c. On *each data node* in the cluster, log in as root and install the PCE RPM:

      ```
      rpm -Uvh <illumio_pce_rpm>
      ```

      For `illumio_pce_rpm`, substitute the path and filename of the software you downloaded from the Illumio Support portal.

   d. On *all nodes*, start each cluster at runlevel 1:

      ```
      sudo -u ilo-pce illumio-pce-ctl start --runlevel 1
      ```

4. **Update the runtime environment file** (`runtime_env.yml`).

   See "What's New in This Release" to determine whether any changes to `runtime_env.yml` are required to upgrade. If changes are required:

   a. On *all nodes* in the cluster, update the `runtime_env.yml` file.

   b. On *all nodes* in the cluster, check the validity of the `runtime_env.yml` file:

      ```
      sudo -u ilo-pce illumio-pce-ctl check-env
      ```

      If any issues are reported by this command, correct them before moving on to the next step.

5. **Migrate the PCE database.**

   a. On *any node* of *every upgraded PCE*, run the following command:

      ```
      sudo -u ilo-pce illumio-pce-db-management migrate --upgrade-type simple
      ```

      If you encounter a "`max id inconsistency`" error at this point, see Fixing Inconsistency Errors [122].

b. The migration might take some time to complete. Check the progress with the following command:

```
sudo -u ilo-pce illumio-pce-db-management supercluster-upgrade-status
```

c. On *any node* in the *first PCE cluster*, bring all PCEs to runlevel 2. Repeat this step on all the other PCEs.

```
sudo -u ilo-pce illumio-pce-ctl set-runlevel 2
```

d. For all leader and member PCE clusters, repeat step b. Verify that all PCEs in the Supercluster are at runlevel 2.

e. Wait until `agent_slony_service` and `login_slony_service` are up and running. These service names appear in bright blue or may have a pound character (#) appended, depending on which color option was chosen when starting the PCE, `--color` or `--no-color`. **Do not restart the PCE.** This step could take some time, depending on how recently you upgraded the PCE software. Run the following command to monitor the progress:

```
sudo -u ilo-pce illumio-pce-ctl cluster-status -w
```

Issue the command again, when needed, until the services are ready.

6. **Bring PCEs back to operational status.**

a. On *any node for each PCE*, set the runlevel to 5 :

```
sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
```

Setting the runlevel can take time to complete.

b. On *any node in all PCE clusters*, verify that the `set-runlevel` command finished and the cluster status is `RUNNING`:

```
sudo -u ilo-pce illumio-pce-ctl cluster-status -w
```

> **NOTE**
> Due to the time it takes to replicate new database tables across all the PCEs, the upgrade might take longer than usual. The delay occurs when you bring the PCE to runlevel 2 or 5 from runlevel 1 after upgrading the software. The wait time depends on the number of new tables that are part of the upgrade. The wait might be up to 20 minutes.

c. Verify that you can log into the PCE web console on each PCE in the Supercluster.

7. The upgrade is complete.

Fixing Inconsistency Errors

Failing to follow the recommended steps prior to upgrading a PCE Supercluster to version 23.5.x and later may lead to the PCE displaying errors like the ones shown below when running the `migrate` command:

```
sudo -u ilo-pce illumio-pce-db-management migrate --upgrade-type simple

Checking max id consistency...
------------------------------------------------------------
Fetching tables from ...
Fetching tables from ...
max_id mismatch found in PCE pair with region_id[1, 7]
Replication max_id is NOT in a consistent state.
```

```
max id check found inconsistency across PCEs.
The migration cannot continue. Please contact Illumio support.
```

Inconsistencies encountered while running `migrate` or the `supercluster-replication-check` before attempting to migrate can be corrected by using the `illumio-pce-ctl` command or the `supercluster_management.rb` script, depending on the version the PCE is being upgraded from.

- **To correct inconsistencies on PCE versions older than 23.5.30**
  Note the name of each table that was reported as mismatched, and correct the mismatch by running the same `supercluster_management.rb` script from the Synchronize Data step for each mismatched table:

  ```
  /opt/illumio-pce/illumio/scripts/supercluster_management.rb supercluster-replication-sync --table-name <table name>
  ```
- **To correct inconsistencies on PCE versions newer than 23.5.30**
  Note the name of each table that was reported as mismatched, and correct the mismatch by running the `supercluster-replication-sync` command for each mismatched table

  ```
  illumio-pce-ctl supercluster-replication-sync --table-name <table_name>
  ```

Supercluster Rolling Upgrade

In a rolling upgrade, the PCEs are upgraded one by one. The PCE that is being upgraded is at runlevel 1, while all the other PCEs are fully operational (runlevel 5).

> **NOTE**
> Supercluster rolling upgrades are supported only for a hotfix or maintenance releases. The major and minor release numbers in the installed and upgrade versions must match. For example, you can do a rolling upgrade from 21.2.0 to 21.2.1.

A Supercluster rolling upgrade follows these general steps:

1. Upgrade the software on all nodes of the leader PCE.
2. Migrate the database on the leader PCE.
3. Bring the leader PCE back to runlevel 5.
4. Repeat these steps for each member PCE.

Steps for Upgrade

1. **Upgrade the software on the leader PCE.**
   a. On *any node* of the leader PCE, stop the PCE cluster:

      ```
      sudo -u ilo-pce illumio-pce-ctl cluster-stop
      ```
   b. The packages to install depend on the type of PCE node:
      - Core nodes: Two packages, the PCE RPM and UI RPM.
      - Data nodes: One package, the PCE RPM.

      On *each core node* in the cluster, log in as root and install the PCE RPM and UI RPM. Be sure to specify both of the RPM file names on the command line:

```
rpm -Uvh illumio_pce_rpm illumio_ui_rpm
```

For `illumio_pce_rpm` and `illumio_ui_rpm`, substitute the paths and filenames of the two RPM files you downloaded from the Illumio Support portal.

c. On *each data node* in the cluster, log in as root and install the PCE RPM:

```
rpm -Uvh <illumio_pce_rpm>
```

For `illumio_pce_rpm`, substitute the path and filename of the software you downloaded from the Illumio Support portal.

d. On *all nodes*, start the cluster at runlevel 1:

```
sudo -u ilo-pce illumio-pce-ctl start --runlevel 1
```

2. **Update the runtime environment file** (`runtime_env.yml`).

See "What's New in This Release" to determine whether any changes to `runtime_env.yml` are required to upgrade. If changes are required:

a. On *all nodes* in the cluster, update the `runtime_env.yml` file.

b. On *all nodes* in the cluster, check the validity of the `runtime_env.yml` file:

```
sudo -u ilo-pce illumio-pce-ctl check-env
```

If any issues are reported by this command, correct them before moving on to the next step.

3. **Migrate the PCE database on the leader PCE.**

a. On *any node* of the *leader PCE*, run the following command:

```
sudo -u ilo-pce illumio-pce-db-management migrate --upgrade-type
rolling
```

b. The migration might take some time to complete. Check the progress with the following command:

```
sudo -u ilo-pce illumio-pce-db-management supercluster-upgrade-status
```

4. **Bring the leader PCE back to operational status.**

a. On *any node of the leader PCE*, set the runlevel to 5 :

```
sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
```

Setting the runlevel can take time to complete.

b. On *any node of the leader PCE*, verify that the `set-runlevel` command finished and the cluster status is `RUNNING`:

```
sudo -u ilo-pce illumio-pce-ctl cluster-status -w
```

5. **Upgrade the software on a member PCE.**

a. On *any node* of the member PCE, stop the PCE cluster:

```
sudo -u ilo-pce illumio-pce-ctl cluster-stop
```

b. On *all nodes* of the member PCE, install the new version of the PCE. For information, see the PCE Installation and Upgrade Guide.

c. On *any node*, start the cluster at runlevel 1:

```
sudo -u ilo-pce illumio-pce-ctl start --runlevel 1
```

6. **Migrate the PCE database on the member PCE.**

a. On *any node* of the *member PCE,* run the following command:

```
sudo -u ilo-pce illumio-pce-db-management migrate
```

b. The migration might take some time to complete. Check the progress with the following command:

```
sudo -u ilo-pce illumio-pce-db-management supercluster-upgrade-status
```

7. **Bring the member PCE back to operational status.**
   a. On *any node of the member PCE*, set the runlevel to 5 :

      ```
      sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
      ```

      Setting the runlevel can take time to complete.
   b. On *any node of the member PCE*, verify that the `set-runlevel` command finished and the cluster status is `RUNNING`:

      ```
      sudo -u ilo-pce illumio-pce-ctl cluster-status -w
      ```
8. Repeat steps 4 through 6 for each additional member PCE.
9. Verify that you can log in to the PCE web console on each PCE in the Supercluster.
10. The upgrade is complete

During Supercluster Rolling Upgrade

During a rolling upgrade, if you log in to one of the PCEs, you will see a banner that states the Supercluster is in the process of a rolling upgrade.

The PCE Health page on the Leader displays the upgrade status for each PCE. The Upgrade Status column shows Pending if the PCE is in the process of being upgraded, and it shows Complete when the upgrade is complete. When the upgrade is finished, the Upgrade Status column no longer appears.

Rolling Upgrade Paths

For rolling upgrades, the major and minor versions must be the same, and only the patch number can be different (21.2.1: major 21, minor 2, patch 1).

This means that only HotFix (HF) or Maintenance Releases (MR) are qualified for Rolling Upgrades. Also, the HF or MR should NOT contain a migration script that changes the replication tables.

The tables below indicate whether Rolling Upgrade paths are allowed.

Table 1. Rolling Upgrade Paths for Consecutive Versions

Upgrade Path	Allowed (Yes/No)	Notes
21.2.0 > 21.2.1	Yes	
21.2.1 > 21.2.2	No	
21.2.2 > 21.2.3	Yes	
21.2.3 -> 21.2.4	Yes	
21.2.4 -> 21.2.7	Yes	
21.5.2	n/a	SaaS only release
21.5.3 → 21.5.10	Yes	
21.5.10 → 21.5.12	Yes	
21.5.12 → 21.5.20	Yes	
21.5.20 → 21.5.21	Yes	
21.5.21 → 21.5.30	Yes	
21.5.30 → 21.5.31	Yes	
21.5.31 → 21.5.32	Yes	
21.5.32 → 21.5.33	Yes	
22.2.1 → 22.2.10	Yes	
22.2.10 → 22.2.20	No	`firewall_settings` adds a column
22.2.20 → 22.2.30	Yes	
22.2.30 → 22.2.40	No	`compatibility_check_reports`
22.5.0 → 22.5.1	Yes	
22.5.1 → 22.5.2	Yes	
22.5.2 → 22.5.10	No	PromoteCompatibilityCheckReports
22.5.10 → 22.5.20	No	AddLogFlowToRules
22.5.20 → 22.5.21	Yes	
22.5.21 → 22.5.22	Yes	
22.5.22 → 22.5.23	Yes	
22.5.23 → 22.5.30	Yes	
22.5.30 → 22.5.31	Yes	

Upgrade Path	Allowed (Yes/No)	Notes
22.5.31 → 22.5.32	Yes	
23.5.10 → 23.5.20	No	Postgres upgrade
23.5.22 → 23.5.31	No	Postgres upgrade

NOTE
This list will be updated for new released versions. Currently, there are no Rolling Upgrades for vesions 24.x.

Table 2. Rolling Upgrade Paths for Skipped Versions

Upgrade Path	Allowed (Yes/No)	Notes
21.2.0 → 21.2.2	No	
21.2.0 → 21.2.3	No	Due to 21.2.2
21.2.0 → 21.2.4	No	Due to 21.2.2
21.2.1 → 21.2.3	No	Due to 21.2.2
21.2.1 → 21.2.4	No	Due to 21.2.2
21.2.2 → 21.2.4	Yes	

Supercluster Listen Only Mode

The PCE *Listen Only* mode allows you stop the PCE from sending policy changes to your VENs. Enabling Listen Only mode for the PCE is typically used in these situations:

- During PCE maintenance windows, and when starting the PCE back up
- After restoring the PCE from a backup
- During maintenance windows for other parts of your network environment

In Listen Only mode, VENs still report updated workload information to the PCE, but the PCE does not modify the firewall rules on any workloads or send any updates from the PCE to the VENs. Also, the PCE does not mark workloads as Offline, and does not remove them from policy when Listen Only mode is enabled.

When this mode is enabled, you can still write policy, pair new workloads, provision policy changes, assign or change workload Labels, but changes will not be sent to the VENs until you disable Listen Only mode. You can disable Listen Only mode when you are ready to resume normal policy operations.

Enable PCE Listen Only Mode

1. On *all nodes* in the PCE cluster, stop the PCE software:

   ```
   sudo -u ilo-pce illumio-pce-ctl stop
   ```
2. On *all nodes* in the PCE cluster, set the node at runlevel 1:

   ```
   sudo -u ilo-pce illumio-pce-ctl start --runlevel 1
   ```
3. On *any data node*, enable Listen Only mode:

   ```
   sudo -u ilo-pce illumio-pce-ctl listen-only-mode enable
   ```
4. Set the PCE runlevel to 5:

   ```
   sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
   ```

Disable PCE Listen Only Mode

> **NOTE**
>
> The command to disable PCE Listen Only mode can be executed at either runlevel 1 or 5

> **IMPORTANT**
>
> Disable PCE Listen Only Mode is not needed for rolling upgrades, but for a simple upgrade it is still needed.

1. On *all nodes* in the PCE cluster, stop the PCE software:

   ```
   sudo -u ilo-pce illumio-pce-ctl stop
   ```
2. On *all nodes* in the PCE cluster, set the node to runlevel 1:

   ```
   sudo -u ilo-pce illumio-pce-ctl start --runlevel 1
   ```
3. On *any data node*, disable Listen Only mode:

   ```
   sudo -u ilo-pce illumio-pce-ctl listen-only-mode disable
   ```
4. Set the PCE runlevel to 5:

   ```
   sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
   ```

PCE Supercluster Deployment

This section describes setting up a PCE Supercluster, a single administrative domain encompassing two or more replicating PCEs. Before you start, understand the concepts explained in Overview of Supercluster Deployment [89].

PCE Supercluster VEN Management

This section describes how Virtual Enforcement Nodes (VENs) are managed by a PCE Supercluster, and what tasks you need to perform to successfully pair VENs and manage them in a Supercluster deployment.

Pair VENs in a Supercluster

A Supercluster allows you to control which PCE you want your workloads to pair with and be managed by, depending on your needs. You can pair one set of workloads with a PCE in Europe, for example, and you can pair another set of workloads with a PCE in the United States.

Sometimes, you might need to reassign some workloads (and their VENs) to be managed by a different PCE than the one they were initially paired with. In some instances of a PCE failure, you might want workloads to temporarily fail over to another healthy PCE. In both cases, another PCE manages a set of workloads.

VENs Paired to Disconnected PCE

A PCE that loses connectivity to its VENs maintains its “online” status from the VEN's perspective and retains the workloads in the policy. This condition can be corrected by taking the following general steps:

1. Determine the cause of the PCE failure and correct it.
   - Restore the failed PCE.
   - In the case of a failed leader, promote a member to the leader.
2. For a failed member, uninstall or unpair the VEN on the affected workloads.
3. (Optional) Delete records of the incorrectly marked “online” VENs using the PCE web console or the REST API. This step is an option because, after the VEN heart beating resumes, the proper state of the VEN will be re-established.

Pair Workloads with Leader or Member

This section discusses pairing your workloads with a Supercluster leader or member.

Pairing workloads with the leader or member follows nearly the same process as a standalone PCE cluster:

You create a pairing profile in the Supercluster leader's PCE web console.

- Member PCEs can be offline when this profile is created.
- Pairing profiles must always be created on the Supercluster leader.
- This pairing profile is propagated to all members.

With this pairing profile, you generate a pairing script.

- The pairing script can be configured to pair either with the Supercluster leader or with a member PCE:
  - A pairing script generated on the leader includes the FQDN of the leader.

  - A pairing script generated on a member includes the FQDN of that member.
- The pairing script includes the option `--management-server` with the domain name and port of the leader or the member.
- The paring script includes a pairing key (`--activation-code` option) that can be used to pair with any member.
- Members can create new pairing keys from pairing profiles replicated from the leader.
- Members can be isolated from the Supercluster but continue to pair with workloads.
- You run the pairing script on the workload to pair.

Pairing Script Examples for Supercluster

Pairing Script to Pair with Leader

The leader's FQDN is `supercluster-pce-LEADER.BigCo.com:8443`.

```
rm -fr /opt/illumio/scripts && umask 026 && mkdir -p /opt/illumio/scripts
&&
curl https://repo.illum.io/sPl1tOExo0FIEphoewIujIucrLaTOAS3/pair.sh -o /opt/
illumio/scripts/pair.sh &&
chmod +x /opt/illumio/scripts/pair.sh && /opt/illumio/scripts/pair.sh
--management-server supercluster-pce-LEADER.BigCo.com:8443
--activation-code xxyyzzyywwxx654321
```

Pairing Script to Pair with Member

The member's FQDN is `supercluster-pce-MEMBER.BigCo.com:8443`.

```
rm -fr /opt/illumio/scripts && umask 026 && mkdir -p /opt/illumio/scripts
&&
curl https://repo.illum.io/sPl1tOExo0FIEphoewIujIucrLaTOAS3/pair.sh -o /opt/
illumio/scripts/pair.sh &&
chmod +x /opt/illumio/scripts/pair.sh && /opt/illumio/scripts/pair.sh
--management-server supercluster-pce-MEMBER.BigCo.com:8443
--activation-code xxyyzzyywwxx654321
```

Run Pairing Script on Workloads

As with the standalone PCE configuration, you run the Supercluster-generated pairing script directly on the workload itself.

Linux environment variables and Windows command-line variables allow you to specify the management server to pair with.

For more information about pairing, see VEN Installation and Upgrade Guide.

Pair Workloads with GSLB PCE

When you rely on a Global Services Load Balancer (GSLB) to control which specific PCE a workload communicates with or to pair workloads to a generic name for the Supercluster, set the FQDN value of the `supercluster_fqdn` parameter in each PCE's `runtime_env.yml` file.

This value is used as the argument to the pairing script's `--management-server` option, which is the name of FQDN you define.

> **NOTE**
> Do not put the port number at the end of the `supercluster_fqdn` value. The system itself adds the port number to the pairing script.

Example

This example from the generated pairing script shows how the `supercluster_fqdn` parameter is set.

```
...
--management-server MyBigSuperclusterFQDN-from-supercluster-fqdn-
parameter.BigCo.com:8444
...
```

VEN Failover After PCE Failure

In rare cases, when you pair workloads with a Supercluster PCEs and that PCE fails immediately after you run the workload pairing script, the information about that workload's pairing is not replicated to the other PCEs in the Supercluster. When that workload's VEN tries to retrieve a policy from the PCE or sends a heartbeat, the VEN receives an HTTP 401 Unauthorized error and is eventually moved into the Lost Agent state.

To recover from this situation, you can perform one of these actions:

- Uninstall the VEN completely from the workload and repair it with a functioning PCE.
- Recover the affected PCE so that it is fully functional and online. After the VEN successfully heartbeats to the recovered PCE, it automatically comes out of the Lost Agent state.

  This second option to recover the PCE only works when the affected PCE had information about that VEN before the failure. When you recover the PCE from a backup taken before the VEN was paired, the VEN will have to be uninstalled and the workload repaired.

Pairing Container Clusters

You can pair workloads as part of a container cluster on supercluster member regions. Container clusters can be managed in a member region as well as all the resources attached to this container cluster: container workloads (pods), virtual services (services), and workloads (nodes).

Manage VENs in a Supercluster

This section describes how to manage VENs in a PCE Supercluster. Some of the management tasks are affected by Supercluster considerations, such as whether the task is performed on a leader or member PCE.

Unmanaged Workloads

When you need to create unmanaged workloads for assets that do not have a VEN installed, they must be created on the leader.

VEN Uptime and Heartbeat in Supercluster

Each workload managed by your Supercluster provides the latest “Uptime” of the workload. Uptime is defined as the amount of time that has passed in seconds since the workload reported its first heartbeat to the PCE, either after being paired or after a workload system restart. In PCE version 22.5.10 and later, you can view the Uptime and Heartbeat Last Received attributes for all workloads paired to PCEs in the Supercluster on the Workload Details page of any PCE in the Supercluster.

(Versions earlier than 22.5.10) Depending on which PCE you are logged into while viewing this information, the Uptime field might display the following:

```
Unavailable. Viewable on nameOfPCE
```

This message means that the PCE that you are currently logged into does not manage this workload. Instead, the Uptime and Last Heartbeat properties on the Workload details page indicate the name of the PCE that this workload was paired with.

Workload Support Reports in Supercluster

When you are logged into the leader of a Supercluster, you can generate and download Workload Support Reports for any workload in the Supercluster. This report includes workloads that have been paired with and are being managed by other members.

From a member PCE you can generate a Workload Support Report for all workloads connected to that PCE. However, you cannot generate a Workload Support Report from a member PCE for any workloads connected to a different PCE.

When the Workload Support Report is finished, you can download it from the leader PCE web console.

For information on running Workload Support Reports from the command line on the host, see the PCE Administration Guide.

Workloads on Leader When Member Fails

When one of your member PCEs goes down, any changes you make to workloads managed by the affected member (while logged into the leader) are immediately reflected in the leader PCE web console, even though the change has not been replicated to the member and applied on the workload.

For example, one member of your Supercluster fails. While you are logged into the leader, you make a change to a workload that was paired with that affected member, such as changing the workload's policy state. The Workload's details page on the leader will show the policy state change. However, the actual workload policy state will not be changed until the member is recovered.

VEN Failover

When a PCE in your Supercluster fails, its workloads continue to enforce the latest policy and buffer traffic data until the PCE is recovered. When you need to modify policy on the workload before the affected PCE can be recovered, you can fail over its workloads to a different PCE in the Supercluster. Workload failover is managed outside the Supercluster and requires either a GSLB [107] or an update to your DNS infrastructure.

To fail over a workload to a different PCE, configure your GSLB or DNS to resolve the FQDN of the workload's target PCE to the public IP addresses of another PCE in your Supercluster.

When you configure the `supercluster.fqdn` parameter in your `runtime_env.yml` file, the target PCE of all workloads is the Supercluster FQDN. The next time the workload resolves this FQDN, it will receive the updated IP addresses and begin heartbeating to and receiving policy from the new PCE.

To validate that the VEN reassignment was successful, check that the active PCE now corresponds to the FQDN the workload should have failed-over to.

VEN Failover Impact on Traffic Data

Be aware that some traffic data can be lost when VENs fail over to a different PCE:

- Traffic data used for Illumination and blocked traffic is lost and will be missing from Illumination.
- Traffic data that is exported to syslog or Fluentd is not lost, as long as the PCE has the capacity to handle all incoming flow summaries from all VENs.

VEN Failover and Certificates

A VEN must be able to validate the certificate of the PCE that is managing it and any other PCEs it will fail over to. When a VEN fails over and cannot validate the certificate of the new PCE, it cannot authenticate and enters the Lost Agent state. In this state, just as in a failure scenario, the VEN is disconnected from the PCE and it cannot receive policy updates. In this scenario, because the PCE that was managing the VEN is still running, it will mark the workload as offline in 1 hour, which in turn isolates it from all other workloads.

Reassign VENs in Supercluster Using REST API

When deploying a Supercluster, you might want to “move” workloads that have been paired to one PCE so that they are managed by a different PCE in the Supercluster. For example, you expand your single standalone PCE into a Supercluster and you want to reassign some of your existing VENs to be managed by the nearest PCE. In this case, you can reconfigure the VENs on paired workloads so that they use a different FQDN to communicate with the proper PCE.

Using the Illumio Agent API (the REST API refers to VENs as “agents”), you change the target PCE of the workload to the PCE you want to reassign the workload to. The PCE that is currently managing the workload sends the workload the FQDN of the new target PCE; the workload begins heartbeating to and receiving its policy updates from that PCE. The active PCE of the workload is now the same as the target PCE.

> **NOTE**
> Manually moving a VEN to a different PCE using the REST API is subject to the object limit `active_agents_per_pce`. For more information, see the "Object Limits and Supercluster" topic.

Active and Target PCE

Before reassigning VENs to another PCE, you need to understand these terms: active PCE and target PCE. These terms correspond to two properties that are added to a workload's VEN on pairing.

- `active_pce_fqdn`: The PCE that is currently managing a workload; namely, the PCE the workload has last heartbeat to.
- `target_pce_fqdn`: The PCE that is configured to manage this workload or the FQDN of the Supercluster (when you configured the `supercluster.fqdn` property in your `runtime_env.yml` file).

Workload Reassignment Workflow

This section assumes you are familiar with the basic concepts and usage of the Illumio Core REST API.

> **IMPORTANT**
> Before reassigning workloads to a new PCE, make sure that the active and target PCE are fully operational and at runlevel 5.

The workflow to reassign workloads to a different PCE consists of these general tasks:

1. **GET workloads:** To find the HREF of the agent on a workload, get a collection of workloads from the PCE. When you already know the HREF of a workload, you can get an individual instance of that workload, which returns the HREF of the agent that was used to pair that workload.
2. **Identify agent HREF:** The workloads' GET response include the `agent` property, which represents the VENs that are installed on the workloads as part of the pairing process. An agent is identified by its HREF.
3. **Identify active PCE FQDN of agent:** The workloads GET schema returns two properties that indicate the FQDN of the PCE that is actively managing the agent (`active_pce_fqdn`) and a second property that allows you to use a different “target” PCE FQDN (`target_pce_fqdn`) to manage the agent.
4. **Change target PCE FQDN of agent:** Update (PUT) the `target_pce_fqdn` property so that the VEN can be managed by a different PCE in your Supercluster.

Get Workloads

To get the HREF of an agent (VEN) on a workload, get a collection of workloads. You can GET up to 500 workloads at a time. When you know the HREF of an individual workload, you can get just the single workload.

To get a collection of workloads, you use this URI:

```
GET [api_version][org_href]/workloads
```

For example, using curl:

```
curl -u api_xxxxxxx64fcee809:'xxxxxx5048a6a85ce846a706e134ef1d4
bf2ac1f253b84c1bf8df6b83c70d95'
```

```
-H "Accept: application/json" -X GET
https://my.pce.supercluster:443/api/v1/orgs/7/workloads
```

Identify Agent HREF in Response

The JSON response from getting workloads provides information about the VEN ("agent") that was installed when the workload was paired with the PCE. In this response, you identify the workload's VEN (`agent`) by its HREF.

For example, the section that begins with the `agent` property shows the HREF of the VEN (`href":"/orgs/3/agents/40916`). In the response, the active PCE (`active_pce_fqdn`) and the target PCE (`target _pce_fqdn`) are the same. This does not change until you perform the reassignment.

```
"agent": {
            "config": {
                "log_traffic": false,
                "visibility_level": "flow_summary",
                "mode": "illuminated",
                "security_policy_update_mode": "adaptive"
            },
            "href": "/orgs/3/agents/40916",
            "status": {
                "uid": "e6c21a34-ebc2-4cf4-834e-3ec5df31d6ed",
                "last_heartbeat_on": "2016-02-11T12:22:32.91936Z",
                "instance_id":
                "perf_instance_1289213668111202403-1821@1455178338188",
                "managed_since": "2016-02-11T08:13:19.482909Z",
                "fw_config_current": false,
                "firewall_rule_count": null,
                "security_policy_refresh_at": null,
                "security_policy_applied_at": null,
                "security_policy_received_at": null,
                "uptime_seconds": 95819257,
                "status": "active",
                "agent_version": "2.10.0-20150715010305",
                "agent_health_errors": {
                    "errors": [],
                    "warnings": []
                },
                "agent_health": [],
                "security_policy_sync_state": "syncing"
            },
            "active_pce_fqdn": current-pce-fqdn.example.com,
            "target_pce_fqdn": current-pce-fqdn.example.com,
```

Change Target PCE

When you have the agent HREF, you can update the the target PCE with the PCE FQDN the VEN will use. In your JSON request body, pass the following data:

```
{
"target_pce_fqdn": "new-pce-fqdn.example.com"
}
```

The URI for this operation:

```
PUT [api_version][agent_href]/update
```

This curl example show how you can pass the `target_pce_fqdn` property containing the FQDN of the new PCE:

```
curl -u  api_xxxxxxx64fcee809:'xxxxxxx5048a6a85ce846
a706e134ef1d4bf2ac1f253b84c1bf8df6b83c70d95'
-H "Accept: application/json"
-H "Content-Type:application/json" -X PUT
-d '{"target_pce_fqdn":"target-pce.example.com"}'
https://my.pce.supercluster:443/api/v1/orgs/3/agents/40916/update
```

Validate VEN Reassignment

To validate that the VEN reassignment was successful, verify the active PCE matches the target PCE. Perform a GET request on the agent again. The target and active PCE FQDN should be the same. When the operation is successful, the response return an HTTP 204 code indicating success.

> **NOTE**
> Reassigning a VEN to a different PCE can take up to 10 minutes to complete.

For example:

```
"agent": {
            "config": {
                "log_traffic": false,
                "visibility_level": "flow_summary",
                "mode": "illuminated",
                "security_policy_update_mode": "adaptive"
            },
            "href": "/orgs/3/agents/40916",
            "status": {
                "uid": "e6c21a34-ebc2-4cf4-834e-3ec5df31d6ed",
                "last_heartbeat_on": "2016-02-11T12:22:32.91936Z",
                "instance_id":
                "perf_instance_1289213668111202403-1821@1455178338188",
                "managed_since": "2016-02-11T08:13:19.482909Z",
                "fw_config_current": false,
                "firewall_rule_count": null,
                "security_policy_refresh_at": null,
                "security_policy_applied_at": null,
                "security_policy_received_at": null,
                "uptime_seconds": 95819257,
                "status": "active",
                "agent_version": "2.10.0-20150715010305",
                "agent_health_errors": {
                    "errors": [],
                    "warnings": []
                },
                "agent_health": [],
```

```
        "security_policy_sync_state": "syncing"
    },
    "active_pce_fqdn": new-pce-fqdn.example.com,
    "target_pce_fqdn": new-pce-fqdn.example.com
```

PCE Supercluster NEN Management

This section describes how to deploy the Network Enforcement Nodes (NENs) in a Supercluster as well as how to reassign a NEN from one PCE to another through either a command line or the Illumio Core REST API.

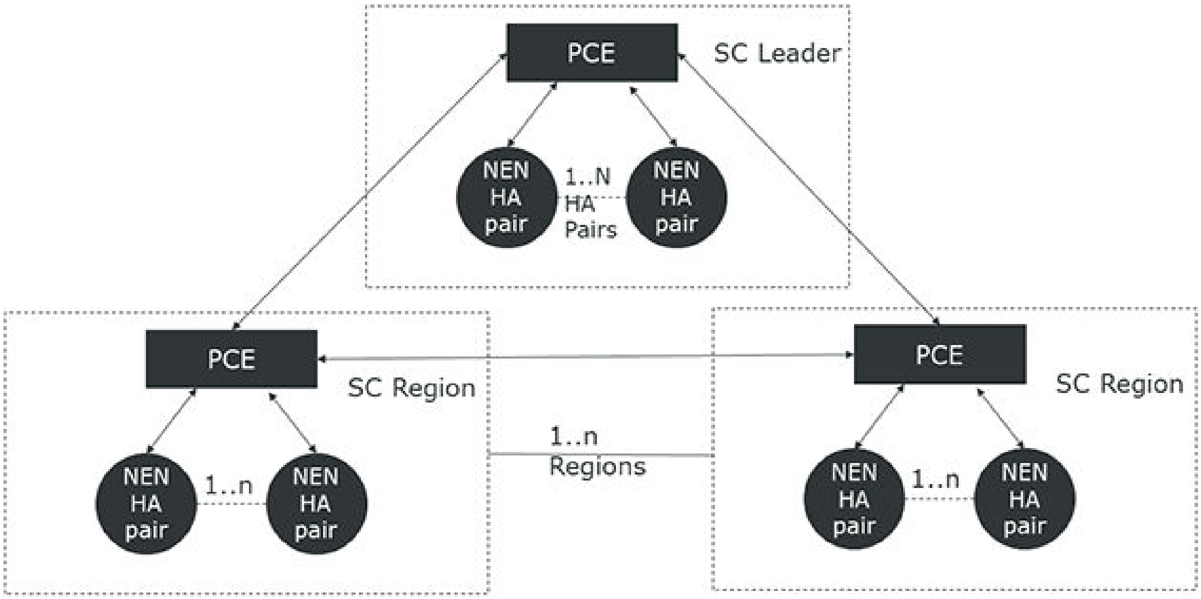
> **IMPORTANT**
> This feature applies only to customers running Illumio Core 22.2.10. If you are running versions 22.2.x or earlier, this feature is unavailable.

Deploy NENs in a Supercluster

> **IMPORTANT**
> This feature applies only to customers running Illumio Core 22.2.10. If you are running versions 22.2.x or earlier, this feature is unavailable.

Follow these guidelines when you deploy NENs in a Supercluster:

- **Where:** You can pair NENs to Supercluster Members or to the Supercluster Leader. Deploy NENs as close as possible to the Server Load Balancers (SLBs) they're managing.
- **How many:** A suggested maximum of three NENs per region, paired to the local PCE.
- **High Availability (HA):** Depending on your scale requirements and the location of your network devices (such as SLBs), you can connect multiple NEN HA pairs to any cluster in a PCE Supercluster deployment (not just the PCE Supercluster leader). This is necessary in environments with many SLBs and virtual servers that are geographically distributed. For information about support for HA NEN deployments in general, see NEN HA Support.



Reassign NENs from One PCE to Another in a Supercluster

When deploying a Supercluster, you might want to “move” Network Enforcement Nodes (NENs) that have been paired to one PCE so that they are managed by a different PCE in the Supercluster. For example, you expand your single standalone PCE into a Supercluster and you want to reassign some of your existing NENs to be managed by the nearest PCE. In this case, you can reconfigure the NENs so that they use a different PCE FQDN to communicate with the proper PCE.

Reassign a NEN in a Supercluster Using a Command Line

You can use a command line to move a NEN from one PCE to another PCE in the same supercluster. When a NEN is moved in this way, associated Server Load Balancers maintain policy for managed virtual servers. After the PCE database is restored, the moved NEN remains connected to the new PCE. The command for moving a NEN is:

```
illumio-nen-ctl pce-host-update <pce-host-addr>:<port>
```

Reassign a NEN in a Supercluster Using the IllumioCore REST API

> **IMPORTANT**
> This feature applies only to customers running Illumio Core 22.2.10. If you are running versions 22.2.x or earlier, this feature is unavailable.

Using the Illumio Network Enforcement Node API, you change the target PCE of the NEN to the PCE you want to reassign the NEN to. The PCE that is currently managing the NEN sends the NEN the FQDN of the new target PCE; the NEN begins heartbeating to – and receiving its policy updates from – that PCE. The active PCE of the NEN is now the same as the target PCE.

> **NOTE**
> Manually moving a NEN to a different PCE using the REST API is subject to the object limit `active_agents_per_pce`. For more information, see the "Object Limits and Supercluster" topic.

Active and Target PCE

Before reassigning NENs to another PCE, you need to understand these terms: active PCE and target PCE. These terms correspond to two properties that are added to a NEN on pairing.

- `active_pce_fqdn`: The PCE that is currently managing a NEN; namely, the PCE the NEN has last heartbeat to.
- `target_pce_fqdn`: The PCE that is configured to manage this NEN or the FQDN of the Supercluster (when you configured the `supercluster.fqdn` property in your `runtime_env.yml` file).

NEN Reassignment Workflow

This section assumes you are familiar with the basic concepts and usage of the Illumio Core REST API.

> **IMPORTANT**
> Before reassigning a NEN to a new PCE, make sure that the active and target PCE are fully operational and at runlevel 5. Also, ensure that both nodes of a NEN HA pair are up and running runlevel 5, and running NEN-2.4.0 or later.

The workflow to reassign a NEN to a different PCE includes these general tasks:

1. **GET network_enforcement_nodes:** To find the HREF of the NEN, get a collection of NENs from the PCE.
2. **Identify NEN HREF:** The NENs' GET response includes the NEN HREF and the associated hostname of the NEN.
3. **Identify active PCE FQDN of NEN:** The NENs GET schema returns two properties that indicate the FQDN of the PCE that is actively managing the NEN (`active_pce_fqdn`) and a second property that allows you to use a different "target" PCE FQDN (`target_pce_fqdn`) to manage the NEN.
4. **Change target PCE FQDN of NEN:** Update (PUT) the `target_pce_fqdn` property so that the NEN can be managed by a different PCE in your Supercluster.

Get NENs

To get the HREF of a NEN, get a collection of NENs. You can GET up to 500 NENs at a time. When you know the HREF of an individual NEN, you can skip this step.

To get a collection of NENs, you use this URI:

```
GET [api_version][org_href]/network_enforcement_nodes
```

For example, using curl:

```
curl -u api_xxxxxxx64fcee809:'xxxxxx5048a6a85ce846
a706e134ef1d4bf2ac1f253b84c1bf8df6b83c70d95'
-H "Accept: application/json" -X GET
https://my.pce.supercluster:443/api/v1/orgs/7/network_enforcement_nodes
```

Identify Agent HREF in Response

The JSON response from getting NENs provides information about the NEN when the NEN was paired with the PCE. In this response, you identify the NEN by its HREF.

For example, the section that shows the NEN, its active PCE (`active_pce_fqdn`), and the target PCE (`target _pce_fqdn`) are the same. This does not change until you perform the reassignment.


```
{
    "href": "/orgs/3/network_enforcement_nodes/05509c61-19b3-
    456a-8336-36a7cf07976b",
    "hostname": "nen1.example.com",
    "public_ip": "nen1.example.com",
    "name": "Illumio Network Enforcement Node - nen1.example.com",
    "software_version": "2.4.0",
    "last_status_at": "2022-06-19T23:43:52.204Z",
    "uptime_seconds": 519237,
    "network_devices": [
        {
            "href": "/orgs/3/network_devices/ffefd599-f169-
            4f54-9377-69fb04b0bc84"
        }
    ],
    "supported_devices": [
        {
            "device_type": "slb",
            "manufacturers": [
                {
                    "manufacturer": "AVI",
                    "models": [
                        {
                            "model": "Vantage"
                        }
                    ]
                },
                {
                    "manufacturer": "F5",
                    "models": [
                        {
                            "model": "Big-IP AFM"
                        },
                        {
                            "model": "Big-IP LTM"
                        }
                    ]
```

```
                }
            ]
        },
        {
            "device_type": "switch",
            "manufacturers": [
                {
                    "manufacturer": "Arista",
                    "models": [
                        {
                            "model": "7000"
                        }
                    ]
                },
                {
                    "manufacturer": "Cisco",
                    "models": [
                        {
                            "model": "9000"
                        }
                    ]
                }
            ]
        }
    ],
    "target_pce_fqdn": "pce1.exmaple.com",
    "active_pce_fqdn": "pce1.example.com",
    "conditions": []
},
{
    "href": "/orgs/3/network_enforcement_nodes/f67d35d5-ea71-
    42da-b40d-8dcc3b1420c2",
    "hostname": "nen2.example.com",
    "public_ip": "nen2.example.com",
    "name": "Illumio Network Enforcement Node - nen2.example.com",
    "software_version": "2.4.0",
    "last_status_at": null,
    "uptime_seconds": null,
    "network_devices": [],
    "supported_devices": [
        {
            "device_type": "slb",
            "manufacturers": [
                {
                    "manufacturer": "AVI",
                    "models": [
                        {
                            "model": "Vantage"
                        }
                    ]

                },
                {
                    "manufacturer": "F5",
                    "models": [
```

```
                        {
                            "model": "Big-IP AFM"
                        },
                        {
                            "model": "Big-IP LTM"
                        }
                    ]
                }
            ]
        },
        {
            "device_type": "switch",
            "manufacturers": [
                {
                    "manufacturer": "Arista",
                    "models": [
                        {
                            "model": "7000"
                        }
                    ]
                },
                {
                    "manufacturer": "Cisco",
                    "models": [
                        {
                            "model": "9000"
                        }
                    ]
                }
            ]
        }
    ],
    "target_pce_fqdn": "pce1.example.com",
    "active_pce_fqdn": "pce1.example.com",
    "conditions": []
}
```

Change Target PCE

When you have the NENH REF, you can update the 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:

```
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:'xxxxxxx5048a6a85ce846a706e134
ef1d4bf2ac1f253b84c1bf8df6b83c70d95'
```

```
-H "Accept: application/json"
-H "Content-Type:application/json" -X PUT
-d '{"target_pce_fqdn":"new-pce.example.com"}'
https://my.pce.supercluster:443/api/v1/orgs/3/
network_enforcement_nodes/f67d35d5-ea71-42da-b40d-8dcc3b1420c2/update
```

Validate NEN Reassignment

To validate that the NEN reassignment was successful, verify that the active PCE matches the target PCE. Perform a GET request on the NEN using the associated NEN HREF. The target and active PCE FQDN should be the same. When the operation is successful, the response returns an HTTP 204 code indicating success.

> **NOTE**
> Reassigning a NEN to a different PCE can take up to 5 minutes to complete.

For example:

```
{
    "href": "/orgs/3/network_enforcement_nodes/f67d35d5-ea71-
    42da-b40d-8dcc3b1420c2",
    "hostname": "nen2.example.com",
    "public_ip": "nen2.example.com",
    "name": "Illumio Network Enforcement Node - nen2.example.com",
    "software_version": "2.4.0",
    "last_status_at": null,
    "uptime_seconds": null,
    "network_devices": [],
    "supported_devices": [
        {
            "device_type": "slb",
            "manufacturers": [
                {
                    "manufacturer": "AVI",
                    "models": [
                        {
                            "model": "Vantage"
                        }
                    ]
                },
                {
                    "manufacturer": "F5",
                    "models": [
                        {
                            "model": "Big-IP AFM"
                        },
                        {
                            "model": "Big-IP LTM"
                        }
                    ]
                }
```

```
                ]
            },
            {
                "device_type": "switch",
                "manufacturers": [
                    {
                        "manufacturer": "Arista",
                        "models": [
                            {
                                "model": "7000"
                            }
                        ]
                    },
                    {
                        "manufacturer": "Cisco",
                        "models": [
                            {
                                "model": "9000"
                            }
                        ]
                    }
                ]
            }
        ],
        "target_pce_fqdn": "new-pce-fqdn.example.com",
        "active_pce_fqdn": "new-pce-fqdn.example.com",
        "conditions": []
}
```

PCE Supercluster Administration

This section explains how to perform common administration tasks for a PCE Supercluster.

Monitor Supercluster Health

You can use these two general methods for monitoring the health of your PCE Supercluster:

- REST API calls to determine the Supercluster leader and a PCE member's health
- The PCE web console to view the health of the entire Supercluster from the leader or for the member you are logged into.

This section discusses health monitoring specifically for a PCE Supercluster. Additionally, follow the PCE health monitoring guidelines in PCE Administration Guide.

REST API for Supercluster Health

You can monitor Supercluster health using the following REST API mechanisms.

REST API `/health`

Using the PCE Health API, you can get current health information about all PCEs in your Supercluster, including the leader and members.

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

REST API to Determine Supercluster Leader

Use this Public Stable REST API request to determine whether the PCE in a Supercluster is a leader or member.

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

Your GSLB can issue this request to monitor the health of the leader.

HTTP Response Code from `/supercluster/leader`

Response	Meaning
202	The PCE is the leader.
404	The PCE is a member.

REST API `/node_available`

After your GSLB determines the Supercluster leader, issue the following REST API request to monitor the leader's availability:

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

HTTP response code from `/node_available`

The Health REST API can take up to 30 seconds to reflect the actual status of the node.

Response	Meaning
202	The node is healthy and is connected to the rest of the cluster.
404 or no response	The node is unhealthy and cannot accept requests. Such a node should be removed from the load balancing pool.

PCE Web Console for Supercluster Health

The Health page in the PCE web console in a Supercluster provides health information about your on-premises PCE, whether you deployed an SNC, 2x2, 4x2, or Supercluster.

- **General PCE Health:** This shows general health information for each PCE in your Supercluster, such as health status, node status and uptime, and system health information for each node (CPU usage, memory, and disk usage). When you deploy a PCE Supercluster, the Health page lists all PCEs in the Supercluster with individual health information for each PCE.
- **Supercluster Leader Health:** Displays the health status of the leader PCE in the Supercluster. You can view the health of each PCE in the Supercluster.
- **Supercluster Member Health:** Shows health information about the member you are logged into, including a timer that indicates the amount of time since Illumination data was synced across the Supercluster. The Health page shows the database replication lag for each PCE

relative to all other PCEs in the Supercluster, indicating how long it took for data to be replicated from one PCE to another.

The PCE health page indicates the current state of database replication across the Supercluster and how recently each member PCE's Illumination data has been synced with the leader.

- **Supercluster Replication (Lag):** Indicates how long it took for one PCE to receive replicated data from another PCE in the Supercluster. For example, a user created a new IP list in the leader and saved it. The change took 4 seconds to replicate to Member1 and Member1's Health page showed that its replication lag is 4 seconds behind the leader. The PCE web console shows replication lag for each PCE in the Supercluster.
- **Supercluster Illumination Sync (Members only):** This shows the last time since a member PCE replicated its Illumination traffic data with the Supercluster leader. This information only appears for members periodically sending traffic data to the leader. This information provides a full picture of Illumination traffic for your entire Supercluster. You can initiate a sync of Illumination data on demand by clicking the link in the lower right of the Illumination map.

Supercluster PCE Health Icon

When the PCE Health button has a badge with a number, one or more of the PCEs in your Supercluster have a health status that is *not* “Normal.” The badge color indicates the type of warning.

For example, a yellow warning badge with the number 1 indicates that one of the PCEs in the Supercluster has a health warning status.

When the badge is red and shows the number 1, one of the Supercluster PCEs has failed or is down.

Supercluster Web Console Health Page

The Supercluster Health page on the leader displays a high-level view of each PCE's health. You can click a PCE to view individual health information. The information on this page is refreshed every 60 seconds.

Individual PCE Health Status

The following table lists the possible health statuses for a PCE: Normal, Warning, or Critical.

Status	Color	Definition
Normal (healthy)	**Green**	A PCE is considered to be in a **normal** state when: • 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	**Yellow**	A PCE is considered to be in a **warning** state when: • One or more nodes are unreachable. • One or more optional services are missing, or one or more required services are 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	**Red**	A PCE is considered to be in a **critical** state when one or more required services are missing. In this scenario, it might not be possible to authenticate to the PCE or get a REST API response depending on which services are missing from the PCE.

PCE Health on Workload Details

When your workloads have been paired with a Supercluster leader or member, you can view PCE health on the Summary tab of the Workload details page. This page includes the PCE section, which lists the hostname and health of the PCE with which this workload is paired.

PCE Health on Illumination Command Panel

When you select a workload in the Illumination map in a Supercluster, the command panel that displays workload details includes the health of the PCE with which the workload is paired. For example, you can see the health status of the PCE the workload is paired with in the PCE Health field.

Command to Show All Supercluster Members

On *any core node* or the *data0 node* in a cluster, run the following command to display the leader and all member PCEs of the Supercluster.

```
sudo -u ilo-pce illumio-pce-ctl supercluster-members
```

Back Up Supercluster

You need to perform regular backups on all PCEs in the Supercluster.

Different data is backed up depending on whether you run the backup from the Supercluster leader or a member:

- **Leader backup:** policy database, which has information for all the regions; traffic redis instance; login information; reporting database; supercluster configuration data.

- **Member backup:** login information, traffic redis instance, reporting database, supercluster configuration data.
- **All PCE nodes' runtime environment file:** The `runtime_env.yml` is not included in the backup and must be backed up separately for each node. The default location of the PCE Runtime Environment File is `/etc/illumio-pce/runtime_env.yml`. When the location is different on your system, you can find it by checking the value of the `ILLUMIO_RUNTIME_ENV` environment variable.
- **Traffic database:** The traffic database dump can be very large, depending on the traffic datastore size. Therefore, the Supercluster database dump on leader and member PCEs does not include the traffic data. A separate procedure is provided. See the "Back Up the Traffic Database" section of the "PCE Database Backup" topic in the PCE Administration Guide.

When to Back Up

Follow your own organization's policies and procedures for backup, including frequency (such as, hourly, daily, or weekly) and retention of backups offsite or on a system other than any of the Supercluster nodes.

Illumio recommends taking backups in the following situations:

- Before and after a PCE version upgrade
- After pairing a large number of VENs
- After updating a large number of workloads (such as, changing workload policy state or applying labels)
- After provisioning major policy changes
- After making major changes in your environment that affect workload information (such as, an IP address changes)
- Before and after adding new PCEs to your Supercluster
- After you assign a new leader
- On-demand backups before the procedures documented in this guide, such as migration and upgrade

Back Up Each PCE's Data

For the leader and every member PCE in your Supercluster, perform these steps:

1. Create a directory for the backup file that is not one of the PCE software's installation directories.
2. Grant both the `ilo-pce` user and the user who will run the backup command Read and Write permissions to this directory.
3. Run the following command:

   ```
   sudo -u ilo-pce install_root/illumio-pce-db-management
   supercluster-data-dump --file desired_location_of_backup_file
   ```
4. Repeat these steps for every PCE in the Supercluster.

Copy Leader Backup to Members

Copy the backup file that you just made on the leader PCE to the data0 node of each member PCE. In this way, if it becomes necessary to restore the entire Supercluster, the leader's data is readily available to every member so the data can be restored more quickly. (The leader PCE backup is not needed if only a single PCE is to be restored; in that case, the PCE's own member backup is sufficient.) You can copy the leader backup file to any file system location of the member data0 node, except for the PCE software's installation directories. Be sure that all member PCEs have the same version of the leader PCE backup.

Using different versions of the leader PCE backup can cause data replication to fail after the Supercluster restore is complete.

Back Up Leader and Member Runtime Environment Files

Store a copy of each node's `runtime_env.yml` file on a system that is not part of the Supercluster. By default, the PCE Runtime Environment File is stored in `/etc/illumio-pce/runtime_env.yml`. When the location is different on your system, locate the file by checking the `ILLUMIO_RUNTIME_ENV` environment variable.

Assign New Leader

A Supercluster can only have one leader at a time. The following section describes how to choose a new leader or temporarily assign a new leader when the leader has failed and you need to change the Supercluster before it can be recovered.

Assign Leader When a Leader Is Connected

You can choose a new Supercluster leader when the existing leader is still running and connected to the rest of the Supercluster. When you choose a new leader, the former leader becomes a member in the Supercluster.

1. Decide which PCE member you want to be the new leader.
2. On *any node of each PCE* in the Supercluster, set the runlevel to 2:

   ```
   $ sudo -u ilo-pce illumio-pce-ctl set-runlevel 2
   ```

   Make sure you wait until the software is running before you proceed.
3. Check the progress to see when the status is `RUNNING` on all nodes:

   ```
   $ sudo -u ilo-pce illumio-pce-ctl cluster-status -w
   ```
4. On *any node on the new leader*, promote it to be the leader:

   ```
   $ sudo -u ilo-pce illumio-pce-ctl supercluster-assign-leader
   ```
5. On *any node* on the new leader <u>and</u> the former leader, bring both PCEs to runlevel 5:

   ```
   $ sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
   ```
6. Check the progress to see when the status is `RUNNING` on all nodes:

   ```
   $ sudo -u ilo-pce illumio-pce-ctl cluster-status -w
   ```

Assign New Leader When Leader Has Failed

When your Supercluster leader has failed, you must drop the failed leader from the Supercluster before you can assign a new leader.

WARNING

When the new leader is promoted, you must isolate the former leader from the network and not allow it to be brought back online. When the former leader is not isolated, it will incorrectly re-join the Supercluster as the leader. Having two leaders in a Supercluster is not supported and can lead to data corruption.

When you are ready to restore the failed PCE and rejoin it to the Supercluster, follow the procedures in Restore a PCE or Entire Supercluster [152], which will bring the PCE back as a member. After it has been brought back as a member, you can assign it to be the leader again.

To drop the failed leader and assign a new leader:

1. On a *core node* of *each surviving PCE* in the Supercluster, set the runlevel to 2:

   ```
   $ sudo -u ilo-pce illumio-pce-ctl set-runlevel 2
   ```
2. On the PCE you *assigned as the new leader*, drop the failed leader from the Supercluster:

   ```
   $ sudo -u ilo-pce illumio-pce-ctl supercluster-drop failed_PCE_fqdn
   ```
3. Check the progress to see when the status is `RUNNING`:

   ```
   $ sudo -u ilo-pce illumio-pce-ctl cluster-status -w
   ```
4. On the *newly designated leader PCE*, assign is as the new Supercluster leader:

   ```
   $ sudo -u ilo-pce illumio-pce-ctl supercluster-assign-leader
   ```
5. On the *new leader PCE*and*all member PCEs*, set them to runlevel 5:

   ```
   $ sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
   ```

   Make sure you wait until the software is running before you proceed.
6. Check the progress to see when the status is `RUNNING` on all nodes:

   ```
   $ sudo -u ilo-pce illumio-pce-ctl cluster-status -w
   ```

Add a New Member to an Existing Supercluster

This topic explains how to add one or more new members to an existing Supercluster.

Before You Begin: Runtime Configuration

Before you add a new member to your PCE Supercluster, be aware of the following `runtime_env.yml` configurations:

- The value of the parameter `service_discovery_encryption_key` in the `runtime_env.yml` file must be exactly the same on all nodes on all PCEs in your Supercluster.
- You do not need to configure the public IP addresses of other PCEs under the `cluster_public_ips` parameter. Supercluster PCEs automatically exchange their configured public IP addresses with each other, which get programmed by the VEN to allow workloads to migrate between PCEs.

Optional

Depending on your deployment environment, you might need to make the following changes to the `runtime_env.yml` file on each PCE in the Supercluster.

When the nodes of each PCE use multiple IP addresses or they use IP addresses other than the one advertised on the node for communication with other PCEs, such as having a NAT between the PCEs in your Supercluster, configure this optional parameter:

- `supercluster.node_public_ip`: The public IP address of this node is advertised to other PCEs in your Supercluster deployment. This IP address must be reachable from all other Supercluster PCEs that you want to join. This parameter must be set on *all* nodes in *each* PCE. When your PCE is deployed in a public cloud, such as AWS, this must be a public IP address.

Install Members

Install each new member of your Supercluster by following the exact same procedures you use when installing a standalone PCE, except *do not* create a domain during deployment.

For information about installing a PCE, see the PCE Installation and Upgrade Guide.

Join Each Member to the Supercluster

> **IMPORTANT**
> If you are adding multiple new PCEs, you must join only one member at a time, and complete all steps before joining the next member. Ensure that each member is at runlevel 2 before joining.

1. If necessary, on *any node*, bring all nodes to runlevel 2:

   ```
   sudo -u ilo-pce illumio-pce-ctl set-runlevel 2
   ```
2. On *any node*, run the following command while you wait for all nodes to reach runlevel 2:

   ```
   sudo -u ilo-pce illumio-pce-ctl status --wait
   ```
3. On *any core node* or the *data0 node of the member cluster*, join the member to the Supercluster (identified by the leader's FQDN):

   ```
   sudo -u ilo-pce illumio-pce-ctl supercluster-join leader_pce_fqdn
   ```

   While this command runs, the PCE temporarily sets the runlevel to 1. If the command is interrupted, you might unexpectedly see runlevel 1.

   > **IMPORTANT**
   > Running this command can take an hour or more depending on the number of PCEs in your Supercluster and size of the PCE database. If this command fails due to network latency, do not proceed until you can run the command again and it executes successfully.

4. Repeat step 3 for all members you want to join to the Supercluster.
5. On *all PCEs*, restart the PCEs in the Supercluster:

   ```
   sudo -u ilo-pce illumio-pce-ctl cluster-restart
   ```
6. On *all PCEs*, bring the PCEs to runlevel 5:

   ```
   sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
   ```

Restore a PCE or Entire Supercluster

This section describes how to restore a single failed PCE, either leader or member, and rejoin it to a Supercluster. It also describes how to restore the entire Supercluster.

- Restore a Single PCE in a Supercluster [152]. Follow this procedure when a single leader or member PCE has failed and needs to be restored.
- Restore an Entire Supercluster [155]. Follow this procedure when more than one PCE has failed.

Restore a Single PCE in a Supercluster

This section explains how to restore a leader or member PCE in a Supercluster. Isolate that PCE from the Supercluster, restore it, and rejoin it to the Supercluster.

Summary

The following steps are an overview of how to restore a single PCE in a Supercluster. For detailed instructions, read the rest of this section.

1. Preparation:
   a. Have the backups and the copy of the affected PCE's `runtime_env.yml` configuration file ready to use.
   b. Know the IP address, ports, and DNS name of the affected PCE. You must use the same values when you rejoin the PCE to the Supercluster.
2. Isolate the affected PCE from the Supercluster.
3. Install the PCE on new hardware or reuse the installation and `runtime_env.yml` file of the affected PCE.
4. Restore the Supercluster data from backup.
5. Join the repaired PCE to the Supercluster.

Prepare to Restore a Single PCE

Have the following items available:

- Back up the failed PCE; see Back Up Supercluster [147] for information.
- Back up the failed PCE's `runtime_env.yml` file.
- Make a list of the IP address, ports, and FQDN of the failed PCE. You will use these values to reconfigure the repaired PCE.

Isolate the Affected PCE

Before restoring a single PCE, isolate that PCE from the Supercluster.

1. On *all nodes*, shut down the affected PCE:

   ```
   sudo -u ilo-pce illumio-pce-ctl stop
   ```
2. On *a core node of each surviving PCE* in the Supercluster, set the PCE to runlevel to 2:

   ```
   sudo -u ilo-pce illumio-pce-ctl set-runlevel 2
   ```

   > **NOTE**
   > You *must* set all PCEs to runlevel 2 before proceeding to the next step.
3. On *any core node of a surviving PCE*, drop the failed PCE from the Supercluster:

   ```
   sudo -u ilo-pce illumio-pce-ctl supercluster-drop fqdn_of_failed_pce
   ```

Install New PCE or Reuse Affected PCE

Decide whether to completely reinstall the PCE on new hardware or reuse the PCE installation that is already on the affected system.

> **NOTE**
> In both cases, you must reestablish the FQDN of the affected PCE so that VENs can continue to communicate with the Supercluster. When you have any VENs in enforcement, or you rely on DNS-based load balancing, the new IP addresses of the PCE nodes can be different, as long as the new IP addresses were already in the appropriate settings in the `runtime_env.yml` file on all PCE core nodes. See Pre-configure New IP addresses [115]for information.

- To reinstall the PCE on new hardware, see Deploy a PCE Supercluster [110].
- To reuse the affected PCE installation, complete the following steps.

When you decide to reuse the PCE's pre-failure installation, refresh the installation as a standalone PCE:

1. Power on the PCE nodes.
2. On *all nodes of the affected PCE*, run the following command to delete pre-failure directories:

   ```
   sudo -u ilo-pce illumio-pce-ctl reset
   ```

   > **NOTE**
   > You must run this command on *all nodes* before proceeding to the next step.
3. Copy your backed-up copy of the failed PCE's `runtime_env.yml` file to its location on the newly repaired PCE. See Back Up Leader and Member Runtime Environment Files [149] for information.

The default location of the PCE Runtime Environment File is `/etc/illumio-pce/runtime_env.yml`. When the location is different on your system, locate the file by checking the value of the `ILLUMIO_RUNTIME_ENV` environment variable.

4. On *all nodes*, bring the nodes to runlevel 1:

```
sudo -u ilo-pce illumio-pce-ctl start --runlevel 1
```

5. On *any node*, verify the nodes are at runlevel 1:

```
sudo -u ilo-pce illumio-pce-ctl cluster-status -w
```

6. On *any node*, run the following command:

```
sudo -u ilo-pce illumio-pce-db-management setup
```

Restore Affected PCE's Supercluster Data

1. On *any node of the affected PCE*, verify the runlevel is still 1:

```
sudo -u ilo-pce illumio-pce-ctl cluster-status -w
```

2. On the *data0 node*, run the following command:

```
sudo -u ilo-pce illumio-pce-db-management supercluster-data-restore --
local-pce-file path_to_backup_file --restore-type single_pce
```

The restore operation can take some time to complete. Wait until it finishes before proceeding to the next step.

3. On the *data1 node*, run the following command:

```
sudo -u ilo-pce illumio-pce-db-management supercluster-data-restore
--skip-db-restore --local-pce-file path_to_backup_file --restore-type
single_pce
```

The `--skip-db-restore` option prevents the command from unnecessarily repeating work that has already been done by previous commands.

4. On any core node, it will be necessary to reinstall any previously installed VEN bundles per the compatibility matrix because they will not be restored with the rest of the supercluster data. See Ways to Install the VEN.

Rejoin PCE to Supercluster

1. On *all Supercluster PCEs*, set the runlevel to 2:

```
sudo -u ilo-pce illumio-pce-ctl set-runlevel 2
```

Setting the runlevel might take some time to complete.

2. Check the progress to see when the status is `RUNNING` on all nodes:

```
sudo -u ilo-pce illumio-pce-ctl cluster-status -w
```

3. Rejoin the PCE to the Supercluster. This command can take some time, depending on the number of PCEs in the Supercluster and the size of the PCE databases.

Choose one of the following options, depending on whether you are working on a leader or member.

Rejoining the Leader PCE

On *any core node*, run the following command:

```
sudo -u ilo-pce illumio-pce-ctl supercluster-restore
fqdn_of_failed_cluster --restore-type single_pce
```

While this command is running, the PCE temporarily sets the runlevel to 1. When the command is interrupted, you might see runlevel 1 unexpectedly.

Rejoining a Member PCE

On *any core node*, run the following command:

```
sudo -u ilo-pce illumio-pce-ctl supercluster-restore
fqdn_of_failed_cluster fqdn_of_supercluster_leader --restore-type
single_pce
```

While this command is running, the PCE temporarily sets the runlevel to 1. If the command is interrupted, you might see runlevel 1 unexpectedly.

4. On *every PCE*, set the runlevel to 5:

```
sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
```

5. Verify the run level:

```
sudo -u ilo-pce illumio-pce-ctl cluster-status -w
```

6. Verify that the restored PCE has rejoined the Supercluster and is fully operational:
   a. Log in to the leader PCE web console.
   b. Go to the PCE Health page and verify that the PCE health status is Normal.
7. Check the status of the paired VENs on each PCE. From the PCE web console, choose Workloads and VENs > VENs. After all VENs change status from Active (Syncing) to Active, run the following command *on one PCE at a time*:

```
sudo -u ilo-pce illumio-pce-ctl listen-only-mode disable
```

Restore an Entire Supercluster

Restoring an entire Supercluster follows this high-level process:

1. Preparation:
   a. Have the backups of all PCEs ready to use. For each member PCE, have that member's backup and, on the data0 node, a copy of the backup file from the leader. The leader only needs its own backup.
   b. Have copies of every PCE's `runtime_env.yml` configuration file ready to use.
   c. Know the IP address, ports, and DNS name of all PCEs in the Supercluster. You must use the same values when you rejoin the PCEs to the Supercluster.
2. Shut down the entire Supercluster.
3. Restore the PCEs. Repeat the following steps for all PCEs in the Supercluster, either one at a time or in parallel:
   a. Reinstall the PCE on new hardware or reuse the installations and `runtime_env.yml` files.
   b. Restore the Supercluster data from backup.
4. Join the repaired PCEs to the Supercluster one at a time.

Prepare to Restore Entire Supercluster

Have the following items ready:

- Backup of each PCE, and the leader's backup copied to each member. See Back Up Supercluster [147]for information.
- Copy of each PCE's `runtime_env.yml` file.
- List of the new IP address, ports, and DNS name for all Supercluster members.

Shut Down Entire Supercluster

On *all nodes of every PCE in the Supercluster*, run the following command:

```
sudo -u ilo-pce illumio-pce-ctl stop
```

Install New PCEs or Reuse PCEs

Decide whether you want to completely reinstall the PCEs on new hardware or to reuse the PCE installations.

> **NOTE**
> In both cases, you must reestablish the FQDN of the affected PCE so that VENs can continue to communicate with the Supercluster. When you have any VENs in enforcement, or you rely on DNS-based load balancing, the new IP addresses of the PCE nodes can be different, as long as the new IP addresses were already in the appropriate settings in the `runtime_env.yml` file on all PCE core nodes. See Pre-configure New IP addresses [115]for information.

- To reinstall the PCEs on new hardware, see Deploy a PCE Supercluster [96] for information.
- To reuse the PCE installations, complete the following steps.

When you decide to reuse the PCE's pre-failure installation, refresh the installation as a standalone PCE:

1. On *all nodes of the PCE*, reset the nodes:

   ```
   sudo -u ilo-pce illumio-pce-ctl reset
   ```

   > **NOTE**
   > You must run this command on *all nodes* before proceeding to the next step.
2. Copy your backed-up copy of the failed PCE's `runtime_env.yml` file to its location on the newly repaired PCE. See Back Up Leader and Member Runtime Environment Files [149]. The default location of the PCE Runtime Environment File is `/etc/illumio-pce/runtime_env.yml`. When the location is different on your system, locate the file by checking the value of the `ILLUMIO_RUNTIME_ENV` environment variable.
3. Bring *all nodes* to runlevel 1:

   ```
   sudo -u ilo-pce illumio-pce-ctl start --runlevel 1
   ```
4. On *any node*, verify runlevel 1:

   ```
   sudo -u ilo-pce illumio-pce-ctl cluster-status -w
   ```
5. On *any node*, run the following command:

   ```
   sudo -u ilo-pce illumio-pce-db-management setup
   ```
6. Repeat these steps for all PCEs in the Supercluster.

Restore Supercluster Data

Perform the following steps for *all PCEs in the Supercluster* one at a time or all in parallel.

1. On *any node of the PCE*, verify the runlevel is still 1:

   ```
   sudo -u ilo-pce illumio-pce-ctl cluster-status -w
   ```
2. On the *data0 node*, run the following depending on whether you are restoring a member or leader PCE.

   Member PCE

   In `--local-pce-file`, enter the path to the member PCE's backup file. In `--restoring-pce-file`, enter the path to the leader PCE's backup file, which should already be present on the PCE from when you followed the steps in Copy Leader Backup to Members [148].

   > **NOTE**
   > If necessary, copy the leader PCE's backup file to the data0 node of this PCE before running this command. Also, be sure that all member PCEs are using the same version of the leader PCE backup file. Using different versions of the leader PCE backup can cause data replication to fail after the supercluster restore is complete.

   ```
   sudo -u ilo-pce illumio-pce-db-management supercluster-data-
   restore --local-pce-file path_to_backup_file --restoring-pce-file
   path_to_leader_pce_backup_file  --restore-type entire_supercluster
   ```

   Leader PCE

   In `--local-pce-file`, enter the path to this leader PCE's backup file:

   ```
   sudo -u ilo-pce illumio-pce-db-management supercluster-data-restore --
   local-pce-file path_to_backup_file --restore-type entire_supercluster
   ```

   > **NOTE**
   > The restore operation can take some time to complete.
3. On *the data1 node*, run the following command after the restore operation finishes:

   ```
   sudo -u ilo-pce illumio-pce-db-management supercluster-data-restore
   --skip-db-restore --local-pce-file path_to_backup_file --restore-type
   entire_supercluster
   ```

   The `--skip-db-restore` option prevents the command from unnecessarily repeating work that has already been done by previous commands.
4. Set the runlevel to 2:

   ```
   sudo -u ilo-pce illumio-pce-ctl set-runlevel 2
   ```

   Setting the run level might take some time to complete.
5. Check the progress to see when the status is `RUNNING` on all nodes:

   ```
   sudo -u ilo-pce illumio-pce-ctl cluster-status -w
   ```

For all PCEs in the Supercluster, you must complete all steps in Install New PCEs or Reuse PCEs [156] and Restore Supercluster Data [156]. When finished, proceed to the next task.

Rejoin the PCEs to the Supercluster

Rejoin the leader PCE, then rejoin the member PCEs one at a time in any order.

1. Rejoin the leader PCE to the Supercluster. This command can take some time depending on the number of PCEs in the Supercluster and size of the PCE databases.
   On *any core node*, run the following command:

   ```
   sudo -u ilo-pce illumio-pce-ctl supercluster-restore
   fqdn_of_failed_cluster --restore-type entire_supercluster
   ```

   While this command is running, it temporarily sets the runlevel to 1. When the command is interrupted, you might see runlevel 1 unexpectedly.
2. Rejoin each member PCE to the Supercluster. This command can take some time depending on the number of PCEs in the Supercluster and size of the PCE databases.
   On *any core node*, run the following command:

   ```
   sudo -u ilo-pce illumio-pce-ctl supercluster-restore
   fqdn_of_failed_cluster fqdn_of_supercluster_leader --restore-type
   entire_supercluster
   ```

   While this command is running, it temporarily sets the runlevel to 1. When the command is interrupted, you might see runlevel 1 unexpectedly.
3. Repeat step 2 until all PCEs are rejoined to the Supercluster.

Finish and Verify Full Supercluster Restore

After rejoining all PCEs in the Supercluster:

1. On *all PCEs*, set the runlevel to 5:

   ```
   sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
   ```
2. On *all PCEs*, verify the runlevel:

   ```
   sudo -u ilo-pce illumio-pce-ctl cluster-status -w
   ```
3. Verify that the restored PCEs have rejoined the Supercluster and are fully operational.
   a. Log into the leader PCE web console.
   b. Go to the PCE Health page and verify that the PCE health status is Normal.
4. Check the status of the paired VENs on each PCE. From the PCE web console, choose Workloads and VENs > VENs. After all VENs change status from Active (Syncing) to Active, run the following command *on one PCE at a time*:

   ```
   sudo -u ilo-pce illumio-pce-ctl listen-only-mode disable
   ```

Import Database to Another Supercluster

This topic explains how to import data from one Supercluster to another Supercluster. For example, you might want to synchronize a test Supercluster with production Supercluster data.

The procedure makes use of two scripts, `remap_supercluster_backup.rb` and `update_supercluster_login.rb`, which are found in `$INSTALL_ROOT/illumio/scripts`.

Back Up Source Supercluster

Back up each PCE in the source Supercluster. See Back Up Supercluster [147] for information.

Restore Backup to Target Supercluster

To import the database, you use a procedure that is similar to restoring a backup with a few extra steps.

Prepare Target Supercluster

1. On *each node in the target Supercluster*, install the PCE software. Use the same software version on the target Supercluster that was installed on the source Supercluster.
2. Copy the backup files to the data0 node of each corresponding PCE in the target Supercluster. When the target Supercluster has fewer PCEs, decide which backups you want to restore.
3. Collect the public IP addresses for each target PCE. You can find them in the `cluster_public_ips` section of each PCE's Runtime Environment File.
4. When you have configured the following settings, verify that they are identical on the source and target Superclusters:
   - `front_end_https_port`
   - `front_end_event_service_port`
   - `front_end_management_https_port`

Remap Supercluster Backup Files

Perform the following steps on each PCE in the target Supercluster.

1. On the *data0 node of the PCE in the target Supercluster*, run the script `remap_supercluster_backup.rb` with the name of the backup file to be remapped and the name of the file in which to write the remapped backup data. You can optionally include the flag `--pce-fqdns-to-skip` with a comma-separated list of fictitious FQDNs that you do not want to include in the remapped database.

   ```
   sudo -u ilo-pce remap_supercluster_backup.rb --pce-fqdns-to-skip
   FQDN1,FQDN2... source_backup_fileremapped_backup_file
   ```

   For each source PCE FQDN in the backup, the script prompts for the following values:
   - The corresponding target FQDN
   - The corresponding target `cluster_public_ips` from that PCE's runtime environment file

     When you have more source PCE FQDNs than target FQDNs, use fictitious names and IP addresses for the extras.
2. When prompted, enter the FQDN of the target PCE where this backup will be restored.
3. After the remapped target backup file is written, copy the file to the data1 node.
4. Repeat these steps on each PCE in the target Supercluster.

Restore Remapped Backup Files

Follow the steps for restoring an entire Supercluster in Restore a PCE or Entire Supercluster [152], but do not bring the PCEs to runlevel 5. Leave the PCEs at runlevel 2.

Update Login Service on Leader

1. On the *leader PCE*, update login service properties by running the script `update_supercluster_login.rb` and specify the full path to the remapped backup file generated by `remap_supercluster_backup.rb`:

   ```
   sudo -u ilo-pce update_supercluster_login.rb remapped_backup_file
   ```

2. Bring the PCEs to runlevel 5:

```
sudo -u ilo-pce illumio-pce-ctl set-runlevel 5
```

Remove PCE from Supercluster

1. Migrate any VENs that are paired to the PCE you intend to remove. Use the procedure in Reassign VENs in Supercluster Using REST API [133].

   Be sure to use this procedure to unpair the VENs. Do not use a DNS move.
2. On *all PCEs in the supercluster*, bring the PCE to runlevel 2:

   ```
   sudo -u ilo-pce illumio-pce-ctl set-runlevel 2
   ```
3. On the *core node* of the PCE to be removed, run the following command:

   ```
   sudo -u ilo-pce illumio-pce-ctl supercluster-leave
   ```

VEN Install Upgrade

Overview of VEN Installation

This section provides an overview on how to install the VEN on your hosts. It explains the two ways to install the VEN. When you've chosen an installation method, you can skip to the content for that method.

About This Installation Guide

Before installing VENs on the hosts in your environment, ensure that you meet the necessary technical background.

How to Use This Guide

This guide explains how deploy the Virtual Enforcement Node (VEN) on your distributed, on-premises systems.

The guide provides the details to complete the following tasks:

- An explanation of the VEN installation methods, namely, by using the VEN Library in the PCE versus the VEN Control Interface (CTL)
- How to install VENs using the VEN Library
- How to install VENs using packaging technology and the workload operating systems' native command line interface
- How to uninstall, upgrade, activate, and deactivate VENs by using the VEN CTL
- How to set up the PCE to install VENs by using the PCE web console and the VEN Library

Before Reading This Guide

Illumio recommends that you be familiar with the following topics before you follow the procedures in this guide:

- Your organization's security goals
- The Illumio Core platform
- General computer system administration of Linux and Windows operating systems, including startup/shutdown, and common processes or services
- Linux/UNIX shell (bash) and Windows PowerShell
- TCP/IP networks, including protocols and well-known ports

Notational Conventions in This Guide

- Newly introduced terminology is italicized. Example: *activation code* (also known as pairing key)
- Command-line examples are monospace. Example: `illumio-ven-ctl --activate`
- Arguments on command lines are monospace italics. Example: `illumio-ven-ctl --activate activation_code`
- In some examples, the output might be shown across several lines but is actually on one single line.

- Command input or output lines not essential to an example are sometimes omitted, as indicated by three periods in a row. Example:

```
...
some command or command output
...
```

Ways to Install the VEN

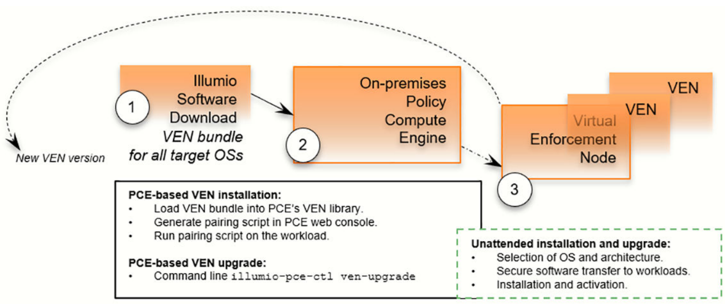
You can install the VEN two ways. These two ways are nearly identical and achieve the same goal: VEN installation and upgrade.

- Using the VEN Library integrated into the PCE: This method is documented in the topics about installing and upgrading the VEN Library.
- Manual VEN installation on individual workloads with your own software deployment tools: This method is documented in the topics about installing and upgrading the VEN using the VEN CTL.

VEN Installation Using the VEN Library

> **NOTE**
> The VEN Library installation and upgrade feature in the PCE is available for the RPM, Debian, and Windows distributions of the VEN software. Other workload operating systems are not supported.

Using the VEN Library in the PCE to install the VEN is a more automated approach than installing the VEN CTL but it gives you less control over optional aspects of VEN installation and upgrade.



The VEN Library method of installation utilizes a *VEN software bundle*. A VEN software bundle is a collection of a particular VEN software version for all supported workload operating systems.

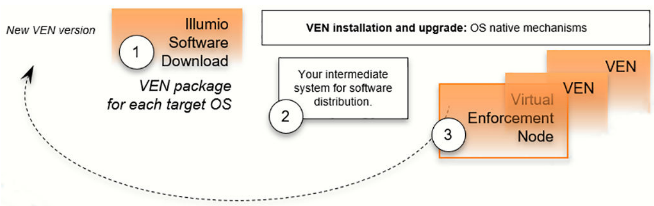
- In the PCE, you load a VEN software bundle into the *VEN library*. The VEN library is a collection of all VEN software versions you have loaded.
- For VEN installation:

  - In the PCE web console, you set a default VEN version.
  - In the PCE web console, you generate a pairing script to install and activate the VEN on target workloads.
  - You copy the pairing script to the target workload and run it.
  - The pairing script:
    - Determines the OS and CPU architecture of the target workload.
    - Securely transfers the VEN software to the target workloads.
    - Installs the VEN software.
    - Pairs the VEN with its PCE.
- For VEN upgrade, use the VEN Library in the PCE to upgrade all workloads or selective workloads.
- Some features are not available with VEN Library method, such as Kerberos-based authentication and custom settings with environment variables.

> **NOTE**
> Setting up the VEN Library in the PCE is required only for Illumio On-premises customers. If you are an Illumio Cloud customer, Illumio Operations performs this task for you.

VEN Installation Using the VEN CTL

This method gives you greater control over optional aspects of VEN installation, pairing, and upgrade.



The VEN installation method using the CTL starts with downloading a *VEN package*. A VEN package is the VEN software for a single supported workload OS and CPU architectures. Installation and upgrade rely on package managers, which are standard, native OS tools.

- For VEN installation with this method:
  - Determine the OS and CPU architecture of the target workloads.
  - Download the appropriate VEN packages.

    For example, installing a VEN on CentOS 8 x86-64 requires you to download the VEN package `illumio-ven-XXX.c8.x86_64.rpm`.

    > **NOTE**
    > You are responsible for securely transferring the VEN software to the target workload with your own software deployment mechanisms.

  - Optionally, set the following environment variables or command-line options:
    - Custom installation directories

    - Custom user and group names
    - Kerberos-based authentication for VEN-to-PCE communications
  - Run the native OS installation mechanism.

    For example: `rpm -ihv illumio-ven*.rpm`
  - Pair the VEN with its PCE.
    - You can pair the VEN during installation or after installation using the VEN CTL activate command (`illumio-ven-ctl activate <options>`)
    - You can use a "prepare script" to install the VEN software on machine images and activate it at the next boot.
- If you installed the VEN with the VEN CTL and packaging CLI and customized installation options (such as, a custom installation directory or alternate VEN user), you cannot later upgrade the VEN by using the VEN Library in the PCE. You must upgrade the VEN using the workload's OS package upgrade process.

> **TIP**
> If you try to upgrade a VEN using the VEN Library in the PCE but nothing happens, verify whether the VEN was installed by using the VEN CTL.

When to Use Which Method

You can use both methods at different stages of your VEN installation.

Installation Method	Use Cases
VEN Library in the PCE	• To demonstrate the ease of VEN installation and assess installing Linux VENs using the VEN Library • To evaluate and certify new versions of the VEN
VEN CTL	To obtain more control over VEN installation and upgrade with a proprietary software distribution method

VEN-to-PCE Authentication

Illumio Core has the following mechanisms for authentication between the VEN and the PCE:

- VEN pairing with the PCE
- Kerberos authentication with the PCE

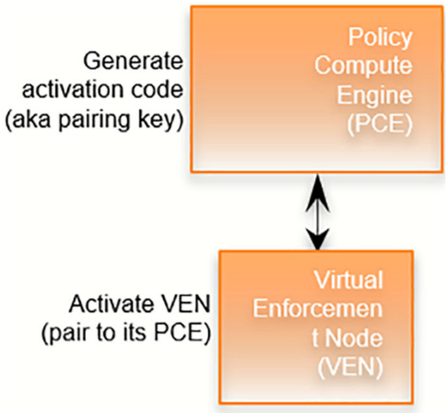
Use one or both mechanisms across your organization, but they are mutually exclusive for the same workload.

> **IMPORTANT**
> This guide assumes that you already have a functional Kerberos service with which to authenticate.

VEN Authentication by Pairing with PCE

This is the default mechanism. When you install a VEN on a workload, the VEN is activated with an activation code generated by the PCE. The activation code is an identifier passed to the VEN software at activation.

After the VEN is activated, it communicates with the PCE over a secure connection. This process of activating a VEN is referred to as pairing it with the PCE. The term *activation* also applies when installing the VEN package directly on a workload by using the VEN CTL.



About the VEN Activation Code

The activation code is an identifier passed to the VEN software at activation. It is obtained from the pairing key. An activation code can be created for one-time use for a single workload or multiple uses for many workloads.

You can get an activation code in the following ways:

- In the PCE web console, create a Pairing Profile. In the profile, you can specify one-time use or unlimited use for the activation code.
- With the REST API. For information, see "Create a Pairing Key" in the *REST API Developer Guide*.

Activation Details

An activation code is used only after initially installing the VEN. During activation, the PCE generates an agent token. The VEN stores the agent token in a local file on the workload. The PCE stores the hash of the agent token. The VEN uses the agent token to uniquely authenticate itself to PCE. Only the agent token is used in VEN-to-PCE communication from that point on.

The VEN communicates with the PCE using HTTPS over Transport Layer Security (TLS) for REST calls and TCP over TLs for the events channel. Additionally, a clone token is generated. When an agent token is mistakenly or maliciously reused on another workload, the clone token is used to detect the condition and disambiguate the hosts. The clone token is periodically rotated. The agent token is never rotated.

NOTE

Automatic Cloned VEN Remediation

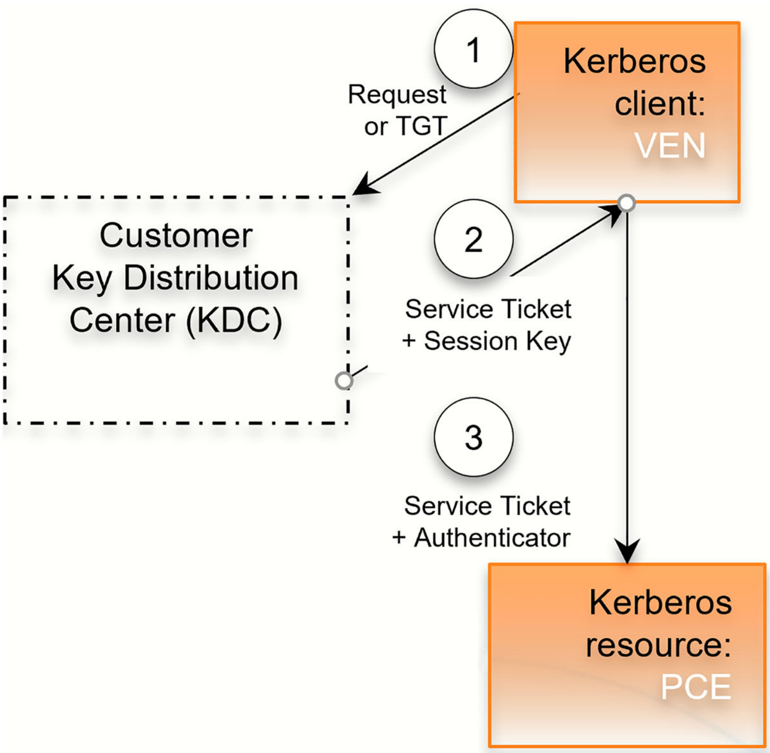
For on-prem domain joined Windows workloads, cloned VENs support automatic clone remediation by detecting changes to the workload's domain Security identifier (SID). After the VEN reports such changes to the PCE, the PCE tells the clone to re-activate itself, after which the cloned VEN is remediated and becomes a distinct agent from the original VEN.

VEN Authentication via Kerberos

You can configure the PCE and VEN to rely on authentication by a pre-configured Kerberos-based system, such as Microsoft Active Directory.

NOTE

Kerberos-based authentication is supported when you install the VEN by using the VEN CTL. It is not supported when you use the VEN Library in the PCE to install the VEN.



The Key Distribution Center (KDC) is your pre-configured Kerberos service; the VEN is a Kerberos client; and the PCE is a Kerberos resource.

1. The VEN requests a session key or passes its ticket granting ticket (TGT).
2. The KDC returns a service ticket and session key.
3. The VEN passes the authenticated service ticket to the Kerberos-protected PCE.

For information about setting up Kerberos for VEN authentication with the PCE, see Set up Kerberos Authentication on PCE [190].

For information about pairing workloads via Kerberos for each operating system, see the following topics:

- Kerberos for Windows VEN-to-PCE Authentication [223]
- Kerberos for Linux VEN-to-PCE Authentication [228]

Additionally, you can use the Illumio Core REST API to set up VEN authentication with the PCE via Kerberos. See the "Workload Operations" and "Bulk Traffic Loader" topics in the *REST API Developer Guide* .

VEN-unactivated Golden Masters

When you create machine images for faster deployment of the VEN, consider preparing them to pair the VEN with the PCE the first time the workload is booted. See Prepare Golden Master for Workload Installation [191]for information.

Upgrading from pre-20.2 to Later Versions

When upgrading from a PCE version pre-20.2 to a later version, stopped VENs that have sent a *goodbye* message to the PCE will have their status value set to *stopped*. During the upgrade, this procedure can cause a burst of events to be emitted to the PCE event stream.

Reduced Banners During VEN Upgrades

The user interface experience on VENs has been enhanced. You will no longer see multiple banners during VEN upgrades. In lieu of an additional banner being displayed, a tally will indicate current upgrade status, show what process is suspended, or display what process is experiencing issues during the upgrade.

MSI to EXE Package Format

Starting with the 21.2.1 Illumio Core release, the Windows VEN installer switched from using the MSI package format to using the EXE package format. Customers using the PCE-based VEN deployment must take an extra step for the transition. Specifically, Illumio Core customers running older, MSI-based Windows VENs must upgrade to 19.3.6+H1-VEN or 21.2.0+H2-VEN before upgrading their VENs to 21.2.1 or a later version. This release contains the necessary VEN changes to handle the transition in the VEN packaging from MSI to EXE.

Prepare for VEN Installation

This section provides information that you need to know before installing the VEN software. For a smooth and successful installation of the VEN in your environment, meet the prerequisites outlined in this section.

Details about VEN installation

This topic describes how VEN installation works. Before installing VENs on workloads, review this section.

VEN Pairing and Activation

The term **pairing** applies when using the VEN Library through the PCE web console to install VENs on workloads. The term **activation** applies when installing the VEN package directly on a workload.

A workload pairs or activates with the PCE to become part of the distributed security system using one of these methods:

- By sending a pairing key or activation code
- With a Kerberos service principal name (SPN)
- With a PKI certificate

A pairing key or activation code is used only after initially installing the VEN. During pairing or activation, an agent token is generated and stored in a local file on the workload. The hash of the token is stored on the PCE. The VEN uses the agent token to uniquely authenticate itself to PCE. Only the agent token is used in VEN-to-PCE communication from that point on.

The VEN communicates with the PCE using HTTPS over Transport Layer Security (TLS). Additionally, a clone token is generated. When an agent token is mistakenly or maliciously reused on another workload, the clone token is used to detect the condition and disambiguate the hosts. The clone token is periodically rotated. The agent token is never rotated.

> **NOTE**
>
> **Automatic Cloned VEN Remediation**
>
> For on-prem domain joined Windows workloads, cloned VENs support automatic clone remediation by detecting changes to the workload's domain Security identifier (SID). After the VEN reports such changes to the PCE, the PCE tells the clone to re-activate itself, after which the cloned VEN is remediated and becomes a distinct agent from the original VEN.

Linux Packages and Kernel Modules

Some packages, such as SecureConnect StrongSwan for enforcing IPsec, are included as part of the VEN package. Other packages are installed on the host itself if they are not already present.

If the following packages are not installed on the workload via RPM dependencies the VEN installation downloads and installs them.

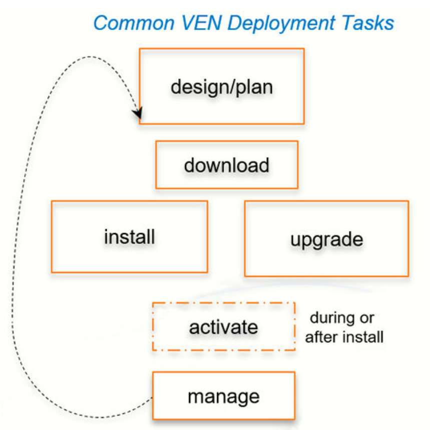
- `curl`: Used for HTTPS client functionality
- `uuid-runtime`: Used for generating UUIDs

- `ipset`: Used for ipset functionality
- `libnfnetlink0`: Used for communicating with the Net Filter module
- `libcap2:` Used for selectively enabling/disabling capabilities
- `libgmp10`: Used for multi-precision arithmetic
- `bind-utils`: Used for DNS client functionality
- `iptables` and `iptables-ipv6`: Used for iptables functionality
- `apt-transport-https` (for apt-based OS): Used for HTTPS transport for apt

NOTE
If the ipset kernel module is not installed, the VEN downloads and installs it.

Workflows for VEN Installation

The following diagram illustrates the workflow for performing common VEN installation and upgrade tasks.



VEN Installation Planning Checklist

This checklist summarizes VEN planning considerations and requirements detailed in this guide. It compares the requirements and considerations of both deployment methods.

Tasks	VEN Library	VEN CTL
1. Select VEN installation method.	✓	✓
2. Select VEN-to-PCE authentication mechanism: Activation Code or Kerberos.		✓
3. Select whether to activate the VEN to the PCE during or after VEN installation.		✓
4. Select whether to use a single-use or unlimited-use pairing key		✓
5. Review VEN-to-PCE communication requirements.	✓	✓
6. Review VEN workload disk sizing requirements for hosts.	✓	✓
7. Review OS and package dependencies.		✓
8. Determine VEN software package for workload CPU architecture.	✓ (Automatic)	✓
9. Remove parameters `ven_repo_url` and `ven_repo_ips` from PCE runtime_env.yml.	✓	
10. (Linux only) Configure mount `tracefs` or `debugfs`.	✓	✓
11. Download the VEN software: • VEN Library: All VEN versions in a VEN software bundle • Installation using the CLI and VEN CTL: All VEN versions in a package of VEN software for specific OS versions and CPU architectures.	✓	✓
12. (Optional) Verify signature of downloaded packages against Illumio's public key.		✓
13. Generate VEN pairing profiles and pairing key.	✓	✓
14. Securely copy VEN paring script to the workload.	✓	
15. (Optional) Prepare VEN-unactivated golden master machine Images.		✓

Prerequisites for VEN Installation

Before installing VENs on the workloads in your environment, you must understand and meet the following prerequisites.

PATH Environment Variable for `illumio-ven-ctl`

For more information about using the VEN CTL, see "illumio-ven-ctl General Syntax" in the VEN Administration Guide.

For easier invocation of `illumio-ven-ctl` and other control scripts, set your `PATH` environment variable to the directories where they are located:

- Linux: default location is `/opt/illumio_ven`
- Windows: default location is `C:\Program Files\Illumio`

Windows VEN PowerShell Version

All supported releases of the Windows VEN 23.2.x and earlier (both Core and Endpoint) require PowerShell 5.1 or earlier for installation and normal functioning; PowerShell Core versions 6.0 and higher are not sufficient, though they may be installed alongside earlier versions. For more information, see this Illumio Knowledge Base article.

VEN OSs and Package Dependencies

- The VEN software package formally declares dependencies on certain other required software packages. Some of these other packages must be installed before the VEN is installed, while others are installed concurrently during VEN installation.
- The VEN software also makes use of certain other software packages without formally declaring dependencies on them. This allows the same VEN software to be installed on various workloads regardless of which other packages may also be installed. The VEN automatically detects and configures the other software as needed, just-in-time, and reports an error if it encounters any problems with the other software.

For the complete list of package dependencies by operating system, see the VEN OS Support and Package Dependencies page on the Illumio Support portal.

Minimum key length with RHEL8+ cryptographic policy set to FUTURE

If the cryptographic policy is set to FUTURE on RHEL8, RHEL9, and related operating systems, the RSA key length must be 3072 bits or greater. For more information, see this Knowledge Base article.

VEN-to-PCE Communication

Illumio Core uses Transport Layer Security (TLS) version 1.2 by default for VEN-to-PCE communications.

- The PCE default minimum version is TLS 1.2.
- For VEN versions 18.1 and later, all VENs use TLS 1.2.

For more information about the TLS requirements for VEN-to-PCE communication, TLS Versions for Communication [38].

Before installing a VEN, the workload must meet the following requirements for VEN-to-PCE communication:

- The workload can validate its certificate's chain of trust back to the root Certificate Authority (CA) of the server certificate on the PCE.
- The VEN can reach the PCE on the ports configured for the PCE in the PCE Runtime Environment File `runtime_env.yml`. Contact your Illumio Support representative for more information.
- To prevent time drift between the PCE and VENs, Network Time Protocol (NTP) must be installed and working on the PCE and the VENs.

Workload Disk Size Requirements

Illumio recommends that you reserve **10GB** of disk space on workloads for the VEN. The amount of disk space ultimately used will vary depending on the size and complexity of your microsegmentation environment.

Application logs are rotated from primary to backup when their size reaches 15 MB. Application log files are preserved at reboot, because application logs are stored in files on a workload.

IP Address Support

In Illumio Core 20.2.0 and later releases, the VEN supports both IPv4 and Ipv6 address versions and the IP address version appears correctly in the PCE; for example, in the Workload section of the VEN summary page in the PCE web console.

You can configure how the PCE treats IPv6 traffic from workloads. For more information, see "Allow or Block IPv6 Traffic" in the PCE Administration Guide.

Obtain the VEN Packages

PCE-based VEN software bundle

If you are an Illumio On-premises customer (you are running the PCE in your corporate data center), download the VEN packages to your PCE by running the `illumio-pce-ctl` from your PCE. For more information, see VEN Library Setup in the PCE [182].

> **NOTE**
> Illumio Cloud customers don't have shell access to the PCE; therefore, the Illumio Operations team downloads and sets up the PCE-based VEN software bundle for customers. They download all necessary VEN packages for customers.

CLI-based VEN software packages

All VEN software is available for download from the Illumio Support portal. A VEN package is downloadable from the Illumio Support portal for each version of the VEN. Illumio provides the package as a tar file that contains a version of the VEN for all supported operating systems.

To download the VEN package:

1. Go to the Illumio Support site (login required).
2. Select **Software > Download** under the VEN section > VEN version.
   The Download VEN page appears.
3. In the VEN Packages row of the VEN table, click the filename for the VEN tar file.
4. Download the file to a convenient location.

VEN Package CPU Architecture

For VEN installation using the VEN CTL, after you have downloaded and unpacked the software, determine the VEN appropriate for your operating systems and hardware architecture.

See the Supported Operating Systems for Illumio VEN table - CPU Architecture Identifier in Filename column on the Illumio Support portal.

(Optional) Verify Package Signature

For additional security, verify the identity of the downloaded VEN packages against the Illumio public key.

> **NOTE**
>
> - You can verify the signature of the VEN RPM packages for CentOS, Red Hat Enterprise Linux (RHEL), Ubuntu, and SUSE Linux Enterprise Server.
> - Signature verification is not support for AIX, Debian, Solaris, and Windows VEN packages.

The Illumio public key is available on the Download VEN page of the Illumio Support portal (login required).

For information about using a public key to verify package signatures, see Checking a Package's Signature on the Red Hat Customer Portal.

Firewall Tampering Protection on Linux

To enable faster host firewall tampering protection (within approximately three seconds) for Linux firewalls, make sure that:

- `tracefs` is mounted (newer Linux distributions)
- `debugfs` is mounted (older Linux distributions that include `tracefs` in `debugfs`)

For information, see "VEN Firewall Tampering Detection" in the VEN Administration Guide.

> **NOTE**
>
> Faster host firewall tampering protection is enabled for Windows automatically.

VEN Compatibility Check

In additional to meeting the requirements in this topic and being aware of the limitations for installing VENs on workloads, you can use the VEN Compatibility Check feature to verify the functionality of the VEN on a workload. The compatibility information for the VEN is available only while the VEN is in Idle mode.

For information about this feature, see VEN Compatibility Check [252].

SecureConnect Setup on Workloads

For information about SecureConnect requirements for VENs, see "SecureConnect" in the Security Policy Guide.

MSI to EXE Package Format

Starting with the 21.2.1 Illumio Core release, the Windows VEN installer switched from using the MSI package format to using the EXE package format. Customers using the PCE-based VEN deployment must take an extra step for the transition. Specifically, Illumio Core customers running older, MSI-based Windows VENs must upgrade to 19.3.6+H1-VEN or 21.2.0+H2-VEN before upgrading their VENs to 21.2.1 or a later version. This release contains the necessary VEN changes to handle the transition in the VEN packaging from MSI to EXE.

VEN Support for Red Hat 5

This section describes Illumio VEN support for the Red Hat 5 operating system.

For more about OS support for the VEN, see VEN OS Support and Dependencies in the Illumio Support portal.

History of Illumio Red Hat 5 Support

- Illumio's initial support for Red Hat Enterprise Linux (RHEL) 5.5 or greater began in VEN release **18.2.x**.
- Illumio discontinued support for RHEL 5.x in VEN **18.3.x** and later releases.
- Illumio re-introduced support for RHEL 5.x in VEN release **23.2.10** and later releases.
- Beginning in VEN release 24.2.20:
  - Illumio supports Oracle Linux 5 UEK
  - VEN activation fails if the ipset userspace binary is not installed on RHEL 5 workloads.

Requirements

- VEN support for RHEL 5 requires VEN release 23.2.10 or later.
- If no version number is listed for an OS package dependency, use the version included with the OS distribution.
- If you are using a public Certificate Authority to sign your OnPrem PCE, or if you've deployed a SaaS PCE, then to install the VEN on RHEL 5.x, make sure that the latest Certificate Authorities are installed with your operating system.
- Download the VEN software package (`illumio-ven-pkgs`) from the Illumio Support portal and use the VEN CTL to install and activate the VEN on the hosts in your environment.

Limitations for VEN Support on RHEL 5.*x*

Installing the VEN on RHEL 5 workloads presents the following limitations:

- **Must download VEN software package (RPM) and install and activate using the VEN CTL**

  For RHEL 5 VENs, only the VEN CTL installation and activation methods are supported (see Linux: Install and Upgrade with CLI and VEN CTL). Download the VEN software package (`illumio-ven-pkgs`) from the Illumio Support portal. VEN installation using the VEN Library on the PCE is not supported.
- **On VEN release 24.2.20, VEN activation fails if ipset userspace binary isn't installed on the RHEL 5 workload**

  The ipset userspace binary is not installed by default on RHEL 5 workloads. If the binary is missing while attempting to activate a 24.2.20 VEN on a RHEL 5 VEN workload, activation fails and the following Runtime Environment error occurs. To remedy, you must install the provided ipset userspace binary on the workload and then try again to activate the VEN:

```
 Cannot find ipset command. Please install /opt/illumio_ven/etc/extras/
pkgs/ipset-<version>.x86_64.rpm
```

On VEN releases from 23.2.10 to 24.2.10, VEN activation doesn't fail if the ipset userspace binary isn't detected on the workload but a failure may occur later when the VEN tries to apply policy.

- **Incompatible ipset modules may have been loaded on the workload**
  If one or more incompatible ipset modules is loaded on the RHEL 5 workload, the VEN enters an error state and the following error message appears in the `platform.log`. To remedy, you must remove the incompatible ipset module from the workload.

```
ERROR:: ilo_ipsets load returned non-zero. /opt/illumio_ven/bin/
ilo_ipsets load -w workload/ .... /new 2>&1 insmod: error inserting
'/opt/illumio_ven/etc/extras/modules/el5uek/5.11/ipset/ip_set_iphash.ko':
-1 Unknown symbol in module Cannot load ip_set_iphash module. Cannot load
ip_set_iphash module. Please unload incompatible modules ip_set.
```

- **Missing ipset modules can cause policy application failure**
  On all VEN releases that support RHEL 5 (23.2.10 and later), applying policy fails if compatible ipset modules are missing. Beginning with VEN release 24.2.20, the VEN tries to resolve the failure if it occurs. VEN release 24.2.20 also provides the correct ipset modules for Oracle Linux 5 UEK.
- **VEN tampering may occur if the iptables service loads rules in iptables at boot**
  Illumio recommends that you disable the iptables service before you install the VEN on RHEL workloads. This is necessary because the iptables service executes a script that loads rules in `/etc/sysconfig/iptables` at boot, which is undesirable for two reasons: (1) non-VEN rules may conflict with rules the VEN applies to the firewall, and (2) the VEN may regard these non-VEN rules as firewall tampering.
- **The VEN cannot enforce FQDN policy (DNS-based rules).**
- **The VEN doesn't support IPv6 or IPv6 ipsets**
- **The VEN doesn't support byte counting.**
- **AdminConnect (also known as Machine Auth) is not supported**.

ULOG for Traffic Flows

VENs on RHEL 5 workloads use ULOG to log traffic flows.

Illumio Changes to iptables for VENs on RHEL 5.x Workloads

> **NOTE**
> These changes to iptables affecting Red Hat 5 also affect other Linux distributions using iptables, including Red Hat 6 and 7, Ubuntu, Debian, and SUSE.

In its support for RHEL 5, Illumio has changed the type of set the VEN uses for ipset.

- To run the VEN on RHEL6/7, Ubuntu, Debian, SUSE versions, the VEN supports using only `hash:net` for ipset, used to store a single IP address or the CIDR.
- To run the VEN on RHEL 5.x, for every ipset, you can select from two ipset types: `iphash`, used to store a single IP address, or `nethash`, used to store a CIDR.

To summarize:

Changes that Illumio made to support VENs on RHEL 5.x include:

- The VEN uses `iphash` and `nethash` set types. Both have a limit of 65536 elements.
- `iphash` can store only single IP addresses; `nethash` can store only CIDRs.

Changes that Illumio made to support VENs on RHEL 6/7, Ubuntu, Debian, SUSE and later versions include:

- The VEN release 23.2.10 and later uses `hash:ip` to store single IP addresses and `hash:net` to store CIDRs.

Example to Illustrate the iptables Changes in Linux

The following example compares how Illumio generated iptables before and after VEN release 23.2.10 in RHEL 7.

RHEL 7 ipset prior to the RHEL 5 change[1]	RHEL 7 ipset after the RHEL 5 change[2]	RHEL 5 ipset beginning in VEN release 23.2.10[3]
create ILON-F535B4CAC2EB950D **hash:net**	create ILON-F535B4CAC2EB95000 **hash:ip**	-N ILON-F535B4CAC2EB95000 **iphash**
family inet maxelem 6 hashsize 6	family inet maxelem 3 hashsize 3	
add ILON-F535B4CAC2EB950D 10.10.10.10/32	add ILON-F535B4CAC2EB95000 10.10.10.10	-A ILON-F535B4CAC2EB95000 10.10.10.10
add ILON-F535B4CAC2EB950D 20.20.20.20/32	add ILON-F535B4CAC2EB95000 20.20.20.20	-A ILON-F535B4CAC2EB95000 20.20.20.20
add ILON-F535B4CAC2EB950D 30.30.30.30/32	add ILON-F535B4CAC2EB95000 30.30.30.30	-A ILON-F535B4CAC2EB95000 30.30.30.30
add ILON-F535B4CAC2EB950D 40.40.40.40/24	create ILON-F535B4CAC2EB95001 **hash:net** family inet maxelem 3 hashsize 3	-N ILON-F535B4CAC2EB95001 **nethash**
add ILON-F535B4CAC2EB950D 50.15.50.50/28	add ILON-F535B4CAC2EB95001 40.40.40.40/24	-A ILON-F535B4CAC2EB95001 40.40.40.40/24
add ILON-F535B4CAC2EB950D 60.60.60.60/31	add ILON-F535B4CAC2EB95001 50.50.50.50/28	-A ILON-F535B4CAC2EB95001 50.50.50.50/28
	add ILON-F535B4CAC2EB95001 60.60.60.60/31	-A ILON-F535B4CAC2EB95001 60.60.60.60/31

- [1] Prior to VEN release 23.2.10, this is how Illumio generated RHEL 7 for ipset.
- [2] Beginning in VEN release 23.2.10, there is no need to change how you define IP Lists in the PCE; the VEN handles everything for you.
- [3] Beginning with VEN release 23.2.10, this is how Illumio generates RHEL 7 for ipset.

VEN Proxy Support

This section describes how to enable proxy support for the VEN on all supported operating systems: Windows, Linux, AIX, and Solaris.

Enforce an allow rule for proxy connectivity

CAUTION

Enforce an allow rule for proxy connectivity

If your environment includes a proxy server, make sure your Illumio policy includes an allow rule for the proxy's **IP:port** before applying a new policy in Selective or Full Enforcement mode. Otherwise, if the VEN discovers that no allow rule is in place allowing the proxy connection, it reports a policy sync error and tries continually to sync policy. In that circumstance, the VEN and the PCE will not be able to communicate.

NOTE

Proxy support setup for the VEN is different between Unix-based versus Windows operating systems due to platform differences. The VENs for Unix-based operating systems do not require system wide proxy setting. For Unix-based VENs, each application obtains the proxy settings from the user, for example, `curl --proxy myproxy:80`. On Windows, the operating system provides proxy settings; for example, the Chrome browser uses the same proxy setting as Microsoft Edge. See the topics below for the details of setting up VEN proxy support by platform.

VEN Connections via Windows Proxy Servers

For Windows workloads only, Illumio Core supports a VEN-to-PCE connection through proxy servers.

- The default proxy configuration on the OS is used and proxy configuration might not be required or available on the VEN.
- Only non-authenticated proxy is supported, which might require that you add an exception for the PCE address.
- Only HTTP proxy is supported. The VEN will detect the proxy automatically and configuration or mode change will not be required.

Configuration for a Windows Proxy Server

- If the network environment supports WPAD protocol, the VEN will automatically use WPAD to discovery proxies and no special configuration is required.
- If proxy configuration is done via a PAC file, you will have to import Internet Explorer's (IE) proxy setting with the PAC file URL to the LocalSystem user (S-1-5-18). The VEN only supports `http://` PAC file URL. It does not support `file://` URLs.
- If proxies are statistically configured, you can configure using one of the following two methods:
  - Using `netsh winhttp set proxy` command. This method takes precedence.

    For `netsh winhttp` usage, refer to https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc731131(v=ws.10).

- Importing IE setting with static proxies setting to the LocalSystem user.
  For importing IE settings for the VEN, refer to https://serverfault.com/questions/34940/how-do-i-configure-proxy-settings-for-local-system.

> **NOTE**
> Both IE-based proxy setting and `netsh winhttp` can be pushed to the endpoints (PCs) using Group Policy.

For information about the proxy string format to use for Windows proxy servers, see also WINHTTP_PROXY_INFO (winhttp.h) - Win32 apps in the Microsoft documentation for information.

VEN Connections via Unix-based Proxy Servers

Release 21.1.0 and later releases extend VEN proxy support from Windows to Linux, AIX, and Solaris systems.

In comparison with Windows, the following limitation affects this feature for Unix-based proxy servers. This release doesn't support the Web Proxy Auto Discovery (WPAD) protocol or proxy discovery via the Proxy Auto Discovery (PAC) file for Unix-based proxy servers. This limitation occurs because VENs use LibCurl as the HTTP transport library, but LibCurl does not provide JavaScript execution capability needed to run proxy scripts. For a workaround, see Proxies - Everything curl.

Configuration for Unix-based Proxy Servers

To set up your environment for a Unix-based proxy server, perform the following steps:

1. Set the proxy string during activation using the `--proxy-server` option. For example, use `illumio-ven-ctl activate --proxy-server proxy-string` as shown:

   ```
   root@qual-solaris11-L:/opt/illumio# /opt/illumio_ven/illumio-ven-ctl
   activate --management-server example.com:8443 --activation-code <code> --
   proxy-server 172.24.88.114:3128
   Checking Runtime Environment..........             Activating Illumio
                 ------------------
   Storing Activation Configuration .....
   Starting Illumio Processes............               Pairing Status
                   --------------
   Pairing Configuration exists ......SUCCESS
   VEN Manager Daemon running ........SUCCESS
   Master Configuration retrieval ....SUCCESS
   VEN Configuration retrieval .......SUCCESS
   VEN has been SUCCESSFULLY paired with Illumio
   root@qual-solaris11-L:/opt/illumio# /opt/illumio_ven/illumio-ven-ctl
   show-proxy
   proxy_server: 172.24.88.114:3128
   ```
2. Set or modify the proxy string using `illumio-ven-ctl set-proxy proxy-string` and clear the proxy setting using `illumio-ven-ctl reset-proxy` as shown:

   ```
   [root@ven-rhel illumio_ven]# ./illumio-ven-ctl set-proxy http://
   proxy.example.com:3128
   ```

```
Updating proxy to http://proxy.example.com:3128. VEN restart needed.
[root@ven-rhel illumio_ven]# ./illumio-ven-ctl restart
Shutting down illumio-control:
 - venAgentMonitor Stopping venAgentMonitor:               [  OK  ]
<snip>

Starting illumio-control:
 - Environment Setting up Illumio VEN Environment:         [  OK  ]
<snip>

[root@ven-rhel illumio_ven]# ./illumio-ven-ctl show-proxy
proxy_server: http://proxy.example.com:3128
[root@ven-rhel illumio_ven]# ./illumio-ven-ctl reset-proxy
Resetting proxy. VEN restart needed.
[root@ven-rhel illumio_ven]# ./illumio-ven-ctl restart
Shutting down illumio-control:
 - venAgentMonitor Stopping venAgentMonitor:               [  OK  ]
<snip>
Starting illumio-control:
 - Environment Setting up Illumio VEN Environment:         [  OK  ]
<snip>
[root@ven-rhel illumio_ven]# ./illumio-ven-ctl show-proxy
No proxy is set
```

3. Restart the VEN after the proxy is set, modified, or cleared, except when the proxy is enabled using `--proxy-server` during activation. Query your current proxy setting using the `illumio-ven-ctl show-proxy` command.
4. Use the proxy string format: `[<scheme>"://"]<server>[":"<port>]`

   In the string format, `[]` indicates optional values in the command and `<>` indicates required values in the command; therefore, specifying either `--proxy-server 172.24.88.114:3128` or `http://172.24.88.114:3128` are both valid.

   NOTE

   When specified, only the “http” scheme is supported. Schemes such as “https” or any other schemes are not supported. For example, `http://my-proxy:8080` or `http://10.0.0.2:80`.

For Linux RPM (or AIX `installp`) installation, you can set the proxy string by setting and exporting the proxy string from the `VEN_PROXY_SERVER` shell variable before invoking the RPM (or `installp`) command.

For Solaris `pkgadd`, you can set the proxy string by setting the `VEN_PROXY_SERVER` variable to an answer file (typically created using the `pkgask` command).

Linux Pairing Script Activation for Proxy Servers

Typically, VENs are paired with the PCE directly. However, if a workload is behind a Web Proxy, you must follow these steps to enable your Linux/Unix VEN to successfully pair to your PCE:

1. From the PCE web console menu, choose **Workloads and VENs > Pairing Profile**.
2. Copy the pairing line from the Linux/Unix OS Pairing Script window.
3. Paste this pairing line into a text file so that you can edit it.

4. Edit the pairing line to make the following two changes (displayed in **bold**):
   a. Add **–x <proxy-string>** to the curl command to indicate the proxy string.
   b. Add **--proxy-server <proxy-string>** to the switch to pass the proxy string to the pairing script.

```
rm -fr /opt/illumio_ven_data/tmp && umask 026 &&
mkdir -p /opt/illumio_ven_data/tmp && curl –x <proxy-
string> --tlsv1 "https://example.com:8443/api/v18/software/ven/
image?pair_script=pair.sh&profile_id=1" -o /opt/illumio_ven_data/tmp/
pair.sh && chmod +x /opt/illumio_ven_data/tmp/pair.sh && /opt/illu-
mio_ven_data/tmp/pair.sh --management-server <server fqdn> --proxy-
server <proxy-string>
```

5. Paste the revised script into the Linux/Unix terminal and press **Enter**.

The workload starts the pairing process. As the pairing script runs, you will see success messages appear. Wait until you see the message "Workload has been SUCCESSFULLY paired with Illumio," which means your VEN (behind a proxy server) and the PCE are paired.

Configure a VEN-specific Windows Proxy

Beginning in VEN release 22.5, you must explicitly configure the Windows proxy for the VEN. In previous VEN releases, you didn't need to configure a proxy on Windows operating systems; instead, the VEN discovered proxy configurations automatically using the WPAD protocol or the Internet Explorer browser PAC file.

CAUTION

Enforce an allow rule for proxy connectivity

If your environment includes a proxy server, make sure your Illumio policy includes an allow rule for the proxy's **IP:port** before applying a new policy in Selective or Full Enforcement mode. Otherwise, if the VEN discovers that no allow rule is in place allowing the proxy connection, it reports a policy sync error and tries continually to sync policy. In that circumstance, the VEN and the PCE will not be able to communicate.

IMPORTANT

This topic applies to VENs deployed on servers or virtual machines and Endpoint VENs (VENs deployed on endpoints, such as Windows laptops.)

Ways to Configure a VEN-specific Windows Proxy

There are two ways to configure a VEN-specific Windows Proxy:

- Configure Windows Proxy by Editing the Pairing Script [181]
- Configure Windows Proxy using CTL Commands [181]

Configure Windows Proxy by Editing the Pairing Script

> **NOTE**
> For details about generating a pairing key and pairing script, see "Pairing Profiles and Scripts" in the section *VEN Installation & Upgrade Using VEN Library* in this guide.

You can configure a VEN-specific Windows proxy by adding the `-proxy-server` parameter to the pairing script.

In the following example, the `proxy-server` parameter is placed at the end of the script after the `-activation-code` parameter:

```
illumio-ven-ctl.exe show-proxy PowerShell -Command "&amp; {Set-
ExecutionPolicy -Scope process remotesigned -Force; Start-Sleep -s 3; Set-
Variable -Name ErrorActionPreference -Value SilentlyContinue;
[System.Net.ServicePointManager]::SecurityProtocol=[Enum]::ToObject([System.
Net.SecurityProtocolType], 3072); Set-Variable -Name ErrorActionPreference
-Value Continue; (New-Object System.Net.WebClient).DownloadFile('https://
example-server.io:443/api/v26/software/ven/image?
pair_script=pair.ps1&amp;profile_id=60', (echo $env:windir\temp\pair.ps1));
&amp; $env:windir\temp\pair.ps1 -management-server example-server:443.io
-activation-code <code> -proxy-server <proxy_server:port>;}"
```

Configure Windows Proxy using CTL Commands

Use the following CTL commands to explicitly configure a Windows proxy.

Installation:

```
<VEN Installation Directory>\illumio-ven-22.2.32-xxxx-preview.win.x64.exe /
install VEN_PROXY_SERVER=<proxy_server:port>
```

Activation:

```
<VEN Installation Directory>\illumio-ven-ctl.exe activate -management-
server <pce_server:port> -activation-code <code> -proxy-server
<proxy_server:port>
```

Restart:

You must restart the VEN after setting (or changing) its proxy configuration.

```
 <VEN Installation Directory>\illumio-ven-ctl.exe restart
```

Manage the proxy configuration:

The VEN CTL supports using the `set-proxy`, `reset-proxy`, and `show-proxy` commands to configure a proxy on Windows. Use of these commands takes precedence over `netsh` and discovery using the Internet Explorer PAC file.

For more information about how the Windows VEN supports a proxy server, see "VEN Proxy Support" in the section *Prepare for VEN Installation* in this guide.

The `set-proxy` command sets the proxy server for the VEN to use.

```
<VEN Installation Directory>\illumio-ven-ctl.exe set-proxy
<proxy_server:port>
```

The `show-proxy` command shows the current proxy configuration.

```
<VEN Installation Directory>\illumio-ven-ctl.exe show-proxy
```

The `reset-proxy` command removes the current proxy configuration.

```
<VEN Installation Directory>\illumio-ven-ctl.exe reset-proxy
```

Set up PCE for VEN Installation

When you plan to install or upgrade the VENs in your environment by using the PCE web console, be sure that you complete the PCE setup tasks described in this section before logging into the PCE web console to install VENs.

VEN Library Setup in the PCE

You can use your PCE cluster as a centralized mechanism for distributing, installing, and upgrading VENs in your environment.

> **NOTE**
>
> If you are an Illumio Secure Cloud customer, you do not need to set up the VEN Library in the PCE. Illumio Operations performs these tasks and upgrading VENs using the PCE web console and REST API is available for your environment. See VEN Upgrade Using the PCE Web Console [213] for information.

About the VEN Library in the PCE

You can use the PCE web console to install and upgrade VENs in your environment in the following scenarios:

- To install or upgrade RPM, Debian, Windows, and (starting in 23.2.20) AIX distributions of the VEN software. Other workload operating systems are not supported.

- The PCE and VEN versions are 18.2 and later.

VEN installation from the PCE does not affect any processes you might already have for installing or upgrading VENs directly on workloads, such as installation or activation/pairing with `illumio-ven-ctl`. Those processes can continue until and after you decide to use the PCE to install and upgrade VENs.

This topic primarily describes how to use the PCE web console to install and upgrade VENs. However, you can also use the Illumio Core REST API to upgrade (but not install) VENs. See the *REST API Developer Guide* for information.

VEN Library

Previously, VENs could be deployed from an external VEN repository (VEN repo) or by manually installing the VEN packages directly on your workloads.

From the 18.2.0 release onwards, the PCE can act as a repository for distributing, installing and upgrading the VEN software. The PCE can host multiple VEN versions, allowing you to evaluate and certify new versions of the VEN while continuing to deploy older versions in production.

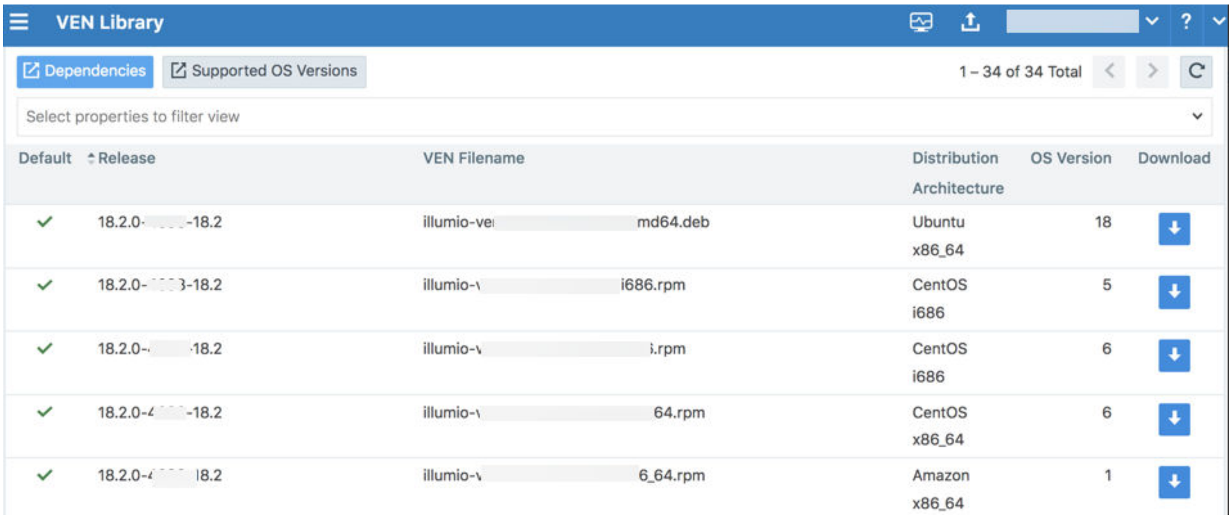
PCE-based installation and upgrade of VENs replaces the use of the external VEN repo, which is no longer supported for VEN version 18.2.0 or higher. A migration path is available for Illumio Secure Cloud customers and on-premises customers with VEN repos upgrading VENs to 18.2.0.

Using the VEN Library to install and upgrade VENs on your workloads has the following benefits:

- The VEN software bundle loaded on a PCE is replicated to all PCE core nodes.
- You can view VEN versions from the VEN Library page in the PCE web console.
- You can download software on workstations.
- Multiple versions of VEN software can exist on the PCE.
- You can specify an initial VEN version in pairing profiles.
- You can specify a default VEN version when the PCE has multiple VEN versions uploaded.
- You can add and remove VEN versions from the PCE.
- You can specify server and endpoint VEN bundles separately.
- You can use the PCE to upgrade all VENs or selected VENs in your environment.

After setting up the VEN software bundle using the PCE control interface `illumio-pce-ctl`, the VEN Library page is available in the **Support > VEN Library** menu. From this page, you can download individual VEN packages and view the dependencies and supported OS versions.

> **NOTE**
>
> You must set an initial VEN version when there is no system default version or an external repository has not been configured. If your PCE has existing pairing profiles created without versions, pairing will fail when use those un-versioned profiles.



Migration to PCE-Based VEN Library

Migration from the central VEN repo or an on-premises VEN repo to the VEN Library should be thoroughly planned and timed to not impact your current operations. Contact Illumio Customer Support for assistance.

PCE Runtime Parameters for PCE-based Installation

After you have migrated from any external VEN repo you might have, remove the following parameters from the PCE `runtime_env.yml` file:

- `ven_repo_url`
- `ven_repo_ips`

These parameters are not needed for the PCE-based installation of the VEN. They are deprecated and should no longer be used.

Workflow for VEN Library Setup

You do not have to make any configuration changes or other settings to enable the VEN Library on the PCE.

Loading the VEN bundle into the PCE VEN Library allows you to use the PCE web console or Illumio Core REST API to install and upgrade VENs in your environment.

To set up the VEN Library, perform the following high-level tasks:

1. Upload the VEN upgrade compatibility matrix to the PCE. See Upload VEN Upgrade Compatibility Matrix [185].

   > **NOTE**
   >
   > The compatibility matrix must be uploaded to the PCE before you upload any VEN software bundles or you will get an error.

2. Download the version of the VEN software bundle from the Illumio Support site.
   a. On the Illumio Support site (login required), go to **Software** > **Download** > **VEN – Download**.
   b. In the Download VEN page, select the radio button for the VEN version you want to set up. From the table, click the filename link for the "VEN Bundle for PCE-based deployment."

   > **TIP**
   > Illumio recommends that you verify the checksum of the VEN software bundle after downloading it.

   The VEN software for PCE-based deployment is a zipped tarball (tar file) of a version of VEN software for all supported workload platforms. This tarball is known as a VEN *software bundle*. The tar file downloads to your local drive.
3. Repeat step 1 for all the VEN versions you want to distribute to your workloads.

   Additionally, when Illumio releases new versions of the VEN software, plan on repeating these steps when you are ready to deploy that VEN version.
4. Copy or move the VEN software bundle tar file to a convenient directory on your PCE core node or to any system that your PCE can reach with HTTP, SFTP, or SCP.

   You do not need to unpack the VEN software bundle tar file.
5. Load the VEN software bundle into one of the PCE core node's VEN Library. From this node, the VEN software bundle is automatically copied to the other nodes.

   See Upload VEN Software Bundle into PCE VEN Library [186] for information.
6. Install or upgrade VENs:
   a. To install the VEN software on workloads, with the PCE web console, generate a pairing script. See Pairing Profiles and Scripts [194] for information.
   b. To upgrade all VEN workloads or selective workloads, use the PCE web console. See VEN Upgrade Using the PCE Web Console [213] for information.

Upload the VEN Upgrade Compatibility Matrix

> **NOTE**
> The compatibility matrix must be uploaded to the PCE before you upload any VEN software bundles or you will get an error.
>
> Alternatively, you can run the compatibility matrix upload command in one line with the command to install a VEN software bundle; for example:
>
> ```
> sudo -u ilo-pce illumio-pce-ctl ven-software-install
> bundle_path --compatibility-matrix matrix_file_path
> ```

As part of setting up the VEN Library in the PCE, you must upload the VEN upgrade compatibility matrix to the PCE. The compatibility matrix contains information about valid VEN upgrade paths and VEN to PCE version compatibility. To use the PCE web console and the Illumio Core REST API, you must upload this matrix for VEN upgrades to be successful.

In Supercluster, VENs are managed from the PCE they are paired to. You must upload VEN bundles and the compatibility matrix to each PCE.

The compatibility is a zipped tarball (tar file). You do not need to unpack the tar file to install it. The tarball contains a set of JSON files specifying the rules for upgrading VENs in your environment.

You can also view these VEN upgrade rules on the Illumio Support site (log in required). Go to **Software** > **Upgrade > VEN** – **Upgrade**. In the Upgrade VEN page, select your current VEN version and the version you want to upgrade to. Click **Find My Upgrade Path**.

IMPORTANT

Until you upload this file, you can only install VENs on workloads when the VEN version is the same as the version of the PCE managing those VENs. Attempting to upgrade a VEN version returns the message: "No valid upgrade paths were found for this release."

To install the compatibility matrix:

TIP

Illumio recommends that you verify the checksum of the compatibility matrix file after downloading it.

1. Download the VEN upgrade compatibility matrix tar file from the Illumio Support site (log in required).
   a. On the Support page, click **SOFTWARE** in the upper right of the page and then select **DOWNLOAD**. (Alternatively, you can click the Software tile on the body of the page.)
   b. On the Software page, click **Compatibility Matrix**, and then in the table that appears click the link for the compatibility tar file. The file downloads to your local drive.
2. Copy or move the tar file to a convenient directory on your PCE core node.
3. To upload the file to the PCE, run this command on the PCE:

```
sudo -u ilo-pce illumio-pce-ctl compatibility-matrix-install
matrix_file_path
```

Upload VEN Software Bundle into PCE

NOTE

Before you upload a VEN software bundle into the PCE, you must first have uploaded the VEN upgrade compatibility matrix. See Upload the VEN Upgrade Compatibility Matrix [185] for information.

Loading the VEN software bundle consists of running `illumio-pce-ctl` on the PCE command line to load the VEN software bundle into the PCE's VEN Library. The VEN Library is then replicated to the other PCE core nodes.

Loading the VEN software bundle into the PCE's VEN Library is what configures the PCE as the VEN installation and upgrade method.

In Supercluster, VENs are managed from the PCE they are paired to. You must upload VEN bundles and the compatibility matrix to each PCE.

You can only upload VEN software bundles into a PCE that are compatible with that PCE. For example, you cannot upload VEN version 21.5.0 software bundles into a PCE version 21.2.0.

To load a VEN software bundle:

1. Copy the downloaded VEN software bundles to a convenient location on your PCE core node or to any system that the PCE can access via HTTP, SFTP, or SCP.
2. To load the VEN software bundle, run the following command on the core node's command line.

   ```
   sudo -u ilo-pce illumio-pce-ctl ven-software-install bundle_path
   ```

   For example:

   ```
   sudo -u ilo-pce illumio-pce-ctl ven-software-install
   protocolAndFqdnOfVenBundleHost/nameOfVenSoftwareBundleFile.tar.bz2
   ```

   Where:
   - `bundle_path` is any of the following locations of the VEN software bundle tar file:
     - The absolute or relative path to the directory on the PCE
     - The HTTP URL to the host and file
     - The SFTP URL to the host and file
     - The SCP URL to the host
   - The filename of the VEN software bundle tar file uses the following format:
     `illumio-ven-repo-someVersionStamp.tar.bz2`
     Where `someVersionStamp` is the version and build number of the Illumio Core release.

Example

The following example assumes you have copied the VEN software bundle into `/var/tmp` on you PCE:

```
sudo -u ilo-pce illumio-pce-ctl ven-software-install /var/tmp/illumio-ven-
repo-someVersionStamp.tar.bz2
Reading /opt/pce_config/etc/runtime_env.yml.

Validating VEN release tarball file contents:
     Valid.
Deploying VEN release tarball to 'PCE's IP address' .

Committing tarball manifest information to database.
```

```
Are you sure you want to continue? [yes/no]: yes

Release version_of_bundle Successful.
```

HTTP and SCP Examples

These examples show HTTP and SCP URLs on the `illumio-pce-ctl ven-software-install` command:

- HTTP:

  ```
  sudo -u ilo-pce illumio-pce-ctl ven-software-
  install http://myVENrepohost.BigCo.com/myRepoDir/pcerepo/illumio-ven-
  repo- someVersionStamp.tar.bz2
  ```
- SCP:

  ```
  sudo -u ilo-pce illumio-pce-ctl ven-software-
  install scp://albert.einstein@myhost.BigCo.com:illumio-ven-repo-
  someVersionStamp.tar.bz2
  ```

Set Default VEN Version in Library

You can set a default version of the VEN software for all workloads or for selected pairing profiles. You can use both methods simultaneously. For example:

- Set a default VEN version for all workloads when you are ready to roll out that specific version.
- Create a separate pairing profile with a specific VEN version for test, evaluation, and certification before general rollout.

Set Default VEN Version for All Workloads

To define the default VEN version for all workloads, run this command on the PCE:

```
sudo -u ilo-pce illumio-pce-ctl ven-software-release-set-default release
```

Where:

`release` is a release identifier like 19.3.0-6623. The PCE uses the default release to determine what release of the VEN to install when you pair a VEN with a workload. You can override the default release for specific pairing profiles. To obtain release IDs, run the `sudo -u ilo-pce illumio-pce-ctl ven-software-releases-list` command.

Set Default VEN Version for Specific Pairing Profile

You can selectively set a VEN version for specific pairing profiles. The profiles that have a defined VEN version create pairing profiles that install that specific VEN version on the workload. Other pairing profiles that have no VEN version set are unaffected.

To set a pairing profile's VEN version, see Configure a Pairing Profile [196].

For information about pairing scripts, see Pairing Profiles and Prepare Scripts [191].

Remove a Release from the VEN Library

To remove a VEN version from the VEN Library on the PCE, run this command on the PCE:

```
sudo -u ilo-pce illumio-pce-ctl ven-software-release-delete release
```

Where:

`release` is a release identifier like 19.3.0-6623. To obtain release IDs, run the `sudo -u ilo-pce illumio-pce-ctl ven-software-releases-list` command.

> **IMPORTANT**
> To remove a VEN version from the PCE database, the PCE cannot be using that VEN version in pairing profiles and it cannot be set as the default VEN version for pairing with workloads. When your orgs no longer use that VEN version, the `ven-software-release-delete` command will remove the VEN software bundle from the PCE file system.

View the VEN Library in the PCE

The VEN loading process with `sudo -u ilo-pce illumio-pce-ctl ven-software-install` prints its success or failure when it completes. You can also verify the successful loading in the following ways:

- In the PCE web console, look at the VEN Library. Navigate to **Support > VEN Library** to see that the bundle has been loaded.
- On the PCE command line, run the following command:

  ```
  sudo -u ilo-pce illumio-pce-ctl ven-software-releases-list
  ```

PCE Maintenance for VEN Library

These are some points to consider about backing up and modifying your PCE cluster for the PCE-based deployment model.

About PCE Backups

Be sure that your backup included the PCE's VEN library and is not earlier than when you loaded the VEN software bundles into the PCE's VEN Library. If you restore from an earlier backup, you need to either reload the VEN library or redeploy from an existing core node.

About Complete PCE failure

In case of a catastrophic failure of the PCE cluster, after rebuilding or reinstalling the cluster, reload the VEN software bundles into a PCE core node's VEN library.

VEN-related Maintenance Commands on PCE

The `illumio-pce-ctl` control script has options for VEN maintenance, such as add new VEN software bundle, remove VEN version, and delete VEN version. See the `illumio-pce-ctl --help` details.

Some of the options for distributing VENs from the PCE show org-id, org-list, and other organization-related arguments. None of the organization-related options or arguments options are needed for distributing VENs from your on-premises PCE and do not need to be specified.

Set up Kerberos Authentication on PCE

You can configure the PCE and VEN to rely on authentication by a pre-configured Kerberos-based system, such as Microsoft Active Directory.

About Enabling Kerberos Authentication

1. Enable Kerberos on the PCE. See About Enabling Kerberos Authentication [190].
2. Configure Kerberos-based authentication of the VEN at installation. Illumio Core supports Kerberos authentication for Linux, Windows, Solaris, and AIX VENs.

   For information, see the following topics:
   - Kerberos for Linux VEN-to-PCE Authentication [228]
   - Kerberos for Windows VEN-to-PCE Authentication [223]

Requirement for Kerberos Authentication

For all VENs to be paired via Kerberos, be sure to add policy rules allowing access to the required Kerberos servers.

Obtain an activation code for the VEN. When installing the VEN by using the VEN CTL, you can use the activation code either during installation or after installation. For information about activation codes for the VEN, see About the VEN Activation Code [165].

About Kerberos Authentication on the PCE

To use Kerberos authentication to pair a workload, you must enable Kerberos authentication on the PCE. Kerberos authentication requires configuring the following parameters in your PCE's runtime_env.yml file:

PCE Runtime Environment File Parameter	Description
`kerberos_device_auth_service_name: kerberos_device_auth_keytab_file`	Kerberos authentication for VENs on devices. These parameters enable Kerberos authentication for the VENs and other devices and provide a Kerberos service name and keytab file. These parameters are only used when the PCE node's role is set to `agent_service`. • The `kerberos_device_auth_service_name` must contain the complete Service Principal Name (SPN); for example, `servicename/fqdn@realm`.
`kerberos_user_auth_service_name: kerberos_user_auth_keytab_file`	Kerberos user authentication for the Login Service. These parameters enable Kerberos authentication for the Login Service and provide a Kerberos service name and keytab. • The `kerberos_user_auth_service_name` must contain the complete Service Principal Name (SPN); for example, `servicename/fqdn@REALM`. Kerberos requires that the `REALM` be in all capital letters. • The `kerberos_user_auth_service_name` is the path to the PCE's Kerberos keytab file. Any key included in keytab can be used for authentication

`Prepare` Scripts

The `prepare` script is used for creating golden images to activate the VEN the first time the image is booted.

Prepare Golden Image for Workload Installation

Many organizations use "golden images" for faster deployment. When using a golden image to install a VEN, you have two options for pairing with the PCE:

- Option 1: [192] Use a modified version of the Illumio Core pairing script called `prepare` to ensure these golden images have the VEN pre-installed.
- Option 2: [192] Use the `illumio-ven-ctl` control script.

IMPORTANT

- **Avoid activating the VEN prematurely:** If you will enable your images with the prepare script, make sure to do so as the last step in your process for building the image. The prepare script takes effect at the next system boot, which means the VEN might be activated prematurely on the image itself. If you have other software to install on the image and the image requires reboot, the VEN is activated at once, which is not desirable.
- **Specify the correct activation code type:** There are two types of activation codes in the pairing profile available in the PCE web console: **one-time use** and **unlimited use**. Be sure to specify the correct type for your needs. For more information, see Configure Pairing Key Usage and Lifespan [198].

Option 1: Prepare the workload using the Pairing Profile/Pairing script

This option relies on a modified version of the Illumio Core pairing script called `prepare` to ensure these golden images have the VEN pre-installed.

1. In the PCE web console, create a pairing profile or select an existing pairing profile. For information, see Pairing Profiles and Scripts [194].
2. Copy the pairing script.
3. In the copy of the script, change all occurrences of `pair` to `prepare`.
4. Execute the modified script on the image.

The `prepare` script installs the VEN on the image. When the prepare script finishes, the VEN is stopped. The script configures the VEN to start the next time the workload is booted.

Option 2: Prepare the workload with illumio-ven-ctl

Use illumio-ven-ctl to place the image in `prepare` mode.

From a command line, execute **`illumio-ven-ctl`**, making sure to include the `prepare` argument and the `management-server` and `activation-code` information shown in the following examples.

Windows: use single dashes (-)

```
<VEN Installation Directory>\illumio-ven-ctl prepare -management-server
<pce_fqdn:port> -activation-code <activation_key>
```

Linux: use double dashes (--)

```
<VEN Installation Directory>/illumio-ven-ctl prepare -–management-server
<pce_fqdn:port> --activation-code <activation_key>
```

Auto Scaling Linux Workloads

The process for enabling Illumio Core to enable auto scaling for Linux workloads follows this general process:

1. Select an existing VM instance that you want to create a new instance for.
2. Inside the PCE web console, create a pairing profile (or select an existing pairing profile).
3. Copy and edit the Linux pairing script:

   ```
   rm -fr /opt/illumio_ven_data/tmp && umask 026 && mkdir -p /opt/
   illumio_ven_data/tmp && curl --tlsv1 "https://
   pce.example.com:8443/api/v18/software/ven/image?
   pair_script=pair.sh&profile_id=1" -o /opt/illumio_ven_data/tmp/pair.sh
   && chmod +x /opt/illumio_ven_data/tmp/pair.sh && /opt/
   illumio_ven_data/tmp/pair.sh --management-server pce.example.com:8443 --
   activation-code
   11a12969c511197eb7ae1e175b9b49382fe1bc011b2a2228c8a184cc6c9f75663325146e5
   d5ac7c5d
   ```

   Change all occurrences of the script where `pair.sh` is used and replace with `prepare.sh`. So that the script looks like this:

```
rm -fr /opt/illumio_ven_data/tmp && umask 026 && mkdir -p /opt/
illumio_ven_data/tmp && curl --tlsv1 "https://
pce.example.com:8443/api/v18/software/ven/image?
pair_script=prepare.sh&profile_id=1" -o /opt/illumio_ven_data/tmp/
prepare.sh && chmod +x /opt/illumio_ven_data/tmp/prepare.sh && /opt/
illumio_ven_data/tmp/prepare.sh --management-server pce.example.com:8443
--activation-code
11a12969c511197eb7ae1e175b9b49382fe1bc011b2a2228c8a184cc6c9f75663325146e5
d5ac7c5d
```

The `prepare.sh` script installs the VEN on the new workload and configures it so the VEN will start running as soon as the new workload is instantiated.

1. Run the modified script on the Linux instance.
2. Configure your auto scaling policy to use an image that contains the `prepare` script.

Auto Scaling for Windows Workloads

The process for enabling Illumio Core to enable auto scaling on Windows workloads follows this general process:

1. Select an existing VM instance that you want to create a new instance for.
2. In the PCE web console, create a pairing profile (or use an existing pairing profile).
3. Copy and edit the Windows pairing script:

   ```
   PowerShell -Command "& {Set-ExecutionPolicy -Scope process remotesigned
   -Force; Start-Sleep -s 3; Set-Variable -Name ErrorActionPreference
   -Value SilentlyContinue;
   [System.Net.ServicePointManager]::SecurityProtocol=[Enum]::ToObject([Syst
   em.Net.SecurityProtocolType], 3072); Set-Variable -Name
   ErrorActionPreference -Value Continue; (New-Object
   System.Net.WebClient).DownloadFile('https://pce.example.com:8443/api/v18/
   software/ven/image?pair_script=pair.ps1&profile_id=1', (echo
   $env:windir\temp\pair.ps1)); & $env:windir\temp\pair.ps1 -management-
   server pce.example.com:8443 -activation-code
   11a12969c511197eb7ae1e175b9b49382fe1bc011b2a2228c8a184cc6c9f75663325146e5
   d5ac7c5d;}"
   ```

   Change all occurrences of the script where `pair.ps1` is used and replace with `prepare.ps1`.
   So that the script looks like this:

   ```
   PowerShell -Command "& {Set-ExecutionPolicy -Scope process remotesigned
   -Force; Start-Sleep -s 3; Set-Variable -Name ErrorActionPreference
   -Value SilentlyContinue;
   [System.Net.ServicePointManager]::SecurityProtocol=[Enum]::ToObject([Syst
   em.Net.SecurityProtocolType], 3072); Set-Variable -Name
   ErrorActionPreference -Value Continue; (New-Object
   System.Net.WebClient).DownloadFile('https://pce.example.com:8443/api/v18/
   software/ven/image?pair_script=prepare.ps1&profile_id=1', (echo
   $env:windir\temp\prepare.ps1)); & $env:windir\temp\prepare.ps1
   -management-server pce.example.com:8443 -activation-code
   11a12969c511197eb7ae1e175b9b49382fe1bc011b2a2228c8a184cc6c9f75663325146e5
   d5ac7c5d;}"
   ```

The `prepare.ps1` script installs the VEN and configures it such that the VEN will start running as soon as the new workload is instantiated.

1. Run the modified script on the Windows instance.
2. Configure your auto scaling policy to use the prepared image.

VEN Installation & Upgrade Using VEN Library

The following topics describe how to install and upgrade the VEN by using the VEN Library in the PCE.

> **NOTE**
> Before you perform the tasks described in this section, the PCE must be set up with the VEN Library. For information, see VEN Library Setup in the PCE [182]

Pairing Profiles and Scripts

A pairing profile contains the configuration for workloads so that you can apply certain properties to workloads as they pair with the PCE, such as applying labels, setting workload policy state, and more.

When you configure a pairing profile, the pairing script contains a unique pairing key at the end of the script (an activation code) that identifies the VEN securely so it can authenticate with the PCE. The pairing key can be set to be used one time or several times, and you can configure its time and use limit.

In the PCE web console, you create a pairing profile with the characteristics to create a script called a pairing script to run on workloads. The pairing script installs the VEN software, activates it, and gets the workloads ready to accept security policy from the PCE. “Pairing” is also known as “installation and activation.”

Workflow for Using Pairing Profiles

Creating and using pairing profiles follows this general workflow:

1. Create a pairing profile.
2. Generate a pairing script.
3. Copy the script to the workload and run it.

The following conditions apply when installing VENs by using pairing profiles:

- An activation code/pairing key is required. In the PCE web console, you can specify either a single, one-time activation code or an unlimited, multi-use activation code.
- The pairing script is not absolutely required. It is an alternative to installing VEN software installation and activation with the VEN CTL (`illumio-ven-ctl`).

Which VEN Version is Installed

When installing and activating a VEN using a pairing script generated by a pairing profile:

- If you set a specific VEN version in the PCE web console for the pairing profile used to generate the pairing script, that specific VEN version is installed on the workload.
- Otherwise, if you have set a default VEN version for all workloads, that VEN version is installed on the workload. The PCE web console shows the current default VEN version in the "Initial VEN Version" dropdown. If no default VEN version is specified, the PCE uses the latest VEN version available in the VEN library.

The Default Pairing Profile

Item	Description
Name	Default
Labels	Role=<Blank> Application=<Blank> Environment=Production Location=<Blank>
Workload State	Visibility Only
Uses per Key	Unlimited
Maximum Key Age	Unlimited
Command Line Overrides	Unlocked (CLI can override anything)

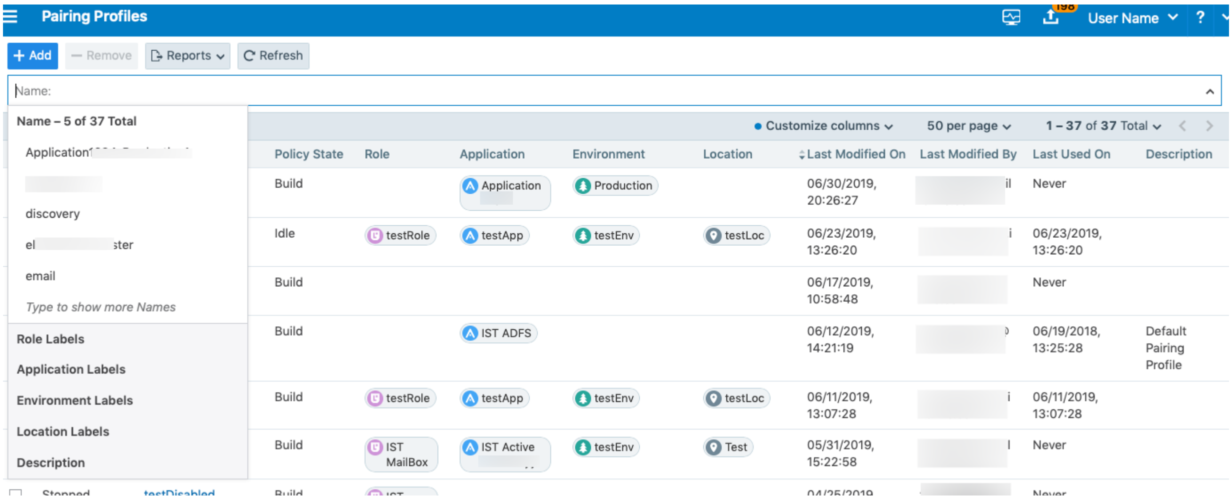
Last Pairing Key Generated and Last VEN Paired

The following information appears on the pairing profile details page:

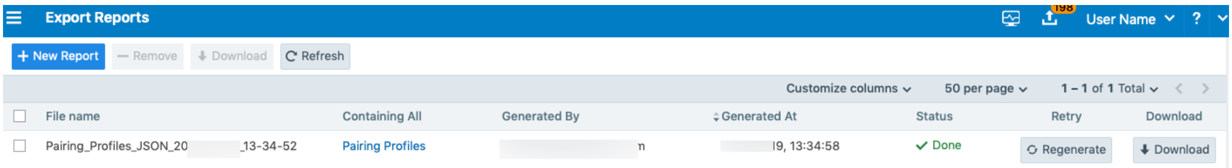
- The last time a pairing key was generated using this pairing profile.
- The last time a VEN was paired using this pairing profile.

Filter the Pairing Profiles List

You can filter the pairing profiles list using the properties filter at the top of the list. You can filter the list by entering a label type to show only those pairing profiles that use the selected labels. You can further filter the list by selecting specific properties of the pairing profiles. For example, you can filter the list by a pairing profile's name.



Click the **Reports** button and select JSON or CSV format to generate the pairing profiles report. Once generated, you can either click the download icon next to Reports to download the generated report or select **Reports > All Export Reports** to view the report details.



Configure a Pairing Profile

You can configure a pairing profile to set the initial workload policy state at the time of pairing. For example, you might want to pair workloads in the Visibility state so you can view network traffic to build policies before enforcing them.

On the other hand, if you are configuring an auto-scale policy and want to pair workloads automatically based on application demands, you can choose to have workloads paired in Full enforcement state.

To configure a pairing profile:

1. From the PCE web console menu, go to **Policy Objects > Pairing Profiles**.
2. Click **Add**.
3. Enter a name and description (optional) for the pairing profile.
4. Configure the following options for the pairing profile:
   Enforcement Mode for Policy

   You can choose one of the enforcement modes for workloads when you pair them:

   - **Idle:** A state in which the VEN does not take control of the workload's iptables (Linux) or WFP (Windows), but uses workload network analysis to provides the PCE relevant details about the workload, such as the workload's IP address, operating system, and traffic flows. This snapshot is taken every four hours.

> **NOTE**
> SecureConnect is not supported on workloads in the Idle policy state. If you activate SecureConnect for a rule that applies to workloads that are in both Idle and non-Idle policy states, it could impact the traffic between these workloads.

- **Visibility:** In the Visibility Only state, the VEN inspects all open ports on a workload and reports the flow of traffic between it and other workloads to the PCE. In this state, the PCE displays the flow of traffic to and from the workload, providing insight into the datacenter and the applications running in it. No traffic is blocked in this state. This state is useful when firewall policies are not yet known. This state can be used for discovering the application traffic flows in the organization and then generating a security policy that governs required communication.
- **Selective:** Segmentation rules are enforced only for selected inbound services when a workload is within the scope of a Selective Enforcement Rule.
- **Full:** Segmentation Rules are enforced for all inbound and outbound services. Traffic that is not allowed by a Segmentation Rule is blocked.

For information about how these enforcement modes impact workload security policy, see the "Enforcement States" and "Enforcement States for Rules" topics in the *Security Policy Guide*.

You can choose one of three modes for the traffic visibility for workloads:

- **Off (no detail):** The VEN does not collect any details about traffic connections. This option provides no Illumination detail and utilizes the least amount of resources from workloads. This state is useful when you are satisfied with the rules that have been created and do not need additional overhead from observing workload communication.
- **Blocked:** The VEN only collects the blocked connection details (source IP, destination IP, protocol and source port and destination port), including all packets that were dropped. This option provides less Illumination detail but also demands fewer system resources from a workload than high detail.
- **Blocked + Allowed:** The VEN collects connection details (source IP, destination IP, protocol and source port and destination port). This applies to both allowed and blocked connections. This option provides rich Illumination detail but requires some system resources from a workload.

Enforcement Node Type

The Enforcement Node Type setting allows you to specify the type of VEN that the pairing profile will activate:

- **Server VEN**: Select to activate a Server VEN with the pairing keys that this pairing profile will generate. When selected, only Server VENs can be activated with this pairing profile.
- **Endpoint VEN**: Select to activate a Endpoint VEN with the pairing keys that this pairing profile will generate. When selected, only Endpoint VENs can be activated with this pairing profile.
- **Specified during VEN activation (deprecated legacy option)**: When Illumio initially released the Endpoint solution, it wasn't yet possible to specify the Endpoint Enforcement Node Type in the PCE UI. Instead, after copying the pairing script from the PCE it was necessary to paste the script into a text editor and enter `-endpoint true` in an appropriate place in the script and then run the script. This option is for that use case.

Initial VEN Version

This is the VEN version that will be installed initially. You can edit the pairing profile later and select a different VEN version. If you don't select an initial VEN version, the pairing script installs the default VEN version. When using the default VEN version, VENs paired using the pairing line may have different versions depending on the default VEN version at pairing time. If you don't specify a default VEN version, VENs paired using the pairing line will use the latest compatible VEN version.

Assign Workload Labels

You can specify in the pairing profile which labels you want the PCE to assign to workloads when they are paired. Labels group workloads into logical categories for use in rulesets.

The PCE provides four types of labels:

- **Role**: The role or 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.
- **Application**: The type of application the workload is supporting (for example, HRM, SAP, Finance, Storefront).
- **Environment**: The stage in the development of the application (for example, production, QA, development, staging).
- **Location**: The physical, geographic location of the workload (for example, Germany, US, Europe, Asia).

For information on creating labels, see Labels and Label Groups in the *Security Policy Guide*.

Configure Pairing Key Usage and Lifespan

You can control the usage and lifespan of the pairing key in the pairing profile.

- **Uses Per Key**: Choose if you want the key generated from this pairing profile to be used an unlimited number of times or only once.
- **Key lifespan:** Specify how long you want the pairing key to be valid, either unlimited (forever) or for a specified time frame.

You can choose from these options to define how you want the pairing profile to be used:

- **Unlimited**: This option provides a pairing script that can be used to pair as many workloads in the organization as you want. Each user in an organization is given the pairing script from this profile regardless of the workload, the application the workload is a part of, the location of the workload (data center or country), or the environment (development, testing, QA, production). Unlimited use pairing profiles can present a security risk because they never change; however, if a pairing script is stolen, a workload could be paired into your environment by an untrusted user.

  > **IMPORTANT**
  >
  > Illumio recommends against configuring unlimited usage of pairing profiles from a security perspective. Instead, determine the appropriate lifespan for the pairing profile to minimize any security risk.

- **Custom Time Range**: If you do not want an unlimited use pairing profile, you can specify that the pairing profile can only be used to pair a workload one time, after which the pairing key cannot be used to pair more workloads.

Key Usage Requirements

The certification Key Usage requirements have changed to `CERT_DIGITAL_SIGNATURE_KEY_USAGE`, `CERT_KEY_ENCIPHERMENT_KEY_USAGE`, and `CERT_DATA_ENCIPHERMENT_KEY_USAGE`, so that the endpoint certificates can be set in the x.509 Windows environment.

Choose Command Line Overrides

For each of these Workload states, you can choose to either allow or block modifications to these settings when the pairing script is executed from the command line:

Workload Policy State

- **Lock Workload policy state assignment**: The policy state of the workloads being paired cannot be changed when the pairing script is run.
- **Allow Workload policy state assignment**: The policy state of the workloads being paired can be changed when the pairing script is run.

Label Assignment
- **Lock Label assignment**: This option prevents a user running the pairing script from assigning labels to workloads during pairing except for what is configured with the pairing profile.
- **Allow custom Labels**: This option permits the user running the pairing script to assign labels to the workloads during pairing using this pairing profile. Selecting this option selects all the Label checkboxes. You can deselect any before saving.

5. Click **Save**.

   The PCE saves the pairing profile and the page refreshes with the values you specified.

Working with a Pairing Profile

Open a pairing profile to display options to perform the following tasks.

Start/Stop Pairing

To enable or disable the pairing profile, click the pairing profile. The pairing profile details page opens and you can click **Stop Pairing** or **Start Pairing** .

Generate Key

Click **Generate Key** at the top of the page to create a unique pairing key that can be used with the pairing script. The key will not be accessible once you close the Pairing Profile details panel.

Every key that is generated under a pairing profile inherits the properties set in the pairing profile. The script can be used to pair Workloads, according to the parameters and time limits set in the pairing profile.

Delete a Pairing Profile

If you want to completely disable the pairing keys generated with a pairing profile, delete the pairing profile.

To delete a pairing profile and its pairing keys:

1. From the PCE web console menu, choose **Policy Objects** > **Pairing Profiles**.
2. Select the checkbox of the pairing profiles you want to delete.
3. Click **Remove**.

All pairing keys that were associated with this pairing profile are no longer valid for pairing workloads.

The same process applies if you are instantiating new VMs in vSphere or Microsoft Azure. You can use the modified Illumio PCE pairing script for preparing your new VMs for auto scaling.

Pairing Script

When you choose to install VENs on workloads by using the VEN Library, you create the pairing profile in the PCE web console and run the pairing script on workloads. (The CLI VEN installation method bypasses the pairing script altogether. In that method, you use the native tools of the operating system to install the package, followed by the VEN CLI to activate the VEN. See VEN Installation & Upgrade with VEN CTL [221].)

Add Options to the Pairing Script

You can add additional pairing options to the pairing profile, such as assign labels to the workload, set the workload policy state, and set logging levels for VEN traffic.

For the complete list of options to use with the pairing script, see VEN Pairing and Activation Command Reference [247].

Linux Pairing Script for VEN Library Installation

For example, if you want to add an Environment label to the workload, such as `--env Production`, include the option at the end of the pairing script as shown below.

```
rm -fr /opt/illumio_ven_data/tmp && umask 026 && mkdir
-p /opt/illumio_ven_data/tmp && curl "https://example.com:8443/api/v18/
software/ven/image?pair_script=pair.sh&profile_id=<pairing_profile_id>"
-o /opt/illumio_ven_data/tmp/pair.sh && chmod +x /opt/illumio_ven_data/tmp/
pair.sh && /opt/illumio_ven_data/tmp/pair.sh --management-server
example.com:8443 --activation-code <code> --env Production
```

Windows VEN Installation without the VEN Library

```
Set-ExecutionPolicy -Scope process remotesigned -Force; Start-Sleep
-s 3; (New-Object System.Net.WebClient).DownloadFile("https://repo.illum.io/
Z3JldGVsbHVuZHl0aGF0Y2hlcjg1dGgK/pair.ps1", "$pwd\Pair.ps1"); .\Pair.ps1
-repo-host repo.illum.io -repo-dir Z3JldGVsbHVuZHl0aGF0Y2hlcjg1dGgK/ -repo-
https-port 443 ` -management-server pce.example.com:8443 -activation-code
<code> –env Production; Set-ExecutionPolicy -Scope process undefined -Force;
```

Running RHEL 5 Pairing Script with cURL

When using a cURL command to run a pairing script for a RHEL 5 VEN, first perform the following additional configuration.

Downgrade the Transport Layer Security (TLS) version that the PCE uses for VEN-to-PCE communications. In RHEL 5, the normal default minimum version, TLS 1.2, can not be used. Set the default minimum version to TLS 1.0 by setting the parameter `min_tls_version` to `tls1_0` in the PCE configuration file `runtime_env.yml`. For details, see TLS Versions for Communications.

Update the CA certificate file on the RHEL 5 machine. Download the latest `cacert.pem` and append it to the `ca-bundle.crt` file. Use the following steps:

1. Download the latest `cacert.pem` to your RHEL 5 machine from https://curl.se/docs/caextract.html. If the download fails because the certificate is expired, perform the download

again on a different machine with a valid certificate and then copy the certificate to the initial machine.
2. Append the certificate using the following command:

```
sudo cat /tmp/cacert.pem >> /etc/pki/tls/certs/ca-bundle.cr
```
3. After changing the TLS version and the CA certificate, pair the VEN using a cURL command as described earlier in this section.

VEN Installation Using VEN Library in PCE

You can install VENs on Windows, Linux, and (starting in 23.2.20) AIX hosts by using the VEN Library in the PCE. To install VENs on Solaris hosts, or on AIX hosts before 23.2.20, download the VEN packages for those platforms and use the VEN CTL.

About VEN Installation, Pairing, and Upgrade

These are some general considerations for installing and upgrading VENs by using the VEN Library in the PCE web console.

- The target VENs can be in any state for installation or upgrade.
- Environment variables supported by the VEN CTL are not supported with when using the VEN Library to install VENs.
- Exact time to install or upgrade a VEN depends on many factors, including the speed of the workload hardware, the speed of its network connections, and its performance load.
- Before installation or upgrade, ensure that all the workloads on which you want to install or upgrade the VEN are online and reachable from the PCE. If they are not reachable when the installation or upgrade is running, they will be skipped.

About Installing VENs by Using the VEN Library

Installing the VEN by using VEN Library in the PCE web console is a two-step process. For each workload, perform the following high-level steps:

1. In the PCE web console, generate a pairing profile. Generating a pairing profile generates a pairing script.
2. Copy that pairing script to the workload and run it.

About Pairing Workloads

Pairing is the process of installing a VEN on a workload.

When you pair a workload, you run a script that installs the VEN on the workload. The VEN then reports detailed workload information to the PCE, such as all services running on the workload, all of its open ports, details about the operating system, workload location, and more.

When you configure and then provision rules, the PCE calculates and configures policy for each paired workload.

When you pair workloads, you can choose to place those workloads in one of these policy states:

To start the workload pairing process:

- Create a Pairing Profile [194] or use the default pairing profile
- Pair workloads

Enforcement Mode for Policy

You can choose one of the enforcement modes for workloads when you pair them:

- **Idle:** A state in which the VEN does not take control of the workload's iptables (Linux) or WFP (Windows), but uses workload network analysis to provides the PCE relevant details about the workload, such as the workload's IP address, operating system, and traffic flows. This snapshot is taken every four hours.

  > **NOTE**
  > SecureConnect is not supported on workloads in the Idle policy state. If you activate SecureConnect for a rule that applies to workloads that are in both Idle and non-Idle policy states, it could impact the traffic between these workloads.

- **Visibility:** In the Visibility Only state, the VEN inspects all open ports on a workload and reports the flow of traffic between it and other workloads to the PCE. In this state, the PCE displays the flow of traffic to and from the workload, providing insight into the datacenter and the applications running in it. No traffic is blocked in this state. This state is useful when firewall policies are not yet known. This state can be used for discovering the application traffic flows in the organization and then generating a security policy that governs required communication.
- **Selective:** Segmentation rules are enforced only for selected inbound services when a workload is within the scope of a Selective Enforcement Rule.
- **Full:** Segmentation Rules are enforced for all inbound and outbound services. Traffic that is not allowed by a Segmentation Rule is blocked.

For information about how these enforcement modes impact workload security policy, see the "Enforcement States" and "Enforcement States for Rules" topics in the *Security Policy Guide*.

You can choose one of three modes for the traffic visibility for workloads:

- **Off (no detail):** The VEN does not collect any details about traffic connections. This option provides no Illumination detail and utilizes the least amount of resources from workloads. This state is useful when you are satisfied with the rules that have been created and do not need additional overhead from observing workload communication.
- **Blocked:** The VEN only collects the blocked connection details (source IP, destination IP, protocol and source port and destination port), including all packets that were dropped. This option provides less Illumination detail but also demands fewer system resources from a workload than high detail.
- **Blocked + Allowed:** The VEN collects connection details (source IP, destination IP, protocol and source port and destination port). This applies to both allowed and blocked connections. This option provides rich Illumination detail but requires some system resources from a workload.

VEN Package Format Changes (Windows only)

In Illumio Core 21.2.1, the Windows VEN installer switched from MSI to EXE package format. Customers using the PCE-based VEN deployment must take an extra step for the transition.

Specifically, Illumio Core customers running older MSI-based Windows VENs must upgrade to 19.3.6+H1-VEN or 21.2.0+H2-VEN before upgrading to their VENs to 21.2.1 or a later version. Older VEN versions are not EXE package aware and cannot upgrade themselves without manual intervention on the CLI.

For the detailed steps required for this transition, see New Windows VEN Installer Starting with 21.2.1, Knowledge Base Article 3561 (login required).

Pair a Windows Workload

Pairing a workload requires running the pairing script on it to install the VEN.

When you first log into the PCE web console, a default pairing profile containing a pairing script is provided so you can begin pairing workloads. You also have the option to create a new pairing profile if you want to configure your own workload pairing settings.

> **NOTE**
>
> When the Illumio VEN is installed on a Windows workload and paired with the PCE in the Full enforcement policy state, all Internet Group Management Protocol (IGMP) traffic will be blocked unless you add a rule to allow it. Windows servers typically use IGMP for things like Windows Internet Name Service (WINS), Windows Deployment Services (WDS), IGMP Router/Proxy mode, and Network Load Balancing (NLB) in multicast mode.

> **WARNING**
>
> Your pairing script to install a Windows VEN on a workload cannot contain colons in the values for command options. Including a colon in a option value causes VEN activation to fail. For example, including the following values in the `-role` option, causes VEN activation to fail:
>
> ```
> -role "R: UNKNOWN" -app "A:UNKNOWN" -env "E: UNKNOWN"
> ```
>
> Activation fails because Windows uses the colon as a special character and cannot interpret the value even when you include quotation marks around the value.

> **NOTE**
>
> You must be logged in as an Administrator user on the Windows workload to run the Illumio pairing script.

To pair a workload on Windows:

1. From the PCE web console menu, go to **Servers & Endpoints > Workloads.**
2. Click **Add** > **Pair Workload with Pairing Profile**.
3. From the drop-down list, select a pairing profile. The list contains the default pairing profile and the pairing profiles you've added.

   To create a new pairing profile, see Configure a Pairing Profile [196].
4. From the Windows OS Pairing Script field, copy the Windows pairing script.
5. On the Windows workload you want to pair, open the Windows PowerShell as an Administrator user.
6. Paste the pairing script you copied into the PowerShell command prompt.

   The following output appears:

```
PS C:\Program Files> PowerShell -Command "& {Set-ExecutionPolicy -Scope
process remotesigned -Force; Start-Sleep -s 3; Set-Variable -Name
ErrorActionPreference -Value SilentlyContinue;
[System.Net.ServicePointManager]::SecurityProtocol=[Enum]::ToObject([Syst
em.Net.SecurityProtocolType], 3072); Set-Variable -Name
ErrorActionPreference -Value Continue; (New-Object
System.Net.WebClient).DownloadFile('https://pce.example.com:8443/api/v18/
software/ven/image?pair_script=pair.ps1&profile_id=1',
'.\Pair.ps1'); .\Pair.ps1  -management-server example.com:8443
-activation-code <code>;}"

             Installing Illumio
             ------------------
Setting up Illumio Repository ........
Retrieving Illumio Package ...........
Installing Illumio Package ...........
Validating Package Installation ......
Pairing with Illumio .................
             Pairing Status
             --------------
Illumio Package installation ......SUCCESS
Pairing Configuration exists ......SUCCESS
VEN Manager Service running .......SUCCESS
Master Configuration retrieval ....SUCCESS
VEN Configuration retrieval .......SUCCESS
VEN has been SUCCESSFULLY paired with Illumio

PS C:\Program Files> cd .\Illumio\
PS C:\Program Files\Illumio> .\illumio-ven-ctl.ps1 status
Service venAgentMgrSvc:                 Running
Service venPlatformHandlerSvc:          Running
Service venVtapServerSvc:               Running
Service venAgentMonitorSvc:             Running
Service venAgentMgrSvc:                 Enabled
Service venPlatformHandlerSvc:          Enabled
Service venVtapServerSvc:               Enabled
Service venAgentMonitorSvc:             Enabled
Agent State: illuminated
```
7. To view the workload after it has finished pairing, choose **Servers & Endpoints > Workloads** from the PCE web console menu.

Unpair a Windows Workload

Unpairing is the process of uninstalling the VEN from a workload so that it no longer reports any information to the PCE and can no longer receive any policy information. After uninstalling the VEN, the PCE will no longer maintain control over the workload.

> **NOTE**
> After you remove a workload from the PCE using the PCE web console or REST API (but not by using the VEN CTL), it remains in policy computation and can continue to appear (for example, in auto complete fields or API responses) until the VEN confirms that it has been uninstalled or a one-hour delay has passed.

Windows Unpair Options

Carefully consider the security state you want to return the Windows workload to after the VEN is uninstalled:

- **Remove Illumio policy**: (Recommended) Remove Illumio Windows Filtering Platform (WFP) filters and activate the Windows firewall.
- **Open all ports**: Uninstalls the VEN and leaves all ports on the workload open to traffic.
- **Close all ports except remote management**: Temporarily allow only RDP/3389 and WinRM/5985, 5986 until the system is rebooted.

> **NOTE**
> When you unpair a workload that uses a Windows GPO policy, the GPO policy overrides local WFP rules.

To unpair a Windows workload:

1. From the PCE web console menu, go to **Servers & Endpoints > Workloads**.
2. Select the Windows workload you want to unpair. You can select as many workloads as you want to unpair.
3. Click **Unpair**.
4. Select an unpair option, and then click **Remove**.

Remove VEN Using Windows Control Panel

You can also use the Windows Control Panel Programs and Features utility to remove the VEN. When you use this utility to remove the VEN, the Windows workload is returned to the “Recommended” state.

Pair a Linux Workload

Pairing a workload requires running the pairing script on it to install the VEN.

When you first log into the PCE web console, a default pairing profile and corresponding pairing script is provided. You can use the default pairing profile to assess installing Linux VENs using the VEN Library.

Or, you can create a new pairing profile to configure your own workload pairing settings. Ultimately, you should create your own Linux pairing profile to designate your own workload pairing settings.

Before you begin, open an SSH connection to the workload you want to pair.

> **NOTE**
> You must be logged in as a user with root permissions to run the Illumio pairing script.

To pair a workload on Linux:

1. From the PCE web console menu, go to **Servers & Endpoints > Workloads**.
2. Click **Add** > **Pair Workload with Pairing Profile**.
3. From the drop-down list, select a pairing profile. The list contains the default pairing profile and the pairing profiles you've added.
   To create a new pairing profile, see Configure a Pairing Profile [196].
4. From the Linux/Unix OS Pairing Script field, copy the Linux pairing script.
5. In a shell window on the workload you want to pair, paste the script you copied from the pairing profile.
   The following output appears:

```
[root@ven-rhel tmp]# rm -fr /opt/illumio_ven_data/tmp
&& umask 026 && mkdir -p /opt/illumio_ven_data/tmp
&& curl --tlsv1 "https://pce.example.com:8443/api/v18/software/ven/
image?pair_script=pair.sh&profile_id=1" -o /opt/illumio_ven_data/tmp/
pair.sh && chmod +x /opt/illumio_ven_data/tmp/pair.sh && /opt/
illumio_ven_data/tmp/pair.sh --management-server pce.example.com:8443 --
activation-code <code>

  % Total  % Received  % Xferd  Average Speed  Time  Time  Time  Current
Dload  Upload   Total   Spent    Left  Speed
  100  40891  100  40891  0  0  93526  0 --:--:-- --:--:-- --:--:--
93572

             Installing Illumio
             ------------------
Retrieving Illumio Packages [x86_64][CentOS][7.4] ..........
Validating sha256 .................
Installing Illumio Packages ..........
EXPECTED_VERSION: <ven_version>
INSTALLED_VERSION: <ven_version>
Starting Illumio processes .................
Starting illumio-control:
- Environment Setting up Illumio VEN Environment:         [  OK  ]
```

```
- venAgentMgr Starting venAgentMgr:                        [  OK  ]
- IPSec Starting IPSec: feature not enabled                [  OK  ]
- venPlatformHandler Starting venPlatformHandler:          [  OK  ]
- venVtapServer Starting venVtapServer:                    [  OK  ]
- venAgentMonitor Starting venAgentMonitor:                [  OK  ]
Pairing with Illumio .................

               Pairing Status
               --------------
Pairing Configuration exists ......SUCCESS
VEN Manager Daemon running ........SUCCESS
Master Configuration retrieval ....SUCCESS
VEN Configuration retrieval .......SUCCESS
VEN has been SUCCESSFULLY paired with Illumio

[root@ven-rhel tmp]#
```

- To view the workload after it has finished pairing, go to **Servers & Endpoints > Workloads**.

Unpair a Linux Workload

Unpairing is the process of uninstalling the VEN from a workload so that it no longer communicates with the PCE. Once unpaired, the PCE no longer controls the workload.

> **NOTE**
> After you remove a workload from the PCE using the PCE web console or REST API (but not by using the VEN CTL), it remains in policy computation and can continue to appear (for example, in auto complete fields or API responses) until the VEN confirms that it has been uninstalled or a one-hour delay has passed.

Linux Unpair Options

Carefully consider the security state you want to return the Linux workload to after the VEN is uninstalled.

- **Remove Illumio policy**: (Recommended) Revert firewall rules to the state previous to pairing.
- **Open all ports**: Uninstalls the VEN and leaves all ports on the workload open to traffic.
- **Close all ports except remote management**: Temporarily allow only SSH/22 traffic until system is rebooted.

> **NOTE**
> If the workload you are unpairing is offline during the unpairing process, the workload may still appear in the workloads list in the PCE web console, even though the workload has been unpaired. The unpaired workload will be removed within 30-35 minutes.

To unpair a Linux Workload:

1. From the PCE web console menu, go to **Servers & Endpoints** > **Workloads**.
2. Select the Linux workload you want to unpair.
3. Click **Unpair**.
4. Select an unpair option, and then click **Remove**.

Pair an AIX Workload

(Available starting in release 23.2.20.)

Pairing a workload requires running the pairing script on it to install the VEN.

When you first log into the PCE web console, a default pairing profile and corresponding pairing script is provided. You can use the default pairing profile to assess installing AIX VENs using the VEN Library.

Or, you can create a new pairing profile to configure your own workload pairing settings. Ultimately, you should create your own AIX pairing profile to designate your own workload pairing settings.

Before you begin, open an SSH connection to the workload you want to pair.

NOTE

You must be logged in as a user with root permissions to run the Illumio pairing script.

Before pairing AIX workloads:

The `curl` utility is not available by default on AIX platforms, so before using a pairing script to install AIX workloads, you must install `curl` from the AIX Toolbox for Open Source Software at https://www.ibm.com/support/pages/node/883796. It may also be necessary to set the environment variable `CURL_CA_BUNDLE=/var/ssl/certs/ca-bundle.crt`.

To pair a workload on AIX:

1. From the PCE web console menu, go to **Servers & Endpoints > Workloads**.
2. Click **Add** > **Pair Workload with Pairing Profile**.
3. From the drop-down list, select a pairing profile. The list contains the default pairing profile and the pairing profiles you've added.

   To create a new pairing profile, see Configure a Pairing Profile [196].
4. From the AIX Pairing Script field, copy the AIX pairing script.
5. In a shell window on the workload you want to pair, paste the script you copied from the pairing profile.

   The following output appears:

```
[root@ven-rhel tmp]# rm -fr /opt/illumio_ven/tmp && umask 026 && mkdir
-p /opt/illumio_ven/tmp && curl --tlsv1 "https://
pce.example.com:8443/api/v26/software/ven/image?
pair_script=pair.aix.sh&profile_id=8" -o /opt/illumio_ven/tmp/pair.sh &&
chmod +x /opt/illumio_ven/tmp/pair.sh && /opt/illumio_ven/tmp/pair.sh --
management-server pce.example.com:8443 --activation-code <code>
        Installing Illumio
        ------------------
Retrieving Illumio Packages [powerpc][AIX][7.2] ..........
Validating sha256 .................
Installing Illumio Packages ..........
/opt/illumio_ven/tmp/illumio-ven.23.2.20.3.bff
+-----------------------------------------------------------------------
-----+
            Pre-installation Verification...
+-----------------------------------------------------------------------
-----+
Verifying selections...done
Verifying requisites...done
Results...

SUCCESSES
---------
  Filesets listed in this section passed pre-installation verification
  and will be installed.

  Selected Filesets
  -----------------
  illumio-ven 23.2.20.3                       # Illumio VEN Agent

  << End of Success Section >>

+-----------------------------------------------------------------------
-----+
            BUILDDATE Verification ...
+-----------------------------------------------------------------------
-----+
Verifying build dates...done
FILESET STATISTICS
------------------
    1  Selected to be installed, of which:
        1  Passed pre-installation verification
   ----
    1  Total to be installed

+-----------------------------------------------------------------------
-----+
            Installing Software...
+-----------------------------------------------------------------------
-----+

installp:  APPLYING software for:
    illumio-ven 23.2.20.3

VEN PRE INSTALL: /opt, Arg [illumio-ven.pre_i]
```

```
VEN POST INSTALL: /opt, Arg [./illumio-ven.post_i]
Finished processing all filesets.  (Total time:  5 secs).

+-----------------------------------------------------------------------
-----+
                Summaries:
+-----------------------------------------------------------------------
-----+

Installation Summary
--------------------
Name                        Level           Part        Event
Result
-----------------------------------------------------------------------
------
illumio-ven                 23.2.20.3       USR         APPLY
SUCCESS
Starting Illumio processes ................
Starting illumio-control:
 - Environment Setting up Illumio VEN Environment:    ...done
 - venAgentMgr Starting venAgentMgr    ...done
 - IPSec Starting IPSec:  feature not enabled    [OK]
 - venPlatformHandler Starting venPlatformHandler    ...done
 - venVtapServer Starting venVtapServer    ...done
 - venAgentMonitor Starting venAgentMonitor:    ...done

Pairing with Illumio ................

              Pairing Status
               --------------
Pairing Configuration exists ......SUCCESS
VEN Manager Daemon running ........SUCCESS
Master Configuration retrieval ....SUCCESS
VEN Configuration retrieval .......SUCCESS

VEN has been SUCCESSFULLY paired with Illumio
```

- To view the workload after it has finished pairing, go to **Servers & Endpoints > Workloads**.

Unpair an AIX Workload

(Available starting in release 23.2.20.)

Unpairing is the process of uninstalling the VEN from a workload so that it no longer communicates with the PCE. Once unpaired, the PCE no longer controls the workload.

> **NOTE**
> After you remove a workload from the PCE using the PCE web console or REST API (but not by using the VEN CTL), it remains in policy computation and can continue to appear (for example, in auto complete fields or API responses) until the VEN confirms that it has been uninstalled or a one-hour delay has passed.

AIX Unpair Options

Carefully consider the security state you want to return the AIX workload to after the VEN is uninstalled.

- **Remove Illumio policy**: (Recommended) Revert firewall rules to the state previous to pairing.
- **Open all ports**: Uninstalls the VEN and leaves all ports on the workload open to traffic.
- **Close all ports except remote management**: Temporarily allow only SSH/22 traffic until system is rebooted.

> **NOTE**
> If the workload you are unpairing is offline during the unpairing process, the workload may still appear in the workloads list in the PCE web console, even though the workload has been unpaired. The unpaired workload will be removed within 30-35 minutes.

To unpair an AIX workload:

1. From the PCE web console menu, choose **Servers & Endpoints** > **Workloads**.
2. Select the AIX workload you want to unpair.
3. Click **Unpair**.
4. Select an unpair option, and then click **Remove**.

Ignored Interfaces

You can now set interfaces from “managed” to “ignored” in the PCE web console. Use this option when you want the workload to ignore visibility and enforcement specific interfaces; for example, on the interconnected interfaces of database clusters, such as Oracle RAC. You can set one or more interfaces to “ignored” during pairing. Using this setting causes the first downloaded firewall to ignore those interfaces. An ignored interface is not be included in the policy configuration and traffic will flow uninterrupted through it without any change in latency. You can see which interfaces are marked as “ignored” on the Workload details page.

> **IMPORTANT**
> Illumio recommends that you designate all private (non-routable) interfaces as ignored interfaces.

To set an interface to ignored:

1. From the PCE web console menu, go to **Servers & Endpoints > Workloads**.
2. Select the workload that has interface you want to ignore.
3. Click **Edit.**
4. In the Network Interfaces section, change the interface to Ignored in the **PCE Action** drop-down menu.

5. Click **Save**.

> **NOTE**
> After you set an interface to Ignored, it will not be included in policy configuration provided by the PCE and traffic will continue to flow uninterrupted through that interface .

VEN Installation Troubleshooting

This topic provides information about troubleshooting VEN installation issues you might encounter when using the PCE web console. For general troubleshooting, see "VEN Troubleshooting" in VEN Administration Guide.

Troubleshoot Pairing Errors

If you execute the pairing script and it fails, the system displays an error message indicating failure. One such error may be:

```
Unable to retrieve Illumio VEN configuration from management server
illumio.example.com:8443 in a timely fashion.
```

If the script fails to install the VEN, the following output displays:

```
Pairing Status
             --------------
Secureware Package installation ......SUCCESS
Pairing Configuration exists .........SUCCESS
VEN Manager Daemon running ...........SUCCESS
Master Configuration retrieval .......SUCCESS
VEN Configuration retrieval ..........FAILED
2014-01-23T18:43:50Z AgentManager 13088
Verify activation code and retry pairing
Workload has FAILED pairing with Illumio
```

Possible Causes of Failure

Failures may be caused by a problem with the VEN communicating with PCE, or by exceeding the hard license limit, or by an attempt to install an Endpoint VEN or Server VEN on an operating system that doesn't support that VEN type.

- The hard license limit has been exceeded.
- An invalid pairing profile is being used.
- The wrong pairing script was copied and run on the workload.
- In the pairing script section, the pairing profile contains invalid pairing keys.
- The pairing profile was disabled.
- The use limit for the pairing key has been exceeded.
- The pairing key has expired.
- Wrong Enforcement Node Type setting (**Servers & Endpoints > Pairing Profiles**):
  - The pairing profile is for a server node (not an endpoint node) but the profile includes the parameter `-endpoint true`.

- The pairing profile used to activate a macOS VEN is set to Server VEN.
- The pairing profile used to activate a Linux VEN is set to Endpoint VEN.

Remediate

- Check the pairing profile in the PCE web console to try to verify the cause of the failure.
- Check the PCE Logs and VEN Logs.
- The VEN Library is available in the PCE web console. You can download VEN software bundle and also view the dependencies and supported OS versions.

VEN Upgrade Using VEN Library in PCE

You can use your PCE cluster as a centralized mechanism for upgrading VENs in your environment.

From the 20.2.0 release on, you can upgrade one or more VENs by using the PCE web console. From the PCE web console menu, go to the **Workloads and VENs > VENs > Upgrade**.

You can also use the Illumio Core REST API to upgrade (but not install) VENs. See the *REST API Developer Guide* for information.

NOTE

Before you use this feature, you must set up the VEN Library in the PCE. See VEN Library Setup in the PCE [182] for information.

If you are an Illumio Cloud customer, Illumio Operations set up the VEN Library in the PCE.

IMPORTANT

For Illumio Core Cloud customers, the VEN Library only provides the option to upgrade to VENs that Illumio designated as a Long Term Support (LTS) release. See Versions and Compatibility in the Illumio Support Portal (login required) for information.

About VEN Upgrade

You can upgrade all VENs, upgrade a selected subset of VENs, or upgrade all VENs that match a set of filters. After you confirm an upgrade from the PCE web console, the VEN will download the new VEN image from the PCE and upgrade itself. The upgrade on the workload host only takes on average a few minutes.

If the VEN does not successfully upgrade within a certain amount of time (this timeout is configurable), the upgrade will time out and the PCE will put the VEN in a warning state.

However, in most cases, the upgrade will complete within this window. To clear this warning, just start another upgrade on the VEN.

You can upgrade up to 25,000 VENs in a single PCE region. Selecting this large a number of VENs in one upgrade will result in some CPU and memory spikes and increased network bandwidth, because the VENs will be communicating with the PCE to request new firewalls and downloading new software versions.

The VEN upgrade feature includes upgrade validation. The PCE will validate that the VEN upgrade path is allowed and that the version of the VEN you are upgrading to is compatible with the version of the PCE. If you attempt to upgrade VENs to a version incompatible with the PCE or Illumio does not support that upgrade path, the PCE web console provides feedback on which VENs can be upgraded.

VEN Package Format Changes

Starting with the 21.2.1 release, the Windows VEN installer switched from MSI to EXE package format. This package format change primarily affects Illumio Core On-Premises customers running older MSI-based Windows VENs. For information about using the VEN Library in the PCE to install Windows VENs on workloads, see Pair a Windows Workload.

Prerequisites and Limitations for VEN Upgrade

Prerequisites

- The PCE must be a version 20.2.0 or later.
- The VENs selected for upgrade must be version 18.2.0 or later.
- The VEN Library must be set up in the PCE. See Upload VEN Software Bundle into PCE VEN Library [186] for information.
- The VEN upgrade compatibility matrix must be installed on the PCE. See Upload VEN Upgrade Compatibility Matrix [185] for information.

Limitations

- You must update the VENs in your environment using a supported upgrade path.
- You cannot upgrade a VEN to a later version than the PCE's current version.
- The VEN Upgrade feature does not support upgrading Solaris VENs or upgrading the C-VEN. In releases earlier than 23.2.20, it does not support upgrading AIX VENs.
- If a VEN has been installed with custom RPM installation options, you cannot use the VEN upgrade feature to upgrade it.

  For example, you cannot upgrade VENs installed with a custom `--prefix` option because options like that aren't persisted when a VEN is upgraded from the PCE, and the VEN will be installed in the default directory. This outcome might cause data loss or operational issues with the VEN, depending on your environment.
- You cannot use the feature to downgrade a VEN to an earlier version.
- Using the PCE CLI to upgrade VENs is no longer supported in Illumio Core 21.2.0 and later releases.
- If you installed the VEN with the VEN CTL and packaging CLI and customized installation options (such as, a custom installation directory or alternate VEN user), you cannot later upgrade the VEN by using the VEN Library in the PCE. You must upgrade the VEN using the workload's OS package upgrade process.
- If you are upgrading a Windows VEN while it's in Full Enforcement and the Illumio ilowfp driver provided in the upgrade differs from the driver that's already installed, the driver

will be unloaded and reloaded, which will block all new connections during the upgrade (existing connections are not affected). **To avoid network disruption during production hours, Illumio recommends that you upgrade VENs only during a maintenance window.**.

VEN Package Format Changes (Windows only)

Starting with the Illumio Core 21.2.1 release, the Windows VEN installer switched from MSI to EXE package format. This package format change affects Illumio Core On-Premises customers running older MSI-based Windows VENs.

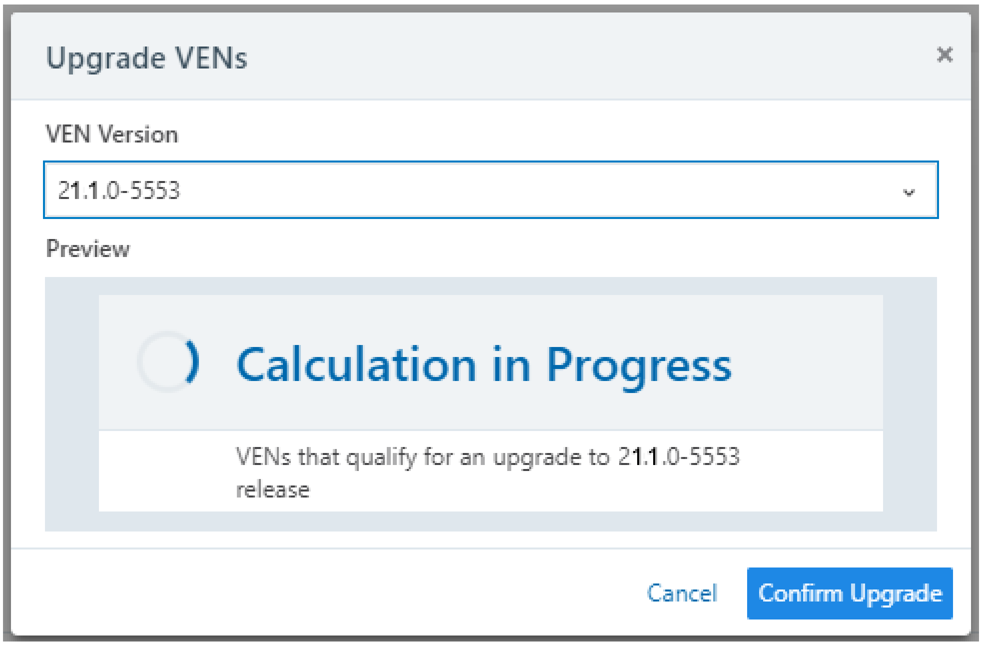
Customers using the VEN Library in the PCE to install or upgrade VENs must take an extra step for the transition. Specifically, Illumio Core customers running older MSI-based Windows VENs must upgrade to Illumio Core 19.3.6+H1-VEN or 21.2.0+H2-VEN before upgrading to their VENs to 21.2.1 or a later version. The 21.2.0+H2-VEN release contained the necessary VEN changes to handle the transition in the VEN packaging from MSI to EXE.

For the detailed steps required for this transition, see New Windows VEN Installer Starting with 21.2.1, Knowledge Base Article 3561 (login required).

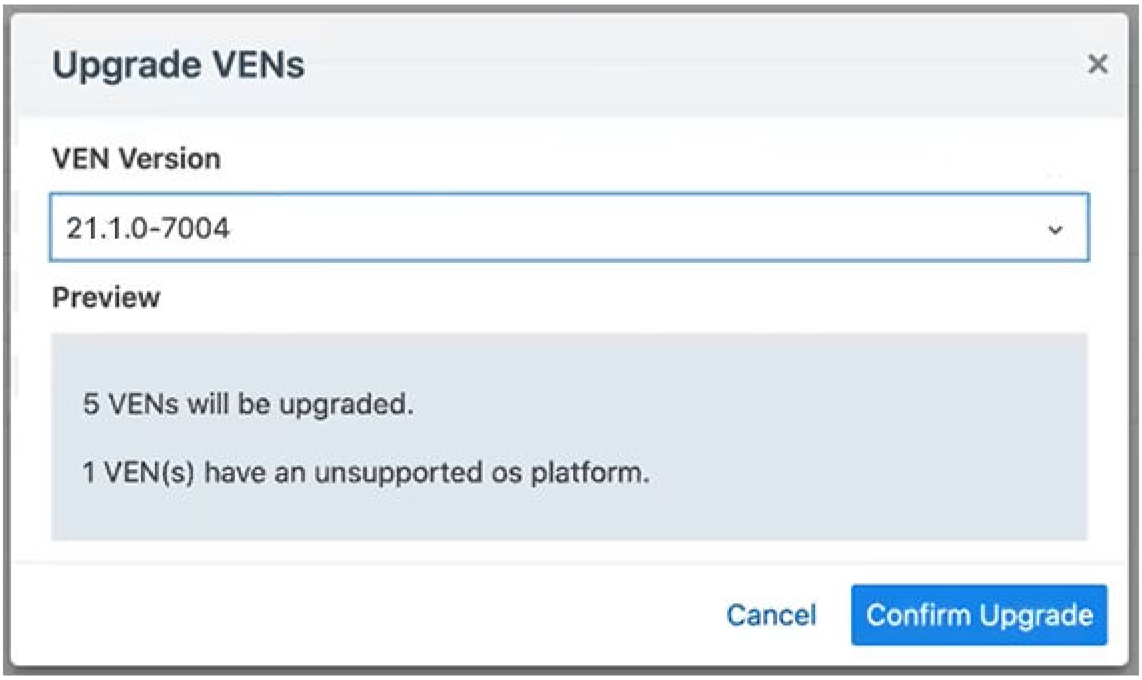
Upgrade All VENs to the Current Version

When you select "Upgrade All," the PCE upgrades all your deployed VENs and not just the VENs that appear in that page of the list (for example, you have 10 pages in the list, the VENs in all 10 pages are upgraded.)

1. From the PCE web console menu, choose **Workloads and VENs > VENs**. The Workloads and VENs – VENs page appears.
2. From the **Upgrade** drop-down menu, choose **Upgrade All**.
   The Upgrade VENs dialog box appears. By default, the most current VEN version in the VEN Library is selected.
3. (Optional) To upgrade to a version other than the current version, select the VEN version from the drop-down menu.
   The PCE calculates the scope of the upgrade.



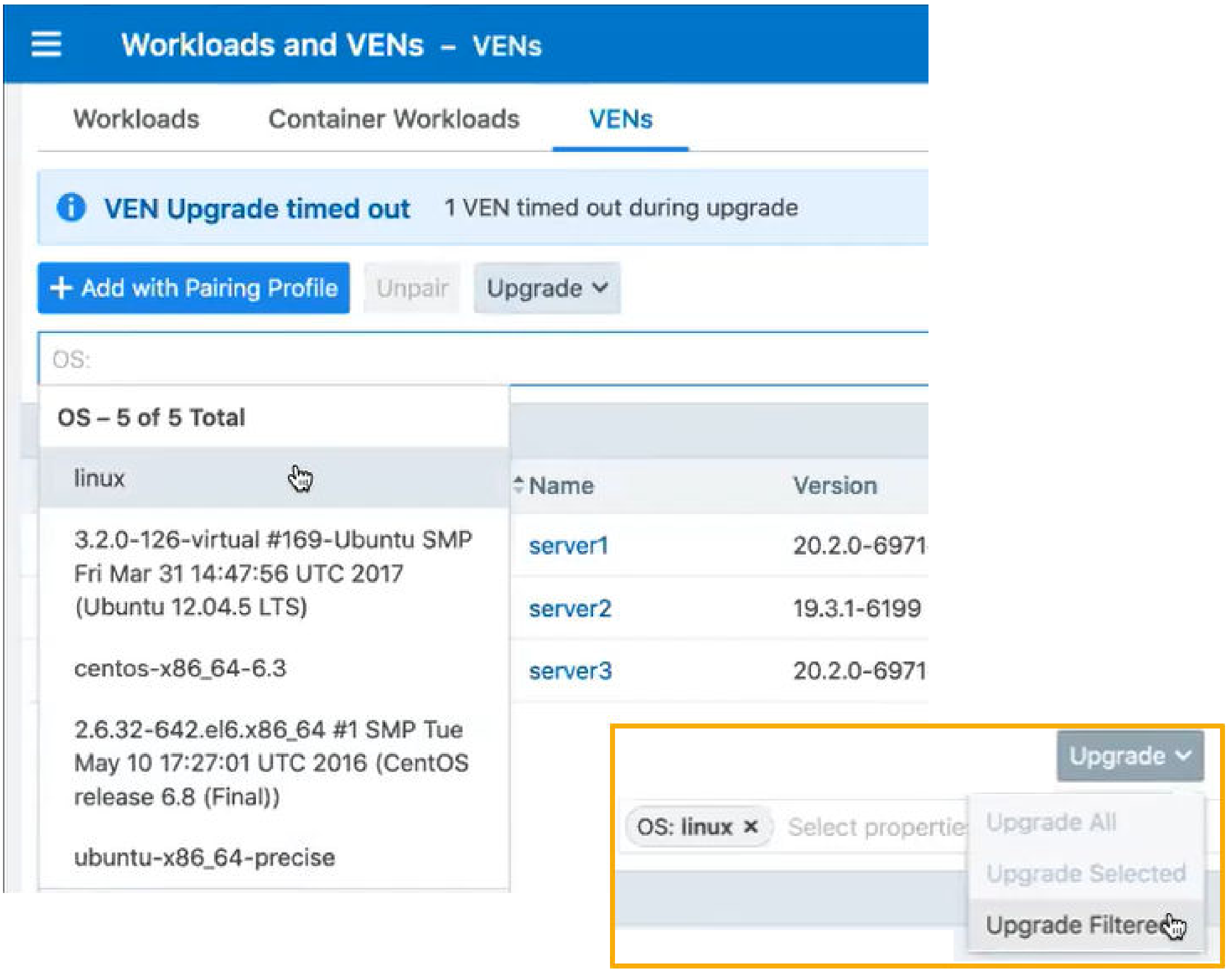
   When the calculation is complete, the dialog box refreshes and informs you which VENs in your environment can be upgraded to the selected version. In this example, one VEN has an unsupported OS platform because it's a VEN running on Solaris, which cannot be upgraded using this feature in this release.



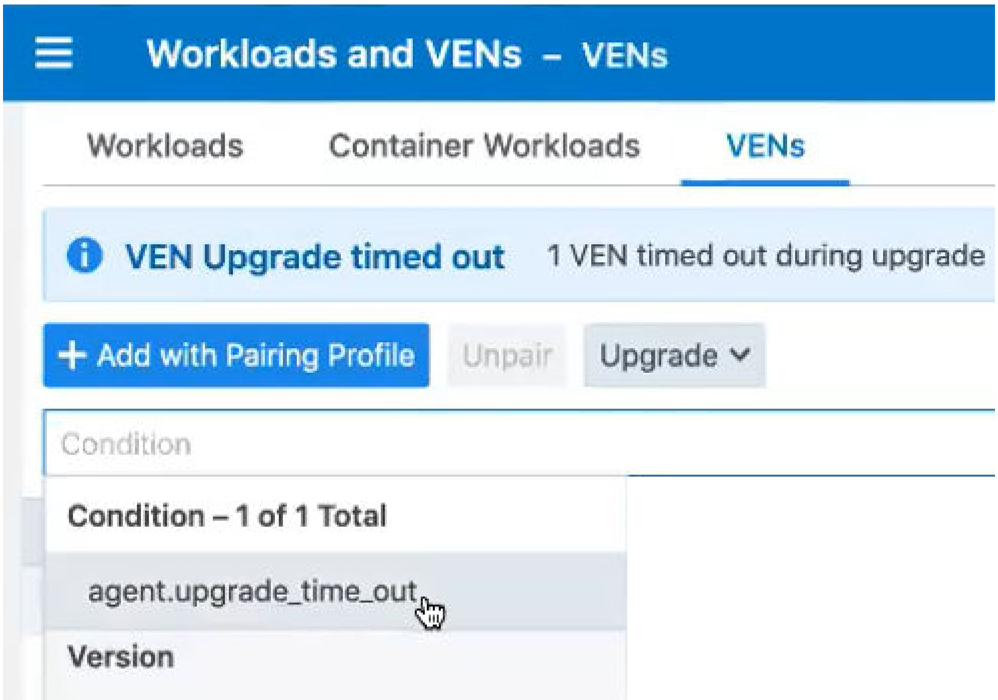
4. (Optional) Specify how much time the VENs have to successfully upgrade by entering a value and units of time in **VEN Upgrade Expiration**. The VEN upgrade timeout value can be specified in minutes, hours, or days. The timeout must be between 15 minutes and 180 days. For server VENs, the recommended upgrade timeout is 1 day. For endpoint VENs, the recommended timeout is 7 days. After the expiration time passes, the PCE will no longer instruct the VEN to upgrade, and the VEN will be in a warning state.
5. Click **Confirm Upgrade**.

Upgrade Selective VENs

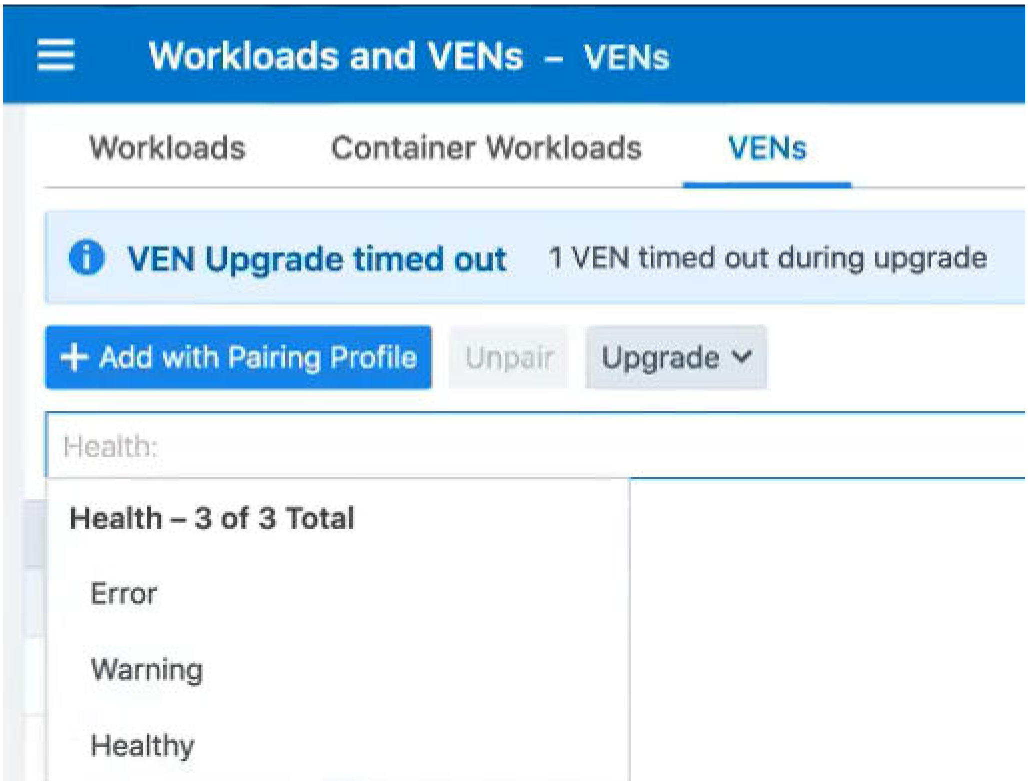
When upgrading, you can use all the filters available in the VEN list to manage the upgrade. For example, you could filter VENs by the location label to only upgrade VENs in a specific datacenter or you can filter the VENs in your environment by their operating systems and upgrade just those VENs.



You can filter and upgrade VENs based on specific conditions. For example, you can locate the VENs that weren't upgraded in your first attempt to upgrade all VENs to the default version; for example, these VENs might have been offline when you ran your first upgrade and the upgrade timed out for those VENs but succeeded on all the others in your environment.



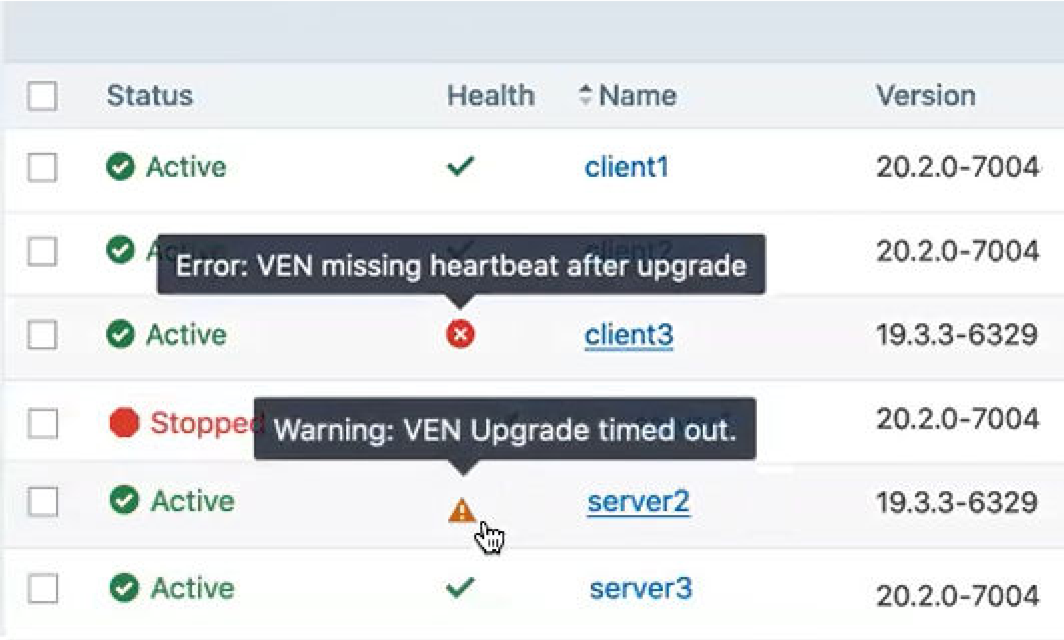
You can filter and upgrade VENs based on their Health status in the PCE.



VENs can have one of three Health statuses that you can filter for:

- **Healthy:** The VEN has no Health conditions.
- **Warning:** The VEN has one or more Warning conditions.

- **Error:** The VEN has one or more Error conditions or has both Error and Warning conditions.

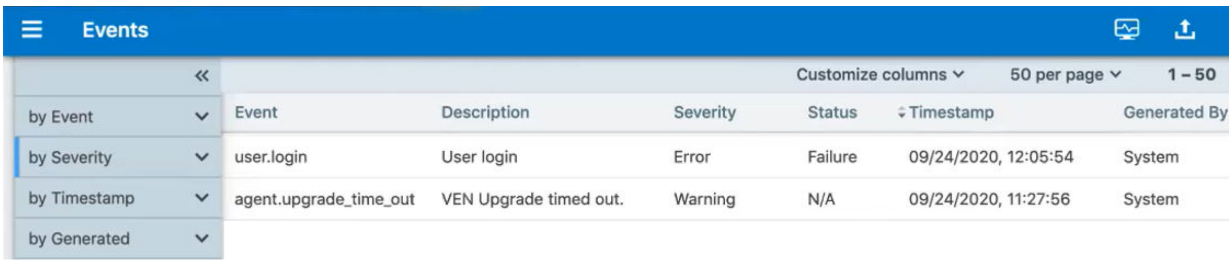


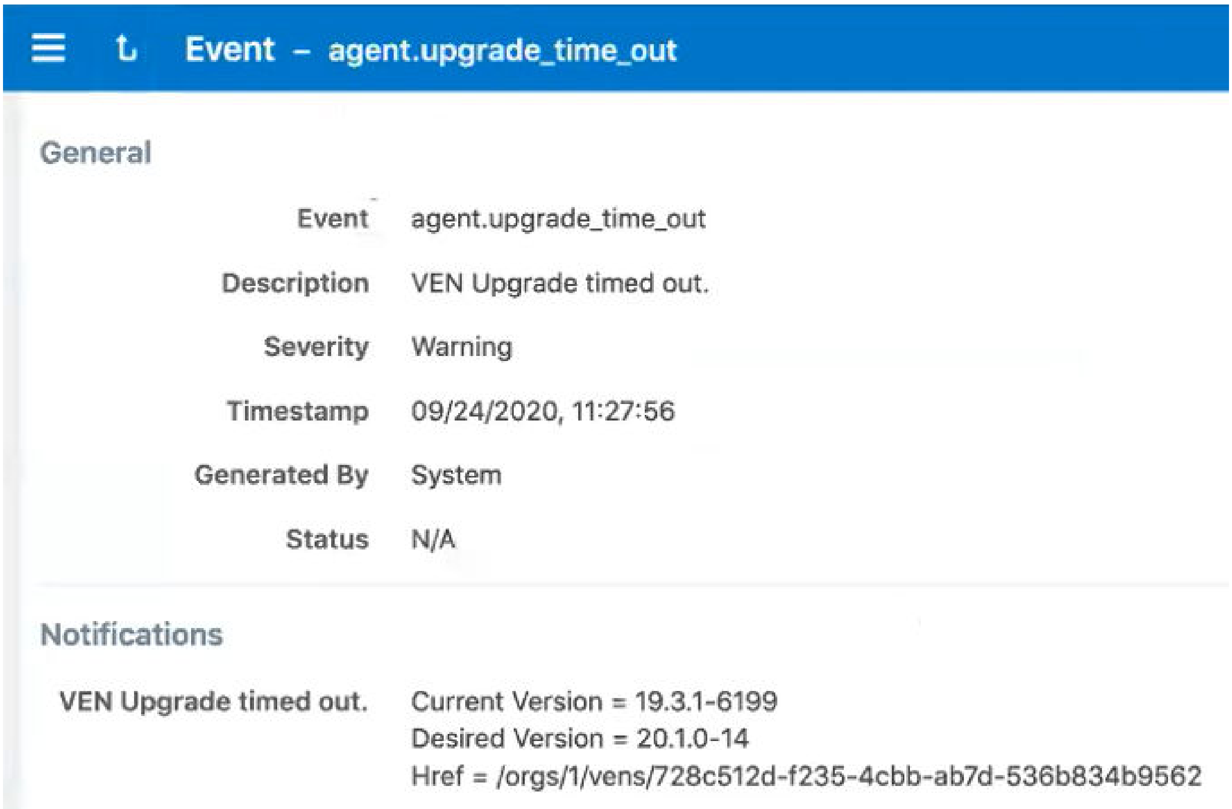
View VEN Upgrade Events

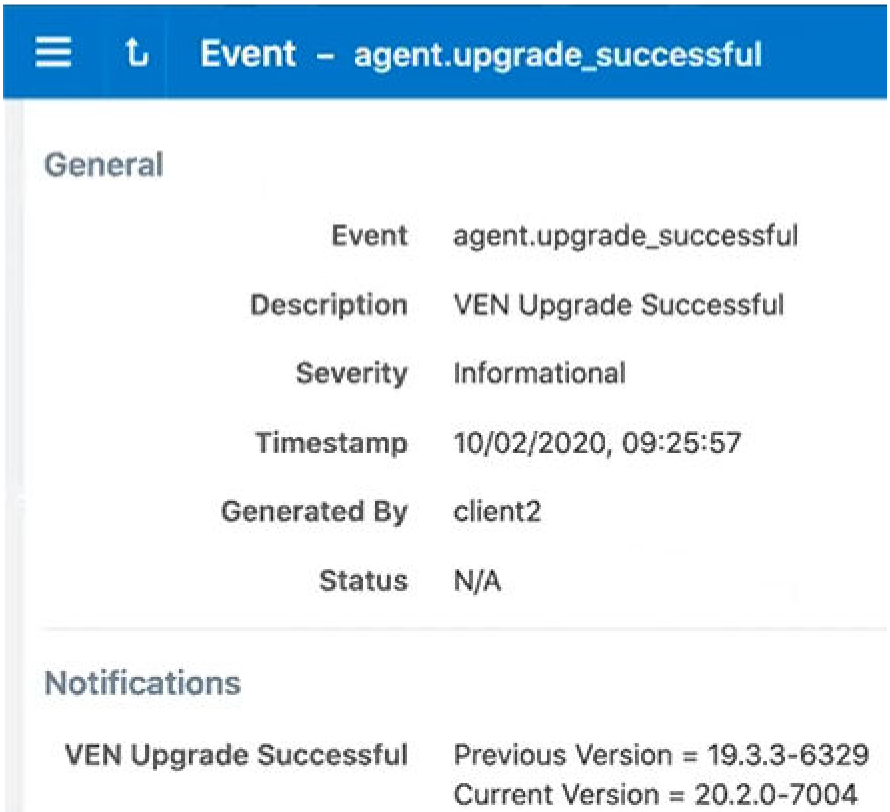
Upgrading VENs using the PCE web console or the REST API generate events that you can view in the Events page.

From the PCE web console menu, choose **Troubleshooting > Events**.

You can find events indicating when upgrades succeeded and when they failed. If you are using the *Upgrade All* option to upgrade large numbers of VENs in your environment, the PCE aggregates the event for `agent.upgrade_requested`; however, the PCE still generates a separate event for each successful upgrade.




Event – agent.upgrade_time_out
General
Event
agent.upgrade_time_out
Description
VEN Upgrade timed out.
Severity
Warning
Timestamp
09/24/2020, 11:27:56
Generated By
System
Status
N/A
Notifications
VEN Upgrade timed out.
Current Version = 19.3.1-6199
Desired Version = 20.1.0-14
Href = /orgs/1/vens/728c512d-f235-4cbb-ab7d-536b834b9562


Event – agent.upgrade_successful
General
Event
agent.upgrade_successful
Description
VEN Upgrade Successful
Severity
Informational
Timestamp
10/02/2020, 09:25:57
Generated By
client2
Status
N/A
Notifications
VEN Upgrade Successful
Previous Version = 19.3.3-6329
Current Version = 20.2.0-7004

VEN Installation & Upgrade with VEN CTL

The following topics describe how to use packages and the VEN CTL to install the VEN on hosts in your environment. To perform the tasks in this section, you must log into the Illumio Support portal to download the VEN software to your local environment.

Windows: Install and Upgrade with CLI and VEN CTL

This section discusses installing and upgrading the VEN for Windows by using packaging technology commands and the VEN CTL.

With the Windows VEN MSI, you have the option of activating (pairing) the VEN either during installation or after installation.

Windows VEN Installation Directories

By default, the Windows VEN is installed in the following directories:

- Installation: `C:\Program Files\Illumio`
- Data: `C:\ProgramData\Illumio`

VEN Package Format Changes

Starting with the Illumio Core 21.2.1 release, the Windows VEN installer switched from MSI to EXE package format. This package format change primarily affects Illumio Core On-Premises customers running older MSI-based Windows VENs.

For information about using the VEN Library in the PCE to install Windows VENs on workloads, see the "Pair a Windows Workload" topic.

Run PowerShell as Administrator

Use Windows PowerShell to run the VEN installation program.

Run PowerShell as Administrator with Execution Policy, because the installation affects the operating system.

Right-click the PowerShell icon and select **Run as Administrator**.

In addition, the VEN control scripts require the proper execution permissions on Windows. In PowerShell, run the following command before installation:

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned
```

Install the Windows VEN Using EXE Package

Starting with the version 21.2.1, the Windows VEN installer format changed from an MSI package to an EXE bundle. The installation file is now executable and `msiexec.exe` is no longer used to install the Windows VEN in Illumio Core 21.2.1 and later releases.

Command Line Interface

The Windows VEN installer supports following command line options:

- `/install`
- `/uninstall`
- `/quiet`
  Disables the interactive installer so that you don't respond to installation prompts.
- `/passive`
  Still displays a minimal user interface but does not provide installation prompts.
- `/norestart`
  Suppresses any attempts at restart.
- `/log`
  Logs installation information to a specific file.

The following installation command lines show how to install the VEN EXE bundle and activate the VEN after installation. See Windows VEN Activation After Installation [223].

Quiet VEN Installation

```
Start-Process -FilePath "<directory_path>\illumio-ven-<ven_version>.
<os_platform>.exe" -ArgumentList "/install","/quiet","/norestart","/log"
 "<directory_path>\VENInstaller.log" -Wait -PassThru
```

For example:

```
Start-Process -FilePath "$env:WinDir\temp\illumio-ven-21.5.0-
xxxx.win.x64.exe" -ArgumentList "/install","/quiet","/norestart","/log",
"$env:WinDir\temp\VENInstaller.log" -Wait -PassThru
```

Quiet VEN Installation with Custom Directories

```
Start-Process -FilePath "$env:WinDir\temp\illumio-ven-<version>-<build>.
win.x64.exe" -ArgumentList "/install","/quiet","/norestart","/log",
"$env:WinDir\temp\VENInstaller.log" INSTALLFOLDER="c:\illumio\ven"
DATAFOLDER="c:\illumio\ven_data" -Wait -PassThru
```

> **CAUTION**
> The VEN EXE installer supports custom installation directories; however, you should only specify the `INSTALLFOLDER` and `DATAFOLDER` parameters when installing the Windows VEN the first time. Do not specify these parameters when upgrading the Windows VEN using the EXE installer or the upgrade will fail.

Interactive VEN Installation

```
Start-Process -FilePath "<directory_path>\illumio-ven-<ven_version>.
<os_platform>.exe" -ArgumentList "/install","/log" "<directory_path>\
VENInstaller.log"
```

Windows VEN Activation After Installation

Be sure that you have the proper administrative permissions. See Run PowerShell as Administrator [221].

To activate the Windows VEN after installation, run the following command:

```
PS C:\Program Files\Illumio> .\illumio-ven-ctl.ps1 activate -activation-code
<activation_code> -management-server <pce_fqdn:pce_portnumber>
<activation_options>
```

Windows VEN Activation Options

You have several activation options you can set while pairing. You can set the workload policy state and apply labels at the time of activation.

This example shows how to activate a Windows workload with the following options:

- Set the VEN policy state to `illuminated` with no traffic logging: `-log_traffic false`
- Set the role as Web service: `-role Web`
- Set the application to HRM: `-app HRM`
- Set the environment to development: `-env Dev`
- Set the location of the VEN to New York City: `-loc NYC`

```
PS C:\Program Files\Illumio> .\illumio-ven-ctl.ps1 activate
-management-server yourPCE.example.com.8443 -activation-code
<activation_code>
-visibility_level flow_summary -log_traffic false -role Web -app HRM -env
Dev
-loc NYC
```

Kerberos for Windows VEN-to-PCE Authentication

To enable Kerberos authentication at installation, set the command-line variable `KERBEROS_PCE_SPN` on the installation program. Use the following value for this variable:

```
illumio-device-auth/<fqdn_of_your_pce>
```

Where:

- The literal `illumio-device-auth/` is required.
- `fqdn_of_your_pce` is the fully qualified domain name (FQDN) of your PCE.

Example:

```
C:\> msiexec.exe /i illumio-ven-<ven_version>.<os_platform>.msi
KERBEROS_PCE_SPN=illumio-device-auth/pce.example.com
```

Activation with Kerberos

On the `illumio-ven-ctl --activate` or in the pairing script, do *not* use any option that sets a label. That is, do not use the `--env`, `--loc`, `--role`, or `--app` options. Labels should be set in the PCE web console. See the *Security Policy Guide* for information.

After installation with the command-line variable, when you activate the VEN, a message similar to the following is displayed:

```
# illumio-ven-ctl activate

      Activating Illumio
...
Enabling Kerberos Authentication .....
...
```

Windows VEN Upgrade for the MSI Package Format

IMPORTANT
Illumio strongly recommends that you upgrade VENs only during maintenance windows.

NOTE
If the VEN was activated prior to the upgrade, it does not need to be activated again after the upgrade completes.

To upgrade the VEN, run this command:

```
PS C:\> msiexec /i illumio-ven-<ven_version>.<os_platform>.msi /qn
```

Windows VEN Uninstallation Using CLI

To uninstall the Windows VEN by using the VEN CTL, see "Deactivate and Unpair VENs" in the VEN Administration Guide.

Offline VEN During Unpairing

If the workload you are unpairing is offline, the workload might still appear in the workloads list in the PCE web console, even though the workload has been unpaired. The unpaired workload is removed from the PCE web console within 30-35 minutes.

Alternative: Remove Windows VEN Using Control Panel

You can also use the Windows Control Panel Programs and Features utility to remove the VEN. When you remove the Windows VEN with the Windows Control Panel, the VEN unpairs the workload with the **Unpair and remove Illumio policy** option. This method removes any current Illumio policy and activates the Windows firewall.

Linux: Install and Upgrade with CLI and VEN CTL

This section discusses installing and upgrading the VEN for Linux by using packaging technology commands and the VEN CTL.

- Installing the VEN on Linux relies native package management commands:
  - For Debian and Ubuntu (referred to as “Debian”): `dpkg`
  - For Red Hat and CentOS (referred to as “Red Hat”): `rpm`
- Root access on the workload is required for installation of the Linux VEN.
- Some of the optional installation features in the RPM are not available with the Debian package. These cases are marked in section titles below with “RPM only.”

About iptables Versions for Red Hat and CentOS

Red Hat Version 6 VENs

If the iptables version already on the workload is older than iptables version 1.4.7-16 or ipsets is older than version 6.11-4, the VEN installation process installs more recent versions of iptables and ipset, including libmnl. These unmodified distribution files (RPMs) are packaged with the VEN itself and installed in `/opt/illumio_ven/etc/extras`.

About Red Hat 8 Support and nftables

In 18.2.x, the VEN supported iptables. In 19.3.0 and later releases, the VEN supports nftables, which is the default host-based firewall used by Red Hat 8. Supporting nftables, which is used by Red Hat 8, simplifies rule writing and allows developers to write fewer rules and do so much more efficiently. The new support for nftables does not change VEN functionality or the VEN feature set because the underlying net filter capabilities are the same. Support for nftables provides the following usability enhancements for the VEN:

- **Simpler syntax:** Uses a simpler syntax, which is very similar to TCP dump.
- **Combined rules:** Rather than write 2 rules for every enforcement point, you can combine them into a single rule; such as combining multiple ports into a single rule, and IPv4 and IPv6 into a single rule.
- **Multiple actions:** A single rule can have multiple actions, such as LOG and DROP.
- **Built-in tracing:** Includes built-in support for named sets. To use lists or sets with `iptables`, you need to install `ipset.nftables` has integrated set support and can be used more naturally within the configuration.

Notes:

- Native support for `nftables` in Red Hat 8 does not change the VEN installation or upgrade process; if you've written installation scripts, they don't require updates.
- `nftables` does not impact Firewall Coexistence and it is still supported.
- Using the `nftrace` tool on Illumio created tables is not supported because it requires specific filtering rules.
- `nftables` support does not impact the PCE; viewing a workload that is running Red Hat 8 in the PCE web console does not change. You can view all the workload details. Creating policy for workloads running Red Hat 8 does not change.

> **IMPORTANT**
> In 19.3.0 and later releases, the VEN continues to support earlier versions of Red Hat with no changes. For the complete list of Red Hat versions supported by the VEN by release, see OS Support and Package Dependencies on the Illumio Support portal.

Linux Default Installation Directories

The Linux VEN is installed into two directories by default:

- `/opt/illumio_ven`
- `/opt/illumio_ven_data`

Directory Ownership Pre- and Post-activation

- All directories are created with mode 0750.
- Post-activation user/group `ilo-ven:ilo-ven` allows processes running as that user to write to the VEN installation directory and VEN data directory.
- At installation, you can set various environment variable to override default settings. See the "Linux Installation with Environment Variables" topic.

VEN Package Format	Path	Default Pre-Activation Owner	Default Post-Activation Owner
RPM	`/opt/illumio_ven`	`root:ilo-ven`	`root:ilo-ven`
	`/opt/illumio_ven_data`	`ilo-ven:ilo-ven`	`ilo-ven:ilo-ven`
DPKG	`/opt/illumio_ven`	`root:ilo-ven`	`root:ilo-ven`
	`/opt/illumio_ven_data`	`root:ilo-ven`	`ilo-ven:ilo-ven`

Dependency Check for Certificates

If your PCE-to-VEN SSL certificate was signed by a private CA and the signing CA's credentials have already been added to the workload's trusted certificate store, the ca-certificates package is not needed. To install the VEN without the dependency check, follow these examples:

- Red Hat: `rpm -vh -nodeps illumio-ven-<ven_version>.<os_platform>.rpm`
- Debian: `dpkg --ignore-depends=illumio-ven-<ven_version>.<os_platform>.deb`

RPM Only: Installation in Non-Default Directory

If you want to change the installation directory during installation or upgrade, you can use environment variable or use the `--prefix` option on the RPM command line.

```
# rpm -ivh <illumio-ven-pkg>.rpm --prefix=/opt/foo/bar
```

CAUTION

The Linux VEN does not support installing the VEN in a directory where the directory is a symbolic link.

Linux Installation with Environment Variables

The following table lists VEN environment variables that you can set for the package installation on Linux.

NOTE

Before installation, set any of the following environment variables.

Variable	Description
VEN_ACTIVATION_CODE	The activation code; see Example: Linux Environment Variables [230] and About the Command Options [247].
VEN_DATA_DIR	Directory where the `illumio_ven_data` directory is created. This option can also be used when you are upgrading a VEN with RPM or Debian.
VEN_DISABLE_MONITOR_RESTART	Disable the VEN agent monitor process. See Disable Agent Monitor cronjob [230].
VEN_INSTALL_ACTION	Activate or prepare the VEN during installation. Valid values: • `activate`: Requires an activation code on the `illumio-ven-ctl` control script or set in the `VEN_ACTIVATION_CODE` environment variable. • `prepare`: Used to defer activation until after installation. For example, see Prepare Golden Master for Workload Installation [191].
VEN_KERBEROS_WORKLOAD_SPN	See the "Kerberos for Linux VEN-to-PCE Authentication" topic.
VEN_KERBEROS_MANAGEMENT_SERVER_SPN	See the "Kerberos for Linux VEN-to-PCE Authentication" topic.
VEN_KERBEROS_LIBRARY_PATH	See the "Kerberos for Linux VEN-to-PCE Authentication" topic.
VEN_MANAGEMENT_SERVER	The FQDN of the PCE server and its port.
VEN_NONPRIV_UID	If `VEN_NONPRIV_USER` is not set, create the `ilo-ven` user with the specified UID.
VEN_NONPRIV_GID	If `VEN_NONPRIV_USER` is not set, create the `ilo-ven` group with the specified GID.
VEN_NONPRIV_USER	Existing username to override the default username `ilo-ven`. The group name of the specified user is the primary existing group name of the specified user. • If `VEN_NONPRIV_USER` is set, any values for `VEN_NONPRIV_UID` and `VEN_NONPRIV_GID` are ignored. • Conversely, if `VEN_NONPRIV_USER` is not set, any values for `VEN_NONPRIV_UID` and `VEN_NONPRIV_GID` take effect.
ILLUMIO_RUNTIME_ENV	If `ILLUMIO_RUNTIME_ENV` is set, read the `runtime_env.yml` from this file path. This environment variable is unique because it is relevant during and after installation.

Kerberos for Linux VEN-to-PCE Authentication

The `illumio-ven-ctl` command does not have any options for Kerberos, but when you activate the VEN with `illumio-ven-ctl`, at installation it honors the Kerberos values that have been set in environment variables.

Before installing the Linux VEN, set the following environment variables.

Environment variable	Value	Notes

`VEN_KERBEROS_WORKLOAD_SPN`	(Optional) See notes.
	
	The default host principal set in the Kerberos keytab file.
	
	The SPN of the server for renewing Ticket Granting Tickets (TGT) for Linux workloads.
	
	Format:
	
	`host/fqdn_of_ven@REALM`
	
	Where:
	
	• The literal `host/` is required.
	• `fqdn_of_ven` is the FQDN of the workload where the VEN is installed.
	• `@REALM` is optional. If not specified, the default realm is used.
		A workload might have more than one host principal in its keytab file, one of them the principal needed for PCE authentication. In this case `VEN_KERBEROS_WORKLOAD_SPN` must be set so that the VEN software knows which principal to use to acquire a TGT.
		
		The VEN relies on the default Kerberos keytab file, typically `/etc/krb5.keytab`. Therefore, the host SPN for PCE authentication must be added to the default keytab file.
		
		Before deploying Kerberos authentication, you can use `kinit` to verify that a TGT for the workload's SPN can be acquired:
		
		`kinit -k`
		
		If the command is successful, use `klist` to verify the TGT has been acquired for the correct host SPN.
		
		If the default SPN is not what you want for PCE authentication, use the following command to verify that you can reach the desired SPN:
		
		`kinit -k host/fqdn_of_ven@REALM`
		
		If the command is successful, set `VEN_KERBEROS_WORKLOAD_SPN` to `host/fqdn_of_ven@REALM`.

`VEN_KERBEROS_MANAGEMENT_SERVER_SPN`	SPN for the PCE Example: `illumio-device-auth/pce.example.com`	GSSAPI Authentication
`VEN_KERBEROS_LIBRARY_PATH`	Absolute path to `libgssapi_krb5.so` Example: `/usr/lib`	The exact path can vary by type of Linux OS. If `libgssapi_krb5.so` does not exist on your system, create a symlink of the same name to point to the `libgssapi_krb5.so.n` file, where `n` is the number on the actual installed shared object library on your workload, like `libgssapi_krb5.so.2`

Activation with Kerberos

On the `illumio-ven-ctl --activate` or in the pairing script, do *not* use any option that sets a label. That is, do not use the `--env`, `--loc`, `--role`, or `--app` options. Labels should be set in the PCE web console. See the *Security Policy Guide* for information.

After installation with the command-line variable, when you activate the VEN, a message similar to the following is displayed:

```
# illumio-ven-ctl activate

      Activating Illumio
```

```
...
Enabling Kerberos Authentication .....
...
```

Example: Linux Environment Variables

To activate the VEN during installation, set the following environment variables before running the installation command.

- VEN_MANAGEMENT_SERVER
- VEN_ACTIVATION_CODE
- VEN_INSTALL_ACTION

For example, to activate a VEN during installation of a VEN package:

```
# VEN_MANAGEMENT_SERVER=pce.example.com:8443 VEN_INSTALL_ACTION=activate
VEN_ACTIVATION_CODE=<activation_code> rpm -ivh illumio-ven-
<ven_version>.<os_platform>.rpm
```

Or

```
# VEN_MANAGEMENT_SERVER=pce.example.com:8443 VEN_INSTALL_ACTION=activate
VEN_ACTIVATION_CODE=<activation_code> dpkg -i illumio-ven-
<ven_version>.<os_platform>.deb
```

Change Default Name of User at Installation

The default username for the VEN installation is `ilo-ven`. With the package installation, you can specify an environment variable to set a different, existing username to override this default. The group name is the specified user's primary group and does not need to be specified.

```
# VEN_NONPRIV_USER=desired_existing_username rpm -ivh illumio-ven-
<ven_version>.<os_platform>.rpm
```

Or

```
# VEN_NONPRIV_USER=desired_existing_username dpkg -i illumio-ven-
<ven_version>.<os_platform>.deb
```

Disable Agent Monitor cronjob

You can disable the agent monitor cronjob before or after VEN installation. When failing to hook into existing `init` systems like `systemd`, `upstart`, or `SysV`, the Linux VEN installation creates a cronjob to check the VEN agent monitor process and restart it if necessary. This cronjob runs every 10 minutes.

Some organizations prefer to rely on their own VEN agent monitoring processes. The Illumio-supplied VEN-checking cronjob might create logs whose size you consider excessive or whose frequency is not right for your needs.

To disable the Linux VEN monitoring cronjob before installation:

Set the following environment variable:

```
export VEN_DISABLE_MONITOR_RESTART=true
```

Any value other than `true` does not have any effect.

To modify or disable the Linux VEN monitoring cronjob after installation:

You have several options:

- Edit your crontab to decrease the cronjob's frequency.
- In your crontab, completely comment out the VEN agent monitoring cronjob.

To substitute your own VEN agent monitor checking process, consider the following points:

- Rely on your own organization's standard mechanisms for monitoring processes.
- Make sure your monitoring restarts the VEN if necessary.
  - Do not restart only the VEN agent monitoring process. Restart the entire VEN:

    ```
    # illumio-ven-ctl restart
    ```
  - Be sure that your monitoring process has sufficient permissions to restart the VEN.

Linux VEN Activation After Installation

To activate the VEN after installation, use the `illumio-ven-ctl` control script with the `activate` argument to activate the workload and pair the VEN with the PCE.

At a minimum, to activate the VEN using the VEN control script, you need the hostname or IP address of the PCE, an activation code (called a pairing key in the PCE web console) generated from a pairing profile, and any other required options, such as the workload policy state, label assignment, and workload name. For example, the following command shows how to activate the VEN that places the workload into the Illumination policy state (`--mode`).

```
# /opt/illumio_ven/illumio-ven-ctl activate --management-server
pce.example.com:8443
 --activation-code <activation_code>
```

Upgrade Linux VEN Using CLI

IMPORTANT

Illumio strongly recommends that you upgrade VENs only during maintenance windows.

NOTE

If the VEN was activated prior to the upgrade, it does not need to be activated again after the upgrade completes.

Custom Username, Installation Directory, VEN Data Directory

If you installed the VEN with your own username, for upgrade you need to specify that same username with the `VEN_NONPRIV_USER` environment variable.

If you previously installed the VEN to non-default installation (RPM only) and data directories with environment variables, specify the same values before upgrade.

See the "Linux Installation with Environment Variables" topic.

RPM Upgrade

```
# rpm -Uvh illumio-ven-<ven_version>.<os_platform>.rpm
```

> **IMPORTANT**
>
> If the `VEN_DATA_DIR` environment variable and the `--prefix` option are not specified during the RPM installation, then the **`illumio_ven`** and **`illumio_ven_data`** directories are created in the `/opt` directory.
>
> This information is important because if you previously installed the VEN to non-default installation and data directories, and if you upgrade without specifying those non-default directories, the VEN will not upgrade to your custom directories.
>
> Therefore, if you specified non-default installation (RPM only) and data directories when you installed the VEN, you need to specify those same directories in the upgrade command.

This example also includes a custom username that was used during VEN installation.

For example, if you installed the VEN with this type of command:

```
# VEN_NONPRIV_USER=ven_install_username VEN_DATA_DIR=/opt/my_data_dir rpm
-ivh <orig-illumio-ven-pkg>.rpm --prefix=/opt/my_ven_dir
```

Then, upgrade the VEN with the following command:

```
# VEN_NONPRIV_USER=ven_install_username VEN_DATA_DIR=/opt/my_data_dir rpm
-Uvh <new-illumio-ven-pkg>.rpm --prefix=/opt/my_ven_dir
```

Debian Upgrade

```
# dpkg -i illumio-ven-<ven_version>.<os_platform>.deb
```

IMPORTANT

If the `VEN_DATA_DIR` environment variable is not specified during VEN installation, then the `illumio_ven_data` directory is created in the `/opt` directory.

This information is important, because if you specified a custom data directory during installation, and if you upgrade the VEN without specifying the custom data directory, the VEN will not upgrade using your custom data directory.

Therefore, if you specified a non-default data directory when you installed, you need to specify the same non-default data directory during upgrade.

This example also includes a custom username that was used during VEN installation.

NOTE

Using `--prefix=/opt/my_ven_dir` to specify a custom installation directory is not supported with Debian.

For example, if you installed the VEN with this type of command:

```
# VEN_NONPRIV_USER=ven_install_username VEN_DATA_DIR=/opt/my_data_dir dpkg
-i
<orig-illumio-ven-pkg>.deb
```

Then, upgrade the VEN with the following command:

```
# VEN_NONPRIV_USER=ven_install_username VEN_DATA_DIR=/opt/my_data_dir dpkg
-i <new-illumio-ven-pkg>.deb
```

Uninstall Linux VEN Using CLI

Unpair a Linux VEN before uninstalling it. See "Deactivate and Unpair VENs" in the VEN Administration Guide.

SUSE Linux: If a SUSE workload is unpaired in the Full enforcement policy state, the uninstallation might not complete when the workload does not have rules that allow it to connect to SUSE repositories. To avoid this issue, change the policy state to Visibility before unpairing the VEN. For more information see "Workload Policy States" in the VEN Administration Guide.

Uninstall the VEN

Security Implications: Production applications on this workload could break because after uninstalling the VEN this workload will no longer allow any connections to it other than SSH on port 22.

To uninstall the VEN, see "Deactivate and Unpair VENs" in the VEN Administration Guide.

AIX: Install and Upgrade with CLI and VEN CTL

The following topic describes how to install and upgrade the AIX VEN by using packaging technology commands and the VEN CTL.

Limitations and Considerations

General

- AIX 5.3 is not supported.
  See VEN OS Support and Package Dependencies for the list of supported operating systems for AIX VENs.
- AIX native IPsec is not supported while the VEN is installed.
- The AIX VEN does not support SecureConnect and SecureConnect Gateway.
- The following directories must be present on the AIX host or the AIX VEN installation will fail. These directories are commonly present on AIX hosts.
  - `/var/lib`
  - `/var/log`
- By default, the AIX VEN is installed in the following directories:
  - `/opt/illumio_ven`
  - `/opt/illumio_ven_data`

  Installing the AIX VEN in a custom directory is not supported. Do not change the default installation directory for the AIX VEN or the AIX VEN installation will fail.

Configuration Options for CA Bundle and CA DIR

> **NOTE**
> The options `trusted_ca_bundle` and `trusted_ca_dir` in `runtime_env.yml` are no longer used.

Core 22.2 introduced new options for configuring the CA bundle or CA directory it uses to verify the PCE TLS certificate. You can specify the new options in the `/etc/default/illumio-agent` file. The options are:

- **TRUSTED_CA_BUNDLE** can point to a specific certificate bundle. For example:

  `TRUSTED_CA_BUNDLE=/etc/ssl/certs/ca-bundle.crt`
- **TRUSTED_CA_DIR** can point to a directory containing certificates. For example:

  `TRUSTED_CA_DIR=/etc/ssl/certs`

IPFilter

- Illumio provides a custom IPFilter package for managing the packet filtering rules. Before you install the AIX VEN, install the Illumio-provided IPFilter package.

> **CAUTION**
> You must use the Illumio customized IPFilter package with the AIX VEN. Do not use IBM's IPFilter package or the AIX VEN will not function correctly.

- Avoid any changes to packet filtering with `genfilt`, `mkfilt` and other such network tools. Do not perform any such operation while VEN software is installed.
- The AIX system firewall's state table limit is 65,536 entries. When that limit is reached, IPFilter drops packets. If you anticipate a high number of network connections, configure higher limits in the IPFilter state table. See "Tuning the IPFilter State Table (AIX/Solaris)" in the VEN Administration Guide.

Change Default Username Before Installation

Before installing the VEN on AIX, you can set an environment variable to change the username that owns the non-privileged portions of the installed software. The privileged portions of the installed software are always owned by root, and the software can only be run as root.

Environment Variable	Description
`VEN_NONPRIV_USER`	Existing username to override the default username `ilo-ven`. The group name of the specified user is the primary existing group name of the specified user.

Boot Scripts Installed at VEN Installation

As part of installation, the VEN creates RC scripts ("run commands") in `/etc/rc3.d` to start the VEN at boot.

Illumio Support for IPFilter

IBM has discontinued support and development of IPFilter and has put IPFilter on GitHub as an open source project. Consequently, Illumio provides its own version of IPFilter for the Illumio AIX VEN version 17.1.2 and later.

> **NOTE**
> Illumio supports *only* its provided version of IPFilter. We do not support installing the AIX VEN with the OEM version of IPFilter. Before installing the AIX VEN, you must install the Illumio-provided IPFilter package.

Illumio supports its version of IPFilter in the following ways:

- The Illumio IPFilter package will not be made public. Permissive licensing of IPFilter does not require that modifications of open source software be made public.
- Illumio can provide IPFilter source code patches for bug-fixes and improvements on request to your Illumio representative.

Download AIX VEN Tar File and IPFilter Package

Download the VEN Packages tar file from the Illumio Support site. The tar file contains the AIX VEN in Backup File Format (BFF) format.

Additionally, you must download the Illumio-provided IPFilter package from the Illumio Support site. The VEN package does *not* contain the required Illumio-provided IPFilter package.

To download the AIX VEN files:

1. Go to the Illumio Support site (login required).
2. Select **Software > Download** under the VEN section > the VEN version.
   The Download VEN page appears. The page contains two tables: "VEN" and "Other"
3. In the VEN Packages row of the VEN table, click the filename for the VEN tar file.
4. In the Other table, click the AIX IPFilter filename (`ipfl.5.3.0.5002.bff`) to download the Illumio-supported IPFilter 5.3.0.5002 package.

Upgrade to Illumio IPFilter

This procedure describes how to perform either of these tasks:

- Upgrade from the IBM IPFilter package to the current Illumio IPFilter 5.3.0.5002 package
- Upgrade from the previous version of the Illumio IPFilter 5.3.0.5001 package to the current 5.3.0.5002 package

The steps in this procedure apply to both of these IPFilter upgrades except for step #4, which applies only when upgrading from IBM IPFilter to the Illumio IPFilter 5.3.0.5002 package.

To upgrade to Illumio IPFilter:

1. Download the Illumio-supplied IPFilter package. See Download AIX VEN Tar File and IPFilter Package [236].
2. Stop the VEN if it's running:

   ```
   illumio-ven-ctl stop
   ```
3. Stop the IBM ipf kernel extension using the following command:

   ```
   /lib/methods/cfg_ipf -u
   ```

   > **NOTE**
   > In some cases, there may be multiple instances of ipf. Confirm there are no running instances by running the above `stop` command again until it returns `no such device`.

   If the command fails with the error `Device Busy`, before continuing these steps, reboot the system
4. **[For Upgrades from IBM IPFilter Only]** If IBM iFIX or ipfl is installed on the host, uninstall them. (In an earlier release, Illumio had recommended installation of some iFIXes.)

> **NOTE**
> Depending on your installed AIX version, you might have installed iFIX version IV89793s5a or IV89793s3a. Remove the version corresponding to the version already installed on your AIX server. Neither version is needed and must be removed with the appropriate `emgr` command. The following command uninstalls only version IV89793s5a.
>
> ```
> emgr -r -L  IV89793s5a.161102.epkg.Z
> ```

5. Change directory to where you downloaded the AIX VEN and the IPFilter package.
6. Upgrade the version of IPFilter with the Illumio custom IPFilter:

   ```
   inutoc . && installp -acYd . ipfl
   ```
7. Proceed to installing or upgrading the AIX VEN.

Install the AIX VEN

1. Download the VEN package from the Illumio Support site. See Download AIX VEN Tar File and IPFilter Package [236].
2. Log in to the AIX host and become superuser.
3. If necessary, upgrade IPFilter on the AIX host to Illumio's custom IPFilter. See Upgrade IBM IPFilter to Illumio IPFilter [236].

   > **IMPORTANT**
   > You must upgrade to Illumio's custom IPFilter before installing the AIX VEN.
4. Copy your trusted root CA certificate in the following directory with a filename `ca-bundle.crt`. This path must be exactly as shown.

   ```
   /var/ssl/certs/ca-bundle.crt
   ```
5. Make `ca-bundle.crt` world-readable.

   ```
   # chmod 644 /var/ssl/certs/ca-bundle.crt
   ```
6. Install the VEN package on the AIX host by entering the following commands, where `path_to_bff_file` is the directory where you copied the AIX VEN BFF file.

   ```
   # inutoc <path_to_bff_file>
   # installp -acXgd path_to_bff_file illumio-ven
   ```

AIX VEN installation is complete. The next step is Activate AIX VEN After Installation [238].

Optional: If you anticipate a high number of network connections, you can configure higher limits in the IPFilter state table. See Tuning the IPFilter State Table (AIX/Solaris)" in the VEN Administration Guide.

Activate AIX VEN After Installation

> **IMPORTANT**
> If you're using the GRE and IPIP protocols, before activating the VEN on AIX, edit the file in the `/etc/protocols` directory to support the GRE and IPIP protocols. If the GRE and IPIP protocol lines are commented out, un-comment them.

After installing the VEN package on the AIX host, activate the VEN. Use the Illumio VEN control script (`illumio-ven-ctl`) with the `activate` option to activate the workload and pair the AIX VEN with the PCE.

At a minimum, to activate the AIX VEN using the VEN control script, you need the hostname or IP address of the PCE, an activation code (called a pairing key in the PCE web console) generated from a pairing profile, and any other available options, such as the workload policy state, label assignment, workload name, and more.

For information about obtaining an activation code from the PCE web console, see "Pairing Profiles" in the Security Policy Guide.

```
# /opt/illumio_ven/illumio-ven-ctl activate --management-server
<pce_fqdn:port> --activation-code <code>
```

See the following example command:

```
# /opt/illumio_ven/illumio-ven-ctl activate --management-server
pce.example.com:8443 --activation-code <code>
```

Upgrade the AIX VEN

> **IMPORTANT**
> Illumio strongly recommends that you upgrade VENs only during maintenance windows.

> **NOTE**
> If the VEN was activated prior to the upgrade, it does not need to be activated again after the upgrade completes.

For the supported upgrade paths for the AIX VEN, see Upgrade VEN on the Illumio Support portal (login required).

1. Download the new version of the VEN package from the Illumio Support site. See AIX Tar File and IPFilter Package [236].
2. If necessary, upgrade the Illumio-supported IPFilter package to version 5.3.0.5002.

   If you are upgrading the AIX VEN from an earlier release, such as 17.1.x, you might be running the Illumio-supported AIX IPFilter package version 5.3.0.5000. See Upgrade to Illumio IPFilter [236].
3. Stop the VEN if it's running:

   ```
   illumio-ven-ctl stop
   ```
4. Upgrade the VEN package on the AIX host by entering the following commands, where `path_to_bff_file` is the directory where you copied the new version of the AIX VEN BFF file.

   ```
   # inutoc <path_to_bff_file>
   # installp -acXgd path_to_bff_file illumio-ven
   ```

Solaris: Install and Upgrade with CLI and VEN CTL

The following topic describes how to install the Solaris VEN by using packaging technology commands and the VEN CTL.

The VEN for Solaris supports two different Solaris machine architectures: SPARC and x86_64. The installation and upgrade steps for both machine architectures are identical but each architecture uses its own VEN package file.

Limitations and Requirements

General

- In Illumio Core 19.3.1 and later releases, the Solaris VEN supports Solaris zones. The Solaris operating system (OS) is architected such that all Solaris zones share the same underlying kernel networking stack of the global zone. Due to OS limitations, the VEN cannot reliably manage non-global zones that are configured with `ip-type: shared` and only non-global zones with `ip-type: exclusive` are fully supported. If you are using the VEN on workloads with a Solaris non-global zone and require additional assistance, contact your Illumio Customer representative.
- By default, the Solaris VEN is installed in the following directories:
  - `/opt/illumio_ven`
  - `/opt/illumio_ven_data`

  Installing the Solaris VEN in a custom directory is not supported. Do not change the default installation directory for the Solaris VEN or the Solaris VEN installation will fail.
- Installing or activating the Solaris VEN on a workload running an LDAP client can take longer than on other workloads without an LDAP client.
- The Solaris VEN requires the bash shell and the Solaris XCU4 utilities (POSIX-compliant tools) be installed on the Solaris host. Verify that both are installed on the host. The XCU4 utilities are installed using the Solaris SUNWxcu4 package, typically in the `/usr/xpg4/bin/` directory. See the Oracle Solaris documentation for information about installing the XCU4 utilities.

IP Filter (Solaris version 11.3 and earlier)

- Avoid making any changes to packet filtering with Packet Filter. Do not use Packet Filter while VEN software is installed.

- The Solaris system's firewall state table limit is 65,536 entries. When that limit is reached, IP Filter drops packets. If you anticipate a high number of network connections, configure higher limits in the IP Filter state table. See "Tuning the IP Filter State Table (AIX/Solaris)" in the VEN Administration Guide.

> **IMPORTANT**
> In Solaris 11.4, Packet Filter replaces IP Filter. When installing the VEN on Solaris 11.4, Illumio only supports Packet Filter. IP Filter is not supported in branded zones starting with Solaris 11.4.

Change Default Username

You can set an environment variable to change the username that owns the non-privileged portions of the installed software. The privileged portions of the installed software are always owned by root, and the software can only be run as root.

Environment Variable	Description
`VEN_NONPRIV_USER`	Existing username to override the default username `ilo-ven`. The group name of the specified user is the primary existing group name of the specified user.

You can reset this environment variable in your customized Solaris Response file or at a prompt during interactive installation.

About Solaris 11.4 Support

Prior to 11.4, Solaris used IP Filter as the firewall. In Solaris 11.4, Packet Filter is the only supported firewall.

The following details apply to Solaris 11.4 support by the VEN:

- Support for Solaris 11.4 does not change the VEN installation or upgrade process on Solaris workloads; if you've written installation scripts, they don't require updates. Package installation remains the same for 11.4 as for earlier supported versions of Solaris.
- Packet Filter support does not impact the PCE; viewing a workload that is running Solaris 11.4 in the PCE web console does not change. You can view all the workload details. Creating policy for workloads running Solaris 11.4 does not change.
- Packet Filter does not support customizable table sizes. However, state tables in Solaris 11.4 use a 1 million state table size.

> **IMPORTANT**
> For the complete list of all Solaris versions supported by the VEN in this release, see OS Support and Package Dependencies on the Illumio Support portal.

About the Solaris Response and Admin Files

In addition to the Solaris VEN, the VEN package includes two files to help with VEN installation on Solaris hosts: the Solaris Administration and Response files. For more information about these files, see Avoiding User Interaction When Adding Packages (pkgadd) in the *Oracle Solaris Administration Guide*.

Solaris Administration File

The Solaris Administration contains information about how the VEN installation or upgrade should proceed on the Solaris host. To perform a non-interactive VEN installation or upgrade (the VEN installation script will *not*

prompt for settings when it runs), you must customize the Administration file.

In addition to settings, the file contains commented-out instructions for changing the settings.

> **CAUTION**
> If you choose to provide custom values in the Administration file, you must delete these commented-out lines or the VEN installation or upgrade will fail. Commented-out lines in a Solaris Administration file are not supported with the Solaris `pkgadd` command.

```
# This file is used in case of upgradation
# instance=ask allows multiple instance of the same software to be installed
# and hence the UPDATE flag is passed to us in procedural scripts of IPS.
mail=
instance=ask
partial=ask
runlevel=ask
# Require that our dependencies are met when installing.
idepend=quit
# However, if someone tries to uninstall us but another package depends on
us,
# we should just warn them & ask if they want to proceed anyway.
rdepend=ask
space=ask
setuid=ask
conflict=ask
action=nocheck
networktimeout=60
networkretries=3
authentication=quit
keystore=/var/sadm/security
proxy=
basedir=default
```

Solaris Response File

The VEN package includes a template for the Solaris Response file. The template contains the environment variables that you can set when installing and upgrading the Solaris VEN.

In addition to the available environment variables, the template contains commented-out instructions for providing custom values for variables.

> **CAUTION**
>
> If you choose to provide custom values in the Response file, you must delete these commented-out lines or the VEN installation or upgrade will fail. Commented-out lines in a Solaris Response file are not supported with the Solaris `pkgadd` command.

```
#Parameter : VEN_NONPRIV_USER
#Type : String
#Description : VEN non-privileged user. If unspecified
(VEN_NONPRIV_USER=""),
then the default account "ilo-ven" is used. If that account does not exist
on the system, it is created automatically. If specified
VEN_NONPRIV_USER="foo"), the provided account is used. If that account does
not exist on the system, then the installer fails. All non-root-owned files
that the VEN creates are owned by that user and that user's primary group.
For further information about this feature, refer to the
Illumio VEN deployment documentation.
VEN_NONPRIV_USER=""
#Parameter : VEN_PKI_CLIENT_CERT
#Type : String
#Description : PKI (public key infrastructure) authentication certificate.
Use with VEN_PKI_CLIENT_KEY. When specified, these fields are appended to
runtime_env.yml. Then, they may be used to activate the VEN. I.e.,
``$ /opt/illumio_ven/illumio-ven-ctl activate'' uses these fields to
authenticate the VEN with the PCE.
VEN_PKI_CLIENT_CERT=""
VEN_PKI_CLIENT_KEY=""
VEN_KERBEROS_MANAGEMENT_SERVER_SPN=""
VEN_KERBEROS_LIBRARY_PATH=""
VEN_ACTIVATION_CODE=""
VEN_MANAGEMENT_SERVER=""
VEN_INSTALL_ACTION=""
#Parameter : VEN_NO_SUSPEND
#Type : Number
#Description : Custom setting to disable suspend. 1 - disable, 0 - default
VEN_NO_SUSPEND=0
```

If you leave the Response file as is, the VEN installation script uses the default values for these environment variables by displaying them at the prompts during an interactive installation or silently during installation (because you're using the Administration file).

To Use Customized Response and Administration Files

To customize the Response and Administration file for your Solaris VEN installation or upgrade, perform these steps:

1. Extract the Response and Administration files from the VEN package. See Installation Preparation [243] for information.
2. Copy and rename the files from the following directories for your machine architecture:

   ```
   sudo bash
   # cp illumio-ven/root/opt/illumio_ven/etc/templates/admin
   ```

```
/tmp/admin.custom
# cp illumio-ven/root/opt/illumio_ven/etc/templates/response
/tmp/response.custom
```

This command example suggests copying the files to the `/tmp` directory; however, you can copy the files to any directory.

3. Edit the files to set your own values. See Solaris Response File [241] and Solaris Administration File [241] for the requirements when customizing these files.

Installation Preparation

1. Download the VEN package from the Illumio Support portal. See the "Obtain the VEN Packages" topic for information.

   The Solaris VEN software downloaded from the Illumio Support portal is provided as a compressed tar archive file that contains one file for each of the supported Solaris machine architectures: SPARC and x86_64:

   - `illumio-ven-<ven_version>.sol5.sparc.pkg`
   - `illumio-ven-<ven_version>.sol5.i386.pkg`
2. Extract the Solaris VEN software:

```
# gunzip illumio-ven-<ven_version>.<architecture>.pkg.tgz
# tar -xvf illumio-ven-<ven_version>.<architecture>.pkg.tar
```

3. Install your trusted root CA certificate in the following directory with this exact specified filename:

```
/etc/certs/ca-certificates.crt
```

Ways to Install the Solaris VEN

You can install the Solaris VEN by specifying the Solaris `pkgadd` command and running the interactive VEN installation script; referred to as a “basic” installation in this topic.

Alternatively, you can use a Solaris Administration file or Solaris Response file (or both) to perform a non-interactive installation or set custom installation values (or both); referred to as an “advanced” installation in this topic.

Behavior During Each Type of Installation

SOLARIS FILES	BEHAVIOR
None	The VEN installation script performs a basic installation wherein you are prompted to set installation values or accept the default values.
Administration file only	The VEN installation script runs without prompting you for installation values (non-interactive) and uses only the default installation values; you cannot specify custom values.
Response file only	The VEN installation script launches an interactive installation; however, the prompts contain your custom values or the default value if not set. Press **Enter** to accept.
Both Administration and Response files	The VEN installation script runs without prompting you for installation values (non-interactive) and uses your custom values or the default value if not set. This method is the most automated of all the ways to install the Solaris VEN.

Basic Installation

1. Complete the tasks to prepare for Solaris VEN installation. See Installation Preparation [243] for information.
2. To install the Solaris VEN, enter the following command:

```
# pkgadd -d . illumio-ven-<ven_version>.<architecture>.pkg
```

> **IMPORTANT**
>
> When installing the Solaris VEN, enter the correct package for the Solaris machine architecture (SPARC or x86_64) you want to install:
>
> - `illumio-ven-<ven_version>.sol5.sparc.pkg`
> - `illumio-ven-<ven_version>.sol5.i386.pkg`

   The interactive VEN installation scripts starts.
3. Provide custom VEN installation and configuration values at the prompts or accept the defaults.

Solaris VEN installation is complete. The next step is to activate the Solaris VEN.

Advanced Installation

1. Complete the tasks to prepare for Solaris VEN installation. See Installation Preparation [243] for information.
2. Prepare the Solaris Administration and Response files for use in the installation. See To Use Customized Response and Administration Files [242] for information.
3. Enter the following command to perform a customized, non-interactive installation:

```
# pkgadd -d . -a pkgadd -a /tmp/admin.custom -r
/tmp/response.custom illumio-ven-<ven_version>.<architecture>.pkg
```

   Where the paths to the customized Administration and the Response files are the same ones you created when you extracted and copied them locally or to a network share. See To Use Customized Response and Administration Files [242] for information.

> **IMPORTANT**
>
> When installing the Solaris VEN, enter the correct package for the Solaris machine architecture (SPARC or x86_64) you want to install:
>
> - `illumio-ven-<ven_version>.sol5.sparc.pkg`
> - `illumio-ven-<ven_version>.sol5.i386.pkg`

Solaris VEN installation is complete. The next step is to activate the Solaris VEN.

Activate a Solaris VEN After Installation

After installing the VEN package on the Solaris host, activate the VEN with the Illumio VEN CTL (`illumio-ven-ctl`). The `--activate` option activates the workload and pairs the Solaris VEN with the PCE.

TIP

You can activate the Solaris VEN by using the VEN CTL or by specifying the values in the appropriate environment variables in the Solaris Response file. See Solaris Response File [241] for information.

NOTE

Activating the Solaris VEN on a workload that is running an LDAP client can take longer than on workloads not using LDAP.

At a minimum, to activate the Solaris VEN using the VEN CTL, you need the hostname or IP address of the PCE, an activation code (called a pairing key in the PCE web console) generated from a pairing profile, and any other available options, such as the workload policy state, label assignment, workload name, and more.

The following example shows how to activate the VEN and set its policy state to Illuminated:

```
# /opt/illumio_ven/illumio-ven-ctl activate --activation-code  <code>
 --management-server <fqdn:port>
```

Upgrade the Solaris VEN

IMPORTANT

Illumio strongly recommends that you upgrade VENs only during maintenance windows.

NOTE

If the VEN was activated prior to the upgrade, it does not need to be activated again after the upgrade completes.

Illumio supports both Solaris machine architectures: SPARC and x86_64. The upgrade steps for both machine architectures are identical but each architecture uses its own VEN package file.

For the supported upgrade paths for the Solaris VEN, see Upgrade VEN on the Illumio Support portal (login required).

Requirement: To upgrade the Solaris VEN, you must perform the upgrade by using the Solaris Administration file. Using the Response file with the upgrade is optional.

1. Download the new version of the VEN package from the Illumio Support site. See "Obtain the VEN Packages" for information.
2. Extract the Solaris VEN software. See Installation Preparation [243] for information.
3. Prepare the Solaris Administration file for the upgrade. See To Use Customized Response and Administration Files [242] for information.

   At a minimum, you must set the following values in the Administration file for the upgrade:

   ```
   mail=
   instance=overwrite
   conflict=nocheck
   action=nocheck
   ```
4. Stop the VEN if it's running:

   ```
   illumio-ven-ctl stop
   ```
5. Enter the following command to perform a non-interactive installation:

   ```
   # pkgadd -d . -a /tmp/admin.custom illumio-ven-<ven_version>.
   <architecture>.pkg
   ```

   Where the path to the customized Administration file is the same one you created when you extracted and copied it locally or to a network share.

   NOTE

   If you also need to customize settings for the upgrade, use a customized Response file for the upgrade and include the `-r` argument in the upgrade command; for example: `-r /tmp/response.custom`. See Solaris Response File [241] for information.

Uninstall the Solaris VEN

IMPORTANT

Before you uninstall the VEN software from a Solaris workload, unpair the VEN running on the workload from the PCE. See "Deactivate and Unpair VENs" in the VEN Administration Guide.

1. Enter the following commands to uninstall the VEN:

   ```
   sudo bash
   # cd /tmp
   # pkgrm illumio-ven
   ```
2. The following command output and prompts appear in the command window. Respond to the required prompts to uninstall the VEN:

   ```
   The following package is currently installed:
      illumio-ven  illumio-ven
                   (i386) <ven_version>.sol5.i386

   Do you want to remove this package? [y,n,?,q] y
   ```

```
## Removing installed package instance <illumio-ven>

This package contains scripts which will be executed with super-user
permission during the process of removing this package.

Do you want to continue with the removal of this package [y,n,?,q] y
## Verifying package <illumio-ven> dependencies in global zone
## Processing package information.
## Executing preremove script.
VEN_DATA : /opt/illumio_ven_data
Stopping venAgentMonitor: ...done
Stopping venAgentMgr:    ...done
Stopping venVtapServer:    ...done
Stopping venPlatformHandler:    ...done
## Removing pathnames in class <none>
.
.
.
## Executing postremove script.
## Updating system information.

Removal of <illumio-ven> was successful.
```

Reference

This section contains useful reference information for installing and upgrading VENs on workloads in your organization.

VEN Activate Command Reference

The following topic describes the commands for activating VENs either during or after installation, and the ways that you can configure the VEN during activation.

About the Command Options

If you are activating with a PCE that has a pairing profile configured to block changes to policy state (the `illumio-ven-ctl` option `--mode`) or label assignment (the `illumio-ven-ctl` options `--env`, `--loc`, `--role`, `--app`), you must not use these options on these blocked configurations or the activation will fail.

WARNING

When you use the VEN CTL or a pairing script to install a Windows VEN on a workload, you cannot include colons in the values for the options. Including a colon in a command value causes VEN activation to fail. For example, including the following values in the `-role` option, causes VEN activation to fail:

```
-role "R: UNKNOWN" -app "A:UNKNOWN" -env "E: UNKNOWN"
```

Activation fails because Windows uses the colon as a special character and cannot interpret the value even when you include quotation marks around the value.

Description of the `activate` Command Options

The options and arguments are the same for Windows and Unix (Linux, Solaris, and Solaris), except the options with two dashes on Unix should be replaced with a single dash on Windows (for example, `--loc` on Linux should be replaced with `-loc` on Windows).

NOTE

The following options are optional unless noted in the description.

Option	Arguments	Description
`activation file`	`<activation_file>`	**REQUIRED:** `--activation-file` is used to specify `activation-code` and `management-server` from a file rather than on the command line. The option `activation-code` is used for sensitive and confidential information. If you enter it on a command line, a non-root user on the machine could see it by running "ps -ef" or similar commands. For greater security, put the `activation-code` in the `activation-file` and specify the file path to the `activate` command.
`activation-code | -a`	`<activation_code>`	**REQUIRED:** Inputs the activation code of the VEN into the pairing script. This code is auto-generated by the pairing profile. Activation code: one-time use or unlimited use In the PCE web console, you can specify that an activation code is for one-time use or for unlimited uses. Be sure you have generated the correct type for your needs. Do not use a single one-time use activation code for more than one workload. Example: `--activation-code 1234567890abcdef`
`management-server | -m`	`<PCE_FQDN:port> | <IPaddress:port>`	**REQUIRED:** Sets the domain name or IP address and port of the host where the VEN can retrieve master configuration information. Example: `--management-server mypce.example.com:8443`
`name | -n`	`<server_friendly_name>`	Sets a friendly name that will be used for this workload when it appears in the PCE web console. Example: `--name "Web Server 1"`
`env`	`<environment_label>`	Assigns an Environment label for this workload. Example: `--env Production`
`loc`	`<location_label>`	Assigns a Location label for this workload. Example: `--loc "US"`
`role`	`<role_label>`	Assigns a Role label for this workload. Example: `--role "Dev Group"`
`app`	`<application_label>`	Assigns an Application label for this workload. Example: `--app "Web Service"`
`proxy_server`	`<proxy-string>`	**[Linux, Solaris, AIX only]** You only need to specify this option when configuring a proxy server for a Linux, Solaris, or AIX workload. Windows automatically detects a proxy server; therefore, you do not need to specify this option when installing a VEN on a Windows workload. For information about configuring a proxy server, see VEN Proxy Support [176].

Option	Arguments	Description
`log-traffic`	`true | false`	Enables or disables traffic logging. If not specified, logging is set to true by default. Default: `true` Interacts with the `visibility-level` option. See Allowable Combinations of log-traffic and visibility-level [250].
`mode`	`illuminated | enforced | idle`	Sets the policy state for the workload. For an explanation of the various states, see "Workload Policy States" in the VEN Administration Guide.
`enforcement_mode`	`full | visibility_only | selective | idle`	Default: `visibility_only` Enables the new `selective` mode for the VEN.
`visbility-level`	`flow_summary | flow_drops | flow_off`	Default: `flow_summary` Defines the extent of the data the VEN collects and reports to the PCE from a workload in the Full enforcement or Visibility policy states, so you can control resource demands on workloads. The higher levels of detail are useful for visualizing traffic flows in greater detail in the Illumination map inside the PCE web console. Interacts with the `--log-traffic` option. See Allowable Combinations of log-traffic and visibility-level [250].

`visibility-level` Arguments

Argument	Value in Policy States	Notes
`flow_summary`	Included in all policy states	Default. The VEN collects traffic connection details for both *allowed* and *blocked* connections: source and destination IP address and port and protocol. This argument creates traffic links in the Illumination map and is typically used initially after installing the VEN to determine the full scope of potential policy impact on the workload.
`flow_drops`	Valid only in full policy state	The VEN collects connection details only for *blocked* traffic: source and destination IP address and port and protocol. This argument produces less detail for Illumination but demands fewer workload system resources than `flow_summary`.
`flow_off`	Valid in all policy states	The VEN does not collect any details about traffic connections. This option produces no details for the Illumination map but requires the fewest number of workload resources. Useful when you are satisfied with policy rules and do not need additional detail.

Allowable Combinations of `log-traffic` and `visibility-level`

The following rules apply to using the `log-traffic` and `visibility-level` options together with the activate command:

- The `visibility-level` argument takes precedence over the `log-traffic argument`.
- `visibility-level flow_off` and `--log-traffic true` is an invalid combination.
- `visibility-level flow_drops` is invalid in Illuminated policy state.

VEN Modes in Illumio Core 20.2.0 and later

In Illumio Core 20.2.0, Illumio introduced a new feature called Selective Enforcement. For an explanation of how this feature changed policy functionality in the release, see Selective Enforcement in *What's New in This Release, 20.2.0*.

> **NOTE**
> This change to the VEN modes affects the VEN in Illumio Core 20.2.0 and later releases.

In particular, the feature changed the workload policy states. To further understand how the policy state changed between releases, see these topics:

- "Workload Policy State" in the *Security Policy Guide*, 19.3.x releases
- "Workload Enforcement States" in the *Security Policy Guide*, 21.2.0 release

The changes to policy states in 20.2.0 impacted the VEN `mode` option that you specify with the `activate` command in the following ways.

> **CAUTION**
> Do not use both the `mode` and `enforcement_mode` options together on the command line because you could specify contradictory options. Specify one or the other. To specify the new Selective Enforcement option, enter the `enforcement_mode selective` option and argument.

- Adds a new option for the `activate` command: `enforcement full|visibility_only|selective|idle`
- Retains the `mode` option from the previous release for backward compatibility
- The arguments for the `mode` option map to those for the `enforcement_mode` option in this way:
  - `illuminated` maps to `visibility_only`
  - `enforced` maps to `full`
  - `idle` is the same in both options
  - `enforcement_mode` option adds the `selective` argument

    Use the `selective` argument to set the VEN mode as Selective Enforcement state. For information about using Selective Enforcement in policy, see "Enforcement Modes for Rules" in the *Security Policy Guide*.
- The `visibility-level` option and argument are unchanged in Illumio Core 20.2.0 and later releases; however, the Selective Enforcement feature separated the workload policy visibility state from the policy enforcement state; in particular, the PCE web console has separate drop-down menus (**Enforcement** and **Visibility**) in the **Workloads** page for these two options.

The arguments for the `visibility-level` option map to the new visibility states in 20.2.0 and later releases in this way:

- `flow_off` maps to Off
- `flow_drops` maps to Blocked
- `flow_summary` maps to Blocked + Allowed

VEN Compatibility Check

This topic explains how to use the VEN Compatibility Check feature after installing VENs on workloads.

About Compatibility Checks

When you pair a 19.3.*x* VEN or later release in the Idle state or change the VEN state to Idle, the VEN performs several compatibility checks and sends the results to the PCE. This process occurs every 24 hours and checks whether the preexisting workload state will have issues when the VEN is moved out of the Idle state.

After reviewing the results of the VEN Compatibility Check, you can determine if the VEN is ready to be moved out of the Idle state or whether you need to resolve any detected issues, such as backing up any system firewall rules.

> **NOTE**
> The VEN Compatibility Check is performed per-workload and is available only for VENs in the Idle state, not Visibility, Selective, or Full states. If a workload reverts from any of these states to the Idle policy state, the VEN Compatibility Check is performed.

All detected issues are categorized as:

- **Red:** Major incompatibility detected
- **Yellow**: A potential incompatibility detected
- **Green**: No major incompatibilities detected

The Compatibility Check results are displayed in the PCE web console. To view the results:

1. Go to **Servers & Endpoints > Workloads** and click the name of the workload whose Compatibility Report you want to see.

   > **NOTE**
   > The workload must be in the Idle state for the Compatibility Report tab to appear (**Edit > Enforcement > Idle** then click **Save**.)

2. Click the **Compatibility Report** tab.

If no incompatibilities are detected on the VEN, the page displays "No Data to Display."

After viewing the results, you can export the report as a text file by clicking **Export**.

Beginning in 22.3.0-PCE, the Compatibility Report displays VEN packages that are required but are either missing or there's a problem with their installation. This information helps you troubleshoot a failed installation.

The compatibility checks vary by the workload's operating system.

Linux Operating Systems

Incompatibility Type	Reason for incompatibility with Illumio Core	Results
IPv4 forwarding enabled	At least 1 iptables forwarding rule is detected in the forwarding chain. VEN removes existing iptables rules in the non-Idle policy state.	Yellow
iptables rule count	At least 1 iptables filter rule is detected. VEN removes existing iptables rules in the non-Idle policy state.	Yellow
IPv6 global scope enabled	IPv6 is enabled for the workload.	Yellow
ip6tables rule count	At least 1 iptables filter rule is detected. VEN removes existing ip6tables rules in the Visibility policy state	Yellow
IPsec service enabled	UDP port 500/4500 is in use by other services. Do not enable SecureConnect for the workload.	Red
Routing table conflict	The StrongSwan routing table setting conflicts with existing networking routing tables. Do not enable SecureConnect for the workload.	Red

Windows Workloads

Incompatibility Type	Reason for incompatibility with Illumio Core	Results
IPv6 enabled	IPv6 is enabled for the workload.	Yellow
Virtual loopback interfaces	Virtual loopback interface is detected. Untested and unsupported configuration.	Yellow
Firewall GPO	Windows firewall Group Policy Object (GPO) is detected. For more information, see KB Article #3545 Firewall GPO Warning Under Compatibility Report (login required).	Yellow
IPsec service enabled	IKEEXT service is disabled. Do not enable SecureConnect for the workload.	Yellow

AIX and Solaris Workloads

Incompatibility Type	Reason for incompatibility with Illumio Core	Results
IPv4 forwarding enabled	IPv4 is enabled for the workload.	Yellow
iptables rule count	At least 1 iptables filter rule is detected. VEN removes existing iptables rules in the non-Idle policy state.	Yellow
IPv6 global scope enabled	IPv6 is enabled for the workload.	Yellow
ip6tables rule count	At least 1 iptables filter rule is detected. VEN removes existing ip6tables rules in the Visibility policy state	Yellow
IPsec service enabled	IPsec service is already in use. Do not enable SecureConnect for the workload.	Red

AIX Workloads only

Incompatibility Type	Reason for incompatibility with Illumio Core	Results
IPv6 active connection count	Complementary check whether IPv6 global scope is enabled.	

Pairing Script and Package Installation

The following information is provided for your reference so that you understand the process and events that occur when you install a VEN by using a pairing script in the PCE or by installing a package with the CLI.

For details about the VEN activate command, see VEN Activate Command Reference [247].

Linux Pairing Script for VEN Library

The following example shows a typical Linux pairing script. The pairing script works with the VEN Library in the PCE web console:

```
rm -fr /opt/illumio_ven_data/tmp && \
umask 026 && mkdir -p /opt/illumio_ven_data/tmp && \
curl --tlsv1 "https://example.com:8443/api/v18/software/ven/
image?pair_script=pair.sh&profile_id=2" -o /opt/illumio_ven_data/tmp/
pair.sh
&& \
chmod +x /opt/illumio_ven_data/tmp/pair.sh && \
/opt/illumio_ven_data/tmp/pair.sh \
--management-server example.com:8443 \
--activation-code <code>
```

This pairing script performs the following actions on the workload:

1. Deletes the `/opt/illumio_ven_data/tmp` directory, if it already exists.
2. Changes `umask` to `026` to prevent the `group-write` and `others-read,write` permissions as it creates the `/opt/illumio_ven_data/tmp` directory.

3. Uses curl to download the pairing script from the VEN repository and store it in the `/opt/illumio_ven_data/tmp` directory.
4. Changes the script permissions to allow execution.
5. Runs the `opt/illumio_ven_data/tmp` script with the following command line options:
   6. `--management-server` to communicate with the PCE
   7. `--activation-code` to authenticate the VEN to the PCE and authorize the VEN to pair with the PCE

The pair script installs the VEN packages on the workload and pairs the VEN with the PCE. The output of pair is captured in `/var/log/illumio_install.log`.

Next, the script performs the following operations:

1. Detects OS release and CPU architecture. Ensure the combination is supported.
2. Downloads the package to `/opt/illumio_ven_data/tmp`.
3. Uses native OS package manager (detected by line 1) to install the package.

   Using native package managers is simpler for newer operating systems. For example, Illumio can use yum to manage package dependencies for the VEN and workloads. For older operating systems, customers have to manage dependencies by manually installing packages.
4. Verifies installation by invoking installed scripts.
5. Invokes `/opt/illumio_ven/bin/init_Platform start`.
6. Generates the activation file `/opt/illumio_ven_data/etc/agent_activation.cfg`.
7. Invokes `/opt/illumio_ven/bin/agent_status.sh` to activate the VEN.

AIX Pairing Script for VEN Library

(Available starting in 23.2.20.)

The following example shows a typical AIX pairing script. The pairing script works with the VEN Library in the PCE web console:

```
rm -fr /opt/illumio_ven/tmp && \
umask 026 && mkdir -p /opt/illumio_ven/tmp && \
curl --tlsv1 "https://pce.example.com:8443/api/v26/software/ven/
image?pair_script=pair.aix.sh&profile_id=8" -o /opt/illumio_ven/tmp/pair.sh
&& \
chmod +x /opt/illumio_ven/tmp/pair.sh && \
/opt/illumio_ven/tmp/pair.sh \
--management-server pce.example.com:8443 \
--activation-code <code>
```

This pairing script performs the following actions on the workload:

1. Deletes the `/opt/illumio_ven_data/tmp` directory, if it already exists.
2. Changes `umask` to `026` to prevent the `group-write` and `others-read,write` permissions as it creates the `/opt/illumio_ven_data/tmp` directory.
3. Uses `curl` to download the pairing script from the VEN repository and store it in the `/opt/illumio_ven_data/tmp` directory.

> **NOTE**
>
> Before pairing workloads for the first time on an AIX host, you must take some preliminary steps to enable `curl` to work. See Pair an AIX Workload. [208]

4. Changes the script permissions to allow execution.
5. Runs the `opt/illumio_ven_data/tmp` script with the following command line options:

   6. `--management-server` to communicate with the PCE

   7. `--activation-code` to authenticate the VEN to the PCE and authorize the VEN to pair with the PCE

The pair script installs the VEN packages on the workload and pairs the VEN with the PCE. The output of pair is captured in `/var/log/illumio_install.log`.

Next, the script performs the following operations:

1. Detects OS release and CPU architecture. Ensure the combination is supported.
2. Downloads the package to `/opt/illumio_ven_data/tmp`.
3. Uses native OS package manager (detected by line 1) to install the package.

   Using native package managers is simpler for newer operating systems. For example, Illumio can use yum to manage package dependencies for the VEN and workloads. For older operating systems, customers have to manage dependencies by manually installing packages.
4. Verifies installation by invoking installed scripts.
5. Invokes `/opt/illumio_ven/bin/init_Platform start`.
6. Generates the activation file `/opt/illumio_ven_data/etc/agent_activation.cfg`.
7. Invokes `/opt/illumio_ven/bin/agent_status.sh` to activate the VEN.

RPM Installation

RPM installation performs the following operations:

- Creates the `ilo-ven` user and group, unless a custom username is specified at installation.

- Prepares and then starts the Illumio Core to perform the following actions:
  a. Loads the necessary kernel modules: `ip_tables`, `iptable_filter`, `nf_conntrack`, `nf_conntrack_ipv4`, `nf_conntrack_ftp`, `ipt_LOG`, `ip_set`, `ip6_tables`, `ip6table_filter`, `nf_conntrack_ipv6`, `ip6t_LOG`
  b. Sets `net.netfilter.nf_conntrack_tcp_timeout_established` to 8 hours (28,800 seconds). See Manage Conntrack Table Size (Linux)" in the VEN Administration Guide.
  c. Takes control of the system firewall.
  d. Disables and stops the system firewall service iptables.

     This action is acceptable because the Illumio services act in place of the iptables service.
  e. Saves existing iptables rules if any.
  f. Loads iptables rules computed from PCE firewall policy.
  g. Starts the VEN components described in "Description of VEN Components" in the *VEN Administration Guide*.

     This step includes monitoring system iptables configuration (similar to the service iptables performed).

Windows Pairing Script

The following example shows a typical Windows pairing script. The pairing script works with the VEN Library in the PCE web console. (*Line breaks have been added for readability only.*)

```
PowerShell -Command "& {Set-ExecutionPolicy -Scope process remotesigned
-Force;
Start-Sleep -s 3;
Set-Variable -Name ErrorActionPreference -Value SilentlyContinue;
[System.Net.ServicePointManager]::SecurityProtocol=[Enum]::ToObject
([System.Net.SecurityProtocolType], 3072);
Set-Variable -Name ErrorActionPreference -Value Continue;
(New-Object System.Net.WebClient).DownloadFile
('https://example.com:8443/api/v18/software/ven/
image?pair_script=pair.ps1&profile_id=1', (echo $env:windir\temp\pair.ps1));
 & $env:windir\temp\pair.ps1
-management-server example.com:8443 -activation-code <code> }"
```

This pairing script performs the following actions on the workload:

1. Changes execution policy of the host PowerShell process to RemoteSigned.
2. Configures the .NET framework `System.Net` class to negotiate TLS 1.2.

   The Windows VEN uses “3072” instead of “Tls12” because the `enum` value is not defined in older Windows operating systems. When `system.net` does not support TLS 1.2, the script fallbacks to using the system default.
3. Using the .NET framework `WebClient` class, downloads `pair.ps1` from the VEN repository and stores it in the `$env:windir\temp` directory.
4. Runs the `pair.ps1` script with the following command line options:
   - `-management-server`: Used by the VEN to communicate with the PCE
   - `-activation-code`: Used by the PCE to authenticate and authorize the VEN during the pairing process
5. The pairing script installs the VEN packages on the workload and pairs the VEN with the PCE. The output of `pair.ps1` is captured in `$env:windir\temp\illumio.log` or `$env:tmp\illumio.log`.

   The script performs the following actions:

   a. Downloads the VEN installer from the VEN repository and installs it.
   b. Generates the `agent_activation.cfg` file with PCE information
   c. Retrieves agent activation status and displays it.

VEN Compatibility with Workloads Hosting Containers

This topic describes how to enable VEN compatibility with workloads hosting containers. It covers how to configure the PCE as well as VENs that are supporting the underlying hosts.

NOTE

This topic describes only how to enable VEN compatibility with workloads hosting containers. It doesn't describe Illumio support for containers.

If you use an overlay network in your environment, Illumio strongly recommends that you use an orchestration platform to manage your containers. Illumio provides a complete solution for Kubernetes-based platforms. See Illumio Core for Kubernetes and OpenShift.

How it works

Illumio Core is compatible with using VENs to protect workloads that host containers. The VEN can apply out-of-band policy to traffic on the forwarding path of the firewall.

The VEN provides a capability called **Container Inherits Host Policy (CIHP)** whereby the VEN applies policies — written for a host workload — to the forwarded traffic. The host, in turn, forwards traffic that is inbound to the container to it's destination, and in this way the container traffic is subjected to the host firewall policy.

Requirements and Limitations

- The VEN's interaction with workloads hosting containers does not represent the container as a workload (map or policy object), and each container is implicitly part of the Docker host (the workload).
- Containers must share the same policies as the host.
- CIHP is supported on RHEL 8 / Oracle Linux 8 / Ubuntu 22 and later with Illumio Core 22.5 and later.
- In Illumio Core 21.5 and later, if you want to enforce traffic on the container's hosts only, allow all the traffic to containers and bypass CIHP by enabling IP forwarding. See Enable IP Forwarding.
- No support for segmentation between the host and containers on the same host.
- No support for segmentation between containers on the same host.

Verified container engines

NOTE

Make sure that the VEN is compatible on your version of Linux. (See the Illumio Support Portal.)

VEN interaction with workloads hosting containers has been verified to be compatible with the following container engines:

Container Engine	Supported Versions
Docker	19.03 and 20.10
Podman	3.0 and 4.0

Enable PCE compatibility with workloads hosting containers

1. Configure Containers to inherit host policy.

   This is disabled by default. Enablment copies the host policy (all Illumio iptables related rules) into the `filter:FORWARD` chain so that packets forwarded to containers are controlled by the Illumio security policy.

   Set a scope of Illumio labels for hosts with containers:

   a. Go to **Settings > Security**.

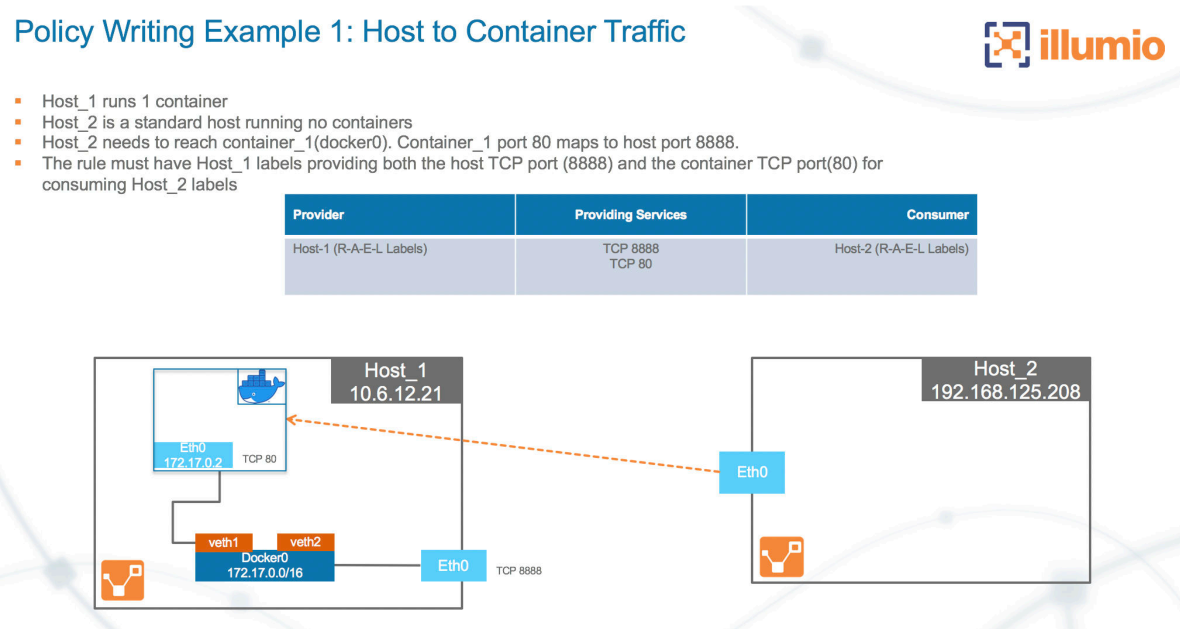
   b. Click the **Containers Policy** tab and then click **Edit**.

   c. Add scope for host with containers.

      You can define a narrow scope with specific label values or a broad scope that encompasses all workloads. For more information, see "Ruleset Scope" in the Security Policy Guide.

2. Pair workloads with containers. If they are already paired, go to the Security section of the workload's details page and verify that the workload's containers settings shows **Container Inherit Host Policy: Yes**

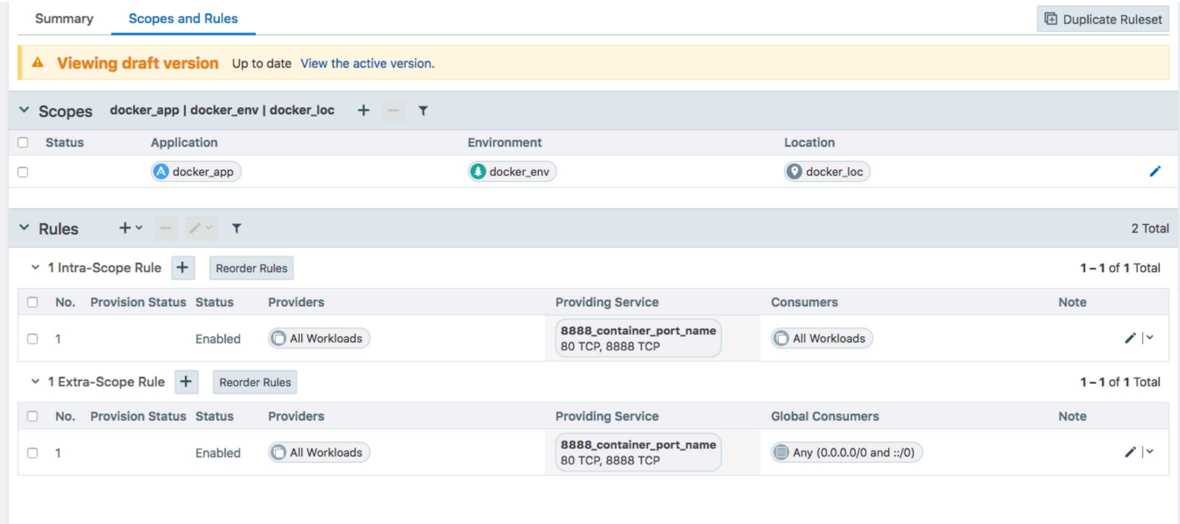
3. Write rules to cover the port mapping between the host and the container. Below is an example scenario of container access:



   To allow access to the container on Host_1 port 80, create an intra-scope rule for access within the application group and extra-scope rule for access from outside of the application group.

   The example rule below depicts any workload being allowed to access container port 80 on Host_1. Notice the service includes port 80 of container port and port 8888 of host port.

   Rule allowing access to container on Host_1:



Docker will have rules to NAT port 8888 to port 80.

4. Verify traffic flows to the containers on the VEN.

   Verify in the log at `/opt/illumio_ven_data/log/vtapdrop.log` that there is no dropped traffic to the containers. In the `/opt/illumio_ven_data/log/vtapflow.log` file, verify that there are flows to the containers on the VEN when workload is enforced.

   The highlighted log entry below below shows the flow between host_2 192.168.125.208:54253 to container on Host_1 172.17.0.2:80:

```
==> /opt/illumio_ven_data/log/vtapflow.log <==
2020-02-03T16:47:15.104-08:00 docker0 O 0 4 192.168.125.208 172.17.0.2
6 64886 80 12286 C 0 U SWID=3c1b9f96-969a-472e-bbbc-9d1c93751ef9 TBI=0
TBO=116
2020-02-03T16:48:15.157-08:00 docker0 O 0 4 192.168.125.208 172.17.0.2
6 64885 80 72339 C 0 U SWID=3c1b9f96-969a-472e-bbbc-9d1c93751ef9 TBI=0
TBO=116
```

   Inbound traffic from Host_2 to Host_1 will not be shown in `vtapflow.log`, only traffic to container IP address from Host_2.

Kubernetes and Openshift

Overview of Containers in Illumio Core

This section describes the architecture, key concepts, and the integration requirements to use Illumio Core with Kubernetes or OpenShift.

Before You Begin

- Prepare your environment
- Create a container cluster in the PCE
- Deploy Kubelink and C-VENs in your cluster
- Configure labels for namespaces, pods, and services
- Configure security policies for containerized environments
- Upgrade and uninstall the C-VEN in your containerized environments
- Migrate to a Helm Chart deployment from a previously-installed C-VEN deployment

Recommended Skills

- Illumio Core
- Linux shell (bash)
- TCP/IP networks, including protocols and well-known ports and a familiarity with PKI certificates
- Docker concepts, such as containers, container images, and docker commands.
  See Get Started with Docker.
- Red Hat OpenShift Container Platform.
  See OpenShift Documentation.
- Kubernetes concepts, such as clusters, Pod, and services.
  See Kubernetes Documentation.

Concepts

This section describes some key concepts of the solution.

Containerized VEN

Containerized VEN (C-VEN) is an Illumio-provided software component, which provides visibility and enforcement on nodes and Pods. In a standard Illumio deployment the Virtual Enforcement Node (VEN) is installed on the host as a package. The C-VEN is not installed on the host but runs as a Pod on the Kubernetes nodes. The C-VEN functions in the same manner as a standard VEN. However, in order to program iptables on the node and Pods namespaces, the C-VEN requires privileged access to the host. For details on the privileges required by the C-VEN, see Privileges [282].

The C-VENs are delivered as a DaemonSet with one replica per host in the Kubernetes cluster. A C-VEN Pod instance is required on each node in the cluster to ensure proper segmentation in your environment. In self-managed deployments, C-VENs are deployed on all nodes in the cluster. In cloud-managed deployments, C-VENs are deployed only on the Worker nodes and not on the Master nodes (Master nodes are not managed by Cloud customers).

Container Cluster

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. Each Pod network(s) that exists on a container cluster is uniquely identified on the PCEin order to handle overlapping subnets. This helps the PCE in differentiating between container workloads that may have the same IP address but are running on two different container clusters. This differentiation is required both for Illumination and for policy enforcement.

You can see the workloads that belong to a container cluster in the PCE Web Console. This mapping between the host workload and the container cluster is done using `machine-ids` reported by Kubelink and C-VEN.

Container Workloads

Container workloads are basic containers (as with Docker), or the smallest resource that can be assimilated within a container in an orchestration system (as with Kubernetes). In the context of Kubernetes and OpenShift, a Pod is a container workload. Similar to workloads, these container workloads (managed Pods) can have labels assigned to them. Container workloads with their associated Illumio labels are also displayed in Illumination. In Illumio Core, containers are differentiated based on whether they are on the Pod network or the host network:

- Containers on the Pod network are considered container workloads and can be managed similarly to workloads.
- Containers sharing the host network stack (Pods that are host networked) are not considered as container workloads and therefore inherit the labels and policies of the host.

To manage container workloads, you can define the policy state (Idle, Visibility Only, Selective, and Full) in container workload profiles.

Container Workload Profiles

A container workload profile maps to a Kubernetes namespace and defines:

- Policy state (Idle, Visibility Only, Selective, and Full) for the Pods and services that belong to the namespace.
- Labels (Role, Application, Environment, and Location) assigned to the Pods and services.

Once Illumio Core is installed on a container cluster, all namespaces that exist on the clusters are reported by Kubelink to the PCE and made visible via Container Workload Profiles. Each time Kubelink detects the creation of a namespace from Kubernetes, a corresponding Container Workload Profile object gets dynamically created in the PCE.

Each profile can either be in a managed or unmanaged state. The default state for a profile is unmanaged. The main difference between both states:

- Unmanaged: no policy applied to Pods by the PCE and no visibility
- Managed: policy is controlled by the PCE and full visibility through Illumination and traffic explorer

A container workload profile is a convenient way to dynamically secure new applications with Illumio Core just by inheriting security policies associated with the scope of that profile.

Kubelink

Kubelink is a software component provided by Illumio to make the integration between the PCE and Kubernetes easier. Kubelink queries Kubernetes APIs to discover nodes, networking details, and services and synchronizes them between the Kubernetes cluster and the PCE.

Kubelink reports network information to the PCE enabling the PCE to understand the cluster network for both the hosts and the Pods in the cluster. This enables the PCE to both accurately visualize the communication flow and create the correct policies for the C-VENs to implement in the iptables of the host and the Pods. It provides flexibility in the type of networking used with the cluster. Kubelink also associates C-VENs with the particular container cluster by matching a unique identifier of the underlying OS called `machine-id` reported by each C-VEN with the one reported by the Kubernetes cluster.

Kubelink is not in the critical path for normal Pod creation. When Pods start, stop, scale up or down, or nodes reboot, these events are all discovered by the C-VEN and the C-VEN provides enough information to the PCE to be able to immediately receive the security policy. There is no dependency on Kubelink to keep that information in sync.

Kubelink is delivered as a Deployment with only one replica within the Kubernetes cluster. One Kubelink Pod instance is required per cluster. There is no node affinity required for Kubelink, so the Kubelink Pod can be spun up on either a Master or Worker node.

Virtual Services

Virtual services are labeled objects and can be utilized to write policies for the respective services and the member Pods they represent.

Kubernetes services are represented as virtual services in the Illumio policy model. Kubelink creates a virtual service in the PCE for services in the Kubernetes cluster. Kubelink reports the list of Replication Controllers, DaemonSets, and ReplicaSets that are responsible for managing the Pods supporting that service.

Workloads

A workload is commonly referred to as a host OS in Illumio Core. In the context of container clusters, a workload is referred to as a node in a container cluster. Usually, a Kubernetes cluster is composed of two types of nodes:

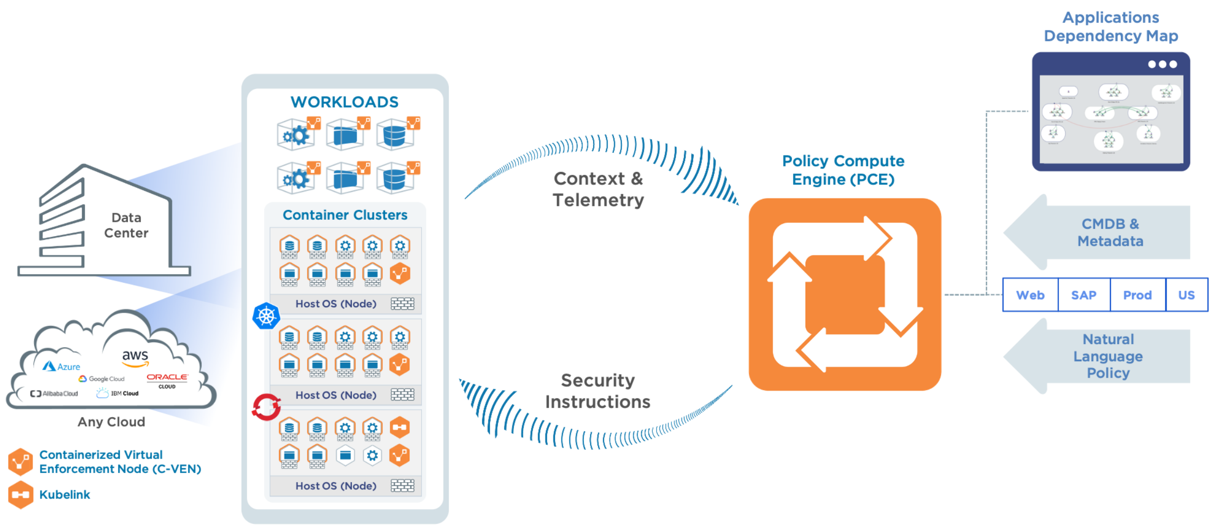
- One or more Master Node(s) - In the control plane of the cluster, these nodes control and manage the cluster.
- One or more Worker Node(s) - In the data plane of the cluster, these nodes run the application (containers).

In Illumio Core, Master and Worker nodes are called workloads and are part of a container cluster. Labels and policies can be applied to these workloads, similar to any other workload that does not run containers. For a managed Kubernetes solution, only the Worker nodes are visible to the administrator and the Master nodes are not displayed in the list of Workloads.

Architecture

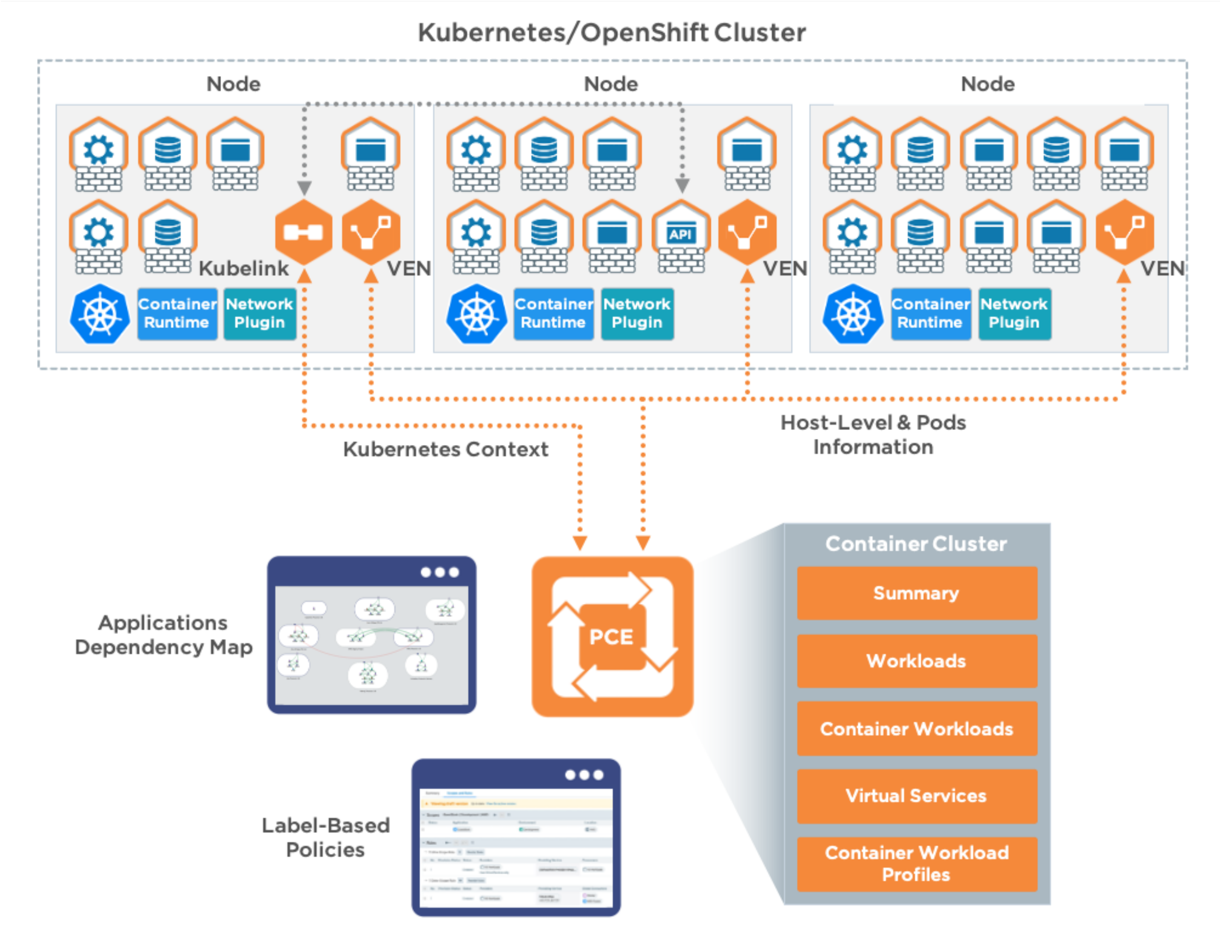
With the increased adoption of containers, the threat of unauthorized lateral movement from vulnerabilities and exploits increases considerably in the east-west attack surface. In addition, consumers and providers may be other containers, bare-metal servers, or virtual machines running on-premises or in the cloud. Multiple disparate solutions create complexity in management and operational workflow, leaving your organization more open to attack.

Illumio Core provides a homogenous segmentation solution for your applications regardless of where they are running - bare-metal servers, virtual machines, or containers. It is a single unified solution with many points of integration, including how you can easily and quickly secure your applications regardless of their location or form.



A container is a loosely defined construct that abstracts a group of processes into an addressable entity, which can run application instances inside it. Containers are implemented using Linux namespaces and cgroups allowing you to virtualize and limit system resources. Since containers operate at a process-level and share the host OS, they require fewer resources than virtual machines. The isolation mechanism provided through Linux namespaces allows containers to have unique IP addresses. Illumio Core uses these mechanisms to program iptables in the network namespace.

Illumio Core for containers orchestrated with Kubernetes or OpenShift, uses the following architecture:



Kubernetes-based orchestration platforms such as native Kubernetes and Red Hat OpenShift integrate with Illumio Core by using the following two components in the cluster:

- Kubelink - An Illumio software component that listens to events stream on the Kubernetes API server.
- Containerized VEN (C-VEN) - An Illumio software component that provides visibility and enforcement on the nodes and the Pods.

Once these components are deployed in the cluster, they both report the following information to the Policy Compute Engine (PCE):

- Summary - Information about the Kubernetes cluster and Illumio components deployed.
- Workloads - Information about Kubernetes nodes.
- Container Workloads - Information about Kubernetes Pods.
- Virtual Services - Information about Kubernetes services.
- Container Workload Profiles - Information about Kubernetes namespaces and policies.

Illumio Core visibility and enforcement occur at the Pod level in Kubernetes and OpenShift, with policies programmed into the iptables in the namespace provided by the Pod. This means only the Pods can be segmented but containers inside a Pod cannot be segmented. The Pod is represented as a single container workload in the PCE, with the C-VEN providing details about the containers that are a part of the Pod.

Configure Labels for Namespaces, Pods, and Services

Once Kubelink is deployed onto the Kubernetes cluster and it gets synced with the PCE, the namespaces within the cluster appear as Container Workload Profiles. By default, all namespaces are unmanaged, which means Illumio does not apply any inbound or outbound controls to the Pods within those namespaces. Any Pods or services within unmanaged namespaces do not show up in the PCE inventory or in Illumination.

Use Container Workload Profiles

The Illumio PCE administrator can change a Kubernetes namespace from unmanaged to managed by modifying the Container Workload Profile. Each profile can be modified even if the Illumio C-VEN is not yet installed on the Kubernetes nodes. If the C-VEN is deployed on the cluster nodes and Container Workload Profile is in the managed state, the Pods and services are displayed in Illumination and they inherit the labels assigned to the Kubernetes namespace. The Pods are represented in Illumio Core as Container Workloads. If Kubernetes services exist in the respective namespace, Illumio Core represents each service as an Illumio Core Virtual Service object.

This section describes how to change a namespace from unmanaged to managed and how to use edit labels and use custom annotations to add more context to your applications. This section also describes how to set enforcement boundaries for your containerized workloads.

1. Log in to the PCE UI and navigate to **Infrastructure** > **Container Clusters**.
2. Select the **Container Cluster** you want to manage.
3. Select the **Container Workload Profiles** tab.
4. You will see a list of all namespaces in the cluster. Select the namespace you want to manage.
5. Click **Edit**:
   a. Enter a Name (optional).
   b. Select a Management state (any state, except unmanaged).
   c. Select an Enforcement mode for how policy rules will be enforced.
   d. Select a Visibility state.
   e. Assign Labels (optional).
   f. Click **Save**.

Configure New Container Workload Profiles

A Container Workload Profile is beneficial when you want to assign labels to resources that are deployed in a namespace and also define the state of the policy created for the scope of labels assigned. A new Container Workload Profile can be created in either of the following ways:

- Dynamically created through the creation of a new namespace in the Kubernetes or OpenShift cluster. This is a *reactive* option in which the Illumio Core Administrator assigns labels and a policy state after the creation of the namespace.
- Manually pre-created to assign labels and a policy state to a namespace that will be created later on. This is a *proactive* option in which the Illumio Core Administrator assigns labels and a policy state before the creation of the namespace. This option offers the best-in-class security mechanism and authenticates each namespace created in the cluster by leveraging the concept of pairing key (same concept that Illumio Core provides in a pairing profile).

TIP

For a best-in-class security deployment, Illumio recommends to *proactively* create pairing profiles and assign labels and a policy state to them. The pairing key for each profile can be provided to the DevOps team for namespaces deployments later on.

When a Container Cluster is created for the first time in the PCE, Kubelink will report the existing namespaces or projects in the cluster. These namespaces will inherit what was defined as part of the Container Workload Profile Template for that cluster.

Dynamic Creation of a Profile

When the team managing Kubernetes or OpenShift clusters creates a namespace in a cluster, this namespace is reported immediately to the PCE via Kubelink. The new namespace will be listed under Container Workload Profiles and the following scenarios can occur:

- A Container Workload Profile Template exists for this cluster - The new namespace will inherit what was defined in the template, as far as Policy state and labels are concerned.
- A Container Workload Profile Template does not exist for this cluster - The new namespace will remain blank until further edited by an Illumio Core Administrator.

The example below shows a new namespace "namespace1" created in a cluster where a Container Workload Profile Template exists with a policy state set to "Build" and a partial label assignment as "Development | Cloud":

NOTE

The namespace is created by the Kubernetes or OpenShift administrator (outside the scope of Illumio Core).

For example, to edit the "namespace1" namespace:

1. Click on it and then click **Edit**.
2. Enter a *Name*.
3. Assign missing *Labels* wherever relevant or modify the existing ones.
   See Labels Restrictions for Kubernetes Namespaces [269].
4. After you are done, click **Save**.

The updates are displayed in the *Container Workload Profiles* list.

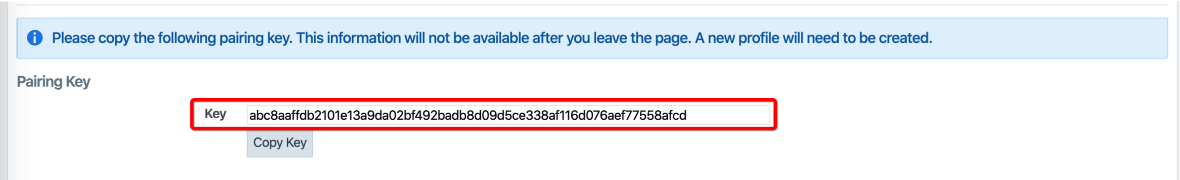
Manual Pre-creation of a Profile

To pre-create a profile:

1. In the Container Workload Profiles page, click **Add**.
2. Enter a *Name*.

3. Select the desired *Management* state.
4. Select the Enforcement state.
5. Choose a **Visibility** state. Note that Enhanced Data Collection is an optional feature that you must contact Illumio Support to enable.
6. Assign *Labels* to the profile.

   See Labels Restrictions for Kubernetes Namespaces [269].
7. Click **Save**.



8. Click **Copy Key** and provide this key to the DevOps team, which will be used as an annotation in a namespace manifest file to authenticate this resource with the PCE.

You can view the newly-created Container Workload Profile. The status is in "Pending" state with the hourglass icon displayed next to it.

To edit the namespace configuration file to include the pairing key in order to authenticate this namespace with the PCE:

1. Navigate to `metadata: > annotations:`. If `annotations:` does not exist, create an `annotations:` section under `metadata:`.
2. Add the `com.illumio.pairing_key:` Illumio label key field under the `annotations:` section.
   - Enter the pairing key obtained during the new Container Workload Profile creation.
   - Save the file and exit.
3. Apply the change using `kubectl` commands.

An example is show below.

```
apiVersion: v1
kind: Namespace
metadata:
  name: namespace2
  annotations:
    com.illumio.pairing_key:
    abc8aaffdb2101e13a9da02bf492badb8d09d5ce338af116d076aef77558afcd
```

The updates are displayed in the *Container Workload Profiles* list.

Set Enforcement

Set an Enforcement Boundary to establish how policy rules affect traffic to and from namespace workloads. Enforcement boundaries can be one of:

- **Visibility Only** - Rules are enforced an any traffic
- **Selective** - Rules are enforced only for selected traffic
- **Full** - Rules are enforced for all traffic

An enforcement boundary can be applied only to Managed workloads, which means Idle workloads cannot have an enforcement state applied to them.

You can change Enforcement for multiple profiles of the same current Enforcement level by selecting the checkboxes for the desired profiles (or by selecting the checkbox in the table heading row to select all profiles), and then hovering over the Enforcement button, which then shows a list of new Enforcement states and how many profiles will be changed to that state. Note that when you change Enforcement to Selective, then Visibility mode must by Blocked & Allowed, which is automatically done for you.

Labels Restrictions for Kubernetes Namespaces

At a high level, creating policy for containerized applications functions in the same basic way as for other types of applications running on bare-metal servers and virtual machines protected by the Illumio Core. Container workloads are assigned multi-dimensional labels to identify their roles, applications, environments, locations (RAEL), or other custom label types. These labels can then be used to apply security policies to specific parts of the containerized application environment. The PCE converts these label-based policies into rules that can be applied to the container workloads.

In previous releases, the PCE supported two options for assigning labels to container workloads:

- When creating or editing a container workload profile in the PCE web console or by using the Illumio Core REST API, an Illumio administrator assigned labels for the resources in that Kubernetes namespace.
- The Illumio administrator did not assign labels in the container workload profile. The DevOps/SRE team could use custom annotations in the service and deployment manifest files (YAML) to apply labels to the pods and services running in a namespace. On receiving this information from Kubelink, the PCE applied these labels to the container workloads, as long as the labels matched existing labels in the PCE.

These two ways of assigning labels for container workloads are sufficient for most container segmentation uses cases; however, this approach lacks the flexibility with label assignment for namespaces requested by Illumio customers. However, there is an alternative in addition to those two options that still allows developers/DevOps teams to assign their own labels for Kubernetes pods and services, but at the same time restricts the list of labels that they can assign. Illumio administrators now have a way to control which labels can be assigned by the developers managing their Kubernetes environments.

Options for Assigning Labels with a Container Workload Profile

You assign labels with container workload profiles in a number of ways:

- By creating a new container workload profile; see Manual Pre-creation of a Profile [267].
- By editing a container workload profile that was dynamically created in the PCE when Kubelink imported a new Kubernetes namespace; see Dynamic Creation of a Profile [267].
- By specifying label assignments in the default settings for the container workload profile template; see Configure New Container Workload Profiles [266].

Previously, four standard label types were predefined (Role, Application, Environment, and Location) for setting labels with a container workload profile. Now you can define custom label types and values in addition to these four predefined labels. You also have the following options:

- Do not allow a label for a specific label type (the “None” option).

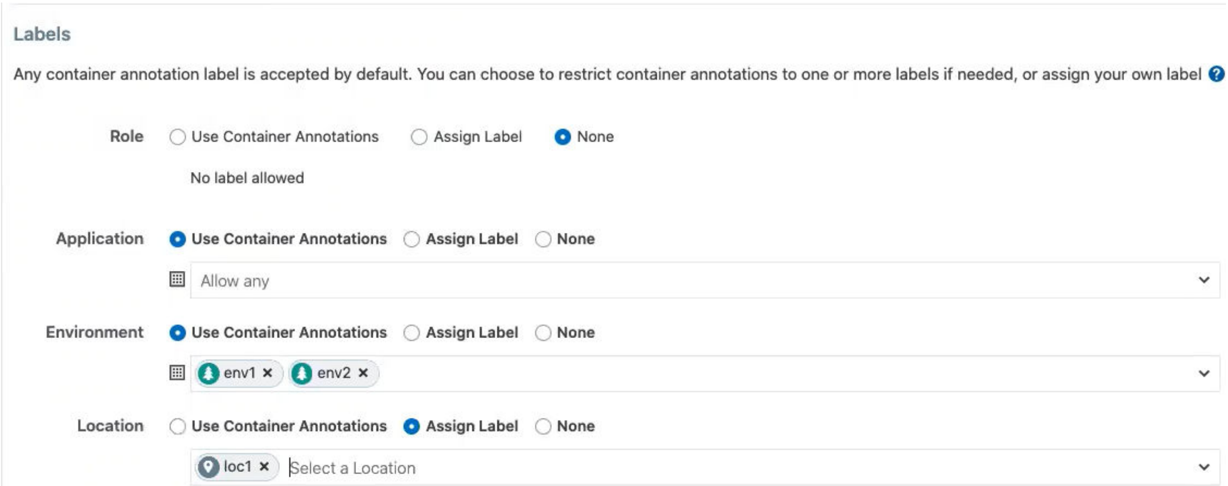
- Allow developers to assign any label from Kubernetes for a specified label type (the "Use Container Annotations" option); so long as the labels match ones in the PCE.

In previous releases, when the PCE administrator left the labels unassigned in the GUI or through the REST API, labels specified in annotations were used. Now the "Use Container Annotations" option is selected by default for all labels in a container workload profile (provided the default settings for the cluster are not configured).

- Specify a list of labels that are allowed for that label type.
- Fix a label to a specific label for that label type (the "Assign Label" option).

Example: Assigning Labels with a Container Workload Profile

The following example shows how you can use each of the four standard predefined options:



The Role label annotation (com.illumio.role) is ignored when passed at runtime and reported by Kubelink to the PCE when "Role" label is set to "None".

Adding, Editing, or Removing Labels

To add one or more labels:

1. Click a profile name, then click **Edit**.
   To apply the same label edits to multiple profiles, click the checkboxes for the desired profiles (or click the topmost checkbox in the table heading to select all profiles), then click **Edit Labels**.
2. Under the Labels heading, add a label type by clicking the field under the Label Type heading and then choose a label type from the list.
3. Choose a Label Assign Type:
   - **Use Container Annotations** - Use label values for container annotations. See the topic "Using Annotations" for more information.
   - **Assign Label** - Explicitly set labels from the values configured for this label type.
   - **No Label Allowed** - Prevent this label type from being used in this profile.
4. Specify labels for this label type by clicking the field under Labels Allowed/Label Assign, and choosing a label from the list.
   Any label type defined from annotations or explicit assignments must also have a label value specified in order to add the label definitions to the profile.

5. Click **Save** when finished.

To remove label types (and their associated label values):

1. Click the profile name.
2. Click **Edit**.
3. Under the Labels heading, choose one or more types to delete from this profile by clicking the checkboxes in front of the Label Type name.
4. Click **Remove**.

To remove or change label values:

1. Click the desired profile name.
2. Click **Edit**.
3. In the Labels table, remove a label value by clicking the small "x" near the label name under the Labels Allowed/Label Assign column.

   You can replace or add label values by clicking the **Select Label** or **Select Labels** field under Labels Allowed/Label Assign column, and then choosing the new label (or in the case of Annotations, multiple labels).
4. Click **Save**.

Possible Labels for the Example

- Developers can specify any label for Applications, so long as the label matches a preexisting label in the PCE.
- For Environment, a list of two labels (env1 and env2) is available. Developers can set either of these labels in Kubernetes. If a developer sets another value for the Environment label as a Kubernetes annotation, the PCE considers it invalid and, as a result, a label is not assigned to that label key. Because the wrong label is assigned, the policy will not allow expected traffic from other services or applications with the Environment label env1.
- The Location label is fixed as the loc1 label. If a developer assigns another Location label (for example, loc3, which is a label in the PCE) or the developer leaves the Location label empty, the PCE overrides what the developer has specified in the annotation and the PCE assigns loc1 for the Location label.

The label assignments for that namespace appears in the Container Clusters list in the PCE web console.

For this example, you can see the label assignments mirrored in the Kubernetes annotation for the namespace:

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: web-app-1
  namespace: demo
spec:
  replicas: 1
  selector:
    matchLabels:
      app: app1
  template:
    metadata:
      annotations:
        com.illumio.role: role1
        com.illumio.app: app1
        com.illumio.env: env1
        com.illumio.loc: loc100
      labels:
        app: app1
    spec:
      containers:
      - name: app1
        image: kodekloud/webapp-conntest
        ports:
        - containerPort: 8080
```

Kubernetes Annotation for Namespaces

where developers set **role1, app1**, **env1**, and **loc100** for the labels in the annotations.

Kubelink passes this data to the PCE at runtime. The PCE ignores the Role label because it's not allowed in the profile. The PCE accepts the Application and Environment labels. It ignores the **loc100** label and uses **loc1** instead.

In the Container Workloads tab, you can see how the label assignments are applied for the pod in this example.

> **NOTE**
> If a developer sets another value for the Environment label as a Kubernetes annotation, the PCE considers it invalid and, as a result, a label is not assigned to that label key. Because the wrong label is assigned, the policy will not allow expected traffic from other services or applications.
>
> For example, if developers leave the Environment label empty or specify env100 in the Kubernetes annotations, the following labels are used for the namespace, and there are no policy for applications or services with the Environment label env1.

Effect of Upgrading the PCE to Core 21.1.0 or Later

After upgrading your PCE to Core 21.1.0 or later, the labels assignments for your Kubernetes namespaces are not impacted operationally. However, you will see changes in the PCE web console and in the REST API.

- The values set in the PCE in the previous Core release are unchanged and the "Assign Label" option is selected in the PCE web console and through the REST API.
- The values left open so that container annotations were used for label assignments are updated to the "Use Container Annotations" option and the label assignments won't be restricted by any settings in the PCE web console or through the REST API.

Using Annotations

> **NOTE**
> Illumio annotations operate differently in CLAS-mode clusters (optionally available starting in Illumio Core for Kubernetes version 5.0.0) than in previous legacy (non-CLAS) environments.
>
> The initial portion of this topic describes how to use annotations in legacy non-CLAS clusters. After this initial portion, in the latter part of this topic, you can find information about using annotations in CLAS-mode clusters, described in the section Using Annotations in CLAS [279].

When assigning labels, you can assign no labels, some labels, or all labels to the namespace. If there is a label that is not assigned, then you can insert annotations in the Deployment configuration (or application configuration) to assign labels. If there is a conflict between a label assigned via the Container Workload Profile and the annotations in the deployment configuration, the label from the Container Workload Profile overrides the deployment configuration file. This security mechanism ensures that a malicious actor cannot spoof labels and get a preferential security policy based on a different scope. Regardless of how you assign labels, it is not required for Pods or services to have all labels in order for the PCE to manage them.

To manually annotate the different resources created in a Kubernetes namespace or OpenShift project, use the steps described in the sections below.

Deployments

1. Edit the Deployment configuration file:
   a. Navigate to `spec: > template: > metadata: > annotations:`. If `annotations:` does not exist, create an `annotations:` section underneath `metadata:`.
   b. The annotation can support any Illumio label key fields, including user-defined label types, as well as the standard set of predefined Illumio labels:
      - `com.illumio.role:`
      - `com.illumio.app:`
      - `com.illumio.env:`
      - `com.illumio.loc:`
   c. Fill in the appropriate labels.
   d. Save the file and exit.
2. Propagate your changes to all Pods.

Services

1. Edit the Deployment configuration file:
   a. Navigate to `metadata: > annotations:`. If `annotations:` does not exist, create an `annotations:` section underneath `metadata:`.
   b. The following Illumio label key fields can be under the `annotations:` section.
      - `com.illumio.role:`
      - `com.illumio.app:`
      - `com.illumio.env:`
      - `com.illumio.loc:`
   c. Fill in the appropriate labels.
   d. Save the file and exit.
2. Propagate your changes to all Pods.

IMPORTANT
When using the annotations method, you should redeploy the Pods or services after saving the changes to the configuration files by using the `kubectl apply` command.

Annotation Examples

Below are examples of namespaces, Pods, and services that use label assignments using either Container Workload Profiles or Container Workload Profiles with annotation insertion.

In the example shown below:

- Kubernetes default services or control plane Pods exist within namespaces such as, `kube-system`. They will inherit the Application, Environment, and Location labels from what has been configured in the Container Workload Profile(s). Kubelink is part of the `illumio-system` namespace, and because the Role label is left blank on the `illumio-system` namespace, you should assign a Role to Kubelink using annotations in the manifest file.

- A new `app1` namespace that contains two different Deployment objects for a two-tier application (Web and Database) is deployed. To achieve tier-to-tier segmentation across the application they will need different Role labels. Therefore, a Role label should be inserted into the annotations of each Deployment configuration.

A snippet of the `illumio-kubelink` Deployment configuration file is shown below, and the "Kubelink" Role label is inserted under the `spec: > template: > metadata: > annotations:` section:

illumio-kubelink-kubernetes.yml

```
spec:
  replicas: 1
  selector:
    matchLabels:
      app: illumio-kubelink
  template:
    metadata:
      annotations:
        com.illumio.role: Kubelink
      labels:
        app: illumio-kubelink
    spec:
#      nodeSelector:
#        node-role.kubernetes.io/master: ""
      serviceAccountName: illumio-kubelink
      tolerations:
      - key: node-role.kubernetes.io/master
        effect: NoSchedule
```

A snippet of the `app1`'s Web Deployment configuration file is shown below, and the "Web" Role label is inserted under the `spec: > template: > metadata: > annotations:` section:

shopping-cart-web.yml

```
spec:
  replicas: 3
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: webapp1
      tier: frontend
  strategy:
    activeDeadlineSeconds: 21600
    resources: {}
    rollingParams:
      intervalSeconds: 1
      maxSurge: 25%
      maxUnavailable: 25%
      timeoutSeconds: 600
      updatePeriodSeconds: 1
    type: Rolling
  template:
```

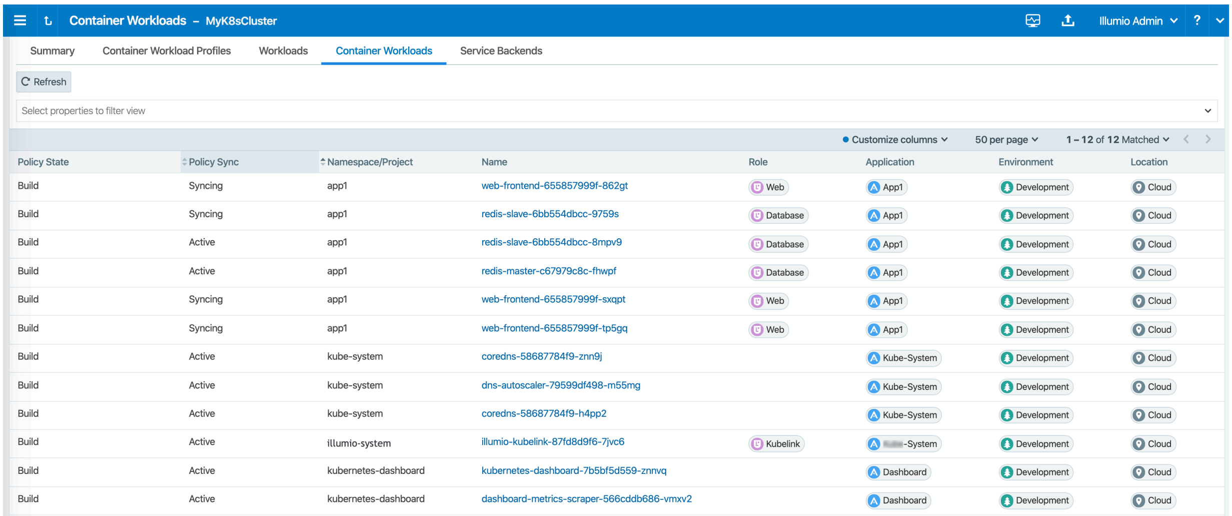
```
    metadata:
      annotations:
        com.illumio.role: Web
      creationTimestamp: null
      labels:
```

A snippet of the `app1`'s Database Deployment configuration file is shown below and the "Database" Role label is inserted under the `spec: > template: > metadata: > annotations:` section:

shopping-cart-db.yml

```
spec:
  replicas: 2
  revisionHistoryLimit: 10
  selector:
    matchLabels:
      app: redis
      role: slave
      tier: backend
  strategy:
    activeDeadlineSeconds: 21600
    recreateParams:
      timeoutSeconds: 600
    resources: {}
    type: Recreate
  template:
    metadata:
      annotations:
        com.illumio.role: Database
      creationTimestamp: null
      labels:
```

Below is the final outcome of the label assignment from the example.



Policy State	Policy Sync	Namespace/Project	Name	Role	Application	Environment	Location
Build	Syncing	app1	web-frontend-655857999f-862gt	Web	App1	Development	Cloud
Build	Syncing	app1	redis-slave-6bb554dbcc-9759s	Database	App1	Development	Cloud
Build	Active	app1	redis-slave-6bb554dbcc-8mpv9	Database	App1	Development	Cloud
Build	Active	app1	redis-master-c67979c8c-fhwpf	Database	App1	Development	Cloud
Build	Syncing	app1	web-frontend-655857999f-sxqpt	Web	App1	Development	Cloud
Build	Syncing	app1	web-frontend-655857999f-tp5gq	Web	App1	Development	Cloud
Build	Active	kube-system	coredns-58687784f9-znn9j		Kube-System	Development	Cloud
Build	Active	kube-system	dns-autoscaler-79599df498-m55mg		Kube-System	Development	Cloud
Build	Active	kube-system	coredns-58687784f9-h4pp2		Kube-System	Development	Cloud
Build	Active	illumio-system	illumio-kubelink-87fd8d9f6-7jvc6	Kubelink	-System	Development	Cloud
Build	Active	kubernetes-dashboard	kubernetes-dashboard-7b5bf5d559-znnvq		Dashboard	Development	Cloud
Build	Active	kubernetes-dashboard	dashboard-metrics-scraper-566cddb686-vmxv2		Dashboard	Development	Cloud

In Illumination Map, the application groups will appear differently if you've assigned labels on resources in the cluster.

DaemonSets and ReplicaSets

The steps described in the above section apply only to services in Kubernetes and OpenShift that are bound to Deployment or DeploymentConfig (existing deployments). This is because Kubelink depends on the Pod hash templates to map resources together, templates that DaemonSet and ReplicaSet configurations do not have. If you discover Pods derived from DaemonSet or ReplicaSet configurations and also discover services bound to those Pods, then Kubelink will **not** automatically bind the virtual service and service backends for the PCE. The absence of this binding will create limitations with Illumio policies written against the virtual service.

To work around this limitation for DaemonSets and ReplicaSets follow the steps below.

1. Generate a random uuid using the `uuidgen` command (on any Kubernetes or OpenShift node, or your laptop).
2. Copy the output of the `uuidgen` command.
3. Edit the DaemonSet or ReplicaSet YAML configuration file.
4. Locate the `spec: > template: > metadata: > labels:` field in the YAML file and create the `Pod-template-hash:` field under the `labels:` section.
5. Paste the new uuid as the value of the `Pod-template-hash:` field.
6. Save the changes.

Repeat steps 1 through 6 for each DaemonSet or ReplicaSet configuration.

The examples below generate a random `pod-template-hash` value and applies it to a DaemonSet configuration.

```
$ uuidgen
9e6f8753-d8ac-11e8-9999-0050568b6a18

$ cat nginx-ds.yml
apiVersion: extensions/v1beta1
kind: DaemonSet
metadata:
  name: nginx-webserver
spec:
  template:
    metadata:
      labels:
        app: nginx-webserver
        pod-template-hash: 9e6f8753-d8ac-11e8-9999-0050568b6a18
    spec:
      containers:
        - name: webserver
          image: rstarmer/nginx-curl
          imagePullPolicy: IfNotPresent
          ports:
          - containerPort: 80
```

Static Pods

Another way of deploying Pods without Deployments or ReplicaSet is by using 'Static Pods'. In this case, a Pod is spun up by not depending on the API server and is managed by an individual node's Kubelet. Static Pods are used to spin up control-plane components such as, kube-apiserver, controller-manager, and scheduler. Static Pods are useful if you want a pod to be running even if the Kubernetes control-plane components fail. Unlike Naked Pods, if a Static Pod is not functional, kubelet spins up a new Static Pod automatically by looking at the manifest file in the `/etc/kubernetes/manifests` directory.

Services for such pods can also be created without any selectors. In which case, you need to manually create the `EndPoint` resources for such services without a selector. For example, the default 'kubernetes' service in the default namespace which binds to the API-Server Pod running on HostNetwork.

If you create Static Pods on an overlay network, you need to create a service without selectors and manually create `EndPoint` resource to map the Pod to see the Container Workload and the Virtual Service on the PCE. You will not see any bindings or backends for this Virtual Service. In order to bind the Static Pods to the Virtual Service, use the '`com.illumio.service_uids`' annotation in the Static Pods manifest and configure the service without selectors and manually create the EndPoints. Once the '`com.illumio.service_uids`' annotation is used, you can bind the Container Workloads to its Virtual Service.

Sample code: Place the Static Pod manifest in the `/etc/kubernetes/manifests` directory

```
[root@qvc-k8s-027-master01 manifests]# pwd
/etc/kubernetes/manifests

[root@qvc-k8s-027-master01 manifests]# cat network-tool.yml
apiVersion: v1
kind: Pod
metadata:
  name: nw-tool1
  annotations:
    com.illumio.service_uids: <numerical-value>
spec:
  containers:
  - name: nw-tool1
    image: praqma/network-multitool
    args: [/bin/sh, -c, 'i=0; while true; do echo "$i: $(date)"; i=$
((i+1)); sleep 10; done']
    imagePullPolicy: IfNotPresent
  restartPolicy: Always

[root@qvc-k8s-027-master01 ~]# cat nw-tool-endpoint.yaml
apiVersion: v1
kind: Endpoints
metadata:
  name: nw-tool-svc
  namespace: default
subsets:
- addresses:
  - ip: <ip-value>
  ports:
  - name: http
    port: 80
    protocol: TCP

[root@qvc-k8s-027-master01 ~]# cat nw-tool-svc.yaml
```

```
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: "2020-05-18T18:39:19Z"
  labels:
    app: nw-tool
  name: nw-tool-svc
  namespace: default
  resourceVersion: "29308511"
  selfLink: /api/v1/namespaces/default/services/nw-tool-svc
  uid: <numerical-value>
spec:
  clusterIP: <ip-value>
  ports:
  - name: http
    port: 80
    protocol: TCP
    targetPort: 80
  sessionAffinity: None
  type: ClusterIP
status:
  loadBalancer: {}
[root@qvc-k8s-027-master01 ~]#
```

IMPORTANT

In the above code sample, you need to modify the following two values based on your configuration:

- `uid: <numerical-value>`
- `clusterIP: <ip-value>`

Using Annotations in CLAS

Illumio annotations in CLAS-mode environments are specified on the Kubernetes Workload, and not on a Pod's template, as is done in legacy non-CLAS environments. This distinction follows from the concept of the Kubernetes Workload in the PCE UI introduced with CLAS-mode, which maps directly to the native Kubernetes concept of a workload resource (that is, Deployments, ReplicaSets, and the like).

Therefore, Kubernetes Workloads on the PCE should be labelled based on the corresponding workload annotations in Kubernetes, instead of on individual pod template annotations in Kubernetes.

This labelling distinction prevents confusion, because Pods from a single Deployment can have different annotations:

```
# kubectl get pod azure-vote-front-6fd8b9b657-6pv8t -n voting-app -o
jsonpath='{.metadata.annotations}' | tr ',' '\n' | grep com.illumio
"com.illumio.app":"A-VotingApp"
```

```
"com.illumio.env":"E-Production"
"com.illumio.loc":"Azure"

# kubectl get pod azure-vote-front-6fd8b9b657-npppz -n voting-app -o
jsonpath='{.metadata.annotations}' | tr ',' '\n' | grep com.illumio
"com.illumio.app":"A-VotingApp"
"com.illumio.env":"Development"
"com.illumio.loc":"Amazon"
"com.illumio.role":"R-Frontend"}
```

Migration

Workloads reporting supports both: Pod template annotations and workload annotations. However, the priority is put on workload, if it contains at least one annotation with a `com.illumio.` prefix.

In the following example, annotations are specified in `metadata.annotations:` and `spec.template.metadata.annotations:`. Annotations specified in `metadata.annotations:` are prioritized.

The resulting annotations mapped to labels are: `app=A-VotingApp` and `env=E-Test` (no merging between the sets of annotations occurs).

```
apiVersion: apps/v1
kind: Deployment
metadata:
  annotations:
    com.illumio.app: A-VotingApp
    com.illumio.env: E-Test
  name: test-deployment
  labels:
    app: nginx
  spec:
    replicas: 2
    selector:
      matchLabels:
        app: test-pod
      template:
        metadata:
          annotations:
            com.illumio.loc: Amazon
            com.illumio.env: test-env
          labels:
            app: test-pod
          spec:
            containers:
              - name: test-pod
                image: nginx:1.14.2
                imagePullPolicy: IfNotPresent
                ports:
                  - containerPort: 80
```

Deployment with Helm Chart (Core for Kubernetes 3.0.0 and Later)

After you set up your clusters, make sure you do the steps in the order provided in this section.

> **NOTE**
> Illumio Core for Kubernetes 3.0.0 and later is a combined release of C-VEN and Kubelink. Starting with C-VEN 21.5.17 and Kubelink 3.0, C-VEN and Kubelink 3.0 will be only used through the combined release. A Helm Chart (via quay.io) is used to deploy all necessary product components. If you are deploying C-VEN 21.5.15 or earlier, instead follow the deployment instructions in Deployment for C-VEN Versions 21.5.15 or Earlier [294].
>
> The installation process is mostly the same for Kubernetes and OpenShift, except a few steps differ. A dedicated section is created for Kubernetes or OpenShift wherever required.

Helm Chart Deployment Overview

Starting with the Illumio Core for Kubernetes 3.0.0 release and later, the product (including C-VEN and Kubelink) is now deployed by using a Helm Chart. The product components and the Helm Chart are downloaded from a public container repository, https://quay.io/repository/illumio/illumio.

The basic steps to deploy via Helm Chart are:

1. Deploy and configure your PCE. (See the PCE Installation and Upgrade Guide.)
2. Create a container cluster. (See Create a Container Cluster in the PCE [288].)
3. Create a pairing profile. (See Create a Pairing Profile for Your Cluster Nodes [290].)
4. Deploy Helm Chart. (See Deploy with Helm Chart [292].) At this stage you can optionally map existing Kubernetes labels to Illumio labels.

Follow the sections in the order provided in the rest of this chapter, including the requirements and environment preparations described next.

Host and Cluster Requirements

To deploy Illumio containers into your environment, you must meet the following requirements.

Supported Configurations for On-premises and IaaS

For full details on all supported configurations for Illumio Core for Kubernetes version 3.0.0 and later, see the Kubernetes Operator OS Support and Dependencies page on the Illumio Support Portal (under Software > OS Support).

Privileges

The Helm Chart deployment process automatically sets all necessary privileges. The privileges listed below must be provided on host-level and cluster-level for the respective components. They are listed here for reference.

Host-Level

C-VEN

C-VEN requires the following privileges on the host:

- C-VEN is a privileged container and requires access to the following system calls:
  - `NET_ADMIN`
  - `SYS_MODULE`
  - `SYS_ADMIN`
- C-VEN requires persistent storage on the host to write iptables rules and logs.
- C-VEN mounts volumes on the local host to be able to operate (mount points may differ depending on the orchestration platform).

Kubelink

Kubelink does not require specific privileges on the host because Kubelink:

- is not a privileged container
- is a stateless container
- does not require persistent storage

Cluster-Level

Namespace

C-VENs and Kubelink are deployed in the `illumio-system` namespace.

C-VEN

C-VEN requires the following privileges on the cluster:

- C-VEN uses the `illumio-ven` ServiceAccount.

Kubelink

Kubelink requires the following privileges on the cluster:

- Kubelink creates a new Cluster Role to list and watch events occurring on the Kubernetes API server for the following elements:
  - `nodes`
  - `hostsubnets`
  - `replicationcontrollers`
  - `services`
  - `replicasets`
  - `daemonsets`
  - `namespaces`
  - `statefulsets`
- Kubelink uses the `illumio-kubelink` ServiceAccount.

Prepare Your Environment

You need to do these steps before creating clusters or pairing profiles in the PCE, or subsequent deployment.

> **CAUTION**
> If the prerequisite steps are not done before deployment, then containerized environments and Kubelink can get disrupted.

Unique Machine ID

Some of the functionality and services provided by the Illumio C-VEN and Kubelink depend on the Linux machine-id of each Kubernetes cluster node. Each machine-id must be unique in order to take advantage of the functionality. By default, the Linux operating system generates a random machine-id to give each Linux host uniqueness. However, there are cases when machine-id's can be duplicated across machines. This is common across deployments that clone machines from a golden image, for example, spinning up virtual machines from VMware templates, creating compute instances from a reference image, or from a template from a Public Cloud provider.

> **IMPORTANT**
> Illumio Core requires a unique machine-id on all nodes. This issue is more likely to occur with on-premises or IaaS deployments, rather than with Managed Kubernetes Services (from Cloud Service Providers). For more information on how to create a new unique machine-id, see Troubleshooting [356].

Create Labels

For details on creating labels, see "Labels and Label Groups" in Security Policy Guide. The labels shown below are used in examples throughout this document. You are not required to use the same labels

Name	Label Type
Kubernetes Cluster	Application
OpenShift Cluster	Application
Production	Environment
Development	Environment
Data Center	Location
Cloud	Location
Kubelink	Role
Node	Role
Master	Role
Worker	Role

NOTE
Starting in Illumio Core for Kubernetes 4.2.0, you can map Kubernetes labels to Illumio labels by using a Container Resource Definition in your **`illumio-values.yaml`** with the Helm Chart deployment. See Map Kubernetes Labels to Illumio Labels [291] for details.

Create a ConfigMap to Store Your Root CA Certificate

This section describes how to implement Kubelink with a PCE using a certificate signed by a private PKI. It describes how to configure Kubelink and C-VEN to accept the certificate from the PCE signed by a private root or intermediate Certificate Authority (CA) and ensure that Kubelink can communicate in a secure way with the PCE.

Prerequisites

- Access to the root CA to download the root CA certificate.
- Access to your Kubernetes cluster and can run `kubectl` commands.
- Correct privileges in your Kubernetes cluster to create resources like a configmaps, secrets, and Pods.
- Access to the PCE web console as a Global Organization Owner.

Download the Root CA Certificate

Before you begin, ensure that you have access to the root CA certificate. The root CA certificate is a file that can be exported from the root CA without compromising the security of the company. It is usually made available to external entities to ensure a proper SSL handshake between a server and its clients.

You can download the root CA cert in the CRT format on your local machine. Below is an example of a root CA certificate:

```
$ cat root.democa.illumio-demo.com.crt
-----BEGIN CERTIFICATE-----
MIIGSzCCBDOgAwIBAgIUAPw0NfPAivJW4YmKZ499eHZH3S8wDQYJKoZIhvcNAQEL
---output suppressed---
wPG0lug46K1EPQqMA7YshmrwOd6ESy6RGNFFZdhk9Q==
-----END CERTIFICATE-----
```

You can also get the content of your root CA certificate in a readable output format by using the following command:

```
$ openssl x509 -text -noout -in ./root.democa.illumio-demo.com.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            fc:34:35:f3:c0:8a:f2:56:e1:89:8a:67:8f:7d:78:76:47:dd:2f
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=US, ST=California, L=Sunnyvale, O=Illumio,
        OU=Technical Marketing,
        CN=Illumio Demo Root CA 1/emailAddress=tme-team@illumio.com
        Validity
            Not Before: Jan 20 00:05:36 2020 GMT
            Not After : Jan 17 00:05:36 2030 GMT
        Subject: C=US, ST=California, L=Sunnyvale, O=Illumio,
        OU=Technical Marketing,
        CN=Illumio Demo Root CA 1/emailAddress=tme-team@illumio.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (4096 bit)
                Modulus:
                    00:c0:e5:48:7d:97:f8:5b:8c:ef:ac:16:a8:8c:aa:
                    68:b8:48:af:28:cd:17:8f:02:c8:82:e9:69:62:e2:
                    89:2b:be:bd:34:fc:e3:4d:3f:86:5e:d7:e6:89:34:
                    71:60:e6:54:61:ac:0f:26:1c:99:6f:80:89:3f:36:
                    b3:ad:78:d1:6c:3f:d7:23:1e:ea:51:14:48:74:c3:
                    e8:6e:a2:79:b1:60:4c:65:14:2a:f1:a0:97:6c:97:
                    50:43:67:07:b7:51:5d:2c:12:49:81:dc:01:c9:d1:
                    57:48:32:2e:87:a8:d2:c0:b9:f8:43:b2:58:10:af:
                    54:59:09:05:cb:3e:f0:d7:ef:70:cc:fc:53:48:ee:
                    a4:a4:61:f1:d7:5b:7c:a9:a8:92:dc:77:74:f4:4a:
                    c0:4a:90:71:0f:6d:9e:e7:4f:11:ab:a5:3d:cd:4b:
                    8b:79:fe:82:1b:16:27:94:8e:35:37:db:dd:b8:fe:
                    fa:6d:d9:be:57:f3:ca:f3:56:aa:be:c8:57:a1:a8:
                    c9:83:dd:5a:96:5a:6b:32:2d:5e:ae:da:fc:85:76:
                    bb:77:d5:c2:53:f3:5b:61:74:e7:f3:3e:4e:ad:10:
                    7d:4f:ff:90:69:7c:1c:41:2f:67:e4:13:5b:e6:3a:
                    a3:2f:93:61:3b:07:56:59:5a:d9:bc:34:4d:b3:54:
                    b5:c6:e5:0a:88:e9:62:7b:4b:85:d2:9e:4c:ee:0b:
                    0d:f4:72:b1:1b:44:04:93:cf:cc:bb:18:31:3a:d4:
                    83:4a:ff:15:42:2d:91:ca:d0:cb:36:d9:8d:62:c0:
                    41:59:1a:93:c7:27:79:08:94:b2:a2:50:3c:57:27:
                    33:af:f0:b6:92:44:49:c5:09:15:a7:43:2a:0f:a9:
                    02:61:b3:66:4f:c3:de:d3:63:1e:08:b1:23:ea:69:
                    90:db:e8:e9:1e:21:84:e0:56:e1:8e:a1:fa:3f:7a:
                    08:0f:54:0a:82:41:08:6b:6e:bb:cf:d6:5b:80:c6:
```

```
                ea:0c:80:92:96:ab:95:5d:38:6d:4d:da:38:6b:42:
                ef:7c:88:58:83:88:6d:da:28:62:62:1f:e5:a7:0d:
                04:9f:0d:d9:52:39:46:ba:56:7c:1d:77:38:26:7c:
                86:69:58:4d:b0:47:3a:e2:be:ee:1a:fc:4c:de:67:
                f3:d5:fe:e6:27:a2:ef:26:86:19:5b:05:85:9c:4c:
                02:24:76:58:42:1a:f8:e0:e0:ed:78:f2:8f:c8:5a:
                20:a9:2d:0b:d4:01:fa:57:d4:6f:1c:0a:31:30:8c:
                32:7f:b0:01:1e:fe:94:96:03:ee:01:d7:f4:4a:83:
                f5:06:fa:60:43:15:05:9a:ca:88:59:5c:f5:13:09:
                82:69:7f
            Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                3D:3D:3D:61:E6:88:09:FE:34:0F:1D:5E:5E:52:72:71:C7:DE:15:92
            X509v3 Authority Key Identifier:
                keyid:3D:3D:3D:61:E6:88:09:FE:34:0F:1D:5E:5E:52:72:71:C7:
                DE:15:92

            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Key Usage: critical
                Digital Signature, Certificate Sign, CRL Sign
    Signature Algorithm: sha256WithRSAEncryption
         28:24:86:91:a6:4a:88:e4:8d:6b:fc:67:2a:68:08:67:35:e5:
         a6:77:ff:07:4b:89:53:99:2e:6d:95:df:12:81:28:6a:8e:6f:
         5a:98:95:5b:4a:21:ae:f0:20:a4:4e:06:b2:4e:5a:67:c1:6a:
         06:f1:0f:c1:f7:7e:f2:e0:b3:9d:d8:54:26:6a:b2:1c:19:b8:
         b5:5c:c7:03:6b:f7:70:9e:72:85:c9:29:55:f9:f4:a4:f2:b4:
         3b:3d:ce:25:96:67:32:1e:8d:e2:00:22:55:4b:05:4f:ee:0e:
         67:ac:db:1b:61:da:5f:9c:10:1c:0c:05:66:c0:5b:5f:b9:95:
         59:a9:58:5b:e7:69:ac:b0:bd:b3:c2:a3:35:58:01:a4:ff:c0:
         8d:ac:1c:19:21:41:50:fb:8e:e0:f5:a9:ad:ec:de:cb:53:04:
         a9:d8:ac:76:8a:09:0d:7c:c6:1a:bc:06:74:bb:10:1c:aa:07:
         f6:cb:b2:1b:0c:0c:65:03:45:2b:51:d5:6e:a0:4d:91:ce:c5:
         ed:8d:a9:e7:f6:37:7d:ab:1b:a4:a2:a3:3b:76:17:5b:d9:3a:
         9c:c1:df:cc:cd:a0:b0:a9:5c:74:61:d7:a0:1d:04:67:68:ee:
         a6:7b:1e:41:a4:02:fc:65:9e:e3:c1:c2:57:b2:2e:b0:ff:a9:
         86:82:35:4d:29:b2:fe:74:2e:b8:37:5d:2b:e8:69:f2:80:29:
         19:f1:1e:7a:5d:e3:d2:51:50:46:30:54:7e:b8:ad:59:61:24:
         45:a8:5a:fe:19:ff:09:31:d0:50:8b:e2:15:c0:a2:f1:20:95:
         63:55:18:a7:a2:ad:16:25:c7:a3:d1:f2:e5:be:6d:c0:50:4b:
         15:ac:e0:10:5e:f3:7b:90:9c:75:1a:6b:e3:fb:39:88:e4:e6:
         9f:4c:85:60:67:e8:7d:2e:85:3d:87:ed:06:1d:13:0b:76:d7:
         97:a5:b8:05:76:67:d6:41:06:c5:c0:7a:bd:f4:c6:5b:b2:fd:
         23:6f:1f:57:2e:df:95:3f:26:a5:13:4d:6d:96:12:56:98:db:
         2e:7d:fd:56:f5:71:b7:19:2b:c9:de:2d:b9:c8:17:cc:20:de:
         7c:19:7a:aa:12:97:1c:80:b7:d3:67:d3:b7:a7:96:f0:c9:4d:
         f5:8b:0e:10:3b:b9:4e:09:90:5a:3b:51:c9:48:a2:ca:9f:db:
         72:44:87:59:db:49:fa:75:44:b5:f6:7f:c5:26:e1:01:ae:7b:
         6f:4a:75:d1:b5:b3:68:c0:31:48:f8:5c:06:c0:f1:b4:96:e8:
         38:e8:ad:44:3d:0a:8c:03:b6:2c:86:6a:f0:39:de:84:4b:2e:
         91:18:d1:45:65:d8:64:f5
```

Create a configmap in Kubernetes Cluster

After downloading the certificate locally on your machine, create a configmap in the Kubernetes cluster that will copy the root CA certificate on your local machine into the Kubernetes cluster.

To create configmap, use the following command:

```
$ kubectl -n illumio-system create configmap root-ca-config \
    --from-file=./certs/root.democa.illumio-demo.com.crt
```

The `--from-file` option points to the path where the root CA certificate is stored on your local machine.

To verify that configmap was created correctly, use the following command:

```
$ kubectl -n illumio-system create configmap root-ca-config \
> --from-file=./certs/root.democa.illumio-demo.com.crt
configmap/root-ca-config created
$
$ kubectl -n illumio-system get configmap
NAME                                 DATA   AGE
root-ca-config                       1      12s
$
$ kubectl -n illumio-system describe configmap root-ca-config
Name:         root-ca-config
Namespace:    illumio-system
Labels:       <none>
Annotations:  <none>

Data
====
root.democa.illumio-demo.com.crt:
----
-----BEGIN CERTIFICATE-----
MIIGSzCCBDOgAwIBAgIUAPw0NfPAivJW4YmKZ499eHZH3S8wDQYJKoZIhvcNAQEL
---output suppressed---
wPG0lug46K1EPQqMA7YshmrwOd6ESy6RGNFFZdhk9Q==
-----END CERTIFICATE-----

Events:  <none>
$
```

`root-ca-config` is the name used to designate configmap. You can modify it according to your naming convention.

Configure Calico in Append Mode

In case your cluster is configured with Calico as the network plugin (usually for Kubernetes and not for OpenShift), both Calico and Illumio Core will write iptables rules on the cluster nodes.

- Calico - Needs to write iptables rules to instruct the host how to forward packets (overlay, IPIP, NAT, and so on).

- Illumio Core - Needs to write iptables rules to secure communications between nodes and/or Pods.

You should establish a hierarchy to make the firewall coexistence work smoothly because Illumio Core and Calico will write rules at the same time. By default, both solutions are configured to insert rules first in the iptables chains/tables and Illumio Core will remove other rules added by a third-party software (in the Exclusive mode).

To allow Calico to write rules along with Illumio without flushing rules from one another, you should:

- Configure Illumio to work in Firewall Coexistence mode (default for workloads that are part of a container cluster).
- Configure Calico to work in Append mode (default is Insert mode).

To configure Calico to work in Append mode with iptables:

1. Edit the Calico DaemonSet:
   ```
   kubectl -n kube-system edit ds calico-node
   ```
2. Locate the `spec: > template: > spec: > containers:` section inside the `YAML` file and change ChainInsertMode by adding the following code block:
   ```
   - name: FELIX_CHAININSERTMODE
     value: Append
   ```
3. Save your changes and exit.
4. Kubernetes will restart all Calico Pods in a rolling update.

For more information on changing Calico ChainInsertMode, see Calico documentation.

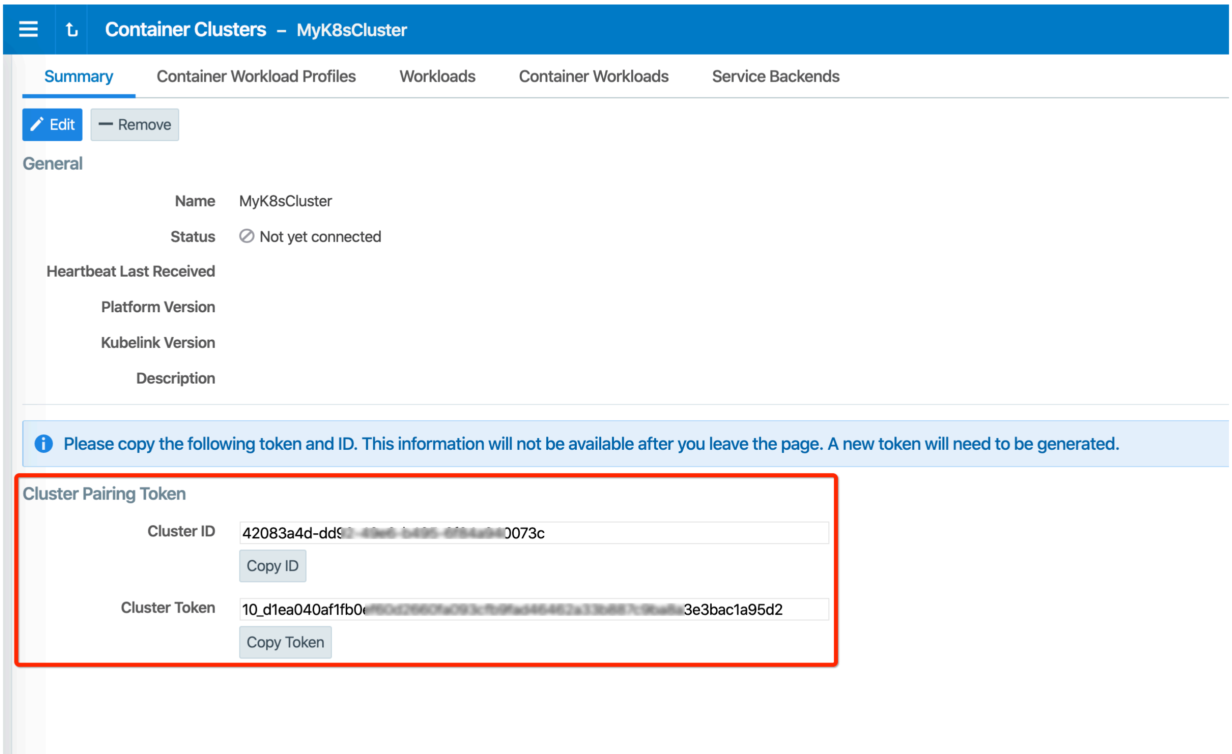
Create a Container Cluster in the PCE

To provide visibility and enforcement to your containerized environment, you first need to create a container cluster in the PCE. Each container cluster maps to an existing Kubernetes or OpenShift cluster.

Create a Container Cluster

To create a new container cluster:

1. Log into the PCE web console as a user with Global Organization Owner privileges.
2. From the PCE web console menu, navigate to **Infrastructure** > **Container Clusters**.
3. Click **Add**.
   a. Add a Name.
   b. **Save** the Container Cluster.
4. You will see a summary page of the new Container Cluster. From the Cluster Pairing Token section, copy the values of the Cluster ID and Cluster Token.
5. After copying and saving the values (in a text editor or similar tool), open the Container Workload Profiles page.



Configure a Container Workload Profile Template

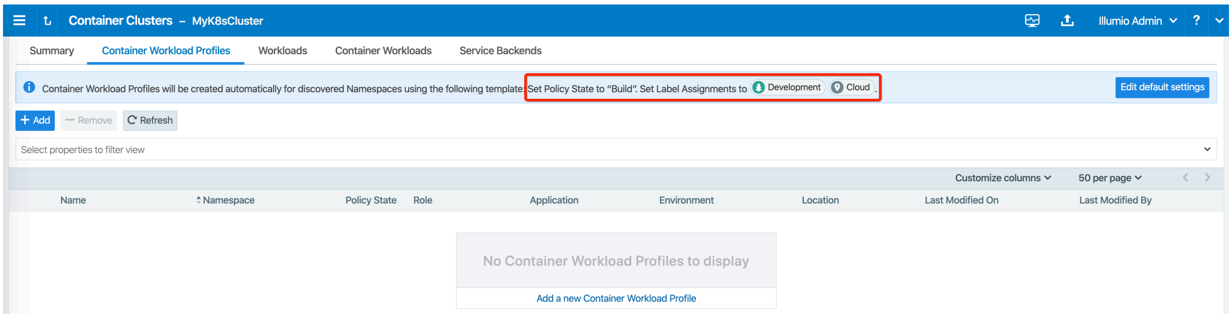
When configuring a new Container Cluster, it is recommended to set the default settings shared by all the Container Workload Profiles. Illumio provides a Container Workload Profile template that can be used for that purpose. By defining the default Policy State and minimum set of labels common to all namespaces in the cluster, you will save time later on when new namespaces are discovered by Kubelink. Each new profile created will inherit what was defined in the template.

> **IMPORTANT**
>
> Illumio does not provide a method to redefine at once all the labels associated with each profile. Hence, it is **strongly recommended** to use the provided template to define the default values for all profiles that are part of the same cluster.

To define the default parameters for all profiles using a template, under *Container Workload Profiles*, click **Edit default settings** and select values for all the fields.

For information about assigning default labels in the template, see the "Labels Restrictions for Kubernetes Namespaces" topic.

After you click OK, the following information is displayed:



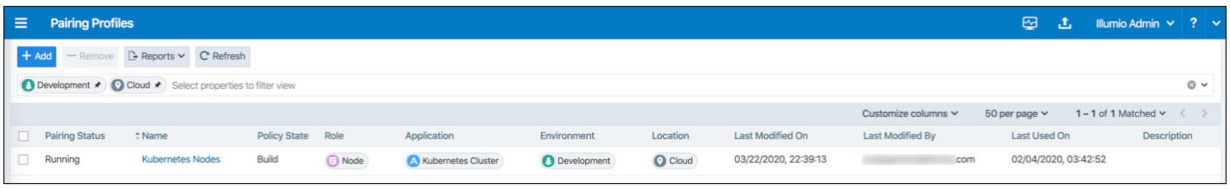
Create a Pairing Profile for Your Cluster Nodes

> **IMPORTANT**
>
> Before deploying the C-VEN, ensure that either of the following two requirements has been met:
>
> - Kubelink is deployed on the Kubernetes cluster and is in sync with the PCE, or
> - Firewall coexistence is enabled.

Before deploying, you should create a pairing profile to pair the cluster nodes with the PCE. You only need to create one pairing profile for all your nodes.

> **NOTE**
>
> You only need to create pairing profiles for Kubernetes or OpenShift nodes and not for container workloads.

For ease of configuration and management, consider applying the same Application, Environment, and Location labels across all nodes of the same Kubernetes or OpenShift cluster. The screenshot below shows an example of a pairing profile for a Kubernetes cluster.



> **TIP**
>
> Illumio recommends all pairing profiles for Kubernetes nodes *not* to use Full Enforcement policy state.
>
> You should only move into Full Enforcement state after you have completed all other configuration steps in this guide.

Map Kubernetes Node Labels to Illumio Labels

Label mapping is a method of mapping some or all existing Kubernetes node labels to Illumio labels. Label maps are a new way to assign Illumio labels to container host workloads in addition to existing methods (such as with container workload profiles and pairing profiles). Labels assigned through label maps take precedence over these other methods -- that is, they overwrite any labels assigned with these other methods.

A label map is defined by a Kubernetes *Custom Resource Definition* (CRD) within a yaml file that is typically installed via a Helm Chart. Installing the Helm Chart then applies the defined labels.

Label Mapping CRD

The CRD is defined in the yaml file with a `kind: LabelMap` declaration, which in turn contains a `nodeLabelMap` section that applies to nodes (host workloads).

Within the `nodeLabelMap` section, Illumio label types are mapped with `fromKey` and `toKey` key-value pairs, where the `fromKey` value specifies a source Kubernetes label, and the `toKey` value paired with it defines the destination Illumio label type.

If an optional `allowCreate: true` is within a `fromKey` and `toKey` pair, the Illumio label type defined in that mapping is created if it does not already exist on the PCE.

An optional `valuesMap:` within a `fromKey` and `toKey` pair specifies one or more label value mappings for that label type, with `from:` value identifying the source Kubernetes label and the `to:` value following it specifying the destination Illumio label value. If no `valuesMap:` is specified, then label values for the mapped label type are not changed. Only the label type is changed in the PCE.

Example Label Map

Note these points about the following example label map:

- The first `nodeLabelMap` item creates a new Illumio `location` label of `Amazon` (if it does not exist, per the `allowCreate: true` declaration) and maps this label to all nodes with the Kubernetes label `topology.kubernetes.io/region` with either value of `eu-west-1` or `eu-west-2`.
- With the second item under `nodeLabelMap`, for every `node-type` Kubernetes label, the map creates Illumio `k8s-node` labels with values based on the existing Kubernetes label values (because there is no associated `valuesMap` mapping definition).

```
kind: LabelMap
apiVersion: ic4k.illumio.com/v1alpha1
metadata:
  name: default
nodeLabelMap:
  - allowCreate: true
    fromKey: topology.kubernetes.io/region
    toKey: loc
    valuesMap:
      - from: eu-west-1
        to: Amazon
      - from: eu-west-2
        to: Amazon
- allowCreate: true
  fromKey: node-type
  toKey: k8s-node
```

The label type has to be created and exist in PCE first before new labels can be created through label mapping.

Deploy with Helm Chart

To deploy via Helm Chart:

1. Install Helm. Refer to https://helm.sh/docs/ for a quick start guide and other relevant information.

   According to official Helm documentation, if your version of Helm is lower than 3.8.0, the following command must be executed in the installation environment:

   ```
   $ export HELM_EXPERIMENTAL_OCI=1
   ```
2. Prepare an **illumio-values.yaml** file with the following mandatory parameters set with values that describe this deployment:

   ```
   pce_url: URL_PORT # PCE URL with port, e.g. mypce.example.com:8443
   cluster_id: ILO_CLUSTER_UUID # Cluster ID from PCE, e.g. cc4997c1-40...
   cluster_token: ILO_CLUSTER_TOKEN # Cluster Token from PCE, e.g. 1_170b...
   cluster_code: ILO_CODE # Pairing Profile key from PCE, e.g. 1391c...
   containerRuntime: containerd # Container runtime engine used in cluster,
   allowed values are [containerd, docker, crio, k3s_containerd]
   containerManager: kubernetes # Container manager used in cluster,
   allowed values are[kubernetes, openshift]
   networkType: flat # CNI type, allowed values are [overlay, flat]
   ```

   where **URL_PORT**, **ILO_CLUSTER_UUID**, **ILO_CLUSTER_TOKEN**, and **ILO_CODE** are placeholders for customer provided variables.

   NOTE

   To add support for flat network CNIs in addition to the default (where pods run on an overlay network), an optional **networkType** parameter is now available in the Helm Chart where you can specify **flat** or **overlay** type. The default value is **overlay**.

3. Optionally map existing Kubernetes labels to desired Illumio labels by adding a Kubernetes *Custom Resource Definition* (CRD) Label Map to your **`illumio-values.yaml`** file. For details on using a label map, see the "Map Kubernetes Labels to Illumio Labels [291]" topic.
4. Install the Helm Chart:

```
helm install illumio -f illumio-values.yaml oci://quay.io/illumio/
illumio --version <ver#> --namespace illumio-system --create-namespace
```

> **IMPORTANT**
> Be sure to explicitly specify the version to install with the `--version <ver#>` option (for example, `--version 4.3.0`), after confirming that the product version you want to install is supported with your PCE version. Verify which PCE versions support the Illumio Core for Kubernetes version you want to deploy at the Kubernetes Operator OS Support and Dependencies page on the Illumio Support Portal.

If the illumio-system namespace already exists, omit the **`--create-namespace`** flag.

Refer to the **README** file included with the Helm Chart for important information, such as additional deployment parameters you can specify in the Helm Chart before installing it.

> **NOTE**
> Kubelink version labeling has changed. Prior to version 3.3.0, Kubelink used a 6-hexit suffix for its release version, like 3.2.1.445a83. In Kubelink 3.3.0 and later, the version suffix is now changed to a numeric build number, like 3.3.0-56.

Re-Label Your Cluster Nodes

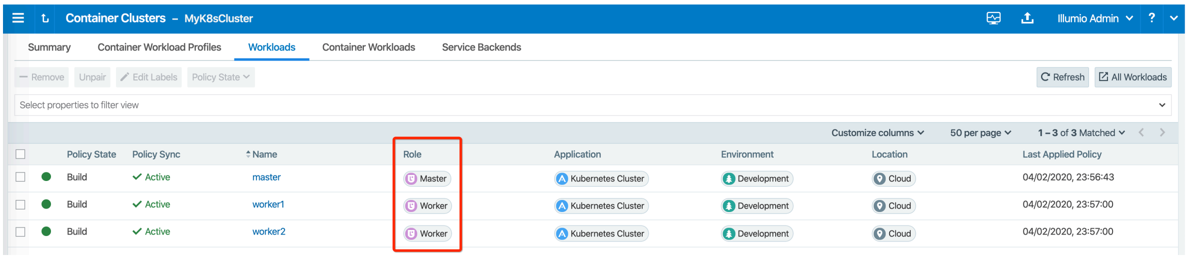
> **NOTE**
> Re-labeling the cluster nodes is optional.

In the case of self-managed deployments in which both Master and Worker nodes are managed, you may want to re-label your nodes to differentiate Master nodes from Worker nodes. Doing this helps when you are writing different policies for the Worker and Master nodes or if you want to segment these nodes differently.

To re-label your cluster nodes:

1. In the PCE UI, go to **Infrastructure > Container Clusters > YourClusterName > Workloads**.
2. Select the workloads you want to re-label.

3. Click **Edit Labels** to assign the new labels (for example, Master and Worker).



4. After re-labeling your cluster nodes, the nodes part of the cluster reflect the updated label(s).

Deployment for C-VEN Versions 21.5.15 or Earlier

After you set up your clusters, make sure you perform the steps in the order provided in this section.

> **NOTE**
>
> Follow these instructions if you are deploying Illumio Core for Kubernetes (C-VEN) versions 21.5.15 or earlier.
>
> If you are deploying the Illumio Core for Kubernetes 3.0.0 release (or later), do not follow these instructions, but instead see Deployment with Helm Chart [281] (Core for Kubernetes 3.0.0 and Higher), which describes how to use a Helm Chart to deploy all necessary product components.
>
> The installation process is mostly the same for Kubernetes and OpenShift, except a few steps differ. A dedicated section is created for Kubernetes or OpenShift wherever required.

Host and Cluster Requirements

To deploy Illumio containers into your environment, you must meet the following requirements.

Supported Configurations for On-premises and IaaS

For full details on all supported configurations for Containerized VEN release 21.5.15 and earlier, see the C-VEN/Kubelink OS Support and Dependencies page on the Illumio Support Portal (under Software > OS Support).

Privileges

The privileges listed below should be provided on host-level and cluster-level for the respective components.

Host-Level

C-VEN

C-VEN requires the following privileges on the host:

- C-VEN is a privileged container and requires access to the following system calls:
  - `NET_ADMIN`
  - `SYS_MODULE`
  - `SYS_ADMIN`
- C-VEN requires persistent storage on the host to write iptables rules and logs.
- C-VEN mounts volumes on the local host to be able to operate (mount points may differ depending on the orchestration platform).

Optionally, you can set the Priority Class to `system-node-critical`. This option is only supported in Kubernetes 1.17 and later, in a namespace other than `kube-system`. For more details, see the v1.17 documentation on Kubernetes.

Kubelink

Kubelink does not require specific privileges on the host because Kubelink:

- Is not a privileged container.
- Is a stateless container.
- Does not require persistent storage.

Cluster-Level

Namespace

C-VENs and Kubelink are deployed in the `illumio-system` namespace. You can modify this namespace name according to your deployment (manifest file modification).

C-VEN

C-VEN requires the following privileges on the cluster:

- C-VEN uses the `illumio-ven` ServiceAccount.

Kubelink

Kubelink requires the following privileges on the cluster:

- Kubelink creates a new Cluster Role to list and watch events occurring on the Kubernetes API server for the following elements:
  - `nodes`
  - `hostsubnets`
  - `replicationcontrollers`
  - `services`
  - `replicasets`
  - `daemonsets`
  - `namespaces`
  - `statefulsets`
- Kubelink uses the `illumio-kubelink` ServiceAccount.

Optionally, you can set the Priority Class to `system-cluster-critical`. This option is only supported in Kubernetes 1.17 and later, in a namespace other than `kube-system`. For more details, see Kubernetes.

Rancher Kubernetes Engine (RKE)

In order for the Illumio solution to work with RKE, the following needs to be set in the `cluster.yml`:

```
kubelet:
    image: ""
    extra_args: {}
    extra_binds:
        - "/etc/machine-id:/etc/machine-id"
        - "/lib/modules:/lib/modules"
    extra_env: []
```

Prepare Your Environment

> **IMPORTANT**
> The following steps for preparing your environment are no longer needed when deploying Illumio Core for Kubernetes version 3.0.0 and beyond, which now uses Helm Chart for deploying C-VEN and Kubelink. This section is included here for backwards compatibility and historical purposes. If you are deploying using Helm Chart, skip this section and follow the instructions in the "Create a Container Cluster in the PCE" topic.

You need to do these steps before C-VEN installation and pairing.

> **CAUTION**
> If the prerequisite steps are not done before C-VEN and Kubelink installation, then containerized environments and Kubelink can get disrupted.

Unique Machine ID

Some of the functionality and services provided by the Illumio C-VEN and Kubelink depend on the Linux machine-id of each Kubernetes cluster node. Each machine-id must be unique in order to take advantage of the functionality. By default, the Linux operating system generates a random machine-id to give each Linux host uniqueness. However, there are cases when machine-id's can be duplicated across machines. This is common across deployments that clone machines from a golden image, for example, spinning up virtual machines from VMware templates, creating compute instances from a reference image, or from a template from a Public Cloud provider.

IMPORTANT

Illumio Core requires a unique machine-id on all nodes. This issue is more likely to occur with on-premises or IaaS deployments, rather than with Managed Kubernetes Services (from Cloud Service Providers). For more information about how to create a new unique machine-id, see Troubleshooting [84].

Create Labels

For details on creating labels, see "Labels and Label Groups" in Security Policy Guide. The labels shown below are used in examples throughout this document. You are not required to use the same labels

Name	Label Type
Kubernetes Cluster	Application
OpenShift Cluster	Application
Production	Environment
Development	Environment
Data Center	Location
Cloud	Location
Kubelink	Role
Node	Role
Master	Role
Worker	Role

Push Kubelink and C-VEN Images to Your Container Registry

In order to install Illumio Core for containers, you first need to upload (or push) Kubelink and C-VEN container images to your container registry. The files in the C-VEN and Kubelink packages you've downloaded are as follows:

C-VEN `illumio-ven-21.5.x-xxxx.k8s.x86_64.tgz` package includes:

- A Docker image
  - `illumio-ven-21.5.x-xxxx.tgz`
- Configuration files:
  - `illumio-ven-secret.yml`
  - `illumio-ven-kubernetes.yml`
  - `illumio-ven-openshift.yml`

Kubelink `illumio-kubelink-2.1.x.tar.gz` package includes:

- A docker image
  - `kubelink-image.tar.gz`
- Configuration files in kube-yaml
  - `illumio-kubelink-secret.yml`
  - `illumio-kubelink-kubernetes.yml`
  - `illumio-kubelink-openshift.yml`
  - `illumio-kubelink-namespace.yml`

> **CAUTION**
> These images are not publicly available and should **not** be posted on a publicly open container registry without Illumio's consent.

In a self-managed deployment, Kubelink and C-VEN images can be pushed to a private container registry. In OpenShift, a container registry is provided as part of the platform, and images can be pushed to this registry for simplicity and better authentication. In the case of Kubernetes, there is no container registry provided by default and must be provided as an external component.

In a cloud-managed deployment, Cloud Service Providers (CSPs) provide integration of private container registries such as, Amazon ECR, Microsoft ACR, and so on. These registries can securely be used to host Illumio's container images for Kubelink and C-VEN. Refer to the documentation provided by the respective CSPs to learn how to push images to those registries.

To push Kubelink and C-VEN container images to your private container registry, use the following commands (based on docker):

1. Log in to your private container registry.

   ```
   docker login <docker-registry>
   ```
2. Load Kubelink and C-VEN container images on your local computer.

   ```
   docker load -i kubelink-image.tar.gz
   docker load -i illumio-ven-yy.y.x-xxxx.tgz
   ```

   Verify that docker images are loaded on your computer.

   ```
   docker image ls
   ```
3. Tag the Kubelink and C-VEN container image IDs with the name of your container registry.

   ```
   docker tag <illumio-kubelink-image-id> <docker-registry>/illumio-
   kubelink:y.y.x.xxxxxx
   docker tag <illumio-ven-image-id> <docker-registry>/illumio-ven:yy.y.x-
   xxxx
   ```

   Verify that images are tagged on your computer and ready to be pushed to your private container registry.

   ```
   docker image ls
   ```

4. Push Kubelink and C-VEN container images on your private container registry.

```
docker push <docker-registry>/illumio-kubelink:y.y.x.xxxxxx
docker push <docker-registry>/illumio-ven:yy.y.x-xxxx
```

After pushing images to your private container registry, proceed to the next section.

Create Illumio Namespace

Illumio Core for containers is deployed in a dedicated namespace illumio-system, by default. This namespace has the minimum privileges in the cluster required to run Illumio Core and can tie into the Kubernetes and OpenShift RBAC models.

To create the `illumio-system` namespace for Kubernetes, use the following command:

```
kubectl create namespace illumio-system
```

NOTE

Illumio provides a yaml manifest file to create the namespace in the Kubelink tarball `illumio-kubelink-namespace.yml`. You can create this namespace by applying this manifest file to your Kubernetes cluster, using the following command:

```
kubectl apply -f illumio-kubelink-namespace.yml
```

To create the `illumio-system` project for OpenShift, use the following command:

```
oc new-project illumio-system
```

Authenticate Kubernetes Cluster with Container Registry

NOTE

Depending on your deployment, the steps in the Authenticate Kubernetes Cluster with Container Registry [299], Create a ConfigMap to Store Your Root CA Certificate [300], and Configure Calico in Append Mode [303] topics are optional.

When storing container images in a private container registry, it is often required and strongly recommended to authenticate against the registry to be able to pull an image from it. In order to do this, the Kubernetes or OpenShift cluster must have the credentials configured and stored in a secret file to be able to pull container images.

To configure a secret to store your container registry credentials, use the following command:

```
kubectl create secret docker-registry <container-registry-secret-name>
-n illumio-system --docker-server=<container-registry>
--docker-username=<username> --docker-password=<password>
```

To verify that the secret has been created, use the following command:

```
kubectl get secret -n illumio-system |
grep <container-registry-secret-name>
```

IMPORTANT

The above commands are valid for deployments with your own private container registry, but may not be valid for a cloud-managed private container registry. For more information, refer to your Cloud Service Provider documentation.

Create a ConfigMap to Store Your Root CA Certificate

This section describes how to implement Kubelink with a PCE using a certificate signed by a private PKI. It describes how to configure Kubelink and C-VEN to accept the certificate from the PCE signed by a private root or intermediate Certificate Authority (CA) and ensure that Kubelink can communicate in a secure way with the PCE.

Prerequisites

- Access to the root CA to download the root CA certificate.
- Access to your Kubernetes cluster and can run `kubectl` commands.
- Correct privileges in your Kubernetes cluster to create resources like a configmaps, secrets, and Pods.
- Access to the PCE web console as a Global Organization Owner.

Download the Root CA Certificate

Before you begin, ensure that you have access to the root CA certificate. The root CA certificate is a file that can be exported from the root CA without compromising the security of the company. It is usually made available to external entities to ensure a proper SSL handshake between a server and its clients.

You can download the root CA cert in the CRT format on your local machine. Below is an example of a root CA certificate:

```
$ cat root.democa.illumio-demo.com.crt
-----BEGIN CERTIFICATE-----
MIIGSzCCBDOgAwIBAgIUAPw0NfPAivJW4YmKZ499eHZH3S8wDQYJKoZIhvcNAQEL
---output suppressed---
wPG0lug46K1EPQqMA7YshmrwOd6ESy6RGNFFZdhk9Q==
-----END CERTIFICATE-----
```

You can also get the content of your root CA certificate in a readable output format by using the following command:

```
$ openssl x509 -text -noout -in ./root.democa.illumio-demo.com.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            fc:34:35:f3:c0:8a:f2:56:e1:89:8a:67:8f:7d:78:76:47:dd:2f
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=US, ST=California, L=Sunnyvale, O=Illumio,
        OU=Technical Marketing, CN=Illumio Demo Root
        CA 1/emailAddress=tme-team@illumio.com
        Validity
            Not Before: Jan 20 00:05:36 2020 GMT
            Not After : Jan 17 00:05:36 2030 GMT
        Subject: C=US, ST=California, L=Sunnyvale, O=Illumio,
        OU=Technical Marketing, CN=Illumio Demo Root
        CA 1/emailAddress=tme-team@illumio.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (4096 bit)
                Modulus:
                    00:c0:e5:48:7d:97:f8:5b:8c:ef:ac:16:a8:8c:aa:
                    68:b8:48:af:28:cd:17:8f:02:c8:82:e9:69:62:e2:
                    89:2b:be:bd:34:fc:e3:4d:3f:86:5e:d7:e6:89:34:
                    71:60:e6:54:61:ac:0f:26:1c:99:6f:80:89:3f:36:
                    b3:ad:78:d1:6c:3f:d7:23:1e:ea:51:14:48:74:c3:
                    e8:6e:a2:79:b1:60:4c:65:14:2a:f1:a0:97:6c:97:
                    50:43:67:07:b7:51:5d:2c:12:49:81:dc:01:c9:d1:
                    57:48:32:2e:87:a8:d2:c0:b9:f8:43:b2:58:10:af:
                    54:59:09:05:cb:3e:f0:d7:ef:70:cc:fc:53:48:ee:
                    a4:a4:61:f1:d7:5b:7c:a9:a8:92:dc:77:74:f4:4a:
                    c0:4a:90:71:0f:6d:9e:e7:4f:11:ab:a5:3d:cd:4b:
                    8b:79:fe:82:1b:16:27:94:8e:35:37:db:dd:b8:fe:
                    fa:6d:d9:be:57:f3:ca:f3:56:aa:be:c8:57:a1:a8:
                    c9:83:dd:5a:96:5a:6b:32:2d:5e:ae:da:fc:85:76:
                    bb:77:d5:c2:53:f3:5b:61:74:e7:f3:3e:4e:ad:10:
                    7d:4f:ff:90:69:7c:1c:41:2f:67:e4:13:5b:e6:3a:
                    a3:2f:93:61:3b:07:56:59:5a:d9:bc:34:4d:b3:54:
                    b5:c6:e5:0a:88:e9:62:7b:4b:85:d2:9e:4c:ee:0b:
                    0d:f4:72:b1:1b:44:04:93:cf:cc:bb:18:31:3a:d4:
                    83:4a:ff:15:42:2d:91:ca:d0:cb:36:d9:8d:62:c0:
                    41:59:1a:93:c7:27:79:08:94:b2:a2:50:3c:57:27:
                    33:af:f0:b6:92:44:49:c5:09:15:a7:43:2a:0f:a9:
                    02:61:b3:66:4f:c3:de:d3:63:1e:08:b1:23:ea:69:
                    90:db:e8:e9:1e:21:84:e0:56:e1:8e:a1:fa:3f:7a:
                    08:0f:54:0a:82:41:08:6b:6e:bb:cf:d6:5b:80:c6:
                    ea:0c:80:92:96:ab:95:5d:38:6d:4d:da:38:6b:42:
                    ef:7c:88:58:83:88:6d:da:28:62:62:1f:e5:a7:0d:
                    04:9f:0d:d9:52:39:46:ba:56:7c:1d:77:38:26:7c:
                    86:69:58:4d:b0:47:3a:e2:be:ee:1a:fc:4c:de:67:
                    f3:d5:fe:e6:27:a2:ef:26:86:19:5b:05:85:9c:4c:
                    02:24:76:58:42:1a:f8:e0:e0:ed:78:f2:8f:c8:5a:
                    20:a9:2d:0b:d4:01:fa:57:d4:6f:1c:0a:31:30:8c:
                    32:7f:b0:01:1e:fe:94:96:03:ee:01:d7:f4:4a:83:
                    f5:06:fa:60:43:15:05:9a:ca:88:59:5c:f5:13:09:
                    82:69:7f
```

```
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                3D:3D:3D:61:E6:88:09:FE:34:0F:1D:5E:5E:52:72:71:C7:
                DE:15:92
            X509v3 Authority Key Identifier:
                keyid:3D:3D:3D:61:E6:88:09:FE:34:0F:1D:5E:5E:52:72:71:
                C7:DE:15:92

            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Key Usage: critical
                Digital Signature, Certificate Sign, CRL Sign
    Signature Algorithm: sha256WithRSAEncryption
         28:24:86:91:a6:4a:88:e4:8d:6b:fc:67:2a:68:08:67:35:e5:
         a6:77:ff:07:4b:89:53:99:2e:6d:95:df:12:81:28:6a:8e:6f:
         5a:98:95:5b:4a:21:ae:f0:20:a4:4e:06:b2:4e:5a:67:c1:6a:
         06:f1:0f:c1:f7:7e:f2:e0:b3:9d:d8:54:26:6a:b2:1c:19:b8:
         b5:5c:c7:03:6b:f7:70:9e:72:85:c9:29:55:f9:f4:a4:f2:b4:
         3b:3d:ce:25:96:67:32:1e:8d:e2:00:22:55:4b:05:4f:ee:0e:
         67:ac:db:1b:61:da:5f:9c:10:1c:0c:05:66:c0:5b:5f:b9:95:
         59:a9:58:5b:e7:69:ac:b0:bd:b3:c2:a3:35:58:01:a4:ff:c0:
         8d:ac:1c:19:21:41:50:fb:8e:e0:f5:a9:ad:ec:de:cb:53:04:
         a9:d8:ac:76:8a:09:0d:7c:c6:1a:bc:06:74:bb:10:1c:aa:07:
         f6:cb:b2:1b:0c:0c:65:03:45:2b:51:d5:6e:a0:4d:91:ce:c5:
         ed:8d:a9:e7:f6:37:7d:ab:1b:a4:a2:a3:3b:76:17:5b:d9:3a:
         9c:c1:df:cc:cd:a0:b0:a9:5c:74:61:d7:a0:1d:04:67:68:ee:
         a6:7b:1e:41:a4:02:fc:65:9e:e3:c1:c2:57:b2:2e:b0:ff:a9:
         86:82:35:4d:29:b2:fe:74:2e:b8:37:5d:2b:e8:69:f2:80:29:
         19:f1:1e:7a:5d:e3:d2:51:50:46:30:54:7e:b8:ad:59:61:24:
         45:a8:5a:fe:19:ff:09:31:d0:50:8b:e2:15:c0:a2:f1:20:95:
         63:55:18:a7:a2:ad:16:25:c7:a3:d1:f2:e5:be:6d:c0:50:4b:
         15:ac:e0:10:5e:f3:7b:90:9c:75:1a:6b:e3:fb:39:88:e4:e6:
         9f:4c:85:60:67:e8:7d:2e:85:3d:87:ed:06:1d:13:0b:76:d7:
         97:a5:b8:05:76:67:d6:41:06:c5:c0:7a:bd:f4:c6:5b:b2:fd:
         23:6f:1f:57:2e:df:95:3f:26:a5:13:4d:6d:96:12:56:98:db:
         2e:7d:fd:56:f5:71:b7:19:2b:c9:de:2d:b9:c8:17:cc:20:de:
         7c:19:7a:aa:12:97:1c:80:b7:d3:67:d3:b7:a7:96:f0:c9:4d:
         f5:8b:0e:10:3b:b9:4e:09:90:5a:3b:51:c9:48:a2:ca:9f:db:
         72:44:87:59:db:49:fa:75:44:b5:f6:7f:c5:26:e1:01:ae:7b:
         6f:4a:75:d1:b5:b3:68:c0:31:48:f8:5c:06:c0:f1:b4:96:e8:
         38:e8:ad:44:3d:0a:8c:03:b6:2c:86:6a:f0:39:de:84:4b:2e:
         91:18:d1:45:65:d8:64:f5
```

Create a configmap in Kubernetes Cluster

After downloading the certificate locally on your machine, create a configmap in the Kubernetes cluster that will copy the root CA certificate on your local machine into the Kubernetes cluster.

> **IMPORTANT**
>
> When running the command to create a configmap, the C-VEN and Kubelink require the file to have the **.crt** extension not work.

To create configmap, use the following command:

```
$ kubectl -n illumio-system create configmap root-ca-config \
    --from-file=./certs/root.democa.illumio-demo.com.crt
```

The `--from-file` option points to the path where the root CA certificate is stored on your local machine.

To verify that configmap was created correctly, use the following command:

```
$ kubectl -n illumio-system create configmap root-ca-config \
> --from-file=./certs/root.democa.illumio-demo.com.crt
configmap/root-ca-config created
$
$ kubectl -n illumio-system get configmap
NAME                             DATA   AGE
root-ca-config                   1      12s
$
$ kubectl -n illumio-system describe configmap root-ca-config
Name:         root-ca-config
Namespace:    illumio-system
Labels:       <none>
Annotations:  <none>

Data
====
root.democa.illumio-demo.com.crt:
----
-----BEGIN CERTIFICATE-----
MIIGSzCCBDOgAwIBAgIUAPw0NfPAivJW4YmKZ499eHZH3S8wDQYJKoZIhvcNAQEL
---output suppressed---
wPG0lug46K1EPQqMA7YshmrwOd6ESy6RGNFFZdhk9Q==
-----END CERTIFICATE-----

Events:  <none>
$
```

`root-ca-config` is the name used to designate configmap. You can modify it according to your naming convention.

Configure Calico in Append Mode

In case your cluster is configured with Calico as the network plugin (usually for Kubernetes and not for OpenShift), both Calico and Illumio Core will write iptables rules on the cluster nodes.

- Calico - Needs to write iptables rules to instruct the host how to forward packets (overlay, IPIP, NAT, and so on).
- Illumio Core - Needs to write iptables rules to secure communications between nodes and/or Pods.

You should establish a hierarchy to make the firewall coexistence work smoothly because Illumio Core and Calico will write rules at the same time. By default, both solutions are configured to insert rules first in the iptables chains/tables and Illumio Core will remove other rules added by a third-party software (in the Exclusive mode).

To allow Calico to write rules along with Illumio without flushing rules from one another, you should:

- Configure Illumio to work in Firewall Coexistence mode (default for workloads that are part of a container cluster).
- Configure Calico to work in Append mode (default is Insert mode).

To configure Calico to work in Append mode with iptables:

1. Edit the calico DaemonSet.

   ```
   kubectl -n kube-system edit ds calico-node
   ```
2. Locate the `spec: > template: > spec: > containers:` section inside the `YAML` file and change ChainInsertMode by adding the following code block:

   ```
   - name: FELIX_CHAININSERTMODE
     value: Append
   ```
3. Save your changes and exit.
4. Kubernetes will restart all Calico Pods in a rolling update.

For more information on changing Calico ChainInsertMode, see Calico documentation.

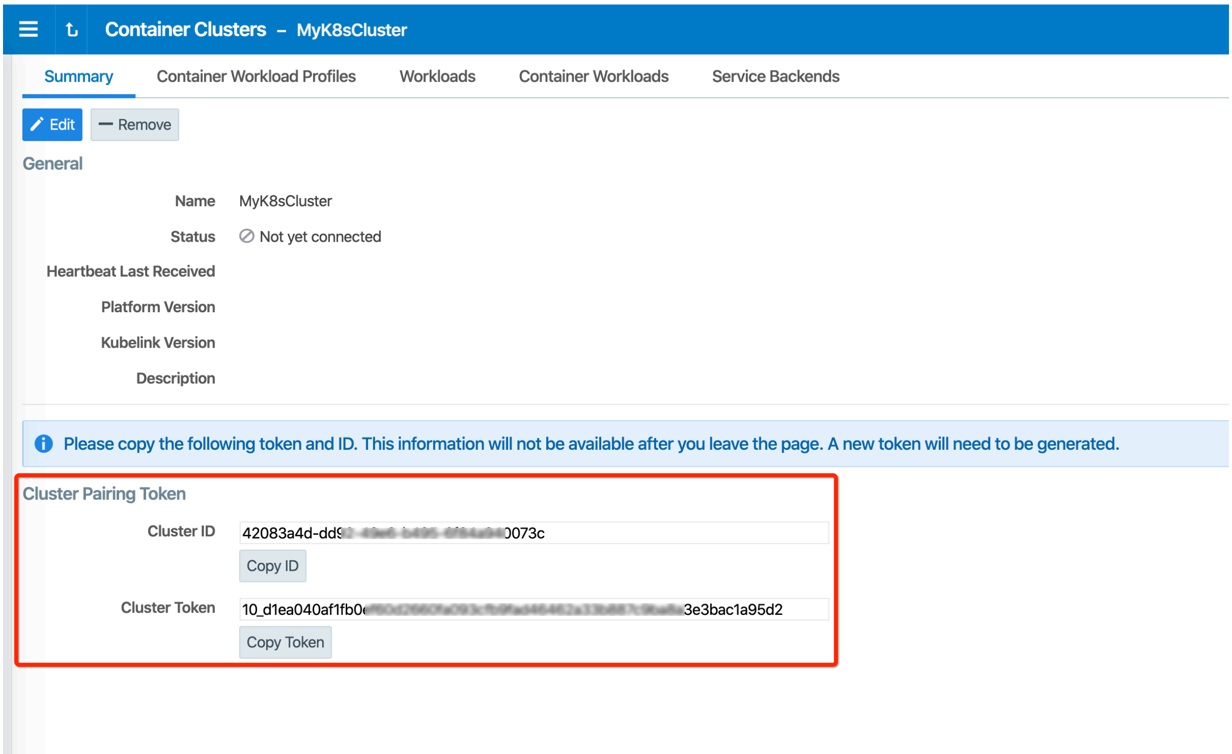
Create a Container Cluster in the PCE

To provide visibility and enforcement to your containerized environment, you first need to create a container cluster in the PCE. Each container cluster maps to an existing Kubernetes or OpenShift cluster.

Create a Container Cluster

To create a new container cluster:

1. Log into the PCE web console as a user with Global Organization Owner privileges.
2. From the PCE web console menu, navigate to **Infrastructure** > **Container Clusters**.
3. Click **Add**.
   a. Add a Name.
   b. **Save** the Container Cluster.
4. You will see a summary page of the new Container Cluster. From the Cluster Pairing Token section, copy the values of the Cluster ID and Cluster Token.
5. After copying and saving the values (in a text editor or similar tool), open the Container Workload Profiles page.



Configure a Container Workload Profile Template

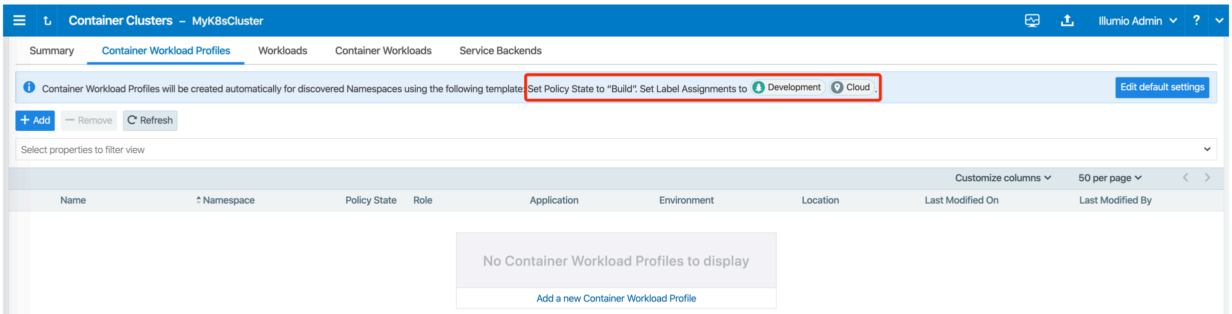
When configuring a new Container Cluster, it is recommended to set the default settings shared by all the Container Workload Profiles. Illumio provides a Container Workload Profile template that can be used for that purpose. By defining the default Policy State and minimum set of labels common to all namespaces in the cluster, you will save time later on when new namespaces are discovered by Kubelink. Each new profile created will inherit what was defined in the template.

> **IMPORTANT**
>
> Illumio does not provide a method to redefine at once all the labels associated with each profile. Hence, it is **strongly recommended** to use the provided template to define the default values for all profiles that are part of the same cluster.

To define the default parameters for all profiles using a template, under *Container Workload Profiles*, click **Edit default settings** and select values for all the fields.

For information about assigning default labels in the template, see the "Labels Restrictions for Kubernetes Namespaces" topic.

After you click OK, the following information is displayed:



Create Illumio Namespace

Create Illumio Namespace

Illumio Core for containers is deployed in a dedicated namespace illumio-system, by default. This namespace has the minimum privileges in the cluster required to run Illumio Core and can tie into the Kubernetes and OpenShift RBAC models.

To create the `illumio-system` namespace for Kubernetes, use the following command:

```
kubectl create namespace illumio-system
```

> **NOTE**
>
> Illumio provides a yaml manifest file to create the namespace in the Kubelink tarball `illumio-kubelink-namespace.yml`. You can create this namespace by applying this manifest file to your Kubernetes cluster, using the following command:
>
> ```
> kubectl apply -f illumio-kubelink-namespace.yml
> ```

To create the `illumio-system` project for OpenShift, use the following command:

```
oc new-project illumio-system
```

Authenticate Kubernetes Cluster with Container Registry

> **NOTE**
>
> Depending on your deployment, the steps in the Authenticate Kubernetes Cluster with Container Registry [306], Create a ConfigMap to Store Your Root CA Certificate [307], and Configure Calico in Append Mode [310] topics are optional.

When storing container images in a private container registry, it is often required and strongly recommended to authenticate against the registry to be able to pull an image from it. In order to do this, the Kubernetes or OpenShift cluster must have the credentials configured and stored in a secret file to be able to pull container images.

To configure a secret to store your container registry credentials, use the following command:

```
kubectl create secret docker-registry <container-registry-secret-name>
 -n illumio-system --docker-server=<container-registry> --docker-
username=<username>
 --docker-password=<password>
```

To verify that the secret has been created, use the following command:

```
kubectl get secret -n illumio-system | grep <container-registry-secret-name>
```

> **IMPORTANT**
> The above commands are valid for deployments with your own private container registry, but may not be valid for a cloud-managed private container registry. For more information, refer to your Cloud Service Provider documentation.

Create a ConfigMap to Store Your Root CA Certificate

This section describes how to implement Kubelink with a PCE using a certificate signed by a private PKI. It describes how to configure Kubelink and C-VEN to accept the certificate from the PCE signed by a private root or intermediate Certificate Authority (CA) and ensure that Kubelink can communicate in a secure way with the PCE.

Prerequisites

- Access to the root CA to download the root CA certificate.
- Access to your Kubernetes cluster and can run `kubectl` commands.
- Correct privileges in your Kubernetes cluster to create resources like a ConfigMaps, secrets, and Pods.
- Access to the PCE web console as a Global Organization Owner.

Download the Root CA Certificate

Before you begin, ensure that you have access to the root CA certificate. The root CA certificate is a file that can be exported from the root CA without compromising the security of the company. It is usually made available to external entities to ensure a proper SSL handshake between a server and its clients.

You can download the root CA certificate in the CRT format on your local machine. Below is an example of a root CA certificate:

```
$ cat root.democa.illumio-demo.com.crt
-----BEGIN CERTIFICATE-----
MIIGSzCCBDOgAwIBAgIUAPw0NfPAivJW4YmKZ499eHZH3S8wDQYJKoZIhvcNAQEL
---output suppressed---
```

```
wPG0lug46K1EPQqMA7YshmrwOd6ESy6RGNFFZdhk9Q==
-----END CERTIFICATE-----
```

You can also get the content of your root CA certificate in a readable output format by using the following command:

```
$ openssl x509 -text -noout -in ./root.democa.illumio-demo.com.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            fc:34:35:f3:c0:8a:f2:56:e1:89:8a:67:8f:7d:78:76:47:dd:2f
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=US, ST=California, L=Sunnyvale, O=Illumio,
        OU=Technical Marketing, CN=Illumio Demo Root
        CA 1/emailAddress=tme-team@illumio.com
        Validity
            Not Before: Jan 20 00:05:36 2020 GMT
            Not After : Jan 17 00:05:36 2030 GMT
        Subject: C=US, ST=California, L=Sunnyvale, O=Illumio,
        OU=Technical Marketing, CN=Illumio Demo Root
        CA 1/emailAddress=tme-team@illumio.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (4096 bit)
                Modulus:
                    00:c0:e5:48:7d:97:f8:5b:8c:ef:ac:16:a8:8c:aa:
                    68:b8:48:af:28:cd:17:8f:02:c8:82:e9:69:62:e2:
                    89:2b:be:bd:34:fc:e3:4d:3f:86:5e:d7:e6:89:34:
                    71:60:e6:54:61:ac:0f:26:1c:99:6f:80:89:3f:36:
                    b3:ad:78:d1:6c:3f:d7:23:1e:ea:51:14:48:74:c3:
                    e8:6e:a2:79:b1:60:4c:65:14:2a:f1:a0:97:6c:97:
                    50:43:67:07:b7:51:5d:2c:12:49:81:dc:01:c9:d1:
                    57:48:32:2e:87:a8:d2:c0:b9:f8:43:b2:58:10:af:
                    54:59:09:05:cb:3e:f0:d7:ef:70:cc:fc:53:48:ee:
                    a4:a4:61:f1:d7:5b:7c:a9:a8:92:dc:77:74:f4:4a:
                    c0:4a:90:71:0f:6d:9e:e7:4f:11:ab:a5:3d:cd:4b:
                    8b:79:fe:82:1b:16:27:94:8e:35:37:db:dd:b8:fe:
                    fa:6d:d9:be:57:f3:ca:f3:56:aa:be:c8:57:a1:a8:
                    c9:83:dd:5a:96:5a:6b:32:2d:5e:ae:da:fc:85:76:
                    bb:77:d5:c2:53:f3:5b:61:74:e7:f3:3e:4e:ad:10:
                    7d:4f:ff:90:69:7c:1c:41:2f:67:e4:13:5b:e6:3a:
                    a3:2f:93:61:3b:07:56:59:5a:d9:bc:34:4d:b3:54:
                    b5:c6:e5:0a:88:e9:62:7b:4b:85:d2:9e:4c:ee:0b:
                    0d:f4:72:b1:1b:44:04:93:cf:cc:bb:18:31:3a:d4:
                    83:4a:ff:15:42:2d:91:ca:d0:cb:36:d9:8d:62:c0:
                    41:59:1a:93:c7:27:79:08:94:b2:a2:50:3c:57:27:
                    33:af:f0:b6:92:44:49:c5:09:15:a7:43:2a:0f:a9:
                    02:61:b3:66:4f:c3:de:d3:63:1e:08:b1:23:ea:69:
                    90:db:e8:e9:1e:21:84:e0:56:e1:8e:a1:fa:3f:7a:
                    08:0f:54:0a:82:41:08:6b:6e:bb:cf:d6:5b:80:c6:
                    ea:0c:80:92:96:ab:95:5d:38:6d:4d:da:38:6b:42:
                    ef:7c:88:58:83:88:6d:da:28:62:62:1f:e5:a7:0d:
                    04:9f:0d:d9:52:39:46:ba:56:7c:1d:77:38:26:7c:
                    86:69:58:4d:b0:47:3a:e2:be:ee:1a:fc:4c:de:67:
```

```
                    f3:d5:fe:e6:27:a2:ef:26:86:19:5b:05:85:9c:4c:
                    02:24:76:58:42:1a:f8:e0:e0:ed:78:f2:8f:c8:5a:
                    20:a9:2d:0b:d4:01:fa:57:d4:6f:1c:0a:31:30:8c:
                    32:7f:b0:01:1e:fe:94:96:03:ee:01:d7:f4:4a:83:
                    f5:06:fa:60:43:15:05:9a:ca:88:59:5c:f5:13:09:
                    82:69:7f
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                3D:3D:3D:61:E6:88:09:FE:34:0F:1D:5E:5E:52:72:71:C7:DE:15:92
            X509v3 Authority Key Identifier:

keyid:3D:3D:3D:61:E6:88:09:FE:34:0F:1D:5E:5E:52:72:71:C7:DE:15:92

            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Key Usage: critical
                Digital Signature, Certificate Sign, CRL Sign
    Signature Algorithm: sha256WithRSAEncryption
         28:24:86:91:a6:4a:88:e4:8d:6b:fc:67:2a:68:08:67:35:e5:
         a6:77:ff:07:4b:89:53:99:2e:6d:95:df:12:81:28:6a:8e:6f:
         5a:98:95:5b:4a:21:ae:f0:20:a4:4e:06:b2:4e:5a:67:c1:6a:
         06:f1:0f:c1:f7:7e:f2:e0:b3:9d:d8:54:26:6a:b2:1c:19:b8:
         b5:5c:c7:03:6b:f7:70:9e:72:85:c9:29:55:f9:f4:a4:f2:b4:
         3b:3d:ce:25:96:67:32:1e:8d:e2:00:22:55:4b:05:4f:ee:0e:
         67:ac:db:1b:61:da:5f:9c:10:1c:0c:05:66:c0:5b:5f:b9:95:
         59:a9:58:5b:e7:69:ac:b0:bd:b3:c2:a3:35:58:01:a4:ff:c0:
         8d:ac:1c:19:21:41:50:fb:8e:e0:f5:a9:ad:ec:de:cb:53:04:
         a9:d8:ac:76:8a:09:0d:7c:c6:1a:bc:06:74:bb:10:1c:aa:07:
         f6:cb:b2:1b:0c:0c:65:03:45:2b:51:d5:6e:a0:4d:91:ce:c5:
         ed:8d:a9:e7:f6:37:7d:ab:1b:a4:a2:a3:3b:76:17:5b:d9:3a:
         9c:c1:df:cc:cd:a0:b0:a9:5c:74:61:d7:a0:1d:04:67:68:ee:
         a6:7b:1e:41:a4:02:fc:65:9e:e3:c1:c2:57:b2:2e:b0:ff:a9:
         86:82:35:4d:29:b2:fe:74:2e:b8:37:5d:2b:e8:69:f2:80:29:
         19:f1:1e:7a:5d:e3:d2:51:50:46:30:54:7e:b8:ad:59:61:24:
         45:a8:5a:fe:19:ff:09:31:d0:50:8b:e2:15:c0:a2:f1:20:95:
         63:55:18:a7:a2:ad:16:25:c7:a3:d1:f2:e5:be:6d:c0:50:4b:
         15:ac:e0:10:5e:f3:7b:90:9c:75:1a:6b:e3:fb:39:88:e4:e6:
         9f:4c:85:60:67:e8:7d:2e:85:3d:87:ed:06:1d:13:0b:76:d7:
         97:a5:b8:05:76:67:d6:41:06:c5:c0:7a:bd:f4:c6:5b:b2:fd:
         23:6f:1f:57:2e:df:95:3f:26:a5:13:4d:6d:96:12:56:98:db:
         2e:7d:fd:56:f5:71:b7:19:2b:c9:de:2d:b9:c8:17:cc:20:de:
         7c:19:7a:aa:12:97:1c:80:b7:d3:67:d3:b7:a7:96:f0:c9:4d:
         f5:8b:0e:10:3b:b9:4e:09:90:5a:3b:51:c9:48:a2:ca:9f:db:
         72:44:87:59:db:49:fa:75:44:b5:f6:7f:c5:26:e1:01:ae:7b:
         6f:4a:75:d1:b5:b3:68:c0:31:48:f8:5c:06:c0:f1:b4:96:e8:
         38:e8:ad:44:3d:0a:8c:03:b6:2c:86:6a:f0:39:de:84:4b:2e:
         91:18:d1:45:65:d8:64:f5
```

Create a ConfigMap in Kubernetes Cluster

After downloading the certificate locally on your machine, create a ConfigMap in the Kubernetes cluster that will copy the root CA certificate on your local machine into the Kubernetes cluster.

To create the ConfigMap, use the following command:

```
$ kubectl -n illumio-system create configmap root-ca-config \
    --from-file=./certs/root.democa.illumio-demo.com.crt
```

The `--from-file` option points to the path where the root CA certificate is stored on your local machine.

To verify that the ConfigMap was created correctly, use the following command:

```
$ kubectl -n illumio-system create configmap root-ca-config \
> --from-file=./certs/root.democa.illumio-demo.com.crt
configmap/root-ca-config created
$
$ kubectl -n illumio-system get configmap
NAME                                 DATA   AGE
root-ca-config                       1      12s
$
$ kubectl -n illumio-system describe configmap root-ca-config
Name:         root-ca-config
Namespace:    illumio-system
Labels:       <none>
Annotations:  <none>

Data
====
root.democa.illumio-demo.com.crt:
----
-----BEGIN CERTIFICATE-----
MIIGSzCCBDOgAwIBAgIUAPw0NfPAivJW4YmKZ499eHZH3S8wDQYJKoZIhvcNAQEL
---output suppressed---
wPG0lug46K1EPQqMA7YshmrwOd6ESy6RGNFFZdhk9Q==
-----END CERTIFICATE-----

Events:  <none>
$
```

`root-ca-config` is the name used to designate the ConfigMap. You can modify it according to your naming convention.

Configure Calico in Append Mode

In case your cluster is configured with Calico as the network plugin (usually for Kubernetes and not for OpenShift), both Calico and Illumio Core will write iptables rules on the cluster nodes.

- Calico - Needs to write iptables rules to instruct the host how to forward packets (overlay, IPIP, NAT, and so on).
- Illumio Core - Needs to write iptables rules to secure communications between nodes and/or Pods.

You should establish a hierarchy to make the firewall coexistence work smoothly because Illumio Core and Calico will write rules at the same time. By default, both solutions are configured to insert rules first in the iptables chains/tables and Illumio Core will remove other rules added by a third-party software (in the Exclusive mode).

To allow Calico to write rules along with Illumio without flushing rules from one another, you should:

- Configure Illumio to work in Firewall Coexistence mode (default for workloads that are part of a container cluster).
- Configure Calico to work in Append mode (default is Insert mode).

To configure Calico to work in Append mode with iptables:

1. Edit the calico DaemonSet.

   ```
   kubectl -n kube-system edit ds calico-node
   ```
2. Locate the `spec: > template: > spec: > containers:` section inside the `YAML` file and change ChainInsertMode by adding the following code block:

   ```
   - name: FELIX_CHAININSERTMODE
     value: Append
   ```
3. Save your changes and exit.
4. Kubernetes will restart all Calico Pods in a rolling update.

For more information on changing Calico ChainInsertMode, see Calico documentation.

Deploy Kubelink in Your Cluster

Download the required resources such as, Kubelink docker image, secret, and deployment files from the Illumio Support portal (login required).

Prerequisites

- Kubelink deployment file provided by Illumio.
- Kubelink secret file provided by Illumio.
- Illumio's Kubelink docker image uploaded to your private docker registry.

Configure Kubelink Secret

This section assumes that you have created a Container Cluster object in the PCE. You will need the Cluster ID and Cluster Token values for the Kubelink secret.

1. Open the Kubelink secret YAML file and modify the following keys that are listed under `stringData`:
   a. `ilo_server` = the PCE URL and port. Example: `https://mypce.example.com:8443`
   b. `ilo_cluster_uuid` = Cluster ID value from previous step. Example: `15643adc-ac09-40f2-be63-fd9a261f41cc`
   c. `ilo_cluster_token` = Cluster Token from previous step. Example: `1_e94c116a4485ab1bb8560728afd6a332182b849c841297f63e73a87bf255cc96`
   d. `ignore_cert` = SSL verification. The value is boolean and is recommended to be set to false so that Kubelink requires PCE certificate verification. Example: '`false`'
   e. `log_level` = Log level where '`0`' for debug, '`1`' for info, '`2`' for warn, or '`3`' for error. Example: '`1`'

> **IMPORTANT**
>
> Illumio does not recommend turning off SSL verification (`ignore_cert`: '`true`'). However, this is an option for deployments in which the PCE uses a self-signed certificate. For PCE deployments using a certificate signed with a private PKI, there is no need to set the `ignore_cert key` to 'false'. For more details, see Create a ConfigMap to Store Your Root CA Certificate [300].

The contents of a modified `illumio-kubelink-secret.yml` file are shown below.

```
#
# Copyright 2013-2021 Illumio, Inc. All Rights Reserved.
#

apiVersion: v1
kind: Secret
metadata:
  name: illumio-config
  namespace: illumio-system
type: Opaque
stringData:
  ilo_server: https://mypce.example.com:8443 # Example:
  https://mypce.example.com:8443
  ilo_cluster_uuid: 42083a4d-dd92-49e6-b495-6f84a940073c #
Example: cc4997c1-408b-4f1d-a72b-91495c24c6a0
  ilo_cluster_token: 10_d1ea040af1fb0ef60d2660fa093cfb9fad46462
  a33b887c9ba8a3e3bac1a95d
# Example: 170b8aa3dd6d8aa3c284e9ea016e8653f7b51cb4b0431d8cbdba
  11508763f3a3
  ignore_cert: 'false' # Set to 'true' to ignore the PCE certificate
  log_level: '1' # Default log level is info
```

> **NOTE**
>
> If you are going to use a private PKI to sign the PCE certificate, see Create a ConfigMap to Store Your Root CA Certificate [300] before deploying Kubelink.

2. Save the changes.
3. Create the Kubelink secret in your Kubernetes or OpenShift cluster.
   - Deploy Kubelink secret in Kubernetes:

   ```
   kubectl apply -f illumio-kubelink-secret.yml
   ```

   - Deploy Kubelink secret in OpenShift:

   ```
   oc apply -f illumio-kubelink-secret.yml
   ```
4. Verify the Kubelink secret creation in your Kubernetes cluster.
   - Verify Kubelink secret in Kubernetes:

   ```
   kubectl get secret -n illumio-system
   ```

   - Verify Kubelink secret in OpenShift:

```
oc get secret -n illumio-system
```

Deploy Kubelink

Modify the Kubelink configuration file to point to the correct Docker image. The example in this document has `illumio-kubelink:2.0.x.xxxxxx` uploaded to `registry.example.com`, so the image link in this example is: `registry.example.com/illumio-kubelink:2.0.x.xxxxxx`

1. Edit the Kubelink configuration YAML file. The file name is `illumio-kubelink-kubernetes.yml` for a Kubernetes cluster or `illumio-kubelink-openshift.yml` for an OpenShift cluster.
   - Locate the `spec: > template: > spec: > containers:` section inside the YAML file. Modify the image link in the `image:` attribute.
2. Save the changes.

   Below is a snippet from an example of the Kubelink configuration for Kubernetes to illustrate the image location.

```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: illumio-kubelink
  namespace: illumio-system
spec:
  replicas: 1
  selector:
    matchLabels:
      app: illumio-kubelink
  template:
    metadata:
      labels:
        app: illumio-kubelink
    spec:
#      nodeSelector:
#        node-role.kubernetes.io/master: ""
      serviceAccountName: illumio-kubelink
      tolerations:
      - key: node-role.kubernetes.io/master
        effect: NoSchedule
      containers:
      - name: illumio-kubelink
        image: registry.example.com/illumio-kubelink:2.0.x.xxxxxx
        imagePullPolicy: Always
        env:
          - name: ILO_SERVER
            valueFrom:
              secretKeyRef:
                name: illumio-kubelink-config
                key: ilo_server
```

3. (Optional) Reference your root CA certificate.

   If you are using a private PKI to sign the PCE certificate, make sure you add the references to the root CA certificate that signed the PCE certificate. By default, the current manifest file provided by Illumio does not include this modification. Open the .yml file and add the following code blocks:
   - `volumeMounts` (under `spec.template.spec.containers`)

- `volumes` (under `spec.template.spec`)

  `root-ca` is the name used to designate the new volume mounted in the container. You can modify it according to your naming convention.

```
        volumeMounts:
        - name: root-ca
          mountPath: /etc/pki/tls/ilo_certs/
          readOnly: false
      volumes:
      - name: root-ca
        configMap:
          name: root-ca-config
```

4. (Optional) Reference your container registry secret. See the Authenticate Kubernetes Cluster with Container Registry [296] section.

   In case you need to authenticate against your container registry when you pull an image from your cluster, you must make reference to the secret previously created for the container registry. Locate the `spec: > template: > spec:` section inside the YAML file and add the following lines:

```
      imagePullSecrets:
      - name: <container-registry-secret-name>
```

   IMPORTANT

   Indentation matters in a YAML file. Make sure there are 6 spaces to the left before inserting the 'imagePullSecrets' keyword and align the '-' character below it with the 'i' of the 'imagePullSecrets' keyword.

5. Deploy Kubelink.
   - To deploy Kubelink for Kubernetes:

     ```
     kubectl apply -f illumio-kubelink-kubernetes.yml
     ```
   - To deploy Kubelink for OpenShift:

     ```
     oc apply -f illumio-kubelink-openshift.yml
     ```
6. Verify your deployment.
   - To check the Kubelink Pod status for Kubernetes:

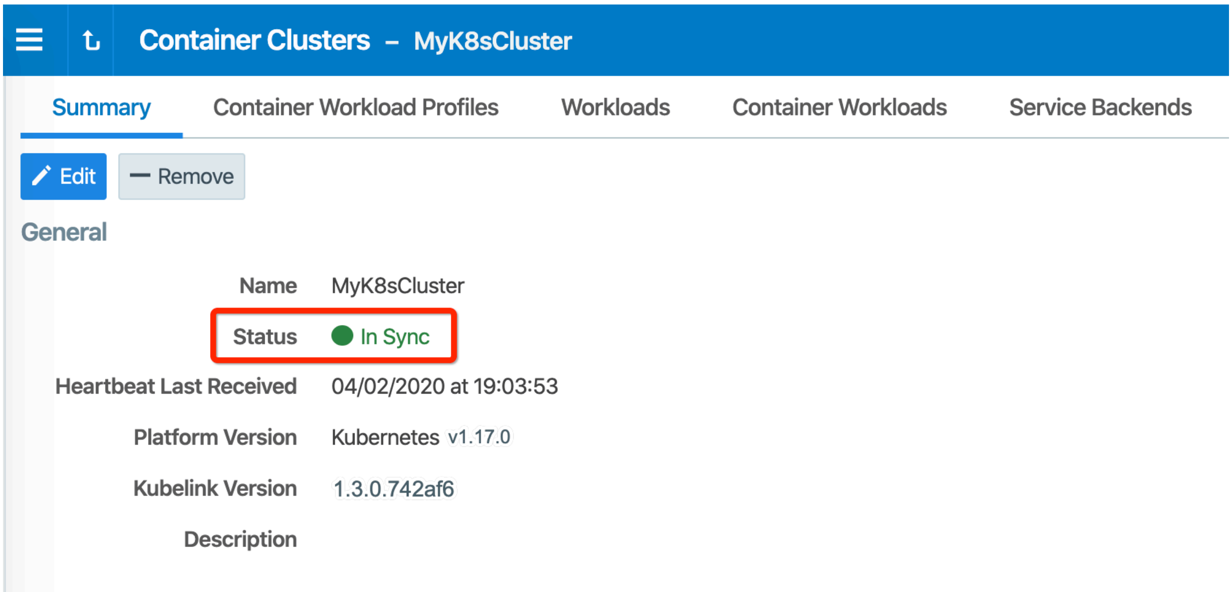
     ```
     kubectl get pods -n illumio-system
     ```
   - To check the Kubelink Pod status for OpenShift:

     ```
     oc get pods -n illumio-system
     ```

The `illumio-kubelink-xxxxxxxxxx-xxxxx` Pod should be in the "Running" state.

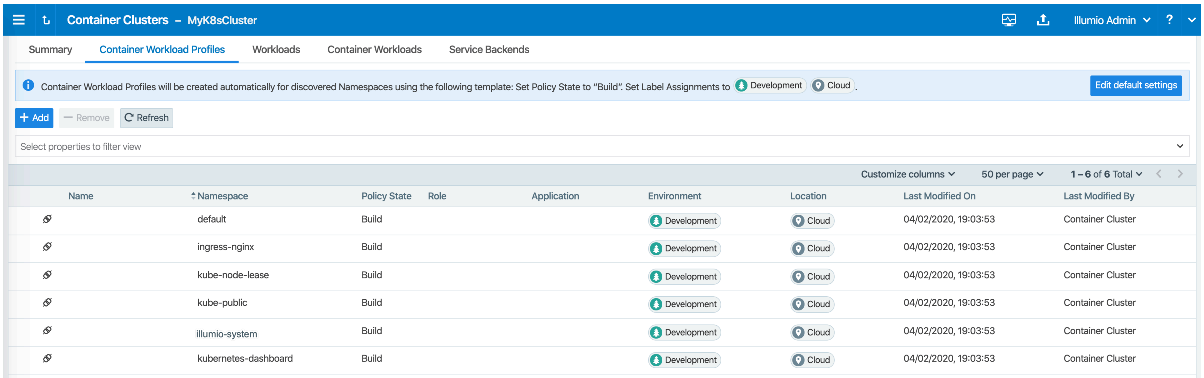
After Kubelink is successfully deployed, you can check the cluster information in the Illumio PCE UI. From the main menu, navigate to **Infrastructure > Container Clusters**.

Below is an example of a healthy container cluster state reported by Kubelink, where Status is "In Sync".

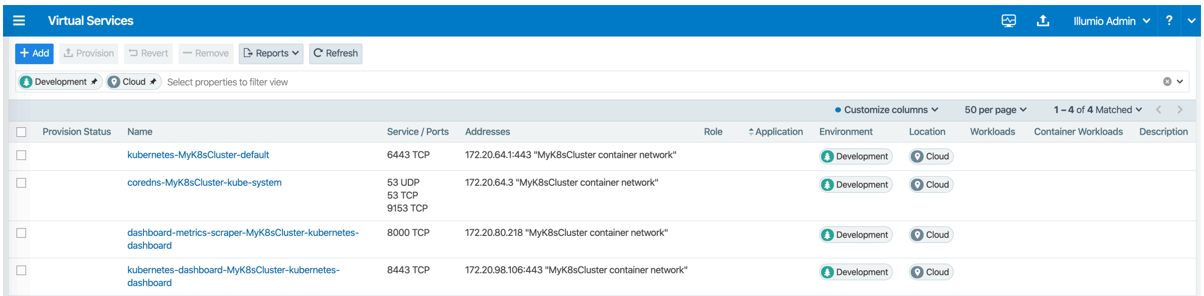


You can also verify in the PCE UI that Kubelink was successfully deployed by checking the following:

- Under the **Container Workload Profiles** tab, namespaces created in your Kubernetes or OpenShift cluster should be listed. An example is shown below.



- Under **Policy Objects > Virtual Services**, services created in your Kubernetes or OpenShift cluster should be listed. An example is shown below.



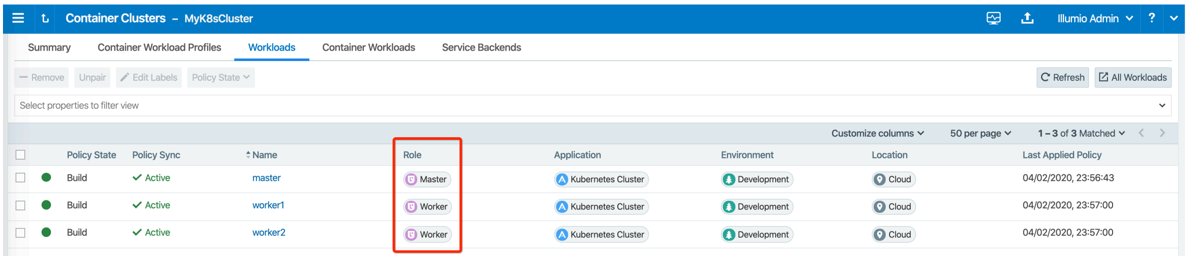
Re-Label Your Cluster Nodes

> **NOTE**
> Re-labeling the cluster nodes is optional.

In the case of self-managed deployments in which both Master and Worker nodes are managed, you may want to re-label your nodes to differentiate Master nodes from Worker nodes. Doing this helps when you are writing different policies for the Worker and Master nodes or if you want to segment these nodes differently.

To re-label your cluster nodes:

1. In the PCE UI, go to **Infrastructure > Container Clusters > YourClusterName > Workloads**.
2. Select the workloads you want to re-label.
3. Click **Edit Labels** to assign the new labels (for example, Master and Worker).



4. After re-labeling your cluster nodes, the nodes part of the cluster reflect the updated label(s).

Deploy C-VENs in Your Cluster

> **IMPORTANT**
> Before deploying the C-VEN, ensure that either of the following two requirements has been met:
>
> - Kubelink is deployed on the Kubernetes cluster and is in sync with the PCE, or
> - Firewall coexistence is enabled.

Prerequisites

- VEN deployment file provided by Illumio.
- VEN secret file provided by Illumio.

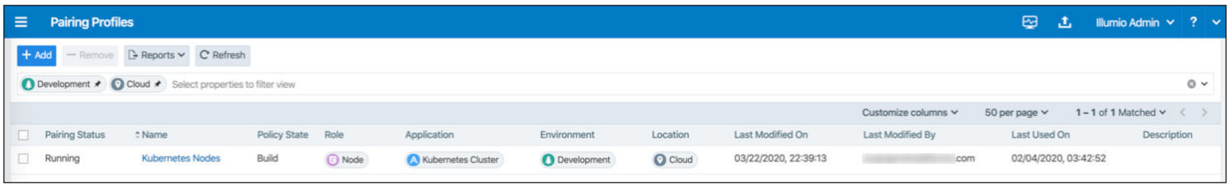
- Illumio's C-VEN docker image uploaded to a private container registry.
- In OpenShift, create the 'illumio-ven' service account in the 'illumio-system' project and add this account to the privileged Security Context Constraint (SCC):
  - `oc create sa illumio-ven`
  - `oc adm policy add-scc-to-user privileged -z illumio-ven -n illumio-system`

Create a Pairing Profile for Your Cluster Nodes

Before deploying the C-VEN in your cluster, you should create a pairing profile to pair the cluster nodes with the PCE. You only need to create one pairing profile for all your nodes.

> **NOTE**
> You only need to create pairing profiles for Kubernetes or OpenShift nodes and not for container workloads.

For ease of configuration and management, consider applying the same Application, Environment, and Location labels across all nodes of the same Kubernetes or OpenShift cluster. The screenshot below shows an example of a pairing profile for a Kubernetes cluster.

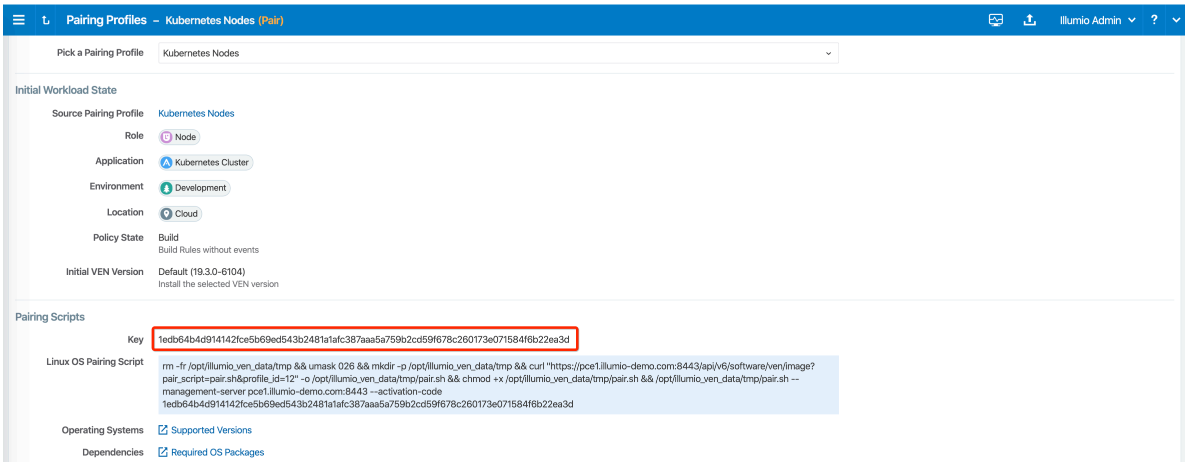


> **TIP**
> Illumio recommends all pairing profiles for Kubernetes nodes *not* to use the Full enforcement policy state. Use Visibility Only mode for initial configuration.
>
> You should only move them into Full enforcement state after you have completed all other configuration steps in this guide, such as setting up Kubelink, discovering services, and writing rules.

Configure C-VEN Secret

This section assumes that you have already created a Pairing Profile in the PCE. You will need the activation code for the C-VEN secret.

1. To retrieve the activation code from the pairing profile, go to **Policy Objects > Pairing Profiles**, open the pairing profile created for your cluster nodes, and click **Generate Key**.



2. After copying and saving the **Key** (in a text editor or similar tool), you can exit the page.
3. Open the C-VEN secret YAML file and modify the following keys (under `stringData`):
   - `ilo_server` = PCE URL and port. Example: mypce.example.com:8443
   - `ilo_code` = Activation code value from Step 1. Example: 1edb64b4d914142fce5b69ed543b2481a1afc387aaa5a759b2cd59f678c260173e071584f6b22ea3d

   Contents of a modified `illumio-ven-secret.yml` file are shown below.

```
#
# Copyright 2013-2021 Illumio, Inc. All Rights Reserved.
#
# VEN 21.5.x-xxxx

apiVersion: v1
kind: Secret
metadata:
  name: illumio-ven-config
  namespace: illumio-system
type: Opaque
stringData:
  ilo_server: mypce.example.com:8443 # Example: mypce.example.com:8443
  ilo_code: 1edb64b4d914142fce5b69ed543b2481a1afc387aaa5a759b2cd59f678c
  260173e071584f6b22ea3d # activation-code
```

> **CAUTION**
> Do not use 'https://' for the value associated with the `ilo_server:` key. This is a known issue and will be fixed in a future release.

4. Save the changes.
5. Create the C-VEN secret using the file.
   - To create the secret for Kubernetes:

```
kubectl apply -f illumio-ven-secret.yml
```

   - To create the secret for OpenShift:

```
oc apply -f illumio-ven-secret.yml
```

6. Verify the C-VEN secret creation in your cluster.
   - To verify the creation of the secret for Kubernetes:

```
kubectl get secret -n illumio-system
```

```
oc get secret -n illumio-system
```

Deploy C-VENs

Modify the C-VEN configuration file to point to the correct Docker image. The example in this document has `illumio-ven:21.5.x-xxxx` uploaded to `registry.example.com:443`, so the image link in this example is: `registry.example.com:443/illumio-ven:21.5.x-xxxx`

1. Edit the C-VEN configuration YAML file. The file name is `illumio-ven-kubernetes.yml` for a Kubernetes cluster and `illumio-ven-openshift.yml` for an OpenShift cluster.
   - Locate the `spec: > template: > spec: > containers:` section inside the YAML file. Modify the image link in the `image:` attribute.
2. Save the changes.

   Below is a snippet from an example of the C-VEN configuration for Kubernetes or OpenShift to illustrate the image location.

```
#
# Copyright 2013-2021 Illumio, Inc. All Rights Reserved.
#
# VEN 21.5.x-xxxx

---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: illumio-ven
  namespace: illumio-system
---
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: illumio-ven
  namespace: illumio-system
  labels:
    k8s-app: illumio-ven
spec:
  selector:
    matchLabels:
      name: illumio-ven
  template:
    metadata:
      labels:
        name: illumio-ven
    spec:
      priorityClassName: system-node-critical
      serviceAccountName: illumio-ven
      hostNetwork: true
      hostPID: true
      tolerations:
      - key: node-role.kubernetes.io/master
        effect: NoSchedule
      containers:
        - name: illumio-ven
          env:
            - name: ILO_SERVER
              valueFrom:
```

```
            secretKeyRef:
              name: illumio-ven-config
              key: ilo_server
        - name: ILO_CODE
          valueFrom:
            secretKeyRef:
              name: illumio-ven-config
              key: ilo_code
      command: [ "/ven-init", "activate" ]
      image: registry.example.com/illumio-ven:21.5.x-xxxx
      imagePullPolicy: IfNotPresent
      <...>
```

3. (Optional) Reference your root CA certificate.

   If you are using a private PKI to sign your PCE certificate, make sure you add the references to the root CA certificate that signed the PCE certificate. If Kubelink is already deployed in the cluster, the ConfigMap used to store the root CA certificate should already be created in the cluster.

   Add the following sections to the C-VEN manifest file to reference the ConfigMap containing the root CA certificate:

   - `volumeMounts` (under `spec.template.spec.containers`)
   - `volumes` (under `spec.template.spec`)

   `root-ca` is the name used to designate the new volume mounted in the container. You can modify it according to your naming convention.

   ```
     volumeMounts:
     - name: root-ca
       mountPath: /etc/pki/tls/ilo_certs/
       readOnly: false
   volumes:
   - name: root-ca
     configMap:
       name: root-ca-config
   ```

4. (Optional) Reference your container registry secret. See the "Authenticate Kubernetes Cluster with Container Registry" topic.

   In case you need to authenticate against your container registry when you pull an image from your cluster, you must make reference to the secret previously created for the container registry. Locate the `spec: > template: > spec:` section inside the YAML file and add the following lines:

   ```
   imagePullSecrets:
   - name: <container-registry-secret-name>
   ```

   IMPORTANT

   Indentation matters in a YAML file. Make sure there are 6 spaces to the left before inserting the 'imagePullSecrets' keyword and align the '-' character below it with the 'i' of the 'imagePullSecrets' keyword.

> **NOTE**
>
> From the 20.2.0 and later releases, the container runtime detection is done automatically. You do not need to manually modify the container runtime socket path. You should do this 'Modify the container runtime socket path' step only if you are using a customized configuration for your container runtime.

5. (Optional) Modify the container runtime socket path.

   In some cases, you have to modify the default socket path the C-VEN relies on to get information about the containers due to the following reasons:
   - A non-conventional or customized container runtime socket path
   - Two concurrent container runtimes

   In this case, you may have to modify the default mount path for the `unixsocks` volume in the C-VEN configuration file.

   For example, you want to listen on the 'containerd' container runtime, however, docker is also used on the nodes. You should modify the file as shown below, so that the C-VEN listens to events on 'containerd':

   ```
             volumeMounts:
             - name: unixsocks
               mountPath: /var/run/containerd/
               <...>
         volumes:
         - name: unixsocks
           hostPath:
             path: /var/run/containerd/
             type: Directory
             <...>
   ```
6. Save the changes.
7. Deploy C-VEN.
   - For Kubernetes:

   ```
   kubectl apply -f illumio-ven-kubernetes.yml
   ```
   - For OpenShift:

   ```
   oc apply -f illumio-ven-openshift.yml
   ```
8. Verify the deployment.
   - For Kubernetes:

   ```
   kubectl get pods -n illumio-system
   ```

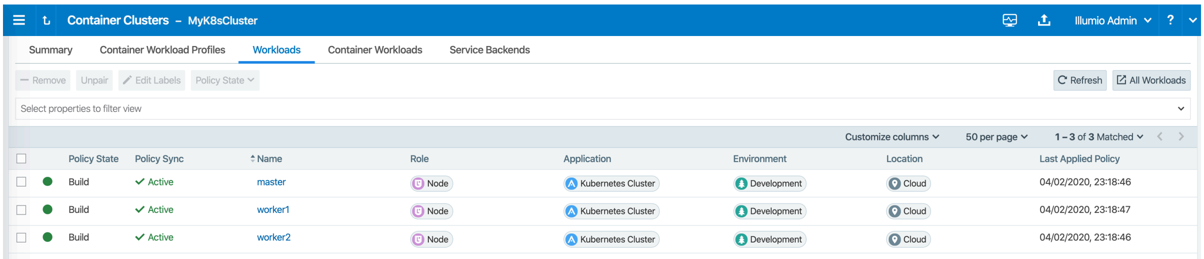
   ```
   oc get pods -n illumio-system
   ```

The `illumio-ven-xxxxxxxxxx-xxxxx` Pods should be in the "Running" state.

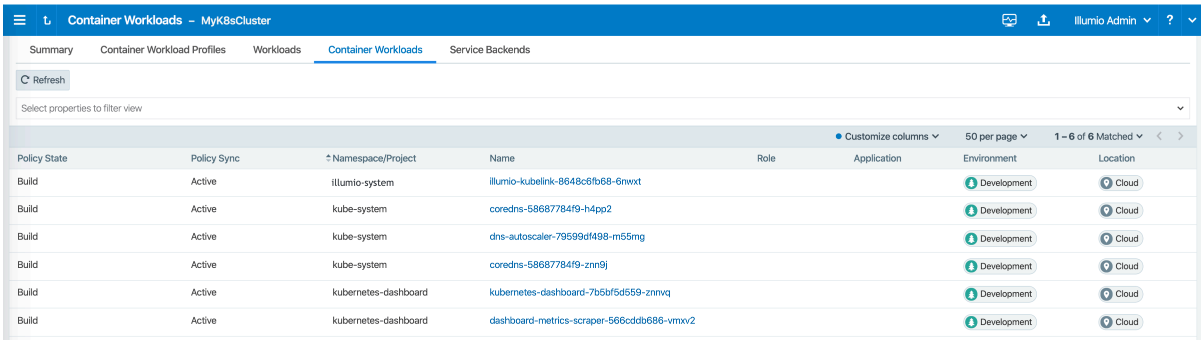
After C-VENs are successfully deployed, you can check the cluster information in the Illumio PCE UI. From the main menu, navigate to **Infrastructure > Container Clusters**.

You can also verify in the PCE UI that the C-VENs were successfully deployed by checking the following:

- Under the **Workload** tab, nodes that are part of your Kubernetes or OpenShift cluster should be listed. An example is shown below.



- Under the **Container Workloads** tab, Pods deployed in your Kubernetes or OpenShift cluster should be listed. An example is shown below.



- Illumination Map now displays system and application Pods running in your cluster.

Configure Security Policies for Containerized Environments

Security policies are a set of rules that you can configure to secure your Kubernetes or Open-Shift environment. You can follow the guidelines and examples described in this section to write rules for your Kubernetes or OpenShift clusters and containerized applications, which you can then modify incrementally.

IP and FQDN Lists

FQDN Services for Kubernetes

There are some basic services that need to be defined as IP lists, such as docker.io or the Kubernetes API server. These FQDNs will be used later in the ring-fence policy for the Kuber-netes cluster. The following FQDNs are commonly found to be dependencies for Kubernetes and should be defined inside Illumio Core's IP list policy objects:

- docker.io
- myregistry.example.com

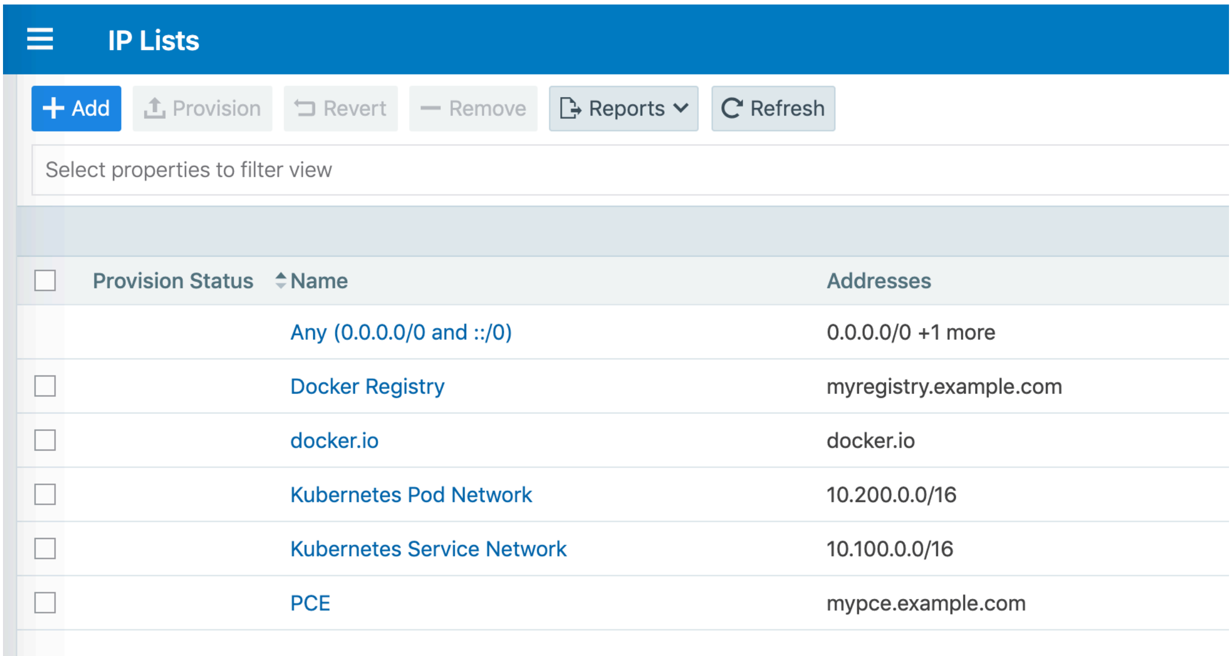
The PCE FQDN is required for Kubelink for example, mypce.example.com.

IP Lists for Kubernetes

Additionally, the following subnets or IP addresses should be defined in the IP list policy objects:

- **Kubernetes Pod Network:** Locate subnet in master node's `/etc/kubernetes/kubeadm-config.yaml` file (Ubuntu) under `networking: > podSubnet:` section, for example, `10.200.0.0/16`
- **Kubernetes Service Network:** Locate subnet in master node's `/etc/kubernetes/kubeadm-config.yaml` file (Ubuntu) under `networking > serviceSubnet` section, for example, `10.100.0.0/16`

The screenshot below displays IP lists created for Kubernetes Infrastructure dependencies.



FQDN Services for OpenShift

There are some basic services that should be defined as IP lists such as docker.io or the Kubernetes API server. These FQDNs will be used later in the ring fence policy for the OpenShift cluster. The following FQDNs are commonly found to be dependencies for OpenShift and should be defined in Illumio IP list policy objects:

- docker.io
- registry.access.redhat.com
- access.redhat.com
- subscription.rhsm.redhat.com
- github.com

The PCE FQDN is required for Kubelink, for example, mypce.example.com.

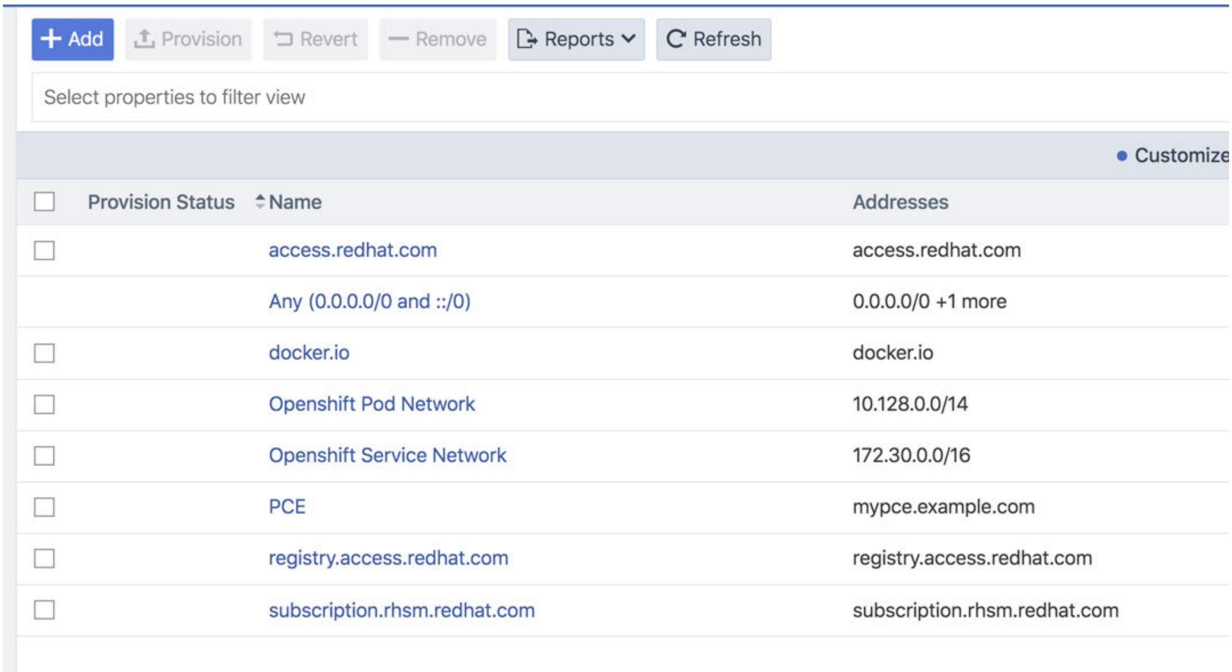
IP Lists for OpenShift

Additionally, the following subnets or IP addresses should be defined in IP list policy objects:

- **OpenShift Pod Network:** Find subnet in master node's `/etc/origin/master/master-config.yaml` file under `networkConfig > clusterNetworkCIDR` section, for example, `10.128.0.0/14`

- **OpenShift Service Network:** Find subnet in master node's `/etc/origin/master/master-config.yaml` file under `networkConfig > serviceNetworkCIDR` section, for example, `172.30.0.0/16`

The screenshot below displays IP lists created for OpenShift Infrastructure dependencies. It references the IP lists which automatically come with the Illumio Segmentation Template.



NOTE

The IP lists mentioned above are for FQDNs and IP addresses that Illumio has found to be necessary for basic Kubernetes or OpenShift deployments. Each deployment varies and may have dependencies on additional FQDNs or IP addresses that are not mentioned in this document.

If your Kubernetes or OpenShift infrastructure needs to communicate with external services that are not mentioned here, then make sure you describe those in the IP lists.

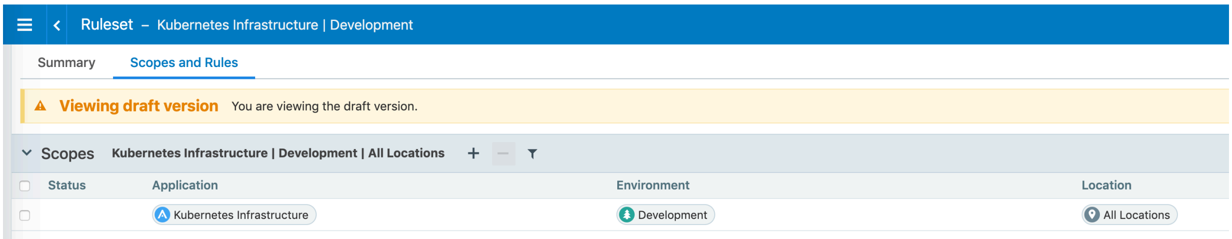
Rules for Kubernetes or OpenShift Clusters

This section assumes the following:

- Kubernetes or OpenShift cluster nodes and infrastructure Pods are activated and managed.
- Labels have been assigned to each workload and container workload.
- All cluster nodes and infrastructure Pods are in the same application group, which means they have been assigned the same application, environment, and location labels.

Kubernetes

Create a ruleset for the Kubernetes cluster and control plane Pods. The labels assigned to all of the Kubernetes nodes and control Pod workloads should fall within the scope.



Add the following lines of policy to the ruleset.

Intra-Scope Rules

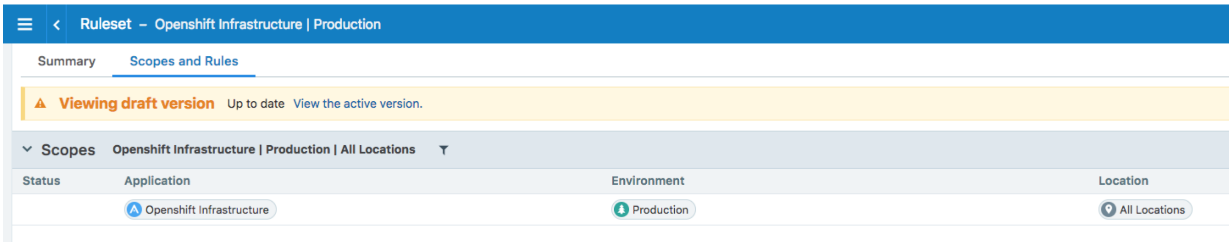
Providers	Services	Consumers	Notes
docker.io (IP List) myregistry.example.com (IP List)	All Services	All Workloads	Containerized environments depend on various external resources to perform basic operations such as pulling a docker image. Illumio has determined that the listed FQDNs are essential to Kubernetes deployments. Each deployment varies and may have dependencies on additional resources. If your container infrastructure has requirements for FQDNs not mentioned in this document, then you should include those FQDNs in this policy line.
Illumio PCE (IP List)	8443 TCP	Kubelink	Kubelink sends context about the Kubernetes cluster to the PCE over TCP 8443 port.
All Workloads	53 TCP 53 UDP	Kubernetes Pod Network (IP List)	The Kubernetes cluster provides internal DNS services to the pods (using coreDNS in this example). This policy enables internal DNS resolution for these tasks.
All Workloads (Uses Virtual Services and Workloads)	All Services	All Workloads	Any communication across all managed Kubernetes nodes or managed infrastructure pods which will be permitted by this policy.
Kubernetes Pod Network (IP List)	All Services	All Workloads	Communications across initiated by any workload which pass through service front ends will be allowed by this policy. It also covers other IP addresses on the Kubernetes pod network which are not discovered by the PCE. Critical for infrastructure functions including but not limited to liveness probes and infrastructure service front ends (Kubernetes).

Extra-scope Rules

Providers	Services	Consumers	Notes
All Workloads	6443 TCP 22 TCP	Any 0.0.0.0/0 (IP List)	Optional: Opens up ports which are purposed for remote management. For example, TCP 22 to provide SSH services to Kubernetes admins. TCP 6443 provides Kubernetes admins with dashboard services. The Dashboard may vary across Kubernetes deployments. The ports can be modified to what is used in your environment and consuming IP list can be changed to corporate subnets or jump servers.
Worker	80 TCP 443 TCP	Any 0.0.0.0/0 (IP List)	This policy assumes Ingress Controllers exist on Worker nodes. If the ingress controllers exist on other nodes, then modify the provider to the host where the Ingress controllers reside. This rule opens default front end ports which are used to access containerized applications from external IP addresses.

OpenShift

Create a ruleset for the OpenShift cluster and control plane Pods. The labels assigned to all of the OpenShift nodes and control Pod workloads should fall within the scope.



Add the following lines of policy to the ruleset.

NOTE

The IP lists referenced in this ruleset are commonly used public registries (e.g., docker.io) for container environments. If you have confirmed that your OpenShift environment does not depend on a public registry shown below, then it is recommended that you remove the IP lists from the ruleset.

Intra-scope Rules

Providers	Services	Consumers	Notes
docker.io (IP List) registry.access.redhat.com (IP List) registry.webscaleone.info (IP LIst) access.redhat.com (IP List) subscription.rhsm.redhat.com (IP List)	All Services	All Workloads	Containerized environments depend on various external resources to perform basic operations such as pulling a docker image. Illumio has determined that the listed FQDNs are essential to OpenShift deployments. Each deployment varies and may have dependencies on additional resources. If your container infrastructure has requirements for FQDNs not mentioned in this doc, then you should include those FQDNs in this policy line.
Illumio PCE (IP List)	8443 TCP	Kubelink	Kubelink sends context about the OpenShift cluster to the PCE over TCP 8443 port.
All Workloads	53 TCP 53 UDP	OpenShift Pod Network (IP List)	The OpenShift cluster in this example uses DNSmasq meaning each cluster node listens on port 53 and provides internal DNS services to the pods. This policy enables internal DNS resolution for these tasks.
All Workloads (Uses Virtual Services and Workloads)	All Services	All Workloads	Any communication across all managed OpenShift nodes or managed infrastructure pods which will be permitted by this policy.
OpenShift Pod Network (IP List) OpenShift Service Network (IP List)	All Services	All Workloads	Communications across initiated by any workload which pass through service front ends will be allowed by this policy. It also covers other IP addresses on the OpenShift pod network which are not discovered by the PCE. Critical for infrastructure functions including but not limited to liveness probes and infrastructure service front ends (Kubernetes).

Extra-Scope Rules

Providers	Services	Consumers	Notes
All Workloads	8443 TCP 22 TCP	Any 0.0.0.0/0 (IP List)	Optional: Opens up ports which are purposed for remote management. For example, TCP 22 to provide SSH services to OpenShift admins. TCP 8443 provides OpenShift admins with webconsole services. Webconsole may vary across OpenShift deployments. The ports can be modified to server other remote management services and consuming IP list can be changed to corporate subnets or jump servers.
Infra (Role)	TCP 80 TCP 443	Any 0.0.0.0/0 (IP List)	This policy assumes the router exists only on dedicated Infra nodes. If the router exists on other nodes, then modify the provider to the host where the router resides. This rule opens default front end router ports which are used to access containerized applications from external IP addresses. As you start to open up application pods to the outside world, you will need to add the application's exposed port to this policy's list of services. For example, you spin up a httpd server and expose that server on TCP 8080. The first step to allow access to the httpd server from outside is to add TCP 8080 to this line of policy.

NOTE

The IP lists referenced in the rulesets are commonly used public registries (for example, docker.io) for container environments. If you have confirmed that your Kubernetes or OpenShift environment does not depend on the public registries mentioned above, then it is recommended that you remove the IP lists from the ruleset.

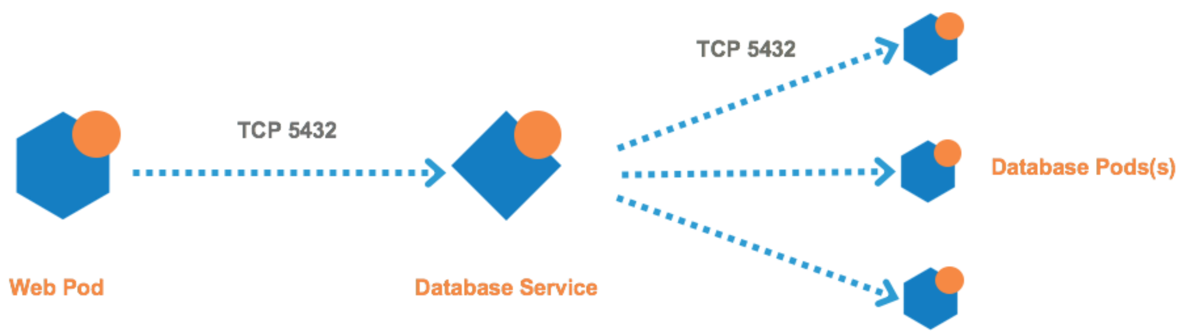
Rules for Containerized Applications

This section covers different scenarios on writing rules for containerized applications.

Access Services from within the Cluster

Access Services from within the Cluster

For connections to a service from within the cluster, the Pods connect to a Service IP and the connections get distributed to the Pods.



Kubernetes

The rules you need to write are:

Example Ruleset

Scope

Application	Environment	Location
Risk Assessment	Development	Cloud

Intra-Scope Rule

Destination	Destination Service	Source	Notes
Database (Virtual Service Role label for database service) + Use Virtual Services Only	Derived from Destination Virtual Service	Web (Role for Web pods)	After the database service gets discovered by the PCE it becomes a virtual service object in the PCE -- not a container workload. The Destination should be the role label of the virtual service plus the "Use Virtual Service Only" option. The Source in this example is the Web pod. Use the Web Role label which describes the pod. Leave the Providing Service empty. Once the rule is saved, it will automatically populate with *Derived from Destination Virtual Service*. **NOTE** This does not allow Web pods to directly access Database pods through the pod IP. This only allows traffic through the service.

OpenShift

The rules you need to write are:

Example Ruleset

Scope

Application	Environment	Location
Risk Assessment	Production	HQ

Intra-Scope Rule

Destination	Destination Service	Source	Notes
Database (Virtual Service Role label for database service) + Use Virtual Services Only	Derived from Destination Virtual Service	Web (Role for Web pods)	After the database service gets discovered by the PCE it becomes a virtual service object in the PCE -- not a container workload. The Destination should be the role label of the virtual service plus the "Use Virtual Service Only" option. The Source in this example is the Web pod. Use the Web Role label which describes the pod. Leave the Providing Service empty. Once the rule is saved, it will automatically populate with *Derived from Destination Virtual Service*. **NOTE** This does not allow Web pods to directly access Database pods through the pod IP. This only allows traffic through the service.

Access Services from Outside the Cluster

Access Services from Outside the Cluster

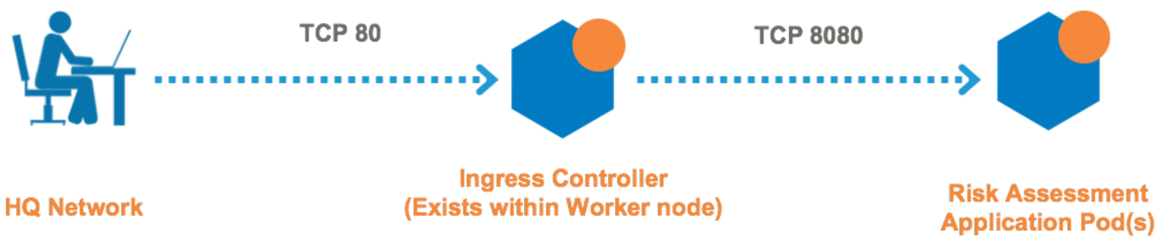
Kubernetes

With Kubernetes, connections to a containerized application from the outside world can be handled in many different ways. In this release, Illumio supports only configurations which expose applications via the Kubernetes NGINX ingress controller (HostNetwork type only). Exposing applications using HostPort are not supported.

In typical Kubernetes deployments, connections to a containerized application from the outside world go through the ingress controllers, then the connection goes directly from controllers to the pods - not the service. Example of scenario and rule coverage are shown below.

Scenario:

- The Kubernetes cluster and containerized applications are in the Development environment
- The containerized application is called RiskAssessment and each Pod within the application listens on TCP 8080
- The RiskAssessment application is exposed to the outside world via the ingress controller. The controller listens on TCP port 80 for the RiskAssessment application
- In Illumio, the RiskAssessment workloads (Pods) provide to the controller on TCP 8080. The controller provides TCP 80 to the outside world.



The rules you need to write are:

Example Ruleset 1

Scope

Application	Environment	Location
Risk Assessment	Development	Cloud

Intra-Scope Rule

Provider	Providing Service	Consumer
All Workloads	All Services	All Workloads

Extra-Scope Rule

Provider	Providing Service	Consumer	Notes
Risk Assessment	TCP 8080	Worker	The consumer should be the role label of the nodes which nest the Ingress controllers.

Example Ruleset 2

The second ruleset opens the ingress controller to the external network. The rule and ruleset below should have been created from the Rules for Kubernetes or OpenShift Cluster [324] section of this guide. You can modify the ruleset as needed.

Scope

Application	Environment	Location	Notes
Kubernetes Infrastructure	Development	Cloud	The scope of the ruleset should match the Kubernetes infrastructure scope.

Intra-Scope Rule

Provider	Providing Service	Consumer	Notes
Worker Node(s)	TCP 80	External Network	This rule should exist from the Rules for Kubernetes or OpenShift Cluster [324] section. The provider should be the Kubernetes node(s) which contain the ingress controller. The consumer can be an IP List such as 0.0.0.0/0 (any), HQ, Corporate, or employee subnet that requires connectivity into the exposed container workloads.

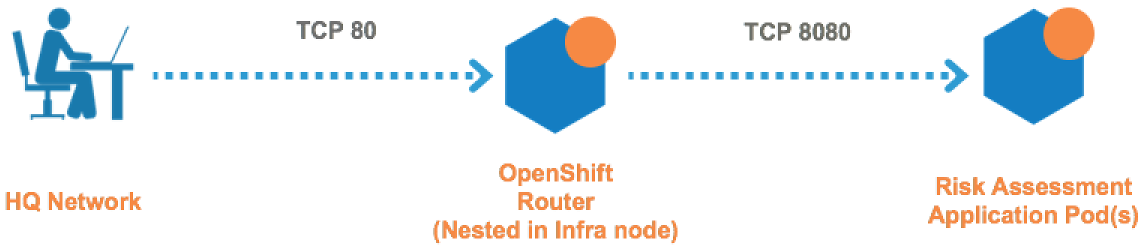
OpenShift

Connections to a containerized application from the outside world go through the OpenShift Router, then the connection goes directly from router to the Pods - not the service. Example of scenario and rule coverage are shown below.

Scenario:

- The OpenShift cluster and containerized applications are in the development environment
- The containerized application is called RiskAssessment and each Pod within the application listens on TCP 8080
- The RiskAssessment application is exposed to the outside world via the router. The router listens on TCP port 80 for the RiskAssessment application

- In Illumio, the RiskAssessment workloads (Pods) provide to the router on TCP 8080. The router provides TCP 80 to the outside world.



The rules you need to write are:

Example Ruleset 1

Scope

Application	Environment	Location
Risk Assessment	Production	HQ

Intra-Scope Rule

Provider	Providing Service	Consumer
All Workloads	All Services	All Workloads

Extra-Scope Rule

Provider	Providing Service	Consumer	Notes
Risk Assess-ment	TCP 8080	IST Infra (Role)	Consumer refers to the Illumio Segmentation Tem-plate. The consumer should be the role label of the node(s) which nest the OpenShift Router.

Example Ruleset 2

The following Ruleset is from the Segmentation Template and you can modify it as needed.

Scope

Application	Environment	Location	Notes
IST OpenShift Infrastructure	IST Production	IST HQ	Ruleset is derived from Illumio Segmentation Template. The scope should match the OpenShift cluster.

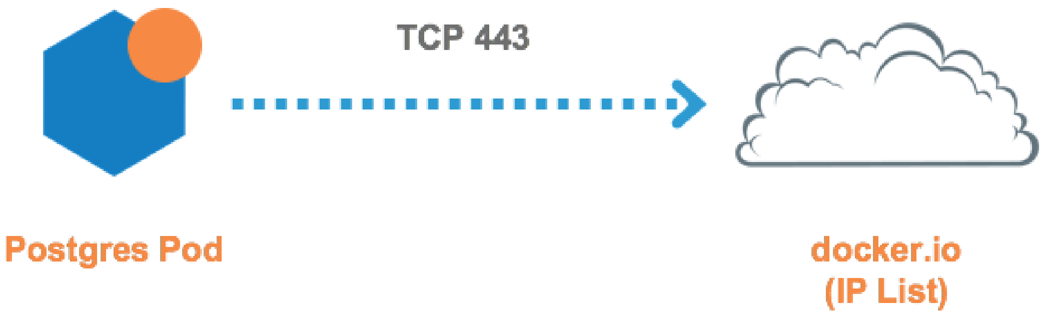
Intra-Scope Rule

Provider	Providing Service	Consumer	Notes
IST Infra (Role)	TCP 80	External Network	This rule is included in Illumio Segmentation Template. The provider should be the OpenShift cluster node(s) which nest the router. The consumer can be an IP list such as 0.0.0.0/0 (any), HQ, Corporate, or employee subnet. The IST default includes 0.0.0.0/0 (any) IP list.

Outbound Connections

Outbound Connections

The outbound connections are required to access repositories.



Kubernetes and OpenShift

The rules you need to write are:

Example Ruleset

Scope

Application	Environment	Location
Risk Assessment	Development	Cloud

Intra-Scope Rule

Provider	Providing Service	Consumer	Notes
docker.io (IP List)	All Services	Database (Role for Postgres Pods)	Once the database service gets discovered by the PCE it becomes a virtual service object in the PCE - not a container workload. The provider should be the role label of the virtual service plus the "Use Virtual Service Only" option. The Consumer in this example is the Web Pod. Use the Web Role label which describes the Pod. Leave the Providing Service empty. Once the rule is saved, it will automatically populate with *Derived from Provider Virtual Service*.

Liveness Probes

Liveness Probes

Containerized applications may require periodic health checks known as liveness probes and readiness probes. Each application includes a health check YAML file that contains liveness and readiness probe configurations. The health checks between the container node and the local container workload may rely on TCP ports. Illumio has included a source object called Container Host for this use case. The Container Host object represents the container node or nodes that host the Pod(s). The example below uses the Container Host object as a source for Liveness and Readiness Probes.

NOTE
The Container Host must always fall under an Extra-Scope rule.

The rules you need to write are:

Kubernetes and OpenShift

The rules you need to write are:

Example Ruleset

Scope

Application	Environment	Location
Risk Assessment	Development	Cloud

Extra-Scope Rule

Destination	Destination Service	Source	Notes
All Workloads	TCP 9090	Container Host (built-in Illumio object)	In this example, the Risk Assessment health check configuration indicates that liveness probe occurs on TCP 9090. Liveness probe ports/protocols may vary across applications. Container Host is an object built into the PCE by default and represents any node that hosts the respective Pod(s).

NodePort Support on Kubernetes and OpenShift

NodePort Support on Kubernetes and OpenShift

Kubernetes (and OpenShift) provide a mechanism to access cluster services from the outside world, of type NodePort. This service exposes a port on all nodes in the cluster on which traffic will be forwarded to any of the backing pods that match the service's selector.

Scenario:

- The Kubernetes cluster and containerized applications are in the Production environment.
- The containerized application is called RiskAssessment, and each Pod within the application listens on TCP 8080.
- The RiskAssessment application is exposed to the outside world via a FrontEnd service with type NodePort.
- The exact NodePort in use is not specified, but is automatically allocated by Kubernetes.
- There may be clients to the FrontEnd service within the cluster or outside the cluster - in both cases, they are labeled as Client.

The rules you need to write are:

Example Ruleset 1: Internal and External Access to Service

Scope

Application	Environment	Location
Risk Assessment	Production	Cloud

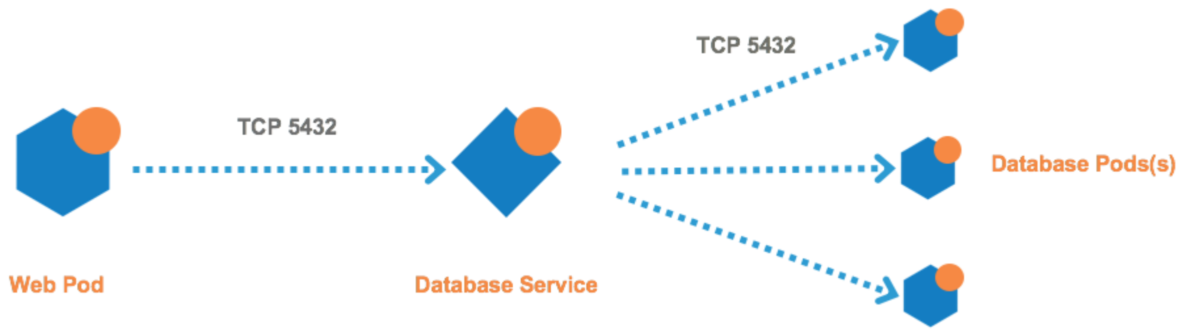
Extra-Scope Rule

Provider	Providing Service	Consumer	Notes
FrontEnd (Virtual Service Role label for Risk Assessment service) + Use Virtual Services Only	Derived from Provider Virtual Service	Client (Role label for Web pods and external workloads)	Once the Risk Assessment service gets discovered by the PCE it becomes a virtual service object in the PCE. The Provider here should be the role label of the virtual service plus the "Use Virtual Service Only" option.

Access Services from within the Cluster

Access Services from within the Cluster

For connections to a service from within the cluster, the Pods connect to a Service IP and the connections get distributed to the Pods.



Kubernetes

The rules you need to write are:

Example Ruleset

Scope

Application	Environment	Location
Risk Assessment	Development	Cloud

Intra-Scope Rule

Destination	Destination Service	Source	Notes
Database (Virtual Service Role label for database service) + Use Virtual Services Only	Derived from Destination Virtual Service	Web (Role for Web pods)	After the database service gets discovered by the PCE it becomes a virtual service object in the PCE -- not a container workload. The Destination should be the role label of the virtual service plus the "Use Virtual Service Only" option. The Source in this example is the Web pod. Use the Web Role label which describes the pod. Leave the Providing Service empty. Once the rule is saved, it will automatically populate with *Derived from Destination Virtual Service*. **NOTE** This does not allow Web pods to directly access Database pods through the pod IP. This only allows traffic through the service.

OpenShift

The rules you need to write are:

Example Ruleset

Scope

Application	Environment	Location
Risk Assessment	Production	HQ

Intra-Scope Rule

Destination	Destination Service	Source	Notes
Database (Virtual Service Role label for database service) + Use Virtual Services Only	Derived from Destination Virtual Service	Web (Role for Web pods)	After the database service gets discovered by the PCE it becomes a virtual service object in the PCE -- not a container workload. The Destination should be the role label of the virtual service plus the "Use Virtual Service Only" option. The Source in this example is the Web pod. Use the Web Role label which describes the pod. Leave the Providing Service empty. Once the rule is saved, it will automatically populate with *Derived from Destination Virtual Service*. **NOTE** This does not allow Web pods to directly access Database pods through the pod IP. This only allows traffic through the service.

Access Services from Outside the Cluster

Access Services from Outside the Cluster

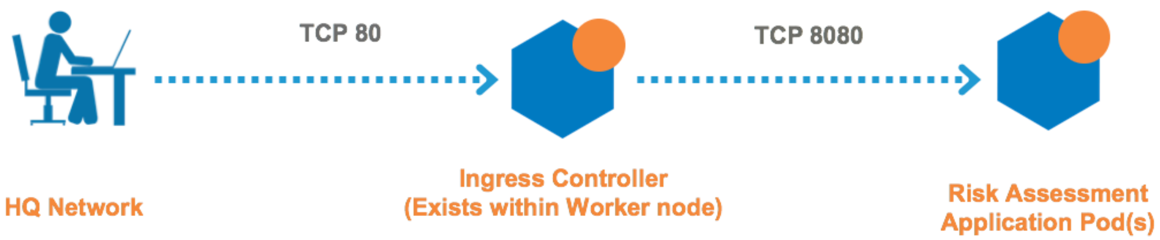
Kubernetes

With Kubernetes, connections to a containerized application from the outside world can be handled in many different ways. In this release, Illumio supports only configurations which expose applications via the Kubernetes NGINX ingress controller (HostNetwork type only). Exposing applications using HostPort are not supported.

In typical Kubernetes deployments, connections to a containerized application from the outside world go through the ingress controllers, then the connection goes directly from controllers to the pods - not the service. Example of scenario and rule coverage are shown below.

Scenario:

- The Kubernetes cluster and containerized applications are in the Development environment
- The containerized application is called RiskAssessment and each Pod within the application listens on TCP 8080
- The RiskAssessment application is exposed to the outside world via the ingress controller. The controller listens on TCP port 80 for the RiskAssessment application
- In Illumio, the RiskAssessment workloads (Pods) provide to the controller on TCP 8080. The controller provides TCP 80 to the outside world.



The rules you need to write are:

Example Ruleset 1

Scope

Application	Environment	Location
Risk Assessment	Development	Cloud

Intra-Scope Rule

Provider	Providing Service	Consumer
All Workloads	All Services	All Workloads

Extra-Scope Rule

Provider	Providing Service	Consumer	Notes
Risk Assessment	TCP 8080	Worker	The consumer should be the role label of the nodes that nest the Ingress controllers.

Example Ruleset 2

The second ruleset opens the ingress controller to the external network. The rule and ruleset below should have been created from the Rules for Kubernetes or OpenShift Cluster [324] section of this guide. You can modify the ruleset as needed.

Scope

Application	Environment	Location	Notes
Kubernetes Infrastructure	Development	Cloud	The scope of the ruleset should match the Kubernetes infrastructure scope.

Intra-Scope Rule

Provider	Providing Service	Consumer	Notes
Worker Node(s)	TCP 80	External Network	This rule should exist from the Rules for Kubernetes or OpenShift Cluster [324] section. The provider should be the Kubernetes node(s) that contain the ingress controller. The consumer can be an IP List such as 0.0.0.0/0 (any), HQ, Corporate, or employee subnet that requires connectivity into the exposed container workloads.

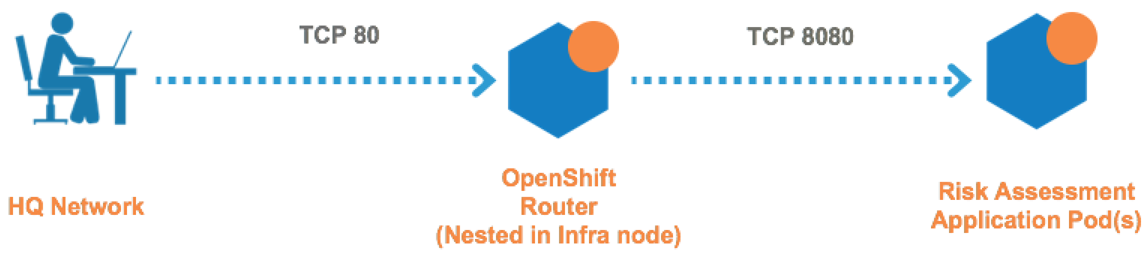
OpenShift

Connections to a containerized application from the outside world go through the OpenShift Router, then the connection goes directly from router to the Pods - not the service. Example of scenario and rule coverage are shown below.

Scenario:

- The OpenShift cluster and containerized applications are in the development environment
- The containerized application is called RiskAssessment and each Pod within the application listens on TCP 8080
- The RiskAssessment application is exposed to the outside world via the router. The router listens on TCP port 80 for the RiskAssessment application

- In Illumio, the RiskAssessment workloads (Pods) provide to the router on TCP 8080. The router provides TCP 80 to the outside world.



The rules you need to write are:

Example Ruleset 1

Scope

Application	Environment	Location
Risk Assessment	Production	HQ

Intra-Scope Rule

Provider	Providing Service	Consumer
All Workloads	All Services	All Workloads

Extra-Scope Rule

Provider	Providing Service	Consumer	Notes
Risk Assessment	TCP 8080	IST Infra (Role)	Consumer refers to the Illumio Segmentation Template. The consumer should be the role label of the node(s) that nest the OpenShift Router.

Example Ruleset 2

The following Ruleset is from the Segmentation Template and you can modify it as needed.

Scope

Application	Environment	Location	Notes
IST OpenShift Infrastructure	IST Production	IST HQ	Ruleset is derived from Illumio Segmentation Template. The scope should match the OpenShift cluster.

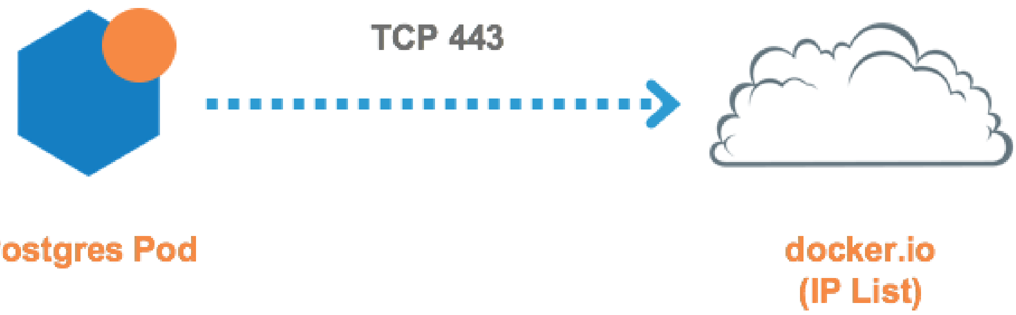
Intra-Scope Rule

Provider	Providing Service	Consumer	Notes
IST Infra (Role)	TCP 80	External Network	This rule is included in Illumio Segmentation Template. The provider should be the OpenShift cluster node(s) which nest the router. The consumer can be an IP list such as 0.0.0.0/0 (any), HQ, Corporate, or employee subnet. The IST default includes 0.0.0.0/0 (any) IP list.

Outbound Connections

Outbound Connections

The outbound connections are required to access repositories.



Kubernetes and OpenShift

The rules you need to write are:

Example Ruleset

Scope

Application	Environment	Location
Risk Assessment	Development	Cloud

Intra-Scope Rule

Source	Service	Destination	Notes
docker.io (IP List)	All Services	Database (Role for Postgres Pods)	Once the database service gets discovered by the PCE it becomes a virtual service object in the PCE - not a container workload. The Source should be the role label of the virtual service plus the "Use Virtual Service Only" option. The Destinationin this example is the Web Pod. Use the Web Role label which describes the Pod. Leave the Providing Service empty. Once the rule is saved, it will automatically populate with *Derived from Source Virtual Service*.

Liveness Probes

Liveness Probes

Containerized applications may require periodic health checks known as liveness probes and readiness probes. Each application includes a health check YAML file that contains liveness and readiness probe configurations. The health checks between the container node and the local container workload may rely on TCP ports. Illumio has included a source object called Container Host for this use case. The Container Host object represents the container node or nodes that host the Pod(s). The example below uses the Container Host object as a source for Liveness and Readiness Probes.

> **NOTE**
> The Container Host must always fall under an Extra-Scope rule.

The rules you need to write are:

Kubernetes and OpenShift

The rules you need to write are:

Example Ruleset

Scope

Application	Environment	Location
Risk Assessment	Development	Cloud

Extra-Scope Rule

Destination	Destination Service	Source	Notes
All Workloads	TCP 9090	Container Host (built-in Illumio object)	In this example, the Risk Assessment health check configuration indicates that liveness probe occurs on TCP 9090. Liveness probe ports/protocols may vary across applications. Container Host is an object built into the PCE by default and represents any node that hosts the respective Pod(s).

NodePort Support on Kubernetes and OpenShift

NodePort Support on Kubernetes and OpenShift

Kubernetes (and OpenShift) provide a mechanism to access cluster services from the outside world, of type NodePort. This service exposes a port on all nodes in the cluster on which traffic will be forwarded to any of the backing pods that match the service's selector.

Scenario:

- The Kubernetes cluster and containerized applications are in the Production environment.
- The containerized application is called RiskAssessment, and each Pod within the application listens on TCP 8080.
- The RiskAssessment application is exposed to the outside world via a FrontEnd service with type NodePort.
- The exact NodePort in use is not specified, but is automatically allocated by Kubernetes.
- There may be clients to the FrontEnd service within the cluster or outside the cluster - in both cases, they are labeled as Client.

The rules you need to write are:

Example Ruleset 1: Internal and External Access to Service

Scope

Application	Environment	Location
Risk Assessment	Production	Cloud

Extra-Scope Rule

Source	Service	Destination	Notes
FrontEnd (Virtual Service Role label for Risk Assessment service) + Use Virtual Services Only	Derived from Source Virtual Service	Client (Role label for Web pods and external workloads)	Once the Risk Assessment service gets discovered by the PCE it becomes a virtual service object in the PCE. The Source here should be the role label of the virtual service plus the "Use Virtual Service Only" option.

Rules for Persistent Storage

This section only applies to deployments which require communication with external storage nodes over NFS, iSCSI, and others for persistent storage. If the cluster or Pods have external storage dependencies, then you need a policy to allow outbound communications to the storage node. The storage node can be represented as an unmanaged workload or IP list.

The following is an example of outbound policy to a NFS node, which is represented by an IP list.

Kubernetes

The following is an example of an outbound policy to an NFS node, which is represented by an IP list:

Example Ruleset 1

Scope

Application	Environment	Location	Notes
Kubernetes Infrastructure	Development	Cloud	Kubernetes cluster

Intra-Scope Rule

Provider	Providing Service	Consumer	Notes
NFS Storage (IP List)	TCP 2049	All Workloads	All Kubernetes nodes and infrastructure Pods can communicate outbound to NFS over the NFS TCP port.

Example Ruleset 2

Scope

Application	Environment	Location	Notes
ERP	Development	Cloud	From httpd example

Intra-Scope Rule

Provider	Providing Service	Consumer	Notes
NFS Storage (IP List)	TCP 2049	All Workloads	All Pods can talk outbound to NFS over the NFS TCP port.

OpenShift

The following is an example of an outbound policy to an NFS node, which is represented by an IP list:

Example Ruleset 1

Scope

Application	Environment	Location	Notes
OpenShift Infrastructure	Development	Cloud	OpenShift cluster

Intra-Scope Rule

Provider	Providing Service	Consumer	Notes
NFS Storage (IP List)	TCP 2049	All Workloads	All OpenShift nodes and infrastructure Pods can communicate outbound to NFS over the NFS TCP port.

Example Ruleset 2

Scope

Application	Environment	Location	Notes
ERP	Development	Cloud	From httpd example

Intra-Scope Rule

Provider	Providing Service	Consumer	Notes
NFS Storage (IP List)	TCP 2049	All Workloads	All Pods can talk outbound to NFS over the NFS TCP port.

Local Policy Convergence Controller

The local policy convergence controller provides a deterministic way of setting the readiness state of pods in your cluster after local policy has converged. By controlling the readiness state of pods, you can prevent them from receiving and sending traffic through Kubernetes until they are ready. Using a controller ensures that the network and security infrastructure is ready for a multi-microservice application.

In this release, the Kubernetes Custom Pod Conditions feature introduced in v1.14 is available for containerized VENs.

About the Controller Behavior

By default, the readiness gate is not specified on a pod spec and the C-VEN does not affect the readiness state of the pod regardless of annotations or Illumio managed state.

When the Illumio readiness gate is specified on a pod spec, the PCE completes the following actions when a new pod is created:

1. Sends the C-VEN policy for the new pod P.
2. When pod P is managed, the C-VEN applies local policy for the new pod P.
3. The C-VEN waits for a timer to expire to allow peers to apply policy on their end (such as, updating the new pod P IP address).
   By default, the timer uses the following values:
   - If the pod is managed by Illumio, the timer is set to 15 seconds.
   - If the pod is not managed by Illumio, the timer is set to 0 seconds.

   > **TIP**
   > To configure a custom value for the timer duration, see Timer Customization [347].
4. The C-VEN sets the readiness gate pod condition to "True."
   The pod is now considered "Ready" by Kubernetes.

Configure the Illumio Readiness Gate

To use a local policy convergence controller, specify the Illumio readiness gate under `readinessGates.conditionType` in the pod spec YAML.

See the following example pod spec YAML file:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deploy
spec:
  selector:
    matchLabels:
      app: my-pod
  replicas: 2
  template:
    metadata:
      labels:
        app: my-pod
    spec:
      readinessGates:
```

```
# <----- declare readiness gates
      - conditionType: "com.illumio.policy-ready"
 # <----- Illumio policy convergence readiness gate
      containers:
      - name: my-pod-web
        image: nginx
        ports:
        - containerPort: 80
```

Timer Customization

You can customize the timer cluster-wide or pre-pod.

> **NOTE**
> When configuring a custom timer by using the DaemonSet environment variable or an annotation, you are limited to specifying 0-300 seconds.

Cluster Wide Timer Customization

To customize the timer duration on a cluster-wide basis, set the readiness gate timer variable in the C-VEN DaemonSet YAML.

See the following YAML file:

```
...
containers:
  - name: illumio-ven
    env:
      - name: ILO_SERVER
        valueFrom:
          secretKeyRef:
            name: illumio-ven-config
            key: ilo_server
      - name: ILO_CODE
        valueFrom:
          secretKeyRef:
            name: illumio-ven-config
            key: ilo_code
      - name: ILO_K8S_NODE_NAME
        valueFrom:
          fieldRef:
            fieldPath: spec.nodeName
      - name: ILO_K8S_READINESS_TIMER
 # <--- custom readiness gate timer across the cluster
        value: "20"
 # <--- timer value
...
```

Pre-pod Timer Customization

To customize the timer duration for specific pods, set the Illumio readiness gate timer annotation on the pod spec.

See the following example deployment:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deploy
spec:
  selector:
    matchLabels:
      app: my-pod
  replicas: 2
  template:
    metadata:
      labels:
        app: my-pod
      annotations:
        com.illumio.readiness-gate-timer: "20"
# <----- custom readiness gate timer for all pods in this deployment
    spec:
      readinessGates:
      - conditionType: "com.illumio.policy-ready"
      containers:
      - name: my-pod-web
        image: nginx
        ports:
        - containerPort: 80
```

Track the State of the Readiness Gate

You can track the state of the readiness gate by running either of the following commands:

- `kubectl get pod -o wide`
- `kubectl get ep -o wide`

Example: State of the Readiness Gate

This example shows a cluster with Kubelink and the C-VEN deployed and running. When you initially deploy or scaled up the Illumio Readiness Gate, you see the following values:

NOTE

The state of gate readiness appears in the "READINESS GATES" column.

```
$ kubectl get pod,ep -o wide
NAME                             READY    STATUS    RESTARTS   AGE
IP            NODE       NOMINATED NODE   READINESS GATES
pod/my-deploy-855dfbf94f-gwz7c   1/1      Running   1          4d20h
172.17.0.7    ubuntu20   <none>           0/1
pod/my-deploy-855dfbf94f-p7czp   1/1      Running   1          4d20h
172.17.0.6    ubuntu20   <none>           0/1

NAME                       ENDPOINTS                         AGE
```

```
endpoints/kubernetes   10.0.2.15:8443                      19d
endpoints/my-service                                       4d22h
```

In this example, the readiness gates are marked as 0/1 for both pods and my-service does not have any available endpoints. After the VEN has processed the policy for the new pods and the timer expires, it sets the readiness gate to “True” for each pod and you see the following output:

```
$ kubectl get pod,ep -o wide
NAME                                 READY   STATUS    RESTARTS   AGE
IP            NODE       NOMINATED NODE   READINESS GATES
pod/my-deploy-855dfbf94f-gwz7c   1/1     Running   1          4d20h
172.17.0.7   ubuntu20   <none>           1/1
pod/my-deploy-855dfbf94f-p7czp   1/1     Running   1          4d20h
172.17.0.6   ubuntu20   <none>           1/1

NAME                   ENDPOINTS                           AGE
endpoints/kubernetes   10.0.2.15:8443                      19d
endpoints/my-service   172.17.0.6:9376,172.17.0.7:9376   4d22h
```

To view greater detail about the pod conditions, run the command `kubectl get pod <pod name> -o yaml`:

```
$ kubectl get pod my-deploy-855dfbf94f-gwz7c -o yaml
...
status:
  conditions:
  - lastProbeTime: null                         // <--
    lastTransitionTime: "2021-05-18T20:26:26Z"  // <--
    message: Pod Policy Ready                     // <-- this pod condition
is set by VEN
    reason: PolicyReady                         // <--
    status: "True"                              // <--
    type: illumio.com/policy-ready              // <--
  - lastProbeTime: null
    lastTransitionTime: "2021-05-18T20:25:51Z"
    status: "True"
    type: Initialized
  - lastProbeTime: null
    lastTransitionTime: "2021-05-19T19:56:24Z"
    status: "True"
    type: Ready
// <-- this is only set to True after all readiness gates are set to True
  - lastProbeTime: null
    lastTransitionTime: "2021-05-19T19:56:24Z"
    status: "True"
    type: ContainersReady
  - lastProbeTime: null
    lastTransitionTime: "2021-05-18T20:25:51Z"
    status: "True"
    type: PodScheduled
...
```

Upgrade and Uninstallation

Follow the steps and sequence described in this section to upgrade or uninstall Illumio Core for Kubernetes components. This section also describes the procedure for migrating from a deployment of C-VEN version 21.5.15 or earlier (which did not use Helm Charts) to a current Helm Chart deployment.

> **IMPORTANT**
> Use the proper upgrade and uninstallation procedures according to the method that was first used to deploy the product. For deployments made with a Helm Chart (typically with Illumio Core for Kubernetes 3.0.0 or later), follow the steps in Upgrade and Uninstall Helm Chart Deployments [352]. For deployments made without using a Helm Chart (for installations of C-VEN 21.5.15 or earlier), follow the steps in Upgrade and Uninstall Non-Helm Chart Deployments. [353]

Migrate from Previous C-VEN Versions (21.5.15 or Earlier)

This section describes the steps to migrate a manually-deployed Illumio installation to a Helm-managed deployment. Manually-deployed (or, non-Helm deployments) were used to configure and deploy C-VEN versions 21.5.15 and earlier, and Kubelink versions earlier than 3.0.

To upgrade an existing Helm installation to a newer version, follow standard Helm practice with **`helm upgrade`** command.

Follow these general steps to migrate from a manually-deployed Illumio Core for Kubernetes to a Helm Chart deployment:

1. Annotate and label resources.
2. Delete C-VEN DaemonSet.
3. Install Helm and the Helm Chart.

Annotate and Label Resources

From Helm version 3.0.0 on, Helm supports adopting already-deployed resources with the correct name, annotations, and labels.

Required annotations and labels are:

```
annotations:
  meta.helm.sh/release-name: illumio
  meta.helm.sh/release-namespace: illumio-system
labels:
  app.kubernetes.io/managed-by: Helm
```

To annotate and label all Illumio resources, use the commands below (provided the names of resources match your deployment). Note the **`--overwrite`** flag which replaces any existing ownership annotations that might be already assigned.

```
kubectl -n illumio-system annotate secret illumio-ven-config meta.helm.sh/
release-name=illumio
 --overwrite
kubectl -n illumio-system annotate secret illumio-ven-config meta.helm.sh/
release-namespace=illumio-system --overwrite
kubectl -n illumio-system label secret illumio-ven-config app.kubernetes.io/
managed-by=Helm --overwrite
kubectl -n illumio-system annotate secret illumio-kubelink-config
meta.helm.sh/
release-name=illumio --overwrite
kubectl -n illumio-system annotate secret illumio-kubelink-config
meta.helm.sh/
release-namespace=illumio-system --overwrite
kubectl -n illumio-system label secret illumio-kubelink-config
app.kubernetes.io/
managed-by=Helm --overwrite
kubectl -n illumio-system annotate serviceaccount illumio-ven meta.helm.sh/
release-name=illumio --overwrite
kubectl -n illumio-system annotate serviceaccount illumio-ven meta.helm.sh/
release-namespace=illumio-system --overwrite
kubectl -n illumio-system label serviceaccount illumio-ven
app.kubernetes.io/
managed-by=Helm --overwrite
kubectl -n illumio-system annotate clusterrole illumio-kubelink
meta.helm.sh/
release-name=illumio --overwrite
kubectl -n illumio-system annotate clusterrole illumio-kubelink
meta.helm.sh/
release-namespace=illumio-system --overwrite
kubectl -n illumio-system label clusterrole illumio-kubelink
app.kubernetes.io/
managed-by=Helm --overwrite
kubectl -n illumio-system annotate clusterrolebinding illumio-ven
meta.helm.sh/
release-name=illumio --overwrite
kubectl -n illumio-system annotate clusterrolebinding illumio-ven
meta.helm.sh/
release-namespace=illumio-system --overwrite
kubectl -n illumio-system label clusterrolebinding illumio-ven
app.kubernetes.io/
managed-by=Helm --overwrite
kubectl -n illumio-system annotate clusterrole illumio-ven meta.helm.sh/
release-name=illumio --overwrite
kubectl -n illumio-system annotate clusterrole illumio-ven meta.helm.sh/
release-namespace=illumio-system --overwrite
kubectl -n illumio-system label clusterrole illumio-ven app.kubernetes.io/
managed-by=Helm --overwrite
kubectl -n illumio-system annotate serviceaccount illumio-kubelink
meta.helm.sh/
release-name=illumio --overwrite
kubectl -n illumio-system annotate serviceaccount illumio-kubelink
meta.helm.sh/
release-namespace=illumio-system --overwrite
kubectl -n illumio-system label serviceaccount illumio-kubelink
app.kubernetes.io/
```

```
managed-by=Helm --overwrite
kubectl -n illumio-system annotate deployment illumio-kubelink meta.helm.sh/
release-name=illumio --overwrite
kubectl -n illumio-system annotate deployment illumio-kubelink meta.helm.sh/
release-namespace=illumio-system --overwrite
kubectl -n illumio-system label deployment illumio-kubelink
app.kubernetes.io/
managed-by=Helm --overwrite
kubectl -n illumio-system annotate clusterrolebinding illumio-kubelink
meta.helm.sh/
release-name=illumio --overwrite
kubectl -n illumio-system annotate clusterrolebinding illumio-kubelink
meta.helm.sh/
release-namespace=illumio-system --overwrite
kubectl -n illumio-system label clusterrolebinding illumio-kubelink
app.kubernetes.io/
managed-by=Helm --overwrite
```

The output should look similar to this:

```
...
clusterrolebinding.rbac.authorization.k8s.io/illumio-kubelink annotated
clusterrolebinding.rbac.authorization.k8s.io/illumio-kubelink annotated
clusterrolebinding.rbac.authorization.k8s.io/illumio-kubelink labeled
```

Delete C-VEN DaemonSet

The next step is removing the C-VEN DaemonSet. Save any custom labels and validations included in the DaemonSet and reapply them later.

```
kubectl delete daemonset illumio-ven -n illumio-system
```

Install Helm

The last remaining step is installing Helm and the Helm Chart for Illumio Core for Kubernetes. Follow the steps in Deploy with Helm Chart [292]. Filling in the fields in **illumio-values.yaml** is still mandatory.

Upgrade and Uninstall Helm Chart Deployments

Deployments of Illumio Core for Kubernetes 3.0.0 or later are performed with Helm Charts.

Upgrade Helm Chart Deployments

To upgrade an existing installation to a newer version after it had been initially deployed with a Helm Chart, follow standard Helm practice with the **helm upgrade** command.

Uninstall Helm Chart Deployments

To completely uninstall an existing installation that had been initially deployed with a Helm Chart:

```
$ helm uninstall illumio --namespace illumio-system

$ kubectl delete namespace illumio-system
```

The uninstallation process also unpairs the C-VENs from the PCE.

Uninstalling the Helm Chart release takes around two minutes to complete.

Upgrade and Uninstall Non-Helm Chart Deployments

This section describes how deployments that were not installed with Helm can be upgraded or uninstalled.

Upgrade Illumio Components

Illumio Core for Kubernetes and OpenShift is a flexible and modular solution that can be upgraded piece by piece.

For minor upgrades, Kubelink can be upgraded independently from the C-VEN and vice versa unless explicitly mentioned in the release notes.

For major upgrades, including PCE, Kubelink, and C-VEN, Illumio recommends the following process:

- Upgrade the PCE to the new desired version.
- Review the compatibility matrix between PCE, Kubelink, and C-VEN on the Illumio support website.
- Upgrade Kubelink.
- Upgrade C-VEN.

Upgrade Kubelink

The supported process to upgrade Kubelink is as follows:

1. Upload the new image to your private container registry.
2. Change the manifest file to point to the latest Kubelink image in the registry. You do not need to change the previously created secret for Kubelink.
3. Apply this new manifest file to the cluster. `illumio-kubelink` follows the default update behavior of Kubernetes. For more information, see Kubernetes Documentation.

You can verify that the upgrade was successful in the PCE UI on the **Container Clusters > Summary** page and checking for the new Kubelink version.

Upgrade C-VEN

The supported process to upgrade C-VENs is as follows:

1. Upload the new image to your private container registry.
2. Change the manifest file to point to the latest C-VEN image in the registry. You do not need to change the previously created secret for C-VEN.
3. Apply this new manifest file to the cluster. `illumio-ven daemonset` follows the default rolling update behavior of Kubernetes. For more information, see Kubernetes Documentation.

You can verify that the upgrade was successful in the PCE UI on the **Container Clusters > Workloads** page and clicking on any workload and checking for the new C-VEN version.

Uninstall Illumio from Your Cluster

To uninstall the Illumio components, you need to contact Illumio Professional Services to unpair the C-VENs and then delete the Illumio resources from your cluster.

Unpair C-VENs

> **IMPORTANT**
>
> Contact Illumio Professional Services to unpair the C-VENs in your Kubernetes or OpenShift clusters.

Deleting C-VENs or DaemonSet will not properly unpair them from the PCE and can cause the following issues:

- Workloads will go offline in the PCE UI after 5 minutes (defined by the default Offline Timers configured in the PCE).
- Workloads will be left in the PCE UI as offline with the button to unpair them grayed out (this action is not supported by Illumio).
- Firewall rules configured on the Host and Pods namespaces will remain untouched and active.

The current way to properly delete these workloads created in the PCE UI by C-VENs is by deleting the entire cluster in the PCE UI.

> **IMPORTANT**
>
> Unpairing an individual C-VEN is not supported. It has to be done at the cluster level (through the DaemonSet), because the cluster is considered as a single entity from a security point of view.

If a node unjoins the cluster for any reason or due to the `kubectl delete node <node_name>` command, the PCE automatically unpairs the C-VEN and deletes the workload and Container workloads associated with the C-VEN that was running on the deleted node.

Delete Illumio Resources

To delete the existing Illumio resources created in your Kubernetes or OpenShift cluster, follow these steps:

Delete C-VEN Resources

1. Contact Illumio Professional Services to unpair the C-VENs and clean up existing iptables rules created by Illumio.
2. Check the Workloads and Container Workloads tabs under **Infrastructure > Container Clusters > YourClusterName** and validate that your nodes and Pods are no longer visible.

3. Delete the resources created during the C-VEN installation by using the following command:

```
kubectl delete -f illumio-ven-kubernetes.yml
kubectl delete -f illumio-ven-secret.yml
```

```
oc delete -f illumio-ven-openshift.yml
oc delete -f illumio-ven-secret.yml
```

Delete Kubelink Resources

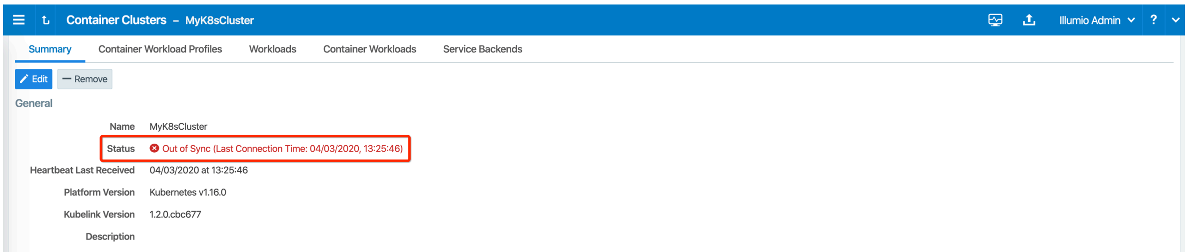
1. Delete the resources created during the Kubelink installation.
2. Delete Kubelink resources from Kubernetes:

```
kubectl delete -f illumio-kubelink-kubernetes.yml
kubectl delete -f illumio-kubelink-secret.yml
```

3. Delete Kubelink resources from OpenShift:

```
oc delete -f illumio-kubelink-openshift.yml
oc delete -f illumio-kubelink-secret.yml
```

4. Check the Summary tab under **Infrastructure > Container Clusters > YourClusterName** and validate that your cluster is "Out of Sync". It takes approximately 10 minutes for the cluster Status to change from "In Sync" to "Out-of-Sync".



5. Finally, delete the container cluster from the PCE UI and verify that there are no resources left in your cluster such as, ConfigMap, Secrets, and others.

Delete Illumio Namespace

- To delete the Illumio namespace in Kubernetes, use the following command:

```
kubectl delete ns illumio-system
```

- To delete the Illumio namespace in OpenShift, use the following command:

```
oc delete project illumio-system
```

Reference: General

This section lists a few known limitation of this release and how to troubleshoot issues that may occur during the installation process.

For more information see these additional topics.

- Troubleshooting
- Troubleshooting CLAS Mode Architecture
- Known Limitations
- Kubelink Monitoring and Troubleshooting

- Aggregating Logs from Kubelink and C-VEN Pods

Troubleshooting

This section describes how to troubleshoot common issues when installing Illumio on Kubernetes or OpenShift deployments.

Failed Authentication with the Container Registry

In some cases, your Pods are in `ImagePullBackOff` state after the deployment:

```
$ kubectl -n kube-system get Pods
NAME                                 READY   STATUS             RESTARTS   AGE
coredns-58687784f9-h4pp2             1/1     Running            8
175d
coredns-58687784f9-znn9j             1/1     Running            9
175d
dns-autoscaler-79599df498-m55mg      1/1     Running            9
175d
illumio-kubelink-87fd8d9f6-nmh25     0/1     ImagePullBackOff   0          28s
```

In this case, check the description of your Pods using the following command:

```
$ kubectl -n kube-system describe Pods illumio-kubelink-87fd8d9f6-nmh25
Name:          illumio-kubelink-87fd8d9f6-nmh25
Namespace:     kube-system
Priority:      0
Node:          node2/10.0.0.12
Start Time:    Fri, 03 Apr 2020 21:05:07 +0000
Labels:        app=illumio-kubelink
               Pod-template-hash=87fd8d9f6
Annotations:   com.illumio.role: Kubelink
Status:        Pending
IP:            10.10.65.55
IPs:
  IP:            10.10.65.55
Controlled By:  ReplicaSet/illumio-kubelink-87fd8d9f6
Containers:
  illumio-kubelink:
    Container ID:
    Image:          registry.poc.segmentationpov.com/illumio-
kubelink:2.0.x.xxxxxx
    Image ID:
    Port:           <none>
    Host Port:      <none>
    State:          Waiting
      Reason:       ImagePullBackOff
    Ready:          False
    Restart Count:  0
    Environment:
      ILO_SERVER:         <set to the key 'ilo_server' in secret
     'illumio-kubelink-config'>
      Optional: false
```

```
      ILO_CLUSTER_UUID:   <set to the key 'ilo_cluster_uuid' in secret
     'illumio-kubelink-config'>
      Optional: false
      ILO_CLUSTER_TOKEN:  <set to the key 'ilo_cluster_token' in secret
     'illumio-kubelink-config'>
      Optional: false
      CLUSTER_TYPE:       Kubernetes
      IGNORE_CERT:        <set to the key 'ignore_cert' in secret
     'illumio-kubelink-config'>
      Optional: true
      DEBUG_LEVEL:        <set to the key 'log_level' in secret
     'illumio-kubelink-config'>
      Optional: true
    Mounts:
      /etc/pki/tls/ilo_certs/ from root-ca (rw)
      /var/run/secrets/kubernetes.io/serviceaccount from
      illumio-kubelink-token-7mvgk (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             False
  ContainersReady   False
  PodScheduled      True
Volumes:
  root-ca:
    Type:      ConfigMap (a volume populated by a ConfigMap)
    Name:      root-ca-config
    Optional:  false
  illumio-kubelink-token-7mvgk:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  illumio-kubelink-token-7mvgk
    Optional:    false
QoS Class:       BestEffort
Node-Selectors:  <none>
Tolerations:     node-role.kubernetes.io/master:NoSchedule
                 node.kubernetes.io/not-ready:NoExecute for 300s
                 node.kubernetes.io/unreachable:NoExecute for 300s
Events:
  Type     Reason          Age                 From               Message
  ----     ------          ----                ----               -------
  Normal   Scheduled       <unknown>           default-scheduler
Successfully assigned kube-system/illumio-kubelink-87fd8d9f6-nmh25 to node2
  Normal   SandboxChanged  45s                 kubelet, node2     Pod
sandbox changed, it will be killed and re-created.
  Normal   BackOff         14s (x4 over 45s)  kubelet,
node2     Back-off pulling image "registry.poc.segmentationpov.com/illumio-
kubelink:2.0.x.xxxxxx"
  Warning  Failed          14s (x4 over 45s)  kubelet, node2     Error:
ImagePullBackOff
  Normal   Pulling         1s (x3 over 46s)   kubelet, node2     Pulling
image "registry.poc.segmentationpov.com/illumio-kubelink:2.0.x.xxxxxx"
  Warning  Failed          1s (x3 over 46s)   kubelet,
node2     Failed to pull image "registry.poc.segmentationpov.com/illumio-
kubelink:2.0.x.xxxxxx": rpc error: code = Unknown desc = Error response
from daemon: unauthorized: authentication required
```

```
  Warning  Failed          1s (x3 over 46s)   kubelet, node2     Error:
ErrImagePull
```

The messages at the end of the output above are self-explanatory that there is a problem with the authentication against the container registry. Verify the credentials you entered in the secret for your private container registry and reapply it after fixing the issue.

Kubelink Pod in CrashLoopBackOff State

In some cases, your Kubelink Pod is in `CrashLoobBackOff` state after the deployment:

```
$ kubectl -n kube-system get Pods
NAME                                READY   STATUS             RESTARTS
AGE
coredns-58687784f9-h4pp2            1/1     Running            8
174d
coredns-58687784f9-znn9j            1/1     Running            9
174d
dns-autoscaler-79599df498-m55mg     1/1     Running            9
174d
illumio-kubelink-8648c6fb68-mdh8p   0/1     CrashLoopBackOff   1
16s
```

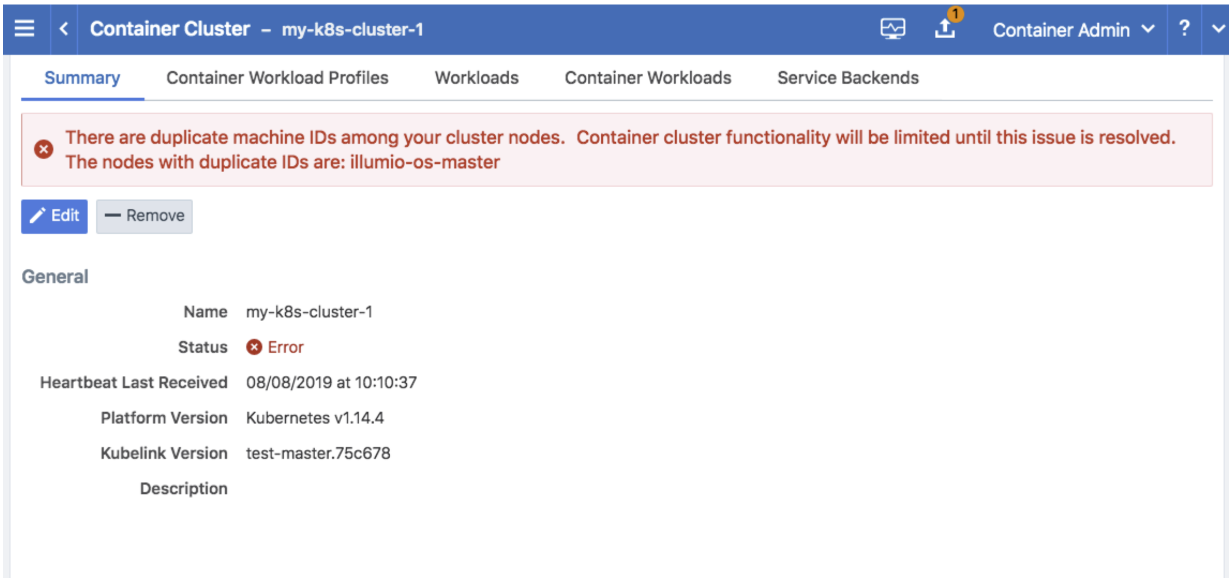
In this case, check the logs of your Pods using the following command:

```
$ kubectl -n kube-system logs illumio-kubelink-8648c6fb68-mdh8p
I, [2020-04-03T01:46:33.587761 #19]  INFO -- : Starting Kubelink for PCE
https://mypce.example.com:8443
I, [2020-04-03T01:46:33.587915 #19]  INFO -- : Found 1 custom certs
I, [2020-04-03T01:46:33.594212 #19]  INFO -- : Installed custom certs to
/etc/pki/tls/certs/ca-bundle.crt
I, [2020-04-03T01:46:33.619976 #19]  INFO -- : Connecting to PCE
https://mypce.example.com:8443
E, [2020-04-03T01:46:33.651410 #19] ERROR -- : Received a non retriable
error 401
/illumio/kubelink.rb:163:in `update_pce_resource': HTTP status code 401 uri:
 https://mypce.example.com:8443/api/v2/orgs/10/container_clusters/
42083a4d-dd92-49e6-b495-6f84a940073c/put_from_cluster, request_id:
21bdfc05-7b02-442d-a778-e6f2da2a462b response:  request_body:
{"kubelink_version":"2.0.x.xxxxxx","errors":[],"manager_type":"Kubernetes
v1.16.0"} (Illumio::PCEHttpException)
    from /illumio/kubelink.rb:113:in `initialize'
    from /illumio/main.rb:39:in `new'
    from /illumio/main.rb:39:in `block in main'
    from /external/lib/ruby/gems/2.4.0/gems/em-synchrony-1.0.6/lib/em-
synchrony.rb:39:in
 `block (2 levels) in synchrony'
```

In the example above, the request is rejected by the PCE because of a wrong identifier. Open your secret file for Kubelink, verify your cluster UUID and token, and make sure you copy-pasted the same string provided by the PCE during cluster creation.

Container Cluster in Error

In some cases, the container cluster page displays an error indicating that duplicate machine IDs were detected and functionality will be limited. See the screenshot below.



To resolve this error, follow the steps in the section below. After following those steps, restart the C-VEN Pod on each of the affected Kubernetes cluster node.

Verify Machine IDs on All Nodes

To verify machine-ids and resolve any duplicate IDs across nodes:

1. Check the machineID of all your cluster nodes with the following command:

   ```
   kubectl get node -o yaml | grep machineID
   ```

   ```
   $ kubectl get node -o yaml | grep machineID
         machineID: ec2eefcfc1bdfa9d38218812405a27d9
         machineID: ec2bcf3d167630bc587132ee83c9a7ad
         machineID: ec2bf11109b243671147b53abe1fcfc0
   ```
2. As an alternative, you can also to check content of the `/etc/machine-id` file on all cluster nodes. The output should be a single newline-terminated, hexadecimal, 32-character, and lowercase ID.
3. If the machine-id string is unique for each node, then the environment is OK. If the machine-id is duplicated across any of the nodes, then you must generate a machine-id for each node which has the same machine-id.
4. Running the following command displays the output of the machine-id:

   ```
   cat /etc/machine-id
   ```

   ```
   root@k8s-c2-node1:~# cat /etc/machine-id
   2581d13362cd4220b20020ff728efff8
   ```

Generate a New Machine ID

If the machineID is duplicated on some or all of the Kubernetes nodes, use the following steps to generate a new machine-id.

- For CentOS or Red Hat:

  ```
  rm -rf /etc/machine-id; systemd-machine-id-setup; systemctl restart
  kubelet
  ```

- For Ubuntu:

```
rm -rf /etc/machine-id; rm /var/lib/dbus/machine-id; systemd-machine-id-
setup;
systemctl restart kubelet
```

> **NOTE**
> Check the machine-id again after doing the above steps to verify that each Kubernetes cluster node has a unique machine-id.

Pods and Services Not Detected

In some cases, the Container Workloads page under **Infrastructure > Container Clusters > MyClusterName** is empty although the Workloads page has all the cluster nodes in it. This issue typically occurs when the wrong container runtime is monitored by Illumio. To resolve this issue:

1. Validate which container runtime is used in your Kubernetes or OpenShift cluster.
2. Open your configuration file for the C-VEN DaemonSet.
3. Modify the `unixsocks` mount configuration to point to the right socket path on your hosts.

> **NOTE**
> This issue typically occurs when containerd or cri-o is the primary container runtime on Kubernetes or OpenShift nodes and there is an existing Docker container runtime on the nodes that is not "active" (the socket still present on the nodes and process still running, mostly some leftover from the staging phase of the servers).

Pods Stuck in Terminating State

In a Kubernetes cluster running containerd 1.2.6-10 as the container runtime, on deleting a Pod while the C-VEN is deployed may result in the Pod being stuck in a terminating state. If you see this error, redeploy the C-VEN and modify the socket path as follows:

Change the `volumeMount` and `hostPath` from `/var/run` to `/var/run/containerd` in the illumio-ven.yaml file

Known Limitations

The known limitations in this release are:

- Kube-proxy mode set to IPVS is currently not supported.

- If a C-VEN on a server hosting containers is paired directly into the Enforced policy state, other nodes may lose connectivity with the master node until policy is synchronized across all the nodes.
- Pods which run on the host network stack (inherit the host IP address) are not reported to the PCE. Any rules written for the host will also be inherited by any *hostNetworked Pods* on the host.
- If you are using an external load balancer, the policy configuration will be dependent on the type of the load balancer used.
- Kubernetes uses NAT tables, which depend on traffic being tracked and stateful. Therefore, it is not recommended to use stateless rules.
- If a Kubernetes service has both port 1234/TCP and port 2345/UDP configured, a rule configured with the Pod as Consumer and the virtual service as Provider will open up both ports 1234/TCP and 2345/TCP, and 1234/UDP and 2345/UDP on the Pod's firewall (outbound rule).

  In case of a Kubernetes service configured with a `port` and `targetPort` statement in the manifest file as shown in the example below:

  ```
  apiVersion: v1
  kind: Service
  metadata:
    name: web-frontend-svc
    namespace: app1
    labels:
      app: app1
      tier: web-frontend
    annotations:
      com.illumio.role: Web
  spec:
    type: ClusterIP
    ports:
    - port: 8080
      targetPort: 80
      protocol: TCP
    - port: 8081
      targetPort: 81
      protocol: UDP
    selector:
      app: app1
      tier: web-frontend
  ```

  This configuration is supported with Illumio Core. In this case, only the port number associated to the `port` statement will show this issue, the port number associated to the `targetPort` statement will not show this issue and will use the `protocol` specified in the Service yaml file.

Kubelink Monitoring and Troubleshooting

If you deployed Illumio Core for Kubernetes 3.0.0 or later, Kubelink is deployed as part of the overall Helm Chart deployment, as described in "Deployment with Helm Chart (Core for Kubernetes 3.0.0 and Higher) [281]." If you deployed an earlier version of the product, refer to the "Deploy Kubelink in Your Cluster [311]" section of the "Deployment for Versions 21.5.15 or Earlier" chapter for details on how to configure and deploy Kubelink on Kubernetes.

Kubelink Process

Kubelink uses a single Ruby process which runs as: **`ruby /illumio/init.rb`**.

Kubelink Startup Log Messages

After deploying Kubelink (whether by Helm Chart or manually), verify your deployment with the **`kubectl get pods -n illumio-system`** command. The **`kubelinkpod`** should be shown with the Running status. In addition, you can review the log file entries after the deployment with the **`kubectl logs`** command pointing to the Kubelink pod name.

```
kubectl logs <kubelink_pod_name> -n illumio-system
```

A typical successful Kubelink deployment produces log entries similar to these:

```
I, [2022-05-23T14:36:53.847248 #10]  INFO -- : Starting Kubelink for PCE
https://192.168.88.127:10443
I, [2022-05-23T14:36:53.847502 #10]  INFO -- : Metrics reporting enabled;
reporting window 30
I, [2022-05-23T14:36:53.847520 #10]  INFO -- : PCE fqdn https://
192.168.88.127:10443
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:36:53.893048 #10]  INFO -- : Successfully connected to PCE
I, [2022-05-23T14:36:53.893170 #10]  INFO -- : begin sync on resource
namespaces
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
 see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:36:53.904369 #10]  INFO -- : Synchronized 6 namespaces.
I, [2022-05-23T14:36:53.904424 #10]  INFO -- : sync on resource namespaces
successful, setting up
resource version to 184232
I, [2022-05-23T14:36:53.904522 #10]  INFO -- : Start watch on namespaces
with version 184232
I, [2022-05-23T14:36:53.905678 #10]  INFO -- : begin sync on resource nodes
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
 see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
 see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:36:53.918093 #10]  INFO -- : Synchronized 1 nodes.
I, [2022-05-23T14:36:53.918143 #10]  INFO -- : sync on resource nodes
successful, setting up
resource version to 184232
I, [2022-05-23T14:36:53.918175 #10]  INFO -- : Start watch on nodes with
version 184232
I, [2022-05-23T14:36:53.919265 #10]  INFO -- : begin sync on resource pods
I, [2022-05-23T14:36:53.935536 #10]  INFO -- : sync on resource pods
```

```
successful, setting up
resource version to 184232
I, [2022-05-23T14:36:53.935601 #10]  INFO -- : Start watch on pods with
version 184232
I, [2022-05-23T14:36:53.936938 #10]  INFO -- : begin sync on resource
services
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
 see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
 see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:36:54.029965 #10]  INFO -- : Synchronized 3 services,
full=true, force=false
I, [2022-05-23T14:36:54.030013 #10]  INFO -- : sync on resource services
successful, setting up
resource version to 184232
I, [2022-05-23T14:36:54.030046 #10]  INFO -- : Start watch on services with
version 184232
I, [2022-05-23T14:36:54.031042 #10]  INFO -- : begin sync on resource
replica_sets
I, [2022-05-23T14:36:54.100090 #10]  INFO -- : Nothing to sync
I, [2022-05-23T14:36:54.100237 #10]  INFO -- : sync on resource
replica_sets successful, setting up
resource version to 184232
I, [2022-05-23T14:36:54.100281 #10]  INFO -- : Start watch on replica_sets
with version 184232
I, [2022-05-23T14:36:54.101226 #10]  INFO -- : begin sync on resource
stateful_sets
I, [2022-05-23T14:36:54.170175 #10]  INFO -- : Nothing to sync
I, [2022-05-23T14:36:54.170220 #10]  INFO -- : sync on resource
stateful_sets successful, setting up
resource version to 184232
I, [2022-05-23T14:36:54.170267 #10]  INFO -- : Start watch on stateful_sets
with version 184232
I, [2022-05-23T14:36:54.171159 #10]  INFO -- : begin sync on resource
daemon_sets
I, [2022-05-23T14:36:54.245866 #10]  INFO -- : Nothing to sync
I, [2022-05-23T14:36:54.246025 #10]  INFO -- : sync on resource daemon_sets
successful, setting up
resource version to 184232
I, [2022-05-23T14:36:54.246210 #10]  INFO -- : Start watch on daemon_sets
with version 184232
I, [2022-05-23T14:36:54.247946 #10]  INFO -- : begin sync on resource
replication_controllers
I, [2022-05-23T14:36:54.324925 #10]  INFO -- : Nothing to sync
I, [2022-05-23T14:36:54.324977 #10]  INFO -- : sync on resource
replication_controllers successful,
setting up resource version to 184232
```

```
I, [2022-05-23T14:36:54.325032 #10]  INFO -- : Start watch on
replication_controllers with version
184232
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
 see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:36:54.505403 #10]  INFO -- : replica_sets MODIFIED
I, [2022-05-23T14:37:24.312086 #10]  INFO -- : Heart beating to PCE
I, [2022-05-23T14:37:24.312191 #10]  INFO -- : Attaching metrics report to
heartbeat:
{:pod_changes=>[{:namespace=>"illumio-system", "added"=>0, "modified"=>0,
"deleted"=>1}],
:service_changes=>[], :duration_seconds=>30}
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
 see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:37:54.343467 #10]  INFO -- : Heart beating to PCE
I, [2022-05-23T14:37:54.343874 #10]  INFO -- : Attaching metrics report to
heartbeat:
{:pod_changes=>[], :service_changes=>[], :duration_seconds=>30}
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:38:24.373847 #10]  INFO -- : Heart beating to PCE
I, [2022-05-23T14:38:24.373924 #10]  INFO -- : Attaching metrics report to
heartbeat:
{:pod_changes=>[], :service_changes=>[], :duration_seconds=>30}
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:38:54.380933 #10]  INFO -- : Heart beating to PCE
I, [2022-05-23T14:38:54.381009 #10]  INFO -- : Attaching metrics report to
heartbeat:
 {:pod_changes=>[], :service_changes=>[], :duration_seconds=>30}
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
```

```
{verify_peer: true}'),
 see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:39:24.401636 #10]  INFO -- : Heart beating to PCE
I, [2022-05-23T14:39:24.401748 #10]  INFO -- : Attaching metrics report to
heartbeat:
{:pod_changes=>[], :service_changes=>[], :duration_seconds=>30}
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:39:54.422494 #10]  INFO -- : Heart beating to PCE
I, [2022-05-23T14:39:54.422595 #10]  INFO -- : Attaching metrics report to
heartbeat:
{:pod_changes=>[], :service_changes=>[], :duration_seconds=>30}
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
 see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:40:24.453077 #10]  INFO -- : Heart beating to PCE
I, [2022-05-23T14:40:24.453217 #10]  INFO -- : Attaching metrics report to
heartbeat:
{:pod_changes=>[], :service_changes=>[], :duration_seconds=>30}
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:40:54.466210 #10]  INFO -- : Heart beating to PCE
I, [2022-05-23T14:40:54.466455 #10]  INFO -- : Attaching metrics report to
heartbeat:
{:pod_changes=>[], :service_changes=>[], :duration_seconds=>30}
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:41:24.296410 #10]  INFO -- : Verify watches for
["namespaces", "nodes", "pods",
 "services", "replica_sets", "stateful_sets", "daemon_sets",
"replication_controllers"]
I, [2022-05-23T14:41:24.296468 #10]  INFO -- : Watch client namespaces
Connection Idle: 270.3355407714844s
I, [2022-05-23T14:41:24.296485 #10]  INFO -- : Watch client nodes
Connection Idle: 179.93679809570312s
I, [2022-05-23T14:41:24.296499 #10]  INFO -- : Watch client pods
Connection Idle: 240.5237274169922s
I, [2022-05-23T14:41:24.296513 #10]  INFO -- : Watch client services
Connection Idle: 270.0260314941406s
I, [2022-05-23T14:41:24.296526 #10]  INFO -- : Watch client replica_sets
Connection Idle: 269.85888671875s
I, [2022-05-23T14:41:24.296542 #10]  INFO -- : Watch client stateful_sets
Connection Idle: 270.0269775390625s
I, [2022-05-23T14:41:24.296573 #10]  INFO -- : Watch client daemon_sets
Connection Idle: 270.02490234375s
I, [2022-05-23T14:41:24.296731 #10]  INFO -- : Watch client
replication_controllers
```

```
Connection Idle: 270.02490234375s
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:41:24.300532 #10]  INFO -- : Synchronized 3 services,
full=true, force=true
I, [2022-05-23T14:41:24.452846 #10]  INFO -- : Heart beating to PCE
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
 see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
W, [2022-05-23T14:41:54.186807 #10]  WARN -- : watch client for
stateful_sets error callback invoked.
Resetting watch ...
W, [2022-05-23T14:41:54.186863 #10]  WARN -- : Watch on stateful_sets
ended. Resetting it after
3 seconds
I, [2022-05-23T14:41:54.441880 #10]  INFO -- : Heart beating to PCE
I, [2022-05-23T14:41:54.441991 #10]  INFO -- : Attaching metrics report to
heartbeat:
{:pod_changes=>[], :service_changes=>[], :duration_seconds=>60}
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
 see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:41:57.193339 #10]  INFO -- : begin sync on resource
stateful_sets
I, [2022-05-23T14:41:57.267375 #10]  INFO -- : Nothing to sync
I, [2022-05-23T14:41:57.267411 #10]  INFO -- : sync on resource
stateful_sets successful,
setting up resource version to 184451
I, [2022-05-23T14:41:57.267424 #10]  INFO -- : Start watch on stateful_sets
with version 184451
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
 see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
I, [2022-05-23T14:42:24.483142 #10]  INFO -- : Heart beating to PCE
I, [2022-05-23T14:42:24.483224 #10]  INFO -- : Attaching metrics report to
heartbeat:
 {:pod_changes=>[], :service_changes=>[], :duration_seconds=>30}
[WARNING; em-http-request] TLS hostname validation is disabled (use 'tls:
{verify_peer: true}'),
see CVE-2020-13482 and https://github.com/igrigorik/em-http-request/
issues/339 for details
```

Verify Kubelink Deployment

To verify your Kubelink deployment.

- To check the Kubelink Pod status for Kubernetes:

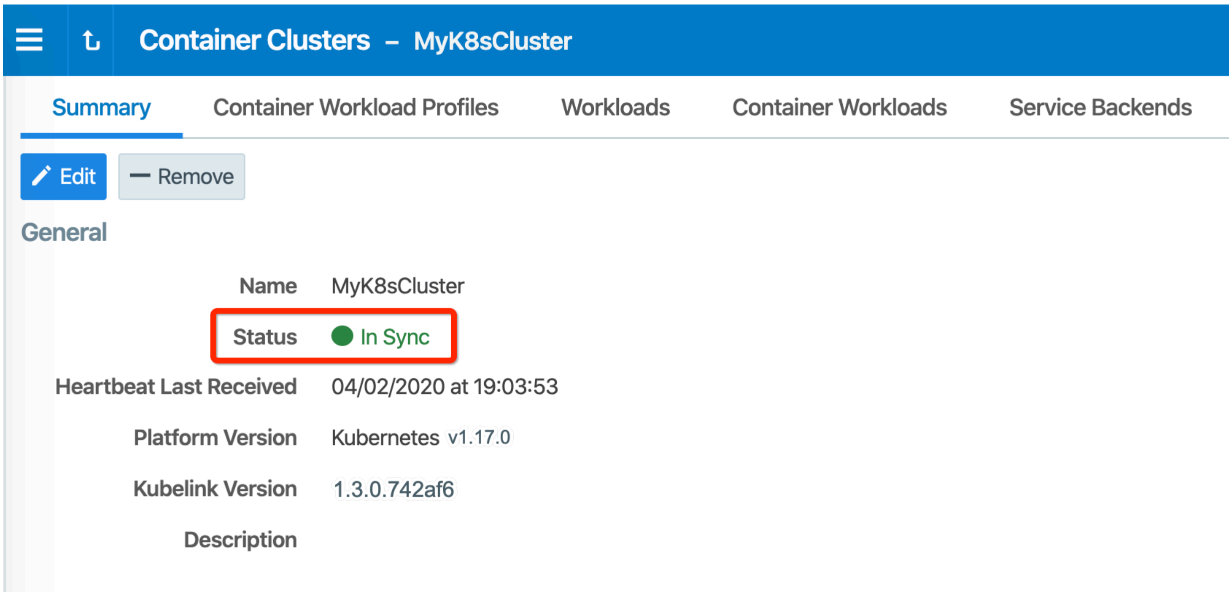
  ```
  kubectl get pods -n illumio-system
  ```
- To check the Kubelink Pod status for OpenShift:

```
oc get pods -n illumio-system
```

The `illumio-kubelink-xxxxxxxxxx-xxxxx` Pod should be in the "Running" state. If the either **`get pods -n illumio-system`** command shows the kubelink pod is not successfully running, check the log file for any ERROR messages.

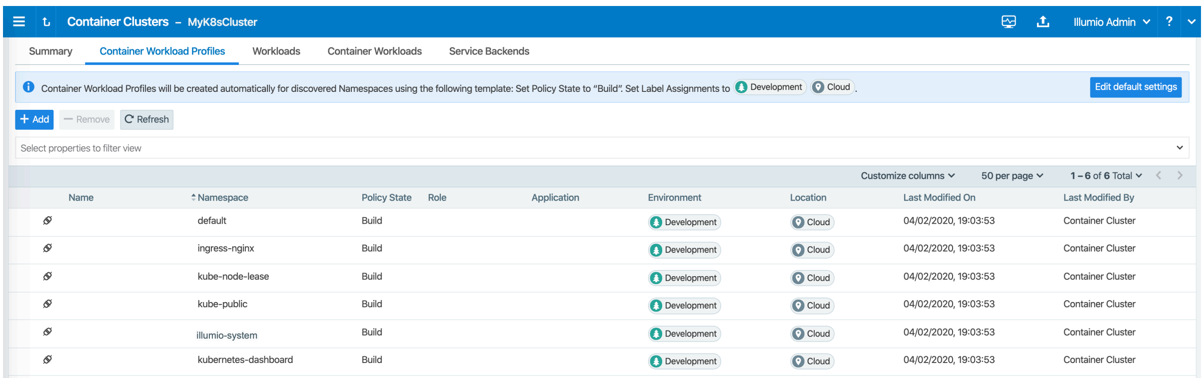
After Kubelink is successfully deployed, you can check the cluster information in the Illumio PCE UI. From the main menu, navigate to **Infrastructure > Container Clusters**.

Below is an example of a healthy container cluster state reported by Kubelink, where Status is "In Sync".

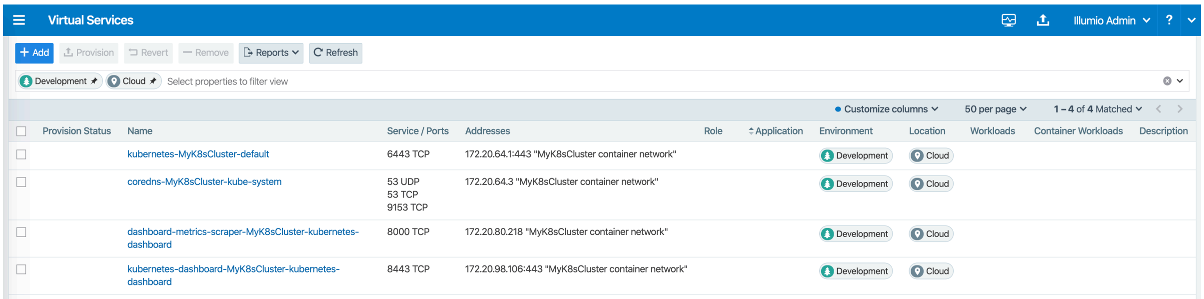


You can also verify in the PCE UI that Kubelink was successfully deployed by checking the following:

- Under the **Container Workload Profiles** tab, namespaces created in your Kubernetes or OpenShift cluster should be listed. An example is shown below.



- Under **Policy Objects > Virtual Services**, services created in your Kubernetes or OpenShift cluster should be listed. An example is shown below.



PCE-Kubelink Connection and Heartbeat

The Kubelink heartbeat to the PCE is logged in its log file. Use the **`kubectl logs`** command, and search for the string **`Heart beating to PCE`** to confirm. To confirm PCE-Kubelink connectivity, check the PCE UI, which will show the Kubelink pod as being offline if the heartbeat is missing 2-3 times (about 10 minutes).

Additional Kubelink Monitoring

Other Kubelink actions that can be confirmed in the Kubelink log file include:

API request succeeds

When Kubelink successfully sets up a watch with the Kubernetes API, the related log entry is:

```
sync on resource <RESOURCE> successful, setting up resource version to
<RESOURCE VERSION>
```

Information sent to PCE

When Kubelink successfully sends information to the PCE, the related log entry is:

```
Synchronized 2 <RESOURCE>, full=..., force=...
```

Setting Log Verbosity

The log verbosity level is set by default to include INFO, WARNING, and ERROR messages in the log. If your log appears to be extremely small (showing only ERRORs, for example), or is extremely large (which could indicate being set at the DEBUG level), you can check the `log_level` setting in the `illumio-kubelink-secret.yml` file. Values for this setting are:

log_level Setting	Description
0	Debug
1	Info (default)
2	Warn
3	Error

Values are cumulative, in that a setting includes any other setting above it. For example, the default setting of '1' includes in the log file all INFO, WARNING, and ERROR messages. Whereas a setting of '3' would only include ERROR messages.

OpenShift Deployment Reference

After you set up your clusters, make sure you perform the steps in the order provided in this section.

1. Prepare OpenShift for Illumio Core.
2. Deploy Kubelink.
3. Implement Kubelink with a private KPI.
4. Install and Pair VENs for Containers.
5. Manage OpenShift Namespaces.

Prepare OpenShift for Illumio Core

If the prerequisite steps are not performed prior to VEN and Kubelink installation, containerized environments and Kubelink may be disrupted.

Unique Machine ID

Some of the functionalities and services provided by the Illumio VEN and Kubelink depend on the Linux machine-id of each OpenShift cluster node. Each machine-id must be unique in order to take advantage of the functionalities. By default, the Linux OS generates a random machine-id to give each Linux host uniqueness. However, there are cases when machine-id's can be duplicated across machines. This is common across deployments that clone machines from a golden image, for example, spinning up virtual machines from VMware templates or creating Amazon EC2 instances from an AMI.

To verify machine-ids and resolve any duplicate machine-ids across nodes:

1. `ssh` into every node of the OpenShift cluster (master, infra, and worker) as the root user.
2. Check the contents of the `/etc/machine-id` file. The output is a string of letters and numbers.
3. If the machine-id string is unique for each node, then the environment is ok. If the machine-id is duplicated across any of the nodes, you must generate a machine-id for each node which has the same machine-id.

You can run the following command to view the output of machine-id:

```
cat /etc/machine-id
```

If the machine-id is duplicated, then run the command listed below to generate a new machine-id. You will also need to restart the `atomic-OpenShift-node` service on each node. If the machine-id is not duplicated, go to the next section.

```
rm -rf /etc/machine-id; touch /etc/machine-id; systemd-machine-id-setup;
service atomic-OpenShift-node restart
```

> **NOTE**
> Check the machine-id again to verify that each machine has a unique machine-id.

Create Labels

For details on creating labels, see "Labels and Label Groups" in the Security Policy Guide.

The labels listed below are used in examples throughout this document. You are not required to use the same labels.

Name	Label type
Openshift Infrastructure	Application
Development	Environment
HQ	Location
Kubelink	Role
Master	Role
Infra	Role
Compute	Role

Create Pairing Profiles

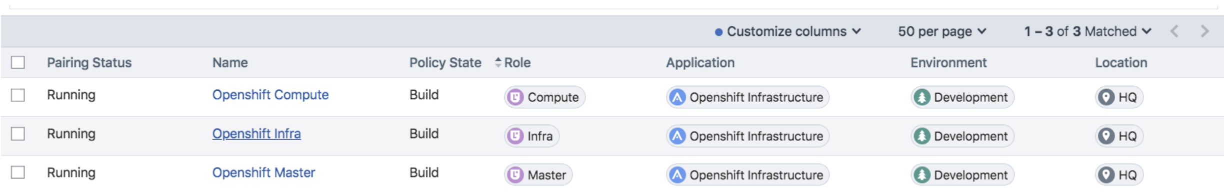
After creating labels for your OpenShift cluster nodes, you can use those labels to create pairing profiles. You do not need to create pairing profiles for container workloads.

For ease of configuration and management, consider applying the same Application, Environment, and Location labels across all nodes of the same OpenShift cluster. The screenshot below show examples of three pairing profiles for one OpenShift Enterprise cluster. The pairing profiles are used for pairing either master, compute, or infrastructure nodes of an OpenShift cluster.

TIP

It is recommended that all pairing profiles for OpenShift nodes **not** use Enforced policy state.

Move into Enforced state after you have completed all other configuration steps in this guide (setup Kubelink, discover services, and write rules).


Customize columns 50 per page 1 – 3 of 3 Matched

Pairing Status	Name	Policy State	Role	Application	Environment	Location
Running	Openshift Compute	Build	Compute	Openshift Infrastructure	Development	HQ
Running	Openshift Infra	Build	Infra	Openshift Infrastructure	Development	HQ
Running	Openshift Master	Build	Master	Openshift Infrastructure	Development	HQ

Deploy Kubelink

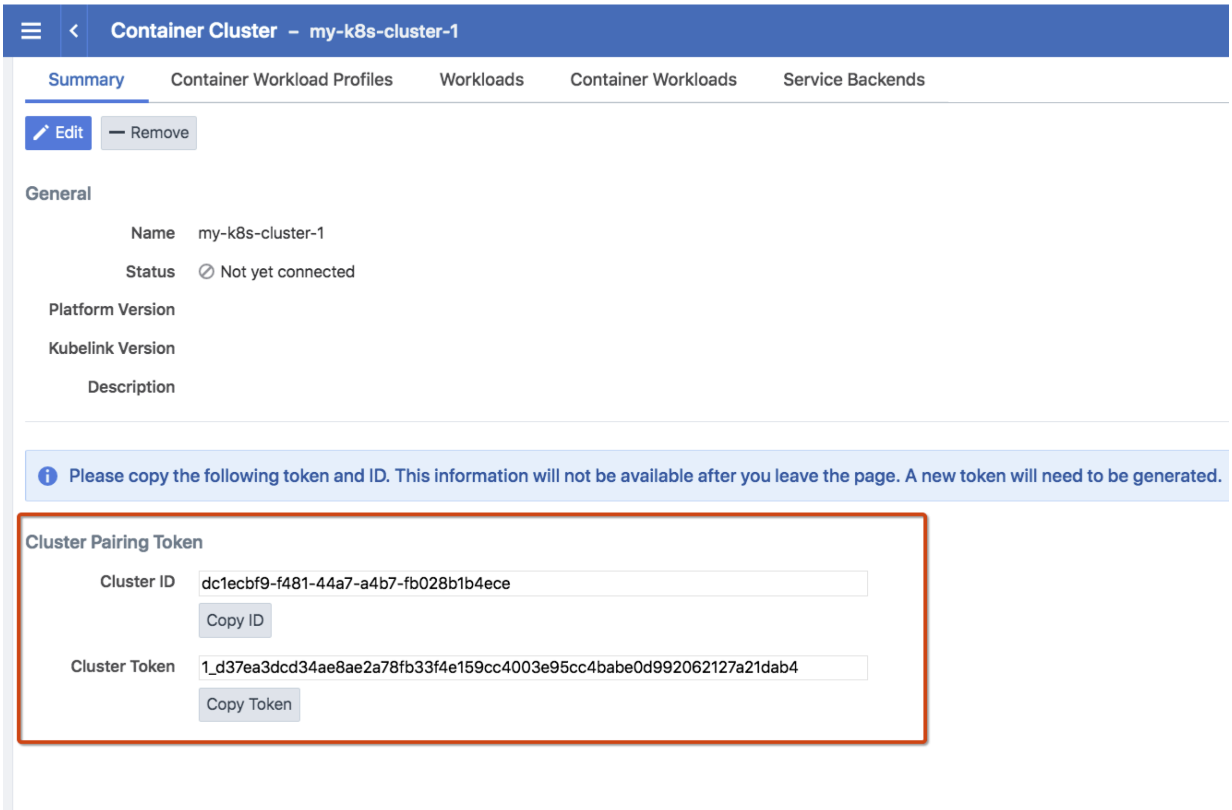
Download the required resources such as Kubelink docker image, secret file, and deployment file from the Illumio Support portal (login required).

Prerequisites

- Kubelink deployment file provided by Illumio. For OpenShift deployments, the file name is `illumio-kubelink-openshift.yml`.
- Kubelink secret file provided by Illumio. This file name is `illumio-kubelink-secret.yml`.
- Illumio's Kubelink docker image uploaded to your private docker registry.

Create Container Cluster

1. Log into the PCE as a user with Global Organization Owner privileges.
2. From the PCE web console menu, choose **Infrastructure** > **Container Clusters**.
3. Click **Add**.
   a. Enter a Name.
   b. **Save** the Container Cluster.
4. You will see a summary page of the new Container Cluster. Copy the values of the Cluster ID and Cluster Token found under the Cluster Pairing Token section.
5. Once you have the values, you can exit the Container Cluster page.



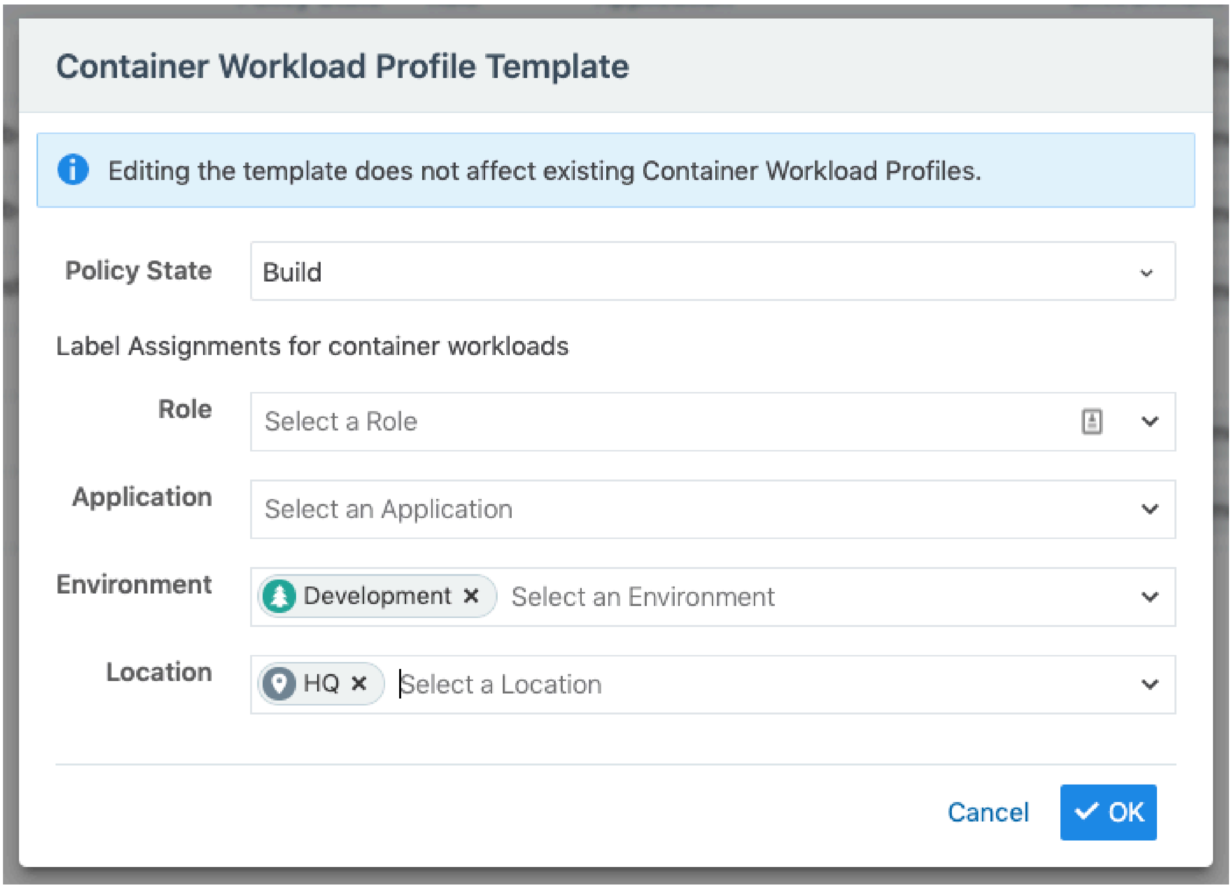
Configure Container Workload Profile

When configuring a new Container Cluster, it is recommended to set the default settings shared by all the Container Workload Profiles. Illumio provides a Container Workload Profile template that can be used for that purpose. By defining the default Policy State and minimum set of labels common to all namespaces in the cluster, you will save time later on when new namespaces are discovered by Kubelink. Each new profile created will inherit what was defined in the template.

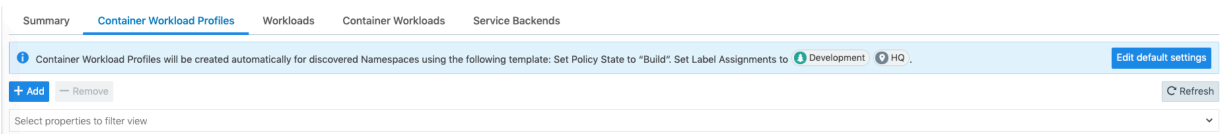
SSL Verification

Illumio does not provide a simple way to redefine all at once the labels associated to each profile all at once in this release, so it is strongly recommended to use this template to define the default values for all profiles part of the same cluster.

To define the default parameters for all profiles using a template, under Container Workload Profiles, click on Edit default settings and fill in the different fields. An example is shown below:



Once you validate, you should see something like the following:



Configure Kubelink Secret

This step assumes that you have created a Container Cluster object in the PCE. You will need the Cluster ID and Cluster Token values for the Kubelink secret.

1. `ssh` to the master node.
2. Open the kubelink secret YAML file and modify the `stringData`.
   a. `ilo_server` = the PCE URL and port. Example: `https://mypce.example.com:8443`
   b. `ilo_cluster_uuid` = Cluster ID value from previous step. Example: `dc1ecbf9-f481-44a7-a4b7-fb028b1b4ece`
   c. `ilo_cluster_token` = Cluster Token from previous step. Example: `1_d37ea3dcd34ae8ae2a78fb33f4e159cc4003e95cc4babe0d992062127a21dab4`
   d. `ignore_cert` = SSL verification. The value is boolean and is recommended to be set to false so that Kubelink requires PCE certificate verification. Example: '`false`'
   e. `log_level` = Log level where '`0`' for debug, '`1`' for info, '`2`' for warn, or '`3`' for error. Example: '`1`'

   SSL Verification

   Illumio does not recommend turning off SSL verification (`ignore_cert`: '`true`'); however, this is an option for deployments in which the PCE uses a self-signed certificate.

   Contents of a modified illumio-kubelink-secret.yml file are shown below.

```
#
# Copyright 2013-2020 Illumio, Inc. All Rights Reserved.
#

apiVersion: v2
kind: Secret
metadata:
  name: illumio-kubelink-config
  namespace: kube-system
type: Opaque
stringData:
  ilo_server: https://mypce.example.com:8443 # Example: https://
mypce.example.com:8443
  ilo_cluster_uuid: dc1ecbf9-f481-44a7-a4b7-fb028b1b4ece
# Example: cc4997c1-408b-4f1d-a72b-91495c24c6a0
  ilo_cluster_token:
1_d37ea3dcd34ae8ae2a78fb33f4e159cc4003e95cc4babe0d992062127a21dab4
# Example:
170b8aa3dd6d8aa3c284e9ea016e8653f7b51cb4b0431d8cbdba11508763f3a3
  ignore_cert: 'false' # Set to 'true' to ignore the PCE certificate
  log_level: '1' # Default log level is info
```

> **NOTE**
> If you are going to use a private PKI to sign the PCE certificate, see Implement Kubelink with a Private PKI [375] before deploying Kubelink.

3. Save the changes.
4. Create the Kubelink secret using the file.

```
oc create -f illumio-kubelink-secret.yml
```

Deploy Kubelink

Modify the Kubelink configuration file to point to the correct docker image. The example in this document has kubelink:<version#> uploaded to registry.example.com:443/illumio, which means the image link in this example is registry.example.com:443/illumio/kubelink:<version#>

1. Edit the Kubelink configuration YAML file. For OpenShift clusters, the file name will be `illumio-kubelink-openshift.yml`.
   - Inside the YAML you will find the `spec: > template: > spec: > containers:` section. Paste the image link in the `image:` attribute. The string should be wrapped in single quotes as shown in the example below.
2. Save the changes.
   Below is a snippet from an example of the Kubelink configuration for OpenShift to illustrate the image location.

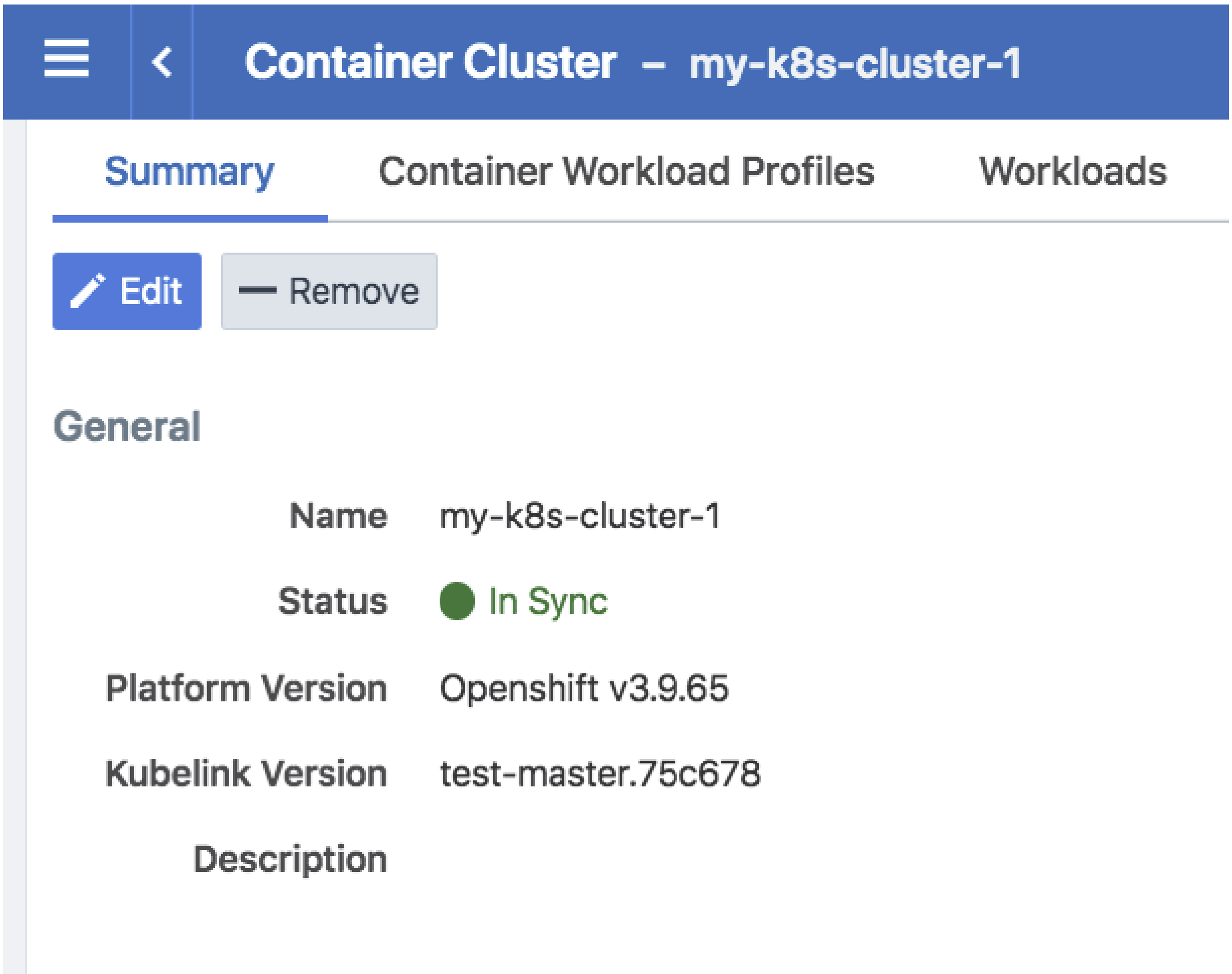
```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: illumio-kubelink
  namespace: kube-system
spec:
  replicas: 1
  selector:
    matchLabels:
      app: illumio-kubelink
  template:
    metadata:
      labels:
        app: illumio-kubelink
    spec:
#      nodeSelector:
#        node-role.kubernetes.io/master: ""
      serviceAccountName: illumio-kubelink
      tolerations:
      - key: node-role.kubernetes.io/master
        effect: NoSchedule
      containers:
      - name: illumio-kubelink
        image: 'registry.example.com:443/illumio/illumio-
kubelink:<version#>'
        imagePullPolicy: Always
        env:
          - name: ILO_SERVER
            valueFrom:
              secretKeyRef:
                name: illumio-kubelink-config
                key: ilo_server
```

3. (Optional) If you're using a private PKI to sign the PCE certificate, make sure you add the references to the root CA certificate that signed the PCE certificate. For more details, see Implement Kubelink with a Private PKI [375].
4. To deploy Kubelink, run the following command:

```
oc apply -f illumio-kubelink-openshift.yml
```

After Kubelink is successfully installed, you can check the cluster information by using the Illumio PCE web console. From the main menu, navigate to **Infrastructure > Container Clusters**.

Below is an example of a healthy container cluster state reported by Kubelink.



Implement Kubelink with a Private PKI

This section describes how to implement Kubelink with a PCE using a certificate signed by a private PKI. It describes how to configure Kubelink to accept the certificate from the PCE signed by a private root or intermediate Certificate Authority (CA) and ensure that Kubelink can communicate in a secure way with the PCE.

> **NOTE**
> The steps described below are not applicable for a PCE using a self-signed certificate.

Prerequisites

- Access to the root CA to download the root CA certificate.
- Access to your Kubernetes cluster and can run `kubectl` commands.
- Correct privileges in your Kubernetes cluster to create resources like a configmaps, secrets, and pods.
- Access to the PCE UI as a Global Organization Owner.

Download the Root CA Certificate

Before you begin, ensure that you have access to the root CA certificate. The root CA certificate is a file that can be exported from the root CA without compromising the security of the company. It is usually made available to external entities to ensure a proper SSL handshake between a server and its clients.

You can download the root CA cert in the CRT format on your local machine. Below is an example of a root CA certificate:

```
$ cat root.democa.illumio-demo.com.crt
-----BEGIN CERTIFICATE-----
MIIGSzCCBDOgAwIBAgIUAPw0NfPAivJW4YmKZ499eHZH3S8wDQYJKoZIhvcNAQEL
---output suppressed---
wPG0lug46K1EPQqMA7YshmrwOd6ESy6RGNFFZdhk9Q==
-----END CERTIFICATE-----
```

You can also get the content of your root CA certificate in a readable output format by running the following command:

```
$ openssl x509 -text -noout -in ./root.democa.illumio-demo.com.crt
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            fc:34:35:f3:c0:8a:f2:56:e1:89:8a:67:8f:7d:78:76:47:dd:2f
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=US, ST=California, L=Sunnyvale, O=Illumio, OU=Technical 
Marketing,
        CN=Illumio Demo Root CA 1/emailAddress=tme-team@illumio.com
        Validity
            Not Before: Jan 20 00:05:36 2020 GMT
            Not After : Jan 17 00:05:36 2030 GMT
        Subject: C=US, ST=California, L=Sunnyvale, O=Illumio, OU=Technical 
Marketing,
        CN=Illumio Demo Root CA 1/emailAddress=tme-team@illumio.com
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (4096 bit)
                Modulus:
                    00:c0:e5:48:7d:97:f8:5b:8c:ef:ac:16:a8:8c:aa:
                    68:b8:48:af:28:cd:17:8f:02:c8:82:e9:69:62:e2:
                    89:2b:be:bd:34:fc:e3:4d:3f:86:5e:d7:e6:89:34:
                    71:60:e6:54:61:ac:0f:26:1c:99:6f:80:89:3f:36:
                    b3:ad:78:d1:6c:3f:d7:23:1e:ea:51:14:48:74:c3:
                    e8:6e:a2:79:b1:60:4c:65:14:2a:f1:a0:97:6c:97:
                    50:43:67:07:b7:51:5d:2c:12:49:81:dc:01:c9:d1:
                    57:48:32:2e:87:a8:d2:c0:b9:f8:43:b2:58:10:af:
                    54:59:09:05:cb:3e:f0:d7:ef:70:cc:fc:53:48:ee:
                    a4:a4:61:f1:d7:5b:7c:a9:a8:92:dc:77:74:f4:4a:
                    c0:4a:90:71:0f:6d:9e:e7:4f:11:ab:a5:3d:cd:4b:
                    8b:79:fe:82:1b:16:27:94:8e:35:37:db:dd:b8:fe:
                    fa:6d:d9:be:57:f3:ca:f3:56:aa:be:c8:57:a1:a8:
                    c9:83:dd:5a:96:5a:6b:32:2d:5e:ae:da:fc:85:76:
                    bb:77:d5:c2:53:f3:5b:61:74:e7:f3:3e:4e:ad:10:
```

```
                    7d:4f:ff:90:69:7c:1c:41:2f:67:e4:13:5b:e6:3a:
                    a3:2f:93:61:3b:07:56:59:5a:d9:bc:34:4d:b3:54:
                    b5:c6:e5:0a:88:e9:62:7b:4b:85:d2:9e:4c:ee:0b:
                    0d:f4:72:b1:1b:44:04:93:cf:cc:bb:18:31:3a:d4:
                    83:4a:ff:15:42:2d:91:ca:d0:cb:36:d9:8d:62:c0:
                    41:59:1a:93:c7:27:79:08:94:b2:a2:50:3c:57:27:
                    33:af:f0:b6:92:44:49:c5:09:15:a7:43:2a:0f:a9:
                    02:61:b3:66:4f:c3:de:d3:63:1e:08:b1:23:ea:69:
                    90:db:e8:e9:1e:21:84:e0:56:e1:8e:a1:fa:3f:7a:
                    08:0f:54:0a:82:41:08:6b:6e:bb:cf:d6:5b:80:c6:
                    ea:0c:80:92:96:ab:95:5d:38:6d:4d:da:38:6b:42:
                    ef:7c:88:58:83:88:6d:da:28:62:62:1f:e5:a7:0d:
                    04:9f:0d:d9:52:39:46:ba:56:7c:1d:77:38:26:7c:
                    86:69:58:4d:b0:47:3a:e2:be:ee:1a:fc:4c:de:67:
                    f3:d5:fe:e6:27:a2:ef:26:86:19:5b:05:85:9c:4c:
                    02:24:76:58:42:1a:f8:e0:e0:ed:78:f2:8f:c8:5a:
                    20:a9:2d:0b:d4:01:fa:57:d4:6f:1c:0a:31:30:8c:
                    32:7f:b0:01:1e:fe:94:96:03:ee:01:d7:f4:4a:83:
                    f5:06:fa:60:43:15:05:9a:ca:88:59:5c:f5:13:09:
                    82:69:7f
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Subject Key Identifier:
                3D:3D:3D:61:E6:88:09:FE:34:0F:1D:5E:5E:52:72:71:C7:DE:15:92
            X509v3 Authority Key Identifier:

keyid:3D:3D:3D:61:E6:88:09:FE:34:0F:1D:5E:5E:52:72:71:C7:DE:15:92

            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Key Usage: critical
                Digital Signature, Certificate Sign, CRL Sign
    Signature Algorithm: sha256WithRSAEncryption
         28:24:86:91:a6:4a:88:e4:8d:6b:fc:67:2a:68:08:67:35:e5:
         a6:77:ff:07:4b:89:53:99:2e:6d:95:df:12:81:28:6a:8e:6f:
         5a:98:95:5b:4a:21:ae:f0:20:a4:4e:06:b2:4e:5a:67:c1:6a:
         06:f1:0f:c1:f7:7e:f2:e0:b3:9d:d8:54:26:6a:b2:1c:19:b8:
         b5:5c:c7:03:6b:f7:70:9e:72:85:c9:29:55:f9:f4:a4:f2:b4:
         3b:3d:ce:25:96:67:32:1e:8d:e2:00:22:55:4b:05:4f:ee:0e:
         67:ac:db:1b:61:da:5f:9c:10:1c:0c:05:66:c0:5b:5f:b9:95:
         59:a9:58:5b:e7:69:ac:b0:bd:b3:c2:a3:35:58:01:a4:ff:c0:
         8d:ac:1c:19:21:41:50:fb:8e:e0:f5:a9:ad:ec:de:cb:53:04:
         a9:d8:ac:76:8a:09:0d:7c:c6:1a:bc:06:74:bb:10:1c:aa:07:
         f6:cb:b2:1b:0c:0c:65:03:45:2b:51:d5:6e:a0:4d:91:ce:c5:
         ed:8d:a9:e7:f6:37:7d:ab:1b:a4:a2:a3:3b:76:17:5b:d9:3a:
         9c:c1:df:cc:cd:a0:b0:a9:5c:74:61:d7:a0:1d:04:67:68:ee:
         a6:7b:1e:41:a4:02:fc:65:9e:e3:c1:c2:57:b2:2e:b0:ff:a9:
         86:82:35:4d:29:b2:fe:74:2e:b8:37:5d:2b:e8:69:f2:80:29:
         19:f1:1e:7a:5d:e3:d2:51:50:46:30:54:7e:b8:ad:59:61:24:
         45:a8:5a:fe:19:ff:09:31:d0:50:8b:e2:15:c0:a2:f1:20:95:
         63:55:18:a7:a2:ad:16:25:c7:a3:d1:f2:e5:be:6d:c0:50:4b:
         15:ac:e0:10:5e:f3:7b:90:9c:75:1a:6b:e3:fb:39:88:e4:e6:
         9f:4c:85:60:67:e8:7d:2e:85:3d:87:ed:06:1d:13:0b:76:d7:
         97:a5:b8:05:76:67:d6:41:06:c5:c0:7a:bd:f4:c6:5b:b2:fd:
         23:6f:1f:57:2e:df:95:3f:26:a5:13:4d:6d:96:12:56:98:db:
```

```
2e:7d:fd:56:f5:71:b7:19:2b:c9:de:2d:b9:c8:17:cc:20:de:
7c:19:7a:aa:12:97:1c:80:b7:d3:67:d3:b7:a7:96:f0:c9:4d:
f5:8b:0e:10:3b:b9:4e:09:90:5a:3b:51:c9:48:a2:ca:9f:db:
72:44:87:59:db:49:fa:75:44:b5:f6:7f:c5:26:e1:01:ae:7b:
6f:4a:75:d1:b5:b3:68:c0:31:48:f8:5c:06:c0:f1:b4:96:e8:
38:e8:ad:44:3d:0a:8c:03:b6:2c:86:6a:f0:39:de:84:4b:2e:
91:18:d1:45:65:d8:64:f5
```

Create a configmap in Kubernetes Cluster

After downloading the certificate locally on your machine, create a configmap in the Kubernetes cluster that will copy the root CA certificate on your local machine into the Kubernetes cluster.

To create configmap, run the following command:

```
$ kubectl -n kube-system create configmap root-ca-config \
    --from-file=./certs/root.democa.illumio-demo.com.crt
```

The `--from-file` option points to the path where the root CA certificate is stored on your local machine.

To verify that configmap was created correctly, run the following command:

```
$ kubectl -n kube-system create configmap root-ca-config \
> --from-file=./certs/root.democa.illumio-demo.com.crt
configmap/root-ca-config created
$
$ kubectl -n kube-system get configmap
NAME                                 DATA   AGE
calico-config                        8      142d
cluster-info                         4      142d
coredns                              1      142d
coredns-autoscaler                   1      142d
crn-info-ibmc                        6      142d
extension-apiserver-authentication   6      142d
iaas-subnet-config                   1      142d
ibm-cloud-cluster-ingress-info       2      142d
ibm-cloud-provider-data              1      142d
ibm-cloud-provider-ingress-cm        6      142d
ibm-master-proxy-config              1      142d
ibm-network-interfaces               1      142d
kube-dns                             0      142d
kubernetes-dashboard-settings        1      44d
metrics-server-config                1      142d
node-local-dns                       1      142d
root-ca-config                       1      12s
subnet-config                        1      142d
$
$ kubectl -n kube-system describe configmap root-ca-config
Name:         root-ca-config
Namespace:    kube-system
Labels:       <none>
Annotations:  <none>
```

```
Data
====
root.democa.illumio-demo.com.crt:
----
-----BEGIN CERTIFICATE-----
MIIGSzCCBDOgAwIBAgIUAPw0NfPAivJW4YmKZ499eHZH3S8wDQYJKoZIhvcNAQEL
---output suppressed---
wPG0lug46K1EPQqMA7YshmrwOd6ESy6RGNFFZdhk9Q==
-----END CERTIFICATE-----

Events:  <none>
$
```

`root-ca-config` is the name used to designate configmap. You can modify it according to your naming convention.

Modify Kubelink Manifest File to Use Certificate

After creating the configmap in your Kubernetes cluster, modify the YAML file that describes Kubelink.

The current manifest file provided by Illumio does not include this modification, by default. Open the .yml file and add the following code blocks:

- volumeMounts (under `spec.template.spec.containers`)
- volumes (under `spec.template.spec`)

```
        volumeMounts:
          - name: root-ca
            mountPath: /etc/pki/tls/ilo_certs/
            readOnly: false
      volumes:
      - name: root-ca
        configMap:
          name: root-ca-config
```

NOTE

In a YAML file, the indentation matters. Make sure that the indentation in the file is as specified.

`root-ca` is the name used to designate the new volume mounted in the container. You can modify it according to your naming convention.

After successfully modifying the manifest file, deploy Kubelink. For more details, see Deploy Kubelink [371].

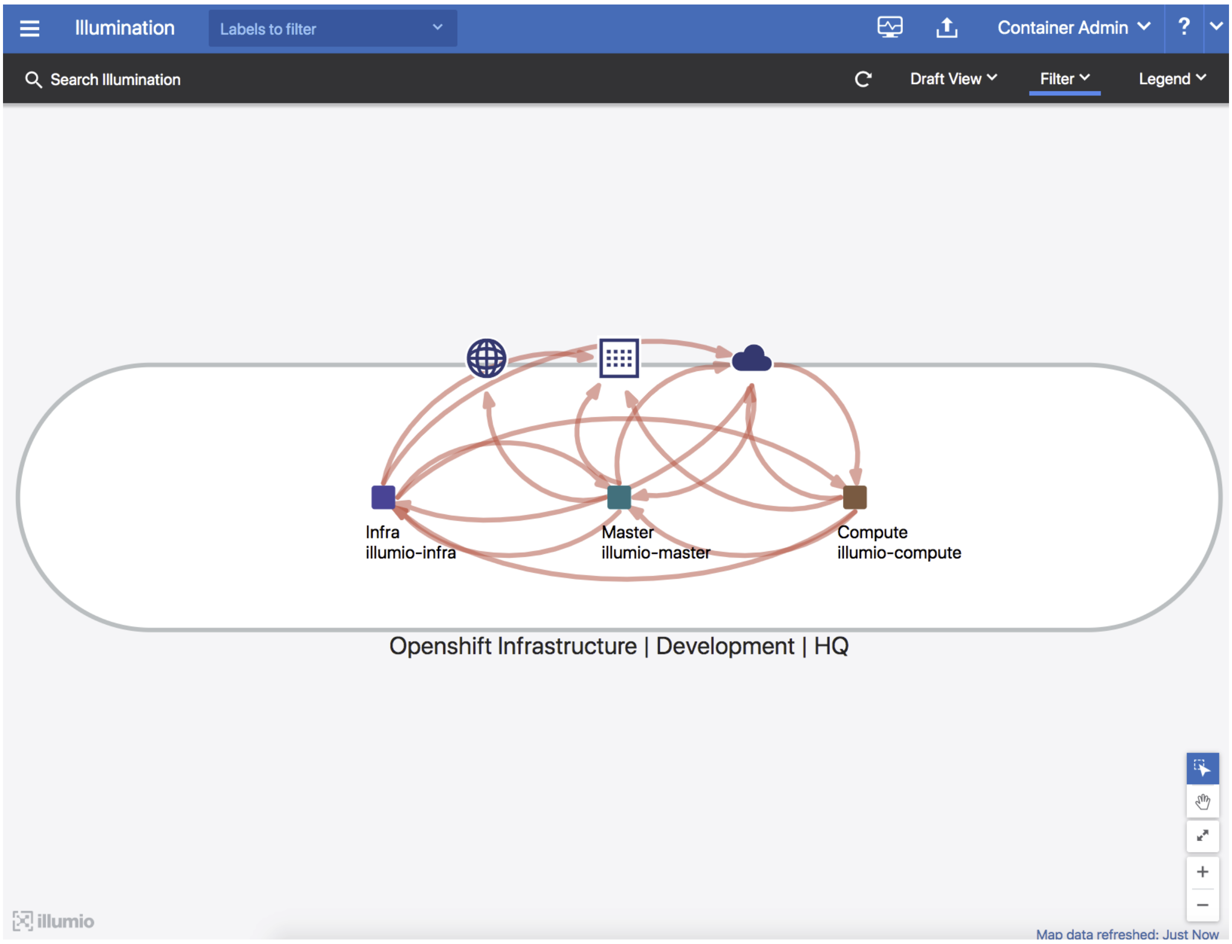
Install and Pair VENs for Containers

Using the pairing profiles mentioned earlier in this guide to install the VEN on each node of your OpenShift cluster. For more information about installing VENs, see the VEN Installation and Upgrade Guide.

Ensure that either of the two requirements below have been met prior to installing the VEN:

- Kubelink is deployed on the OpenShift cluster and in sync with the PCE
- Firewall coexistence is enabled

Below is a screenshot of Illumination with a master, compute, and infra node after deploying and pairing the Illumio VEN.



Manage OpenShift Namespaces

After activating the VENs on the OpenShift cluster nodes and Kubelink is in sync with the PCE, you can start managing the OpenShift projects (or namespaces). By default, all namespaces are unmanaged, which means Illumio Core does not apply any inbound or outbound controls to the pods within those namespaces. Any pods or services within unmanaged namespaces do not show up in the PCE inventory and Illumination.

After an Illumio Core PCE administrator changes an OpenShift namespace from unmanaged to managed, the pods and services will show up in Illumination and inherit the labels of each

OpenShift namespace. The pods are represented in Illumio Core as Container Workloads. If there are frontend services, then Illumio Core represents each one as a Virtual Service.

The following section describes how to change a namespace from unmanaged to managed.

Container Workload Profiles

Log into the PCE Web Console.

1. From the PCE web console menu, choose **Infrastructure** > **Container Clusters**.
2. Select the **Container Cluster** you want to manage.
3. Select the **Container Workload Profiles** tab.
4. You will see a list of all namespaces in the cluster. Select the namespace you want to manage.
5. Click **Edit**:
   a. Name is optional.
   b. Select a Container Workload Policy State (anything other than unmanaged).
   c. Assign Labels (optional).
   d. Click **Save**.

When assigning labels, you can assign no labels, some labels, or all labels to the namespace. If there is a label which is not assigned, then you can insert annotations into the deployment configuration (or application configuration) to assign labels. If there is a conflict between a label assigned via the Container Workload Profile and the annotations in the deployment configuration, then the label from the Container Workload Profile will override the deployment configuration. Regardless of how you assign labels, it is not required for pods or services to have all labels in order for the PCE to manage them. Below are instructions on how to assign labels via the deployment configuration.

Using Annotations

For Deployment Configurations (Pods)

1. Open the OpenShift Web Console.
2. Navigate to the desired deployment/daemon set and click **Edit YAML**.
   a. Inside the configuration YAML navigate to `spec: > template: > metadata: > annotations:`. If `annotations:` does not exist, then create an `annotations:` section underneath `metadata:`.
   b. The following Illumio label key fields which can go under the the `annotations:` section.
      - `com.illumio.role:`
      - `com.illumio.app:`
      - `com.illumio.env:`
      - `com.illumio.loc:`
   c. Fill in the appropriate labels.
   d. Save the file and exit.

For Service Configurations (Services)

1. Open the OpenShift Web Console.
2. Navigate to the desired service and click **Edit YAML**.
   a. Inside the configuration YAML navigate to `metadata: > annotations:`. If `annotations:` does not exist, then create an `annotations:` section underneath `metadata:`.

b. The following Illumio label key fields which can go under the the `annotations:` section.
   - `com.illumio.role:`
   - `com.illumio.app:`
   - `com.illumio.env:`
   - `com.illumio.loc:`

c. Fill in the appropriate labels.

d. Save the file and exit.

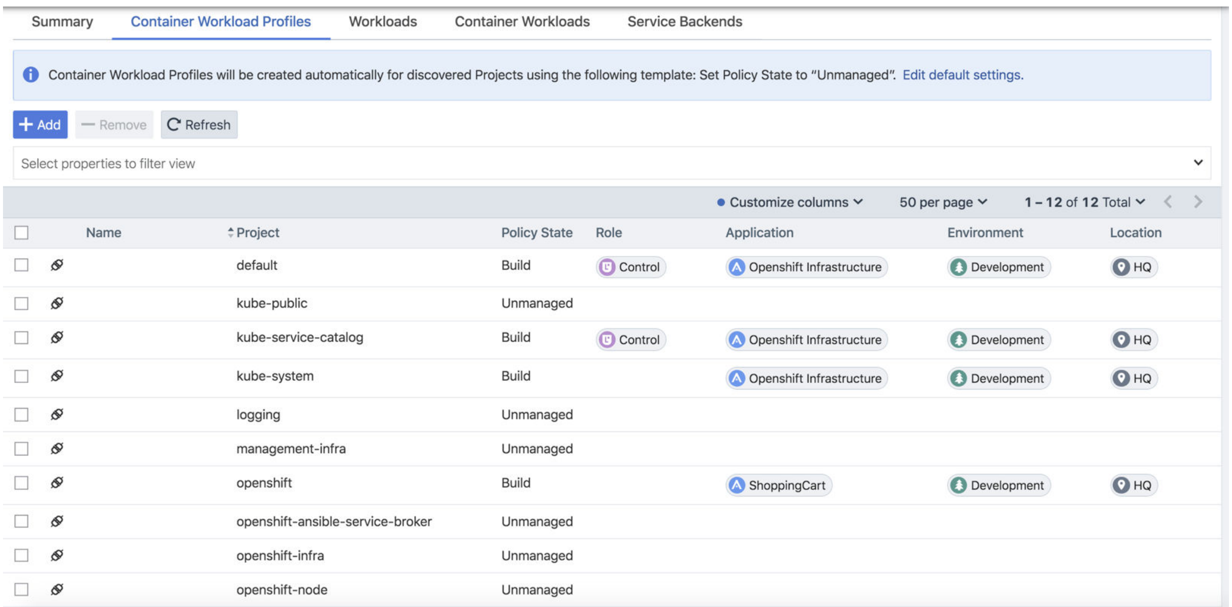
When using the annotations method, you may need to restart the pods or service after saving the changes to the YAML for the labels to get assigned.

Below are examples of pods and namespaces which use label assignments via either Container Workload Profiles or a mix of Container Workload Profiles plus annotation insertion.

This example changes unmanaged namespaces of Openshift infrastructure services (such as apiserver, registry-console, etc.) into managed namespaces.

Things to notice about the example shown below:

- There are Openshift infrastructure services, or control plane pods, that exist within namespaces like **`default`**, **`kube-service-catalog`**, etc. They will inherit all four R-A-E-L labels, including a Role label called "Control", from what has been configured in the Container Workload Profile(s). The Application, Environment, and Location labels are the same as the Openshift cluster nodes. This will minimize the complexity of writing policy which is mentioned later in this guide.
- The Kubelink pod exists in the kube-system. This pod will get the same application, environment, and location labels as the Openshift cluster nodes. The role label is left blank and will be specified later using the annotations. These labels are assigned to the Kubelink pod through the Container Workload Profile associated to the kube-system namespace.
- There is a namespace called **`openshift`** which contains two different deployments or a two-tier shopping cart application (Web and Database). To achieve tier-to-tier segmentation across the application they would need different Role labels; therefore, a Role label will be inserted into the annotations of each deployment configuration.



Name	Project	Policy State	Role	Application	Environment	Location
	default	Build	Control	Openshift Infrastructure	Development	HQ
	kube-public	Unmanaged				
	kube-service-catalog	Build	Control	Openshift Infrastructure	Development	HQ
	kube-system	Build		Openshift Infrastructure	Development	HQ
	logging	Unmanaged				
	management-infra	Unmanaged				
	openshift	Build		ShoppingCart	Development	HQ
	openshift-ansible-service-broker	Unmanaged				
	openshift-infra	Unmanaged				
	openshift-node	Unmanaged				

Snippet of **illumio-kubelink** deployment configuration file shown here. Role label of "Kubelink" inserted under `spec: > template: > metadata: > annotations:` section.

illumio-kubelink-openshift.yml

```
apiVersion: apps/v2
kind: Deployment
metadata:
  name: illumio-kubelink
  namespace: kube-system
spec:
  replicas: 1
  selector:
    matchLabels:
      app: illumio-kubelink
  template:
    metadata:
      labels:
        app: illumio-kubelink
      annotations:
        com.illumio.role: Kubelink
```

Snippet of the Shopping-Cart Web deployment configuration file shown here. Role label of "Web" inserted under `spec: > template: > metadata: > annotations:` section.

shopping-cart-web.yml

```
spec:
  replicas: 3
  revisionHistoryLimit: 10
  selector:
    name: shopping-cart-web
  strategy:
    activeDeadlineSeconds: 21600
    resources: {}
    rollingParams:
      intervalSeconds: 1
      maxSurge: 25%
      maxUnavailable: 25%
      timeoutSeconds: 600
      updatePeriodSeconds: 1
    type: Rolling
  template:
    metadata:
      annotations:
        com.illumio.role: Web
        openshift.io/generated-by: OpenShiftNewApp
      creationTimestamp: null
      labels:
```

Snippet of the Shopping-Cart Database deployment configuration file shown here. Role label of "Database" inserted under `spec: > template: > metadata: > annotations:` section.

shopping-cart-db.yml

```
spec:
  replicas: 2
  revisionHistoryLimit: 10
  selector:
    name: postgresql
  strategy:
    activeDeadlineSeconds: 21600
    recreateParams:
      timeoutSeconds: 600
    resources: {}
    type: Recreate
  template:
    metadata:
      annotations:
        com.illumio.role: Database
        openshift.io/generated-by: OpenShiftNewApp
      creationTimestamp: null
      labels:
```

Below is the final outcome of the label assignment from the example.

Policy State	Policy Sync	Namespace	Name	Role	Application	Environment	Location
Build	Active	default	registry-console-1-r85jf	Control	Openshift Infrastructure	Development	HQ
Build	Active	kube-service-catalog	apiserver-bqf5g	Control	Openshift Infrastructure	Development	HQ
Build	Active	kube-service-catalog	controller-manager-x4vb2	Control	Openshift Infrastructure	Development	HQ
Build	Active	default	docker-registry-6-2jfcn	Control	Openshift Infrastructure	Development	HQ
Build	Active	openshift-template-service-broker	apiserver-x8bx7	Control	Openshift Infrastructure	Development	HQ
Build	Active	kube-system	illumio-kubelink-554688b759-k2mxt	Kubelink	Openshift Infrastructure	Development	HQ
Build	Active	openshift	postgresql-1-2v99p	Database	ShoppingCart	Development	HQ
Build	Active	openshift	shopping-cart-web-1-shgbc	Web	ShoppingCart	Development	HQ
Build	Active	openshift	shopping-cart-web-1-ljq78	Web	ShoppingCart	Development	HQ
Build	Active	openshift	shopping-cart-web-1-ghv2t	Web	ShoppingCart	Development	HQ
Build	Active	openshift	postgresql-1-qbl6z	Database	ShoppingCart	Development	HQ

Daemonsets and Replicasets

The steps above apply only to services in OpenShift which are bound to `deployment` or `deploymentconfig`. This is due to the Kubelink's dependency on pod hash templates which daemonset and replicaset configurations do not have. If you discover pods derived from daemonset or replicaset configurations and also discover services bound to those pods, then Kubelink will **not** automatically bind the virtual service and service backends for the PCE. The absence of this binding will create limitations with Illumio policies written against the virtual service. To get around this limitation for daemonsets and replicasets follow the steps below.

1. Log into the CLI of any OpenShift node and generate a random uuid using the `uuidgen` command.
2. Copy the output of the `uuidgen` command.
3. In the OpenShift web console, navigate to the configuration of the daemonset or replicaset and edit the YAML file.

4. Find the `spec: > template: > metadata: > labels:` field in the YAML and create field called `pod-template-hash:` under the `labels:` section.
5. Paste the new uuid to the value of the `pod-template-hash:` field.
6. Save the changes.

Repeat steps 1 through 6 for each daemonset or replicaset configuration.

See screenshots below for DaemonSet or ReplicaSet reference.

```
[root@master ~]# cat nginx-ds.yaml
apiVersion: extensions/v1
kind: DaemonSet
metadata:
  name: nginx-webserver
spec:
  template:
    metadata:
      annotations:
        com.illumio.pairing_key: 4229fb4a718c62861e1113974958011206 8b35394639954eac02a1395c87888e307d72676fc0
      labels:
        app: nginx-webserver
        pod-template-hash: be85a690-613a-4b24-a7f7-5765befbe11d
    spec:
      containers:
        - name: webserver
          image: rstarmer/nginx-curl
          imagePullPolicy: IfNotPresent
          ports:
          - containerPort: 80
```

NEN Installation and Usage

Introducing the Illumio Network Enforcement Node

This section provides an overview of how the NEN integrates with network devices and presents the new features across several releases.

Overview of the NEN

This section describes the situations where installing an agent (the Illumio VEN) on a device is not possible and how to workaround it by using the NEN.

When Installing a VEN Isn't Possible

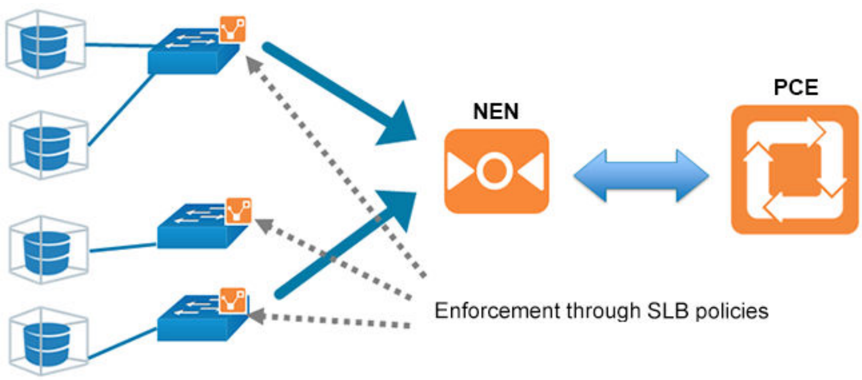
Visibility of communication across applications is critical for segmentation. The optimal method of getting visibility is to use a lightweight agent or a VEN, to report all inbound and outbound communications for each workload.

However, in certain cases a VEN cannot be installed on special purpose systems that provide services to application workloads; for example, IBM Mainframes, NetAPP Filers, legacy Windows machines, or appliances. In other cases, the VEN could be installed on a workload but customers choose not to; for example, installing a VEN might void the vendor's support agreement or the workload is sensitive to latency because it is a high transaction server.

How the NEN Integrates with Network Devices

In cases where a VEN cannot be installed, the NEN extends visualization capabilities to agentless workloads via the network. The NEN is installed as part of an Illumio Core deployment and paired with a PCE. Every IP address associated with the network endpoints managed by the NEN has one workload or virtual server associated with it. The NEN can manage multiple endpoints and enforce policy for those endpoints.

This guide describes how to integrate the NEN with supported load balancers (SLBs) and switches.



Using the NEN, Illumio Core enforces policy on the nearest point to the workload, either:

- A virtual Server on a load balancer in front of the workload
- A switch port on a router in front of the workload

The NEN receives generic policy from PCE and generates policy appropriate to the managed network devices:

- **SLBs:** Firewall policy; for example, the F5 load balancer has two variants of applying policy: AFM and LTM
- **Switches**: ACLs

Until a NEN is paired to the PCE, the switch and load balancer features are deactivated. Using the PCE web console, Illumio users associate unmanaged workloads to the network device endpoints. The NEN syncs its configuration with the PCE every 1 minute. For switch devices, the NEN can be configured to receive traffic flow information from the managed network devices and provide Illumination data to the PCE.

Currently, the NEN does not configure network devices automatically. Network device management has to be done by the user. This process includes applying generated policy by using the Illumio REST API. For information about applying policy to switches, see Apply Policy for Switches [436] and NEN Switch Configuration Using REST API [430]. For information about applying policy for SLBs, see Write SLB Policy [416].

What's New in the Releases

This section describes new features introduced in the following NEN releases.

NEN 2.6.1 New Feature

Support for all Citrix ADC (Netscaler) Load Balancer-supported protocols

With this release, the NEN now supports all of the protocols that Citrix (NetScaler) 13.1 lists in the **Load Balancing > Virtual Servers > Add > Protocol** menu.

NEN 2.6.0 New Features

Support for Citrix ADC (Netscaler) Load Balancer

With this release, the NEN now supports Citrix ADC (Netscaler) Load Balancers and their associated virtual servers that have only a single IPv4 address.

To add a Citrix Software Load Balancer, see *Configure Load Balancers* in Load Balancers and Virtual Servers for the NEN [409].

Support for allowing customers to specify whether disabled VIPs are reported to the PCE

Prior to the release of NEN 2.6.0, if VIP filtering was disabled, all VIPS – including disabled VIPs – were reported to the PCE. You can now disable this reporting using the following new option in the illumio-nen-ctl slb-enable command:

```
--disabled-virtual-server-reporting enabled|disabled
```

To ensure backwards compatibility, the default value is `enabled`.

PCE-provided rule IP addresses and ports now combined into CIDR blocks

NENs now combine rule IP addresses and ports provided by the PCE into CIDR blocks and port ranges. This reduces the number of ACLs that NENs need to generate for switches.

Benefits include:

- Fewer ACLs that the NEN generates for switches.
- Fewer ACLs generated for the IBM iSeries integration with Precisely (current limit: 10k ACLs) allows for optimization of IP addresses into ranges larger than can be covered by a single CIDR block.
- Lower demand on switch TCAM where ACLs are stored.

Support for Rocky Linux 8.7

This release includes support for running standalone NENs on Rocky Linux 8.7.

Support for configuring a PCE policy request timeout

Beginning with NEN 2.5.2.A1, you can configure a PCE policy request timeout. This may be needed If your NEN SLB implementation will involve large policy calculations. The timeout ensures that the NEN doesn't wait too long for the PCE to respond to policy requests in scenarios involving large policy calculations.

To configure the timeout, use the following runtime environment variable:

```
pce_policy_request_timeout_minutes
```

- Default value: 10 minutes
- Minimum value: 3 minutes

NEN 2.5.2 New Feature

Support for AVI NSX Advanced Load Balancer version 21.1.4 2p9

With this release, the NEN now supports AVI NSX Advanced Load Balancer version **21.1.4 2p9**.

NEN 2.5.0 New Feature

IMPORTANT

NEN 2.5.0 is compatible with PCE releases prior to Core 22.3.0-PCE; however, to use the new features available with NEN 2.5.0, you must be using a version of Core 22.3.0-PCE (Illumio Cloud customers only) or later. NEN 2.5.0 is not available for Illumio On-Premises Core customers who are running the PCE in their own data centers.

Enforcement Boundary Support

NEN 2.5.0 now supports the Selective Enforcement policy mode of Enforcement Boundaries by generating deny rule ACLs for the Enforcement Boundary IP addresses. Enforcement Boundaries are a security policy model available in the Core PCE for broadly managing communication across a set of workloads, ports, and/or IP addresses. They allow you to define the end state and then the PCE implements an Enforcement Boundary to create the appropriate native firewall rules. For more information about Enforcement Boundaries, see "Enforcement Boundaries" in the Security Policy Guide.

NEN 2.4.10 New Features

Support for discovering pool groups on AVI SLBs

Beginning with this release, NENs can discover – on AVI Server Load Balancers (SLBs) – virtual servers configured with pool groups instead of server pools. Prior to this release, NENs could discover only virtual servers with server pools and ignored pool groups.

Configurable polling interval for discovering new virtual servers

Beginning in release NEN 2.4.10, you can configure how frequently the NEN polls Server Load Balancers (SLBs) to discover new virtual servers (VS). You do this by adding a field to the runtime_env.yml file. In previous releases the timeout value was fixed at 5 minutes, which was too long for some use cases. SLB discovery events are customer-configurable as follows:

- **Default** = 5 minutes. You don't have to modify the runtime environment file if you want to keep the default setting.
- **Minimum** = 2 minutes
- **Maximum** = none

The NEN reads the timeout value at startup and polls SLBs accordingly. If you add this field and/or update the timeout value in the field, you must restart the NEN for the change to take effect.

Procedure

You can modify the runtime environment file on an already-running NEN or when installing a NEN. For details, see Install a New Standalone NEN [400].

1. Locate the NEN runtime environment file in the following directory:

   ```
   /etc/illumio-nen/runtime_env.yml
   ```
2. If it's not already present, add the line slb_discovery_timeout_minutes to the file.
3. Add a space, a colon (:), and value of 2 or higher at the end of the line. For example, to configure the SLB discovery timeout to **3 minutes**, you'd enter:

   ```
   slb_discovery_timeout_minutes: 3
   ```
4. Restart the NEN for the new setting to take effect.

If you've updated the timeout value on an already-running NEN, you're done at this point. If you've configured the timeout value as part of a new NEN installation, continue to NEXT STEPS [390] below.

NEXT STEPS

1. Activate the NEN with a pairing key from the PCE. See Obtain Pairing Key and Activate the NEN [403].
2. To enable the NEN to integrate with a load balancer, see Enable load balancer support [403].
3. (Optional) To configure the NEN as an HA pair, perform the steps in Configure HA Support for the NEN [404].

NEN 2.4.0 New Features

Support for moving SLBs to a different NEN host (single and super cluster)

> **NOTE**
> Requires Core PCE version 22.2.0 or later.

You can move a Server Load Balancer to a different NEN host from the PCE Web Console. This capability preserves – on the moved SLB – all policies already assigned to the managed virtual server.

1. From the PCE Web Console, go to **Infrastructure > Server Load Balancers**.
2. In the **Name** column, click the link for the SLB you want to move.
3. On the **Summary** tab for the selected SLB, click **Edit**.
4. In **NEN hostname**, click the drop down list to select the destination NEN host where you want to move the SLB.
5. Click **Save**. The PCE recognizes that the SLB has been moved to the chosen NEN host.

Support for moving a NEN from one PCE to another PCE

You can move a NEN from one PCE to another PCE in the same supercluster. When a NEN is moved in this way, associated Server Load Balancers maintain policy for managed virtual servers. After the PCE database is restored, the moved NEN remains connected to the new PCE. The command for moving a NEN is:

```
sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl pce-host-update
<pce-host-addr>:<port>
```

Support for using LTPs instead of iRules on the F5 BIG-LTM

You can use Local Traffic Policies (LTP) on the F5 BIG-IP-LTM. This support is provided in addition to existing support for using iRules.

> **IMPORTANT**
> If you use this functionality, only use LTP rules. Don't use both LTP and iRules together.

1. From the PCE Web Console, go to **Infrastructure > Server Load Balancers**.
2. Select a NEN host. The **Device Type** field appears.
3. In Device Type, select **F5 Big-IP LTM (LTP)**.

Support for maintaining PCE-managed virtual servers when associated SLB virtual servers are disabled

> **NOTE**
> This applies to IPv4 only. IPv6 is not currently supported.

Beginning with this release, the PCE continues to maintain and display PCE-managed virtual servers even when their associated Server Load Balancer (SLB) virtual servers are disabled. This ensures that the PCE doesn't drop or invalidate policy rules for a managed virtual server if the associated SLB virtual server is temporarily disabled. It also ensures virtual servers that were temporarily disabled receive policy updates when they come back online. Previously, when an SLB virtual server was disabled, the associated PCE-managed virtual server showed up as "deletion pending" even after the SLB virtual server was re-enabled.

Support for Red Hat Enterprise Linux (RHEL) 8

This release includes support for running standalone NENs on RHEL 8.

Support for IBM iSeries

Beginning with this release, it's now possible to generate IBM iSeries firewall policies for the Precisely integration using the PCE's capability to generate switch ACLs. For details, see Generate and Download ACLs [436].

Support for Enabling/Disabling Debug Mode Logging

You can now turn debug mode logging on or off. When enabled, debug mode logging provides detail for the network_enforcement_service. The following command allows you to show the current debug mode node status or turn debug logging mode on or off dynamically:

```
sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl debug-mode status/on/off
[--all-nodes]
```

Faster Checks for Policy Tampering for Managed F5 Virtual Servers

Beginning with this release, the NEN sends fewer API calls to the F5 Advanced Firewall Manager SLB to check for policy tampering on Virtual Servers, resulting in faster checking for policy tampering.

Faster Policy Programming for Managed F5 Virtual Servers

Beginning with this release, the NEN sends fewer API calls to the F5 AFM SLB to program policy for managed F5 Virtual Servers, resulting in faster policy programming.

NEN 2.3.10 New Features

NEN discovery of Virtual Servers with Protocol/Ports ANY/ANY

NENs can now discover Virtual Servers (VS) with protocol type ANY and ports ANY. This functionality was added to support configuring Layer 3 Forwarding VIP where the VIP acts as a gateway for servers. In order for outbound traffic from servers to work, these VIPs must be configured to handle protocol type ANY. Prior to this update, VS discovery was limited to SNAT-enabled VSs, VSs that are members of a server pool, or VSs operating on protocol TCP/UDP. To enable discovery of Virtual Servers (VS) with protocol type ANY and ports ANY, disable virtual server filtering with this command:

```
sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl slb-enable --virtual-
server-filtering disabled
```

Support for IBM iSeries Integration (AS/400)

In this release, the NEN supports PCE integration with IBM iSeries (AS/400) computers running Precisely Assure Security. Although the IBM iSeries is not a switch, you will use the PCE switch integration user interface to perform the integration. For more information, see IBM i Series Integration (AS/400) [434].

Support for Enabling/Disabling Debug Mode Logging

You can now turn debug mode logging on or off. When enabled, debug mode logging provides detail for the network_enforcement_service. The following command allows you to show the current debug mode node status or turn debug logging mode on or off dynamically:

```
sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl debug-mode status/on/off
[--all-nodes]
```

Full support for NEN on Supercluster

NEN 2.3.10 supports environments with large numbers of widely distributed SLBs and Virtual Servers. Whereas NEN 2.1.0 supported installing the NEN only on the 2 database nodes of the Supercluster leader (but not on a standalone system or on non-Supercluster leader nodes), NEN 2.3.10 allows deployment of multiple NENs per Supercluster region. Policy is written centrally, similar to VEN deployments.

Scale

- 200 SLBs across all regions
- 32k VIPs, 32k Virtual Servers across all regions
- 6k VIPs, 6k Virtual Servers per NEN cluster, for 2 HA pairs per Supercluster region

Restrictions

- Support only for the standalone NEN (not installed on PCE data nodes).
- No support for moving NENs from one region to another.
- No support for moving SLBs from one NEN to another.

NEN 2.3.0 New Features

> **IMPORTANT**
> NEN 2.3.0 was a Limited Availability (LA) release. However, these features are also available in NEN 2.3.10.

The NEN 2.3.0 release includes the following features and enhancements.

Reduced Load on F5 Authentication

To reduce the load on the F5 login authentication mechanism, beginning with this release NENs now use F5 token authentication for F5 API calls. Prior to this change, the NEN used basic authentication, which requires the F5 to use the login authentication mechanism to validate every API call. In contrast, token authentication creates a 20 minute window during which the NEN can reuse the token repeatedly for API calls until the token expires. When the token expires, the NEN requests a new token.

Faster Checks for Policy Tampering for Managed F5 Virtual Servers

Beginning with this release, the NEN sends fewer API calls to the F5 Advanced Firewall Manager SLB to check for policy tampering on Virtual Servers, resulting in faster checking for policy tampering.

Faster Policy Programming for Managed F5 Virtual Servers

Beginning with this release, the NEN sends fewer API calls to the F5 AFM SLB to program policy for managed F5 Virtual Servers, resulting in faster policy programming.

NEN 2.2.0 New Features

> **IMPORTANT**
> NEN 2.2.0 was a Limited Availability (LA) release. However, these features are also available in NEN 2.3.10.

The NEN 2.2.0 release includes the following features and enhancements.

Standalone NEN configuration with HA support

The NEN 2.2.0 standalone NEN configuration provides a High Availability (HA) architecture with separate standalone Primary and Secondary nodes sharing the work queue. Either node, if it has capacity, can tackle work in the queue. Both nodes can program any SLB as long as the NEN is up and communicating with the SLB.

Unique duties of each role include:

- **Primary node:** Communicates with the PCE; receives configuration information from the PCE and reconciles it with information in its database; determines the work that is placed in the shared work queue.
- **Secondary node:** If the Primary node can't communicate with the PCE for whatever reason, the Secondary node temporarily assumes the role of Primary until communication between the PCE and the original Primary node is re-established.

NEN critical events automatically reported to the PCE console

The NEN automatically reports status about the following events through the PCE console (**Troubleshooting > Events**).

- High CPU usage
- High memory usage
- Critical disk space utilization
- The PCE logs an event if it hasn't received a heartbeat from the NEN in the preceding 15 minutes

NEN health status reporting available through NEN CLI

You can generate a NEN health status report through a CLI. A NEN health report displays onscreen only.

```
illumio-nen-ctl health
```

NEN support report available through the NEN CLI

To help Illumio Support troubleshoot your implementation, you can generate a NEN support report. A NEN support report is a unique file that includes a health report as well as NEN logs.

```
illumio-nen-ctl support-report
```

NEN host selector available when adding an SLB

When adding or editing an SLB from the PCE console (**Infrastructure > Load Balancers**) the new NEN hostname option allows you to select which NEN you want to manage policy programming for this particular SLB.

Support for UDP virtual servers

NEN 2.2.0 supports managing policy programming on Virtual Servers that utilize the UDP transport protocol.

NEN 2.1.0 New Features

The NEN 2.1.0 release includes the following features and enhancements.

Policy on Both Members of SLB cluster

The policy can be applied to both the configured members of an SLB cluster:

- You can create and update rules on both members of an AFM/LTM cluster, with up to two load balancers.
- Both members must be in sync before informing the PCE that the policy has been applied.

- If only one SLB is available, the operation will fail. You can retry to apply the policy only after both are in sync.
- If one member fails to program the rules, you should not retry.

Remove Filtering of F5 VIPs

You can view all types of Virtual Services configured on F5 load balancers, by running a specific command during the NEN installation. To disable (enabled, by default) the built-in filter running on the NEN on the leader PCE cluster, run the following command:

```
sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl slb-enable
--virtual-server-filtering disabled
```

Manage NEN on Supercluster Leader

For Supercluster deployment, you can install the NEN only on the 2 database nodes of the Supercluster leader. You cannot install on a standalone system or on non-Supercluster leader nodes.

Scale

The NEN 2.1.0 release supports up to 500 VIPs and up to 15 SLBs.

NEN 2.0.0 New Feature

The NEN 2.0.0 release includes support for AVI Vantage load balancers.

NEN Installation and Configuration

This section provides an overview of supported NEN architectures and describes how to install the NEN RPM package standalone hosts as well as upgrade the NEN to the latest version.

About NEN Installation and Architecture

This topic explains how the NEN is installed and the supported architectures.

PCE-based versus Standalone NEN Installation

> **IMPORTANT**
> Beginning in NEN 2.3, the NEN is deployed as a standalone NEN installation only. New PCE-based installations are not supported.

In NEN 2.1.*x*, two types of NEN installations were supported:

- **PCE-based installation**
  You installed the NEN on one of the PCE data nodes so that the NEN ran as a service on the PCE. When you installed the NEN as a service on a PCE data node, you had the option

of installing it on both data nodes (data node 0 and data node 1) so that the NEN operated as a high availability (HA) pair.

- **Standalone NEN installation**

  You installed the NEN on a separate Linux host. When you installed a standalone NEN in NEN 2.1.*x*, you did not have the option to configure the NEN deployment as an HA pair.

Beginning in NEN 2.3, you must install the NEN on a separate Linux host (standalone installation). Installing the NEN on a PCE data node isn't supported beginning with NEN 2.3. The new standalone installation has the following benefits:

- Provides full (optional) HA support for Illumio On-Premises customers and Illumio Cloud customers.
- Allows you to deploy NENs closer to your network devices, namely load balancers and switches.
- Supports higher scale with multiple NEN HA pairs paired to a single PCE cluster.

IMPORTANT

Because NEN releases from 2.3 and later don't support a PCE-based installation, customers with existing installations (NEN 1.0.1 through NEN 2.1.0) must upgrade to NEN 2.3 or later. For information, see Upgrade Standalone NEN 2.1.0 to Standalone NEN 2.3.x or later [404].

NEN High Availability Support

Prior to NEN 2.1.0, when NENs had to be installed on a PCE data node, High Availability (HA) on NENs was achieved by using the PCE's HA capabilities. Beginning with the move to a standalone NEN installation in NEN 2.2.0, the NEN now features full HA support independently of the PCE.

The following diagram illustrates how to plan your NEN installation to provide full HA support by installing it on two Linux hosts (node 1 and node 2). In an HA configuration, the primary NEN performs the following actions:

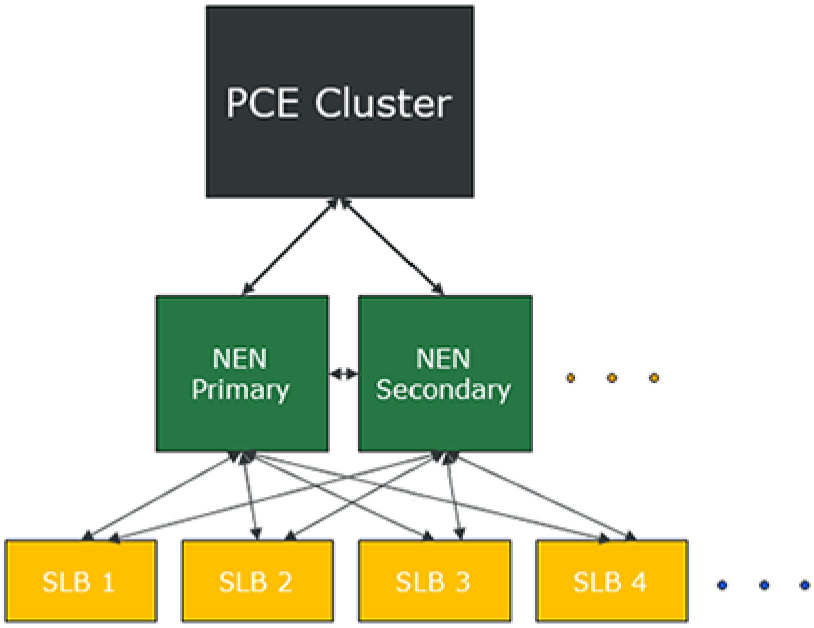
- Retrieves configuration information from the PCE and reconciles it with the PCE database.
- Determines what work needs to go into the work queue for the NEN HA pair.

If the primary NEN (on node 1) loses connectivity to the PCE, the secondary NEN (on node 2) becomes the primary NEN until the NEN on node 1 re-establishes connectivity with the PCE.

NOTE

For hardware requirements in an HA Pair implementation, see CPU, Memory, and Storage Requirements [398] in this topic.

When using the NEN for SLB integration, both NENs (primary and secondary) can program any load balancer because they share the work queue. Either NEN can accept the next job from the work queue depending on their available capacity. This capability is available when the primary NEN has connectivity with the PCE.



A PCE cluster supports multiple NENs per PCE, which can consist of multiple single node NENs, multiple NEN HA pairs, or a combination of both.

NEN Supercluster Support

In NEN 2.1.*x* (when installed as part of Illumio Core 20.2.0, 21.1.0, or 21.2.*x*), Illumio provided limited support for the NEN with PCE Supercluster deployments. For information see, Manage NEN on Supercluster Leader [395] in "NEN 2.1.0 New Features." NEN releases prior to 2.1.0 did not include Supercluster support.

NEN 2.3.10 extended support for installing a NEN within a PCE Supercluster as follows:

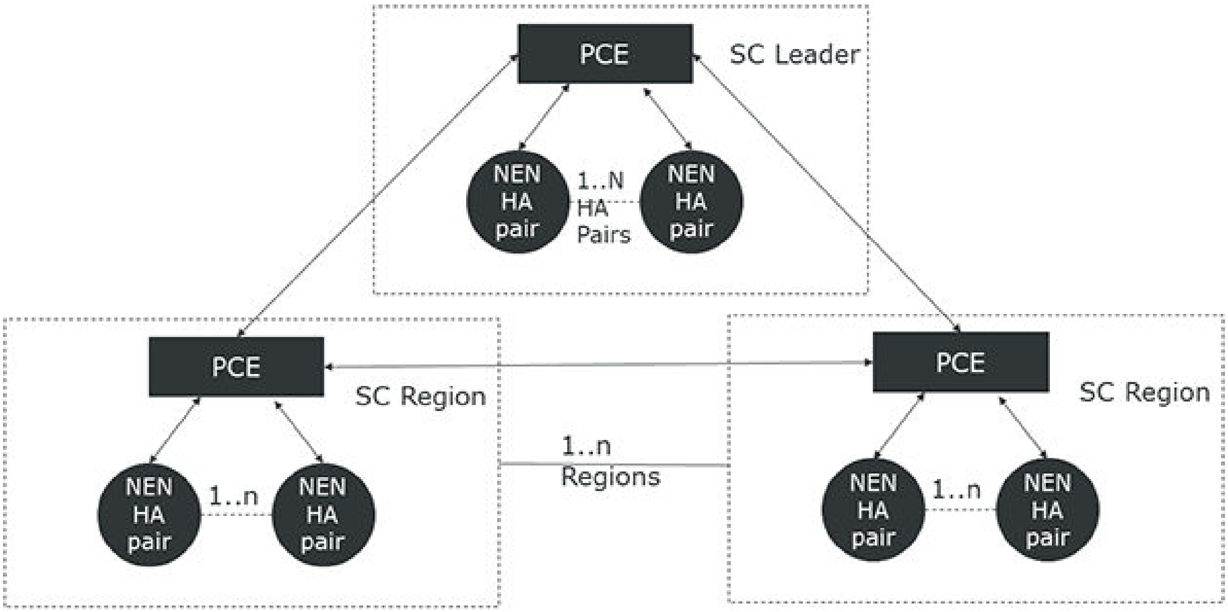
- **NEN Installation on Supercluster Members**
  You can pair the NEN to the other regions in the Supercluster; referred to as Supercluster "members." Prior to NEN 2.3.10, you could only install the NEN on the Supercluster leader. For more information about PCE Supercluster deployment architecture, see "Design Supercluster Deployment" in the PCE Supercluster Deployment Guide.

  > **CAUTION**
  > Plan your NEN installation carefully when you install it as part of a PCE Supercluster deployment. Once installed, you cannot move NENs from one PCE Supercluster member to another member.

- **Multiple NEN HA Pairs in a Supercluster Member**
  Depending on your scale requirements and the location of your network devices (such as SLBs), you can connect multiple NEN HA pairs to any cluster in a PCE Supercluster deployment (not just the PCE Supercluster leader). This enhancement is necessary to support environments with large numbers of SLBs and virtual servers that are geographically distributed.

NOTE
At a minimum, you must install a primary and secondary NEN HA pair in one of the Supercluster regions.

The following diagram illustrates how to plan your NEN installation in a PCE Supercluster deployment:



CPU, Memory, and Storage Requirements

This section presents hardware requirements for supporting SLBs and switches.

Hardware requirements to support SLBs and VIPs

To install NEN(s) to support a given number of server load balancers and Virtual IPs, your hardware must meet the hardware requirements detailed in this section.

Server Load Balancers (SLBs)	Virtual IPs (VIPs)	Cores/Clock Speed[1]	RAM per Node[2]	Storage Device Size[3] and IOPS[4]	Network
Up to 6 SLBs	• Max 1,000 VIPs per SLB • Max 3,000 VIPs across all SLBs	• 2 cores • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or equivalent	8 GB	A single node including both core and data: • 1 x 50 GB • 100 IOPS per device	1 Gb Ethernet
Up to 50 SLBs	• Max 1,000 VIPs per SLB • Max 12,000 VIPs across all SLBs	• 4 cores • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or equivalent	16 GB	A single node including both core and data: • 1 x 50 GB • 100 IOPS per device	1 Gb Ethernet

NEN HA implementation considerations:

The two nodes in an HA pair must match the size specified in this table. This is necessary because in the event of a failover, a single node must be able to manage the entire load.

- For an HA Pair for up to 3000 VIPs, use the sizing detailed in the first row.
- For an HA pair for 3000+ VIPs, use the sizing detailed in the second row.

Footnotes:

[1] CPUs:

- The recommended number of cores is based only on physical cores from allocated CPUs, irrespective of hyper-threading or virtual cores. For example, in AWS one vCPU is only a single hyper-thread running on a physical core, which is half a core. 16 physical cores equates to 32 vCPUs in AWS.
- Full reservations for vCPU. No overcommit.

[2] Full reservations for vRAM. No overcommit.

[3] Additional disk notes:

- Storage requirements for network traffic data can increase rapidly as the amount of network traffic increases. Allocating a separate, large storage device for traffic data can accommodate these rapid changes without potentially interrupting the service.
- Network File Systems (NFS) is not supported.

[4] Input/output operations per second (IOPS) are based on 8K random write operations. IOPS specified for an average of 300 flow summaries (80% unique src_ip, dest_ip, dest_port, proto) per workload every 10 minutes. Different traffic profiles might require higher IOPS.

Install and Activate the NEN

This section describes how to:

- Install and activate a new standalone NEN deployment
- Upgrade a PCE-based NEN installation to the standalone NEN installation required for NEN 2.3.x and later.

Illumio recommends that you have the following knowledge before installing and administering the NEN:

- Your organization's security goals.
- Thorough understanding of Illumio Core.
- When integrating the NEN with your organization's load balancers and switches, how to configure and manage these network devices.

NEN Software

For the complete list of OS support for the NEN, see NEN OS Support and Package Dependencies on the Illumio Support portal.

To download the NEN software:

1. Log into the Illumio Support portal and go to **Software > NEN** .
2. From the **Download NEN Software** page, select the latest version.
3. Click the filename in the table to download the software locally.

(Optional) Configure a PCE policy request timeout

Beginning with NEN 2.5.2.A1, you can configure a PCE policy request timeout. This may be needed If your NEN SLB implementation will involve large policy calculations. The timeout ensures that the NEN doesn't wait too long for the PCE to respond to policy requests in scenarios involving large policy calculations.

To configure the timeout, use the following runtime environment variable:

```
pce_policy_request_timeout_minutes
```

- Default value: 10 minutes
- Minimum value: 3 minutes

Install a New Standalone NEN

> **NOTE**
> This procedure describes how to perform a **new** NEN standalone installation where you have **not** previously installed the NEN as a service on a PCE data node or you have not installed the NEN 2.1.0 standalone service on your own host.

- For the steps to upgrade standalone NEN 2.1.0 to standalone NEN 2.3.x or later, see Upgrade Standalone NEN 2.1.0 to Standalone NEN 2.3.x or later [404].
- For the steps to upgrade a previously-installed PCE-based NEN to NEN 2.3.x or later, see Upgrade a PCE-based NEN 2.1.0 to a Standalone NEN 2.3.10 [406].

To install a NEN as a standalone NEN:

> **NOTE**
> For standalone NEN hardware requirements, see CPU, Memory, and Storage Requirements [398].

1. Download the NEN software from the Illumio Support portal.
2. Run the following command to install the NEN RPM on the host:

```
sudo yum install -y <path_to_Illumio_NEN_rpm>/illumio-nen-
<release_number>
-<build_number>.x86_64.rpm
```

3. Configure the NEN runtime environment settings in **one** of the following ways:
   - By running the NEN `setup` command to launch an interactive installation and answering the prompts to configure the NEN runtime environment. (This method creates the NEN runtime environment file and saves it in the correct NEN directory.)
   - By copying a template of the NEN runtime environment file to the required location and then modifying that file

To perform an interactive installation:

a. Enter the following command to start the installation and run the environment set up:

```
sudo /opt/illumio-nen/illumio-nen-env setup
```

b. Complete the installation by providing the values at the prompts.

To modify the template runtime environment file:

a. Copy the NEN runtime environment file from:

```
/opt/illumio-nen/illumio/config/templates
```

b. Paste it to:

```
/etc/illumio-nen/runtime_env.yml
```

c. Update the file with the host FQDNs and service discovery certificate information.

> **IMPORTANT**
> A standalone NEN cannot communicate with the PCE by using a self-signed service discovery certificate. The NEN requires an X.509 public certificate in PEM format for TLS communication with the PCE.

```
# Configuration generated <timestamp>
```

```
install_root: “/opt/illumio-nen”

runtime_data_root: “/var/lib/illumio-nen/runtime”

persistent_data_root: “/var/lib/illumio-nen/data”

ephemeral_data_root: “/var/lib/illumio-nen/tmp”

log_dir: “/var/log/illumio-nen”

private_key_cache_dir: “/var/lib/illumio-nen/keys”

nen_fqdn: <example.com>

service_discovery_fqdn: <example.com>

cluster_type: snc0

service_discovery_private_key: “/var/lib/illumio-nen/cert/server.key”

service_discovery_certificate: “/var/lib/illumio-nen/cert/server.crt”

service_discovery_encryption_key: <key>
```

Where:

- `nen_fqdn` is the hostname of the node where the NEN is installed.
- `service_discovery_fqdn` is the hostname of the NEN FQDN.
- `service_discovery_private_key` is the directory path of the RSA private key file.
- `service_discovery_certificate` is the directory path of the certificate file.
- `service_discovery_encryption_key` is a 16 byte hexadecimal base-64 encoded value

When adding the encryption key to the template runtime environment file, you create your own value. However, if you are using the interactive NEN installation, the NEN CTL `setup` command automatically creates this value in the file.

> **NOTE**
>
> Beginning in NEN release 2.4.10, you can add a field to the runtime environment file to configure how frequently the NEN polls Server Load Balancers (SLBs) to discover new virtual servers (VS). For details, see Load Balancers and Virtual Servers for the NEN [409].

4. Start the NEN and set the runlevel to 5. The option `-svw` shows the status of the start operation.

```
sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl start --runlevel 5 -svw
```

NEXT STEPS

1. Activate the NEN with a pairing key from the PCE. See Obtain Pairing Key and Activate the NEN [403].
2. To enable the NEN to integrate with a load balancer, see Enable Load Balancer Support [403].
3. (Optional) To configure the NEN as an HA pair, perform the steps in Configure HA Support for the NEN [404].

Obtain Pairing Key and Activate the NEN

When the NEN is installed as part of a NEN HA pair, you only pair the NEN primary node with the PCE.

1. Log into the PCE web console.
2. From the left navigation menu, choose **Workloads and VENS > Workloads**.
3. Click **Add > Pair Workload with Pairing Profile**.
4. Select any existing pairing profile from the “Pick a Pairing Profile” drop-down menu.
5. Copy the pairing **Key** value (alphanumeric).
6. Log in to the NEN host and run the `illumio-nen-ctl activate` command:

```
sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl activate
<pairing_key_value>
--host <pce-address>:<pce-port>
```

Enable load balancer support

After installing the NEN RPM and activating it with the PCE, enable load balancer support by running the following command on the NEN node:

> **NOTE**
> If the NEN is configured as an HA pair, run this command on the primary node.

```
sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl slb-enable
```

Move a NEN from one PCE to another PCE

You can move a NEN from one PCE to another PCE in the same supercluster. When a NEN is moved in this way, associated Server Load Balancers maintain policy for managed virtual servers. After the PCE database is restored, the moved NEN remains connected to the new PCE. The command for moving a NEN is:

```
sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl pce-host-update
<pce-host-addr>:<port>
```

Upgrade Standalone NEN 2.1.0 to Standalone NEN 2.3.x or later

Keep in mind that if you perform this procedure, you **don't** need to:

- Restore the NEN database because the NEN upgrade doesn't impact it.
- Activate the NEN with the PCE if you are upgrading an existing NEN installation; that is, if you are upgrading a NEN 2.1.0 standalone installation to a NEN 2.3.x or later standalone installation.

1. Run the following upgrade command:

   ```
   sudo yum update -y <path to Illumio NEN rpm>/illumio-nen-<release_number>
   -<build-number>.x86_64.rpm
   ```
2. Enable load balancer support by running the following command on the NEN node:

   > **NOTE**
   > If the NEN is configured as an HA pair, run the command on the primary node.

   ```
   sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl slb-enable
   ```
3. (Optional) To configure an HA pair for the NEN in the PCE cluster, see Configure HA Support for the NEN [404]. (The steps are the same whether you are installing a new standalone NEN or upgrading an existing NEN.)

Configure HA Support for the NEN

This optional procedure describes how to install the NEN on a secondary node to provide HA support for the NEN in a PCE cluster. For information about running the NEN as an HA pair, see NEN HA Support. [396] For information about upgrading nodes in an HA pair, see Upgrade a NEN HA pair [405].

Prerequisites for HA support

- You have already installed the NEN on the primary node.
- `service_discovery_fqdn` must be the hostname or IP address of the primary node.
- Network latency between DB nodes must not exceed 10ms.
- `nen_fqdn` in the `runtime_env.yml` file:
  - Both nodes must have the same `nen_fqdn` so that the PCE knows they are part of the same NEN HA pair.
  - The `nen_fqdn` can be anything you choose as long as it is unique among NEN clusters paired to the PCE.
  - The `nen_fqdn` doesn't need to match the actual hostname of either node nor be resolvable via DNS.
- Each NEN node's actual hostname must be resolvable from the actual hostname of the other node in the pair.

> **NOTE**
> You cannot change the `nen_fqdn` once the NEN has been paired to the PCE.

To set up a NEN HA Pair

1. Install the NEN on the secondary node:

   ```
   sudo yum install -y <path_to_Illumio_NEN_rpm>/
   illumio-nen-<release_number>-<build_number>.x86_64.rpm
   ```
2. Set up NEN runtime environment on the secondary node using one of the following methods:
   - Copy `/etc/illumio-nen/runtime_env.yml` file from primary node to `/etc/illumio-nen/runtime_env.yml` on the secondary node, and change the node_type: value to `network_enforcement1`
   - `sudo /opt/illumio-nen/illumio-nen-env setup`.
3. Start the NEN on the secondary node:

   ```
   sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl start
   ```

Upgrade a NEN HA pair

To upgrade the nodes in a NEN pair, you must do so in the proper sequence when the nodes are in the proper state.

Before you begin

- A rolling upgrade is not supported. Perform the upgrade in the order described in the steps below.
- Make sure that the nodes can communicate with each other (that is, that the network connection between them is up). The nodes need to be able to share the same database information. This is to avoid a "split brain" state where both nodes can communicate with the PCE but not with each other.

To upgrade a NEN HA pair:

1. Stop the secondary NEN node.
2. Stop the primary node.
3. Upgrade the primary NEN node.
4. Wait for the primary NEN node to be online (in the RUNNING state).
5. Upgrade the secondary NEN node.

Upgrade a PCE-based NEN 2.1.0 to Standalone NEN 2.3.x or later

If you are upgrading Illumio Core to 21.5.0-PCE or later, you must upgrade the NEN to 2.3.x or later. Illumio Core 21.3.0-PCE is not backwards compatible with NEN 2.1.0 and earlier releases.

Upgrade prerequisites

Before taking the NEN database back up, ensure that no asynchronous jobs have been submitted right before you begin the upgrade. As a best practice, wait until all asynchronous jobs have finished before upgrading the PCEs and associated NENs.

> **NOTE**
>
> When to back up the NEN database and uninstall the NEN software from the PCE
>
> You must back up the NEN database and uninstall the NEN RPM from your PCE-based NEN installation before you upgrade to Illumio Core 21.3.0-PCE and later. Be aware that you must set the PCE to runlevel 1 before backing up the NEN database on the PCE primary data node and uninstalling the NEN RPM from both PCE data nodes.

Notes about the upgrade

- Upgrading the NEN in a single PCE cluster versus a PCE Supercluster deployment

  The steps to install and configure a NEN in a PCE Supercluster deployment are the same as for a single PCE cluster. You perform the procedure to install a NEN in each individual region (PCE Supercluster members).
- Restoring the NEN database in a Supercluster deployment

  When upgrading the NEN that is part of a PCE Supercluster deployment, restore the NEN database from the PCE-based installation; you must restore he NEN database on the NEN paired to the PCE Supercluster leader. You do not need to restore the database for the NENs paired with the PCE Supercluster members.
- When to back up the NEN database and uninstall the NEN software from the PCE

  You must back up the NEN database and uninstall the NEN RPM from your PCE-based NEN installation before you upgrade to Illumio Core 21.3.0-PCE and later. You must set the PCE to runlevel 1 before backing up the NEN database on the PCE primary data node and uninstalling the NEN RPM from both PCE data nodes.

Upgrade NEN 2.1.0 PCE-based installation to NEN 2.2.0 and later standalone installation

1. Back up the NEN database on the PCE primary data node.

   For the requirements and syntax to run PCE commands, see “PCE Control Interface and Commands” in the *PCE Administration Guide*.

```
sudo -u ilo-pce illumio-pce-ctl set-runlevel 1
sudo -u ilo-pce /opt/illumio-pce/illumio-nen-db-management dump
--file <filename>
sudo -u ilo-pce /opt/illumio-pce/illumio-pce-ctl stop
```

2. Uninstall the NEN from the PCE data node(s).

   ```
   sudo rpm -e illumio-nen
   ```
3. Upgrade the PCE to Illumio Core 21.3.0-PCE and later.

   For the steps to upgrade a single PCE cluster, see "Upgrade the PCE" in the *PCE Installation and Upgrade Guide*.

   For the steps to upgrade the PCEs in a PCE Supercluster deployment, see "Upgrade Supercluster" in the *PCE Supercluster Deployment Guide.*

   After upgrading your single PCE cluster or PCE Supercluster, ensure that all PCEs are started at runlevel 5 before installing the NEN RPM package.
4. Install and configure the NEN software. See Install a New Standalone NEN [400]. (This is the procedure for installing a new NEN standalone installation, but the steps are the same whether you are installing a new standalone NEN or upgrading an existing NEN. In the NEN upgrade procedure, you will have uninstalled the previous NEN by this step and must install the new NEN release.)
5. Set the NEN to **runlevel 1** and restore the NEN database that you copied from the PCE primary data node:

   ```
   sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl set-runlevel 1
   sudo -u ilo-nen /opt/illumio-nen/illumio-nen-db-management restore
   --file <path to file to restore>
   ```

   If you are performing this step for a NEN that is part of a PCE Supercluster deployment, restore the NEN database for the NEN node paired with the PCE Supercluster leader. You don't need to restore the NEN database in each Supercluster member region.
6. Set the NEN to runlevel 5 and activate the NEN with a pairing key from the PCE by using the --repair option:

   ```
   sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl set-runlevel 5
   sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl
   activate <pairing-key>
   --host <PCE_host address:port> --repair
   ```

   If you are performing this step for a NEN that is part of a PCE Supercluster deployment, repair the NEN with the Supercluster leader PCE. Additionally, pair any new NEN installations in each region with the Supercluster member in that region.

   For the steps to obtain a pairing key from the PCE, see Obtain Pairing Key and Activate the NEN [403].
7. To configure an HA pair for the NEN in the PCE cluster, see Configure HA Support for the NEN [404]. (The steps are the same whether you are installing a new standalone NEN or upgrading an existing NEN.)

Generate NEN Reports

You can generate NEN health and support reports as well as enable/disable debug mode logging. These provide information useful for troubleshooting issues with your NENs.

Health Report

- Appears onscreen only
- Includes the following information

```
Cluster Mode:   HA Pair
Cluster Status: Normal
Available Time: 0d 4h 27m 24s
PCE FQDN:

Nodes:
------
Hostname                    :
Ip Addr                     :
Node Type                   :        network_enforcement0          network_enforcement1
Runlevel                    :                           5                             5
Report Time                 :   2021-07-30T18:10:04+00:00     2021-07-30T18:10:00+00:00
Node Available Time         :   2021-07-30T13:43:00+00:00
Uptime                      :         21 days 13:09 hours           21 days 13:08 hours
Service Status              :                     RUNNING                       RUNNING
Database Master             :                        true
Cpu                         :                0%  (Normal)                  5%  (Normal)
Disk                        :                8%  (Normal)                  6%  (Normal)
Memory                      :               31%  (Normal)                 27%  (Normal)

Services:
---------
Database Service            :                     RUNNING
Database Slave Service      :
Network Enforcement Service :                     RUNNING                       RUNNING
```

To generate a NEN health report only:

1. Establish a secure shell connection (SSH) to the NEN you want to investigate
2. Issue the following command:

   `sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl health`

Support Report

A Support Report is a unique generated file saved to the /tmp directory. It includes the following information and data:

- A Health Report (see the image above)
- NEN logs

To generate a NEN support report:

1. Establish a secure shell connection (SSH) to the NEN you want to investigate.
2. Run the following command:

   `sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl support-report`
3. On successful completion, the command indicates where you can find the file so that you can copy the support report off the NEN.

Debug Mode Logging

You can turn debug mode logging on or off. When enabled, debug mode logging provides detail for the network_enforcement_service. The following command allows you to show the current debug mode node status or turn debug logging mode on or off dynamically:

```
sudo -u ilo-nen /opt/illumio-nen/illumio-nen-ctl debug-mode status/on/off
[--all-nodes]
```

NEN Integration with Load Balancers

This section describes how to create security policy and apply those policies on the load balancers for use with the NEN.

Load Balancers and Virtual Servers for the NEN

Illumio Core supports activation of enforcement on a number of load balancers as listed below.

Supported Load Balancers

- F5 BIG-IP 11.5x or later
- AVI Vantage 18.23 or later
- Citrix ADC (NetScaler) 13.1 or later

Load Balancer and Virtual Server Concepts

- **Load balancer (SLB):** Either a physical machine or a virtual machine performing load balancing functions. An SLB object represents a standalone device or an HA Pair and includes management of IP/port, user/password, and so on. These values are used by an Illumio NEN to read information from and manage the device. In case of HA, it may include multiple SLB devices.
- **Illumio Virtual Server:** The same as a load balancer Virtual Server.
- **Discovered Virtual Server:** An Illumio NEN queries the load balancer for VIPs and specifies the client-facing VIP with port + protocol combination.
- **Created Virtual Server:** Is a provisionable policy object with labels used in policy writing. In the UI, the Virtual Server creation process is called VIP Management. Virtual Server providers (backend servers) are specified using labels and can optionally specify backend port independently of the port used by the VIP.
  - **VIP:** Is a virtual IP or a local IP (a front-end IP that clients can connect to).
- **SNAP pool:** Is a group of IPs that the Virtual Servers use to connect to the backend servers. A Virtual Server can only have a single VIP connected to it, on a single port. It can also be accessed by the SLBs local IPs.

About Load Balancers

Illumio Core supports activation of enforcement on a number of load balancers listed below.

> **IMPORTANT**
>
> Beginning with Illumio Core 19.3.0 release, the Network Function Controller (NFC) is no longer supported. The F5 interface has been moved from the PCE in to the Network Enforcement Node (NEN). Because the NFC has been discontinued, you need to use the NEN to interface with Load Balancers.

By applying labels to your load balancer's virtual servers, you can write rules that allow client workloads in front of the load balancer to communicate with the virtual IP address of the load balancer's virtual servers. By adding labels to the pool members behind a virtual sever, you can allow communication from the load balancer to the members of the pool. The source for this communication is determined by the load balancer. The Illumio Core programs policies on the load balancer to enforce security policy. The NEN uses the load balancer's REST APIs to read and write security policies to configure security rules.

The PCE supports configuration of two load balancer units if they are configured in Active/Standby or Active/Active modes. The PCE needs to be configured with the user name and password of an administrative user who has read-write access to all configurations on the load balancer.

The NEN configures new objects on the load balancer and does not alter any existing configurations. When an Illumio-created object in the load balancer configuration is modified, the NEN detects it as tampering and modifies the configuration back to the intended state so that the correct security policy is enforced.

The Illumio Core dynamically adjusts policies on the load balancer based on application and topology changes in the datacenter so that the Illumio Core can enforce consistent security policy on load balancers across the datacenter and cloud environments, as well as show the application traffic in Illumination. The Illumio Core keeps track of the policy it programmed and reconfigures policy if it was altered on the load balancer manually or by other means.

The NEN makes use of the following constructs on load balancers:

- **F5 LTM:** iRules or LTP policies on the LTM provide capability to restrict application access. When the LTM is used as enforcement mechanism, the NEN uses virtual-server based iRules/LTP policies and Datagroup Lists.
- **F5 AFM:** AFM provides stateful firewalling on BIG-IP. When AFM is used as an enforcement mechanism, the NEN uses Network Firewall policies in the virtual server section and address-lists in the network firewall. The NEN also supports the F5 BIG-IP Application Services 3 Extension (referred to as BIG-IP AS3) when it is used to define virtual servers on the F56 AFM.
- **AVI:** The NEN uses the Network Security Policy rules to program AVI Vantage.
- **Citrix ADC (NetScaler):** The NEN uses responder policies to control access to the Virtual Servers.

> **NOTE**
> Configuring two SLB units in Active/Standby mode is supported. However, clustering is **not supported**.

F5 BIG-IP Requirements

The NEN uses its REST API to program F5 load balancers, which means that F5 needs to be running a software version that supports REST-API. The requirements include:

- BIG-IP 11.5.x or higher
- Utilize SNAT or Auto-map mode

AVI Vantage Requirements

- AVI Vantage 18.2.3 or higher

Citrix ADC (NetScaler) Requirements

- Citrix ADC 13.1 or higher

Configure Load Balancers

You can add a load balancer using the PCE web console. However, before you add a load balancer, you need to pair the NEN with the load balancer functionality enabled with the PCE.

> **NOTE**
> A load balancer does not need to be provisioned to work. However, the virtual servers you associate with this load balancer do need to be provisioned.

Add an SLB from the PCE Web Console

You can add a load balancer using the PCE Web Console. However, before you add a load balancer, you need to pair the NEN with the load balancer functionality enabled with the PCE.

1. From the PCE Web Console menu, choose **Infrastructure** > **Load Balancers**.
2. Click **Add**.
3. Specify a name for the load balancer and provide a description.
4. From NEN hostname, select the NEN that you want to manage policy programming for this particular SLB.
5. From Device Type, select appropriate load balancer device type.
6. From number of devices, select **(1) Standard** or **(2) HA Pair**.
   The load balancer details are displayed.
7. Specify the following settings to enable the PCE to connect to the load balancer:
   - Management IP address or FQDN of the load balancer
   - Port on which to connect
   - Username
   - Password
8. Select **Verify TLS** to verify the trust of the TLS certificate provided by the load balancer before connecting to it.
9. Click **Save**.

Move an SLB to a different NEN host (single and super cluster)

> **NOTE**
> Requires Core PCE version 22.2.0 or later.

You can move a Server Load Balancer to a different NEN host from the PCE Web Console. This capability preserves – on the moved SLB – all policies already assigned to the managed virtual server.

1. From the PCE Web Console, go to **Infrastructure** > **Server Load Balancers**.
2. In the **Name** column, click the link for the SLB you want to move.
3. On the **Summary** tab for the selected SLB, click **Edit**.
4. In **NEN hostname**, click the drop down list to select the destination NEN host where you want to move the SLB.
5. Click **Save**. The PCE recognizes that the SLB has been moved to the chosen NEN host.

About Virtual Servers

Virtual servers in the Illumio Core contain two parts:

- A virtual IP address (VIP) and port through which the service is exposed
- Local IP address(es) used to communicate with backend servers (pool members).

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 virtual server is identified by a set of labels. The sources (aka consumers) and destinations (aka providers) for a virtual server can be assigned different labels, which could place them in the same group or a different group in Illumination. See **Groups in Illumination** in the Visualization Guide for information.

Providers are allowed to have an incomplete label set (for example, only a Location label), so the providers can be in all groups within the specified location. As a result, a single virtual server can have providers in any group or in any number of groups in Illumination.

Virtual Server Members and Labels

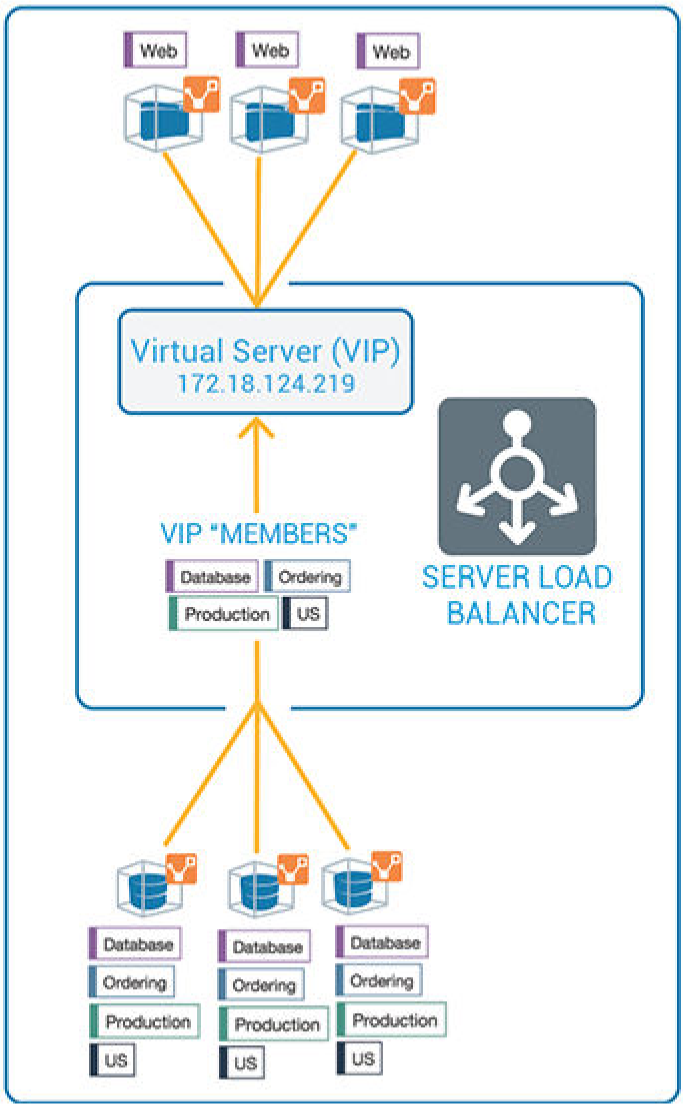
The Illumio Core allows you to write rules to allow communication with workloads managed by a load balancer using labels.

Virtual Server Members

When you configure load balancers in the PCE, it connects to the load balancer using the Illumio Core REST API. The PCE reads all the load balancer virtual servers configurations and populates the Discovered Virtual Servers tab of a load balancer's details page. Any virtual servers associated with the load balancer can be converted to a managed virtual server for use with the PCE. When you configure the virtual server in the PCE web console, you can apply labels to the virtual servers. After configuring a virtual server, you can write a rule that allows other clients to communicate with it.

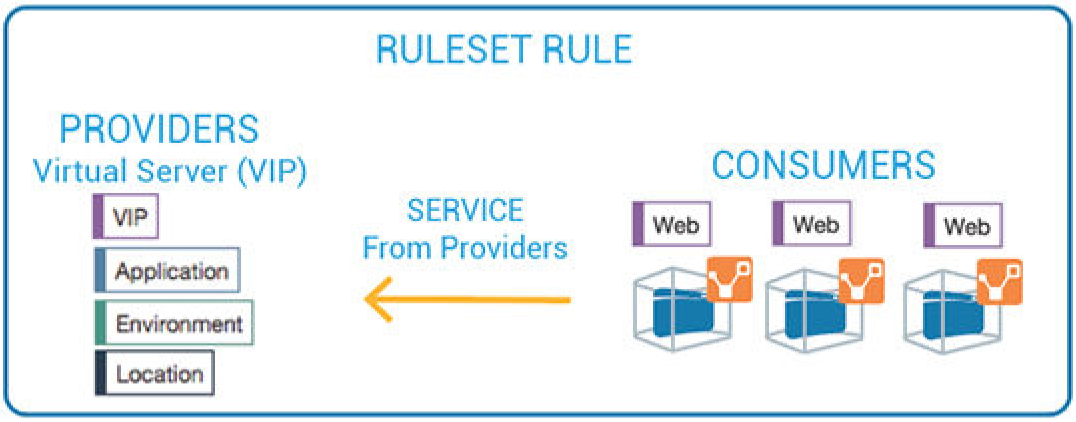
The members behind a virtual server are specified by configuring a set of labels in the virtual server configuration. A set of four Illumio labels can be applied on the Virtual Server Members tab, which is used to match the same set of labels on workloads in the virtual server's pool. If any of the workloads in the virtual server pool share the same set of four labels specified under the Virtual Server Members tab, then any rule you write with the virtual server also applies to the workload members.

In this diagram, you can see how the workloads that belong to the virtual server pool have the same labels specified on the Virtual Server Members tab:



Ruleset Rule for Virtual Server

This diagram illustrates the rule you can write after you label a virtual server and its members:



Configure Virtual Servers

After adding a load balancer to the PCE, you can manage its virtual servers. To each virtual server, you can apply labels through the Virtual Server details page. Applying labels to a virtual server allows you to add the virtual server to a rule.

When the policy is enforced on the virtual server by the NEN, access from any IPs/Workloads to the Virtual Server is controlled according to the rules defined by the policy. The NEN removes the rules from the virtual server when the policy is no longer enforced.

Configuring a load balancer's virtual servers consists of these three settings:

- **Enforced or Not Enforced:** When you select Enforced, any rules you write using the labels associated with the virtual servers and any of its members are enacted. Selecting Not Enforced disables the labels and any policy written that affects the virtual server or its members is disabled.
- **Service:** Select the service to use for the rules that allow access to the virtual server. For example, HTTPD 80 TCP.
- **Labels:** You must apply one each of the four Illumio labels to the virtual server: Role, Application, Environment, and Location. Assigning labels enables the virtual server to be used in rules.

> **NOTE**
> Virtual servers are considered a security policy item, so any changes to a virtual server configuration must be provisioned before any of those changes take effect and become active.

Virtual Server Limitations

- Illumination does not support location-level and application-level maps.
- If a single SNAT pool is shared between multiple virtual servers, the Illumination map does not render correctly.
- SNAT and Auto-map modes of F5 virtual servers are supported. Transparent mode is not supported.

> **NOTE**
> Before any virtual server configuration can go into effect, you need to provision your changes. See the Security Policy Guide for information.

Configure the polling interval for discovering new virtual servers

Beginning in release NEN 2.4.10, you can configure how frequently the NEN polls Server Load Balancers (SLBs) to discover new virtual servers (VS). You do this by adding a field to the runtime_env.yml file. In previous releases the timeout value was fixed at 5 minutes, which was too long for some use cases. SLB discovery events are customer-configurable as follows:

- **Default** = 5 minutes. You don't have to modify the runtime environment file if you want to keep the default setting.
- **Minimum** = 2 minutes
- **Maximum** = none

The NEN reads the timeout value at startup and polls SLBs accordingly. If you add this field and/or update the timeout value in the field, you must restart the NEN for the change to take effect.

Procedure

You can modify the runtime environment file on an already-running NEN or when installing a NEN. For details, see the "Install a New Standalone NEN" topic.

1. Locate the NEN runtime environment file in the following directory:

   ```
   /etc/illumio-nen/runtime_env.yml
   ```
2. If it's not already present, add the line slb_discovery_timeout_minutes to the file.
3. Add a space, a colon (:), and value of 2 or higher at the end of the line. For example, to configure the SLB discovery timeout to **3 minutes**, you'd enter:

   ```
   slb_discovery_timeout_minutes: 3
   ```
4. Restart the NEN for the new setting to take effect.

If you've updated the timeout value on an already-running NEN, you're done at this point. If you've configured the timeout value as part of a new NEN installation, continue to NEXT STEPS [415] below.

NEXT STEPS

1. Activate the NEN with a pairing key from the PCE. See the "Obtain Pairing Key and Activate the NEN" section.
2. To enable the NEN to integrate with a load balancer, see the "Enable Load Balancer Support" section.
3. (Optional) To configure the NEN as an HA pair, perform the steps in "Configure HA Support" for the NEN.

Filter the Virtual Server List

You can filter the Virtual Servers list by using the properties filter at the top of the list. For example, you can filter and search by label. You can also filter and search by the following objects:

- Virtual server mode
- Virtual IP address, the VIP port number, or VIP Protocol
- Server Load Balancer

Configure a Load Balancer's Virtual Servers

1. From the PCE web console menu, choose **Infrastructure** > **Load Balancers**.
2. Select the load balancer for which you want to configure virtual servers.
3. Select the **Virtual Servers** tab.
4. Select one of the load balancer's virtual servers and click **Manage**.
5. Select one of the virtual servers and click **Edit**.
6. Enter a name and description for the virtual server.

7. To enable the virtual server's policy, select **Enforced**.
8. Select a service to associate with the virtual server. The service selected enables that service to be used in rules you write for this virtual server.
9. Select one each of the four labels to assign to the virtual server.
10. Click **Save**.
11. Before any virtual servers can go into effect, they must be provisioned.

Write SLB Policy

Writing a policy for a load balancer is similar to writing a policy for a workload, except for the following differences:

- Leave the service as unspecified and the port and protocol of the discovered VIP will determine the service automatically.
- Specify “Uses Virtual Services” in the rule.

A rule that is provided between a virtual server (or its labels) and a set of sources (aka consumers) implicitly programs two sets of rules:

- Rules between the consuming workloads or labels and the frontend VIP of the F5 on the discovered VIP port and protocol: Traffic flows between consuming workloads and the VIP are enforced on both ends if the virtual server is managed and enforced.
- Rules between the F5 pool and the virtual server providers on the service specified in the virtual server object (usually All Services): These rules are enforced for inbound traffic to the virtual server provider if the virtual server provider workloads are enforced.

SLB Methods

The SLB APIs are used to enable automation for F5 policy management.

Functionality	HTTP	URI
Get the list of SLBs	GET	[api_version][org_href]/slbs
Get a specified SLB	`GET`	`[api_version][org_href]/slbs/:uuid`
Create an SLB object	`POST`	`[api_version][org_href]/slbs`

SLB Parameters

The parameters for the SLB methods are:

Parameter	Description	Type
name	The short friendly name of the server load balancer	String
nfc	Network Function Controller managing this SLB	String
device_type	Device type of the server load balancer	String
devices	Configuration and runtime state of the devices backing this SLB Network VF.	String

Configure an SLB Object

Step 1. Create an SLB object and instruct the NEN to sync with it.

```
POST /api/v2/orgs/{org id}/slbs

{
    "devices" : [
    {
        "config" : {
            "username" : "admin",
            "port" : 443,
            "credential" : "admin",  # never replayed in northbound API
            "host" : "10.2.32.6",
            "credential_type" : "password",
            "check_certificate" : false
        }
    }
],
    "device_type" :“F5 Big-IP LTM”
    "name" : "Illumio Test SLB"
}
```

Step 2. GET an SLB response.

```
GET /orgs/{org id}/slbs/{UUID of SLB object

{
    "name" : "Illumio Test SLB",
    "devices" : [
    {
        "status" : {"connection_state" : "pending"},    # will become
successful when NEN
                 syncs w/ device
        "href" : "/orgs/1/slb_devices/9349ff36-ab38-42bf-909a-eb5aa3baf187",
        "config" : {
        "username" : "admin",
        "check_certificate" : false,
        "credential_type" : "password",
        "host" : "10.2.32.6",
        "credential" : null,
        "port" : 443
    }
}
],
```

```
    "href" : "/orgs/1/slbs/8a82a1b0-c2ce-43ec-abf7-77bd8a3fd22c",
    "device_type" : "f5_bigip_afm"
    [ ... ]  # created_at, updated_at, etc.
}
```

Step 3. GET a list of Discovered Virtual Servers.

```
GET /orgs/1/discovered_virtual_servers

{
    "snat_type" : "snat_pool",
    "dvs_identifier" : "d3b784c2fd24ad364c5adb3319169bd2",
    "mode" : "snat",
    "vip_port" : {"port" : 8080, "protocol" : 6, "vip" : "172.16.27.88" },
    "service_checks" : [{"protocol" : 1}],
    "name" : "Common/QL_VIP_1",
    "slb" : {
        "href" : "/orgs/1/slbs/8a82a1b0-c2ce-43ec-abf7-77bd8a3fd22c"
    },
    "snat_pool_ips" : ["172.16.26.27", "172.16.26.18", "172.16.27.18"],
    "local_ips" : ["172.16.26.18", "172.16.27.18"],
    "href" : "/orgs/1/discovered_virtual_servers/2c460b98-2176-4a44-9ba4-
e77f3eacd0f1"
    [ ... ]  # created_at, updated_at, etc.
}
```

Step 4. Manage a VIP by creating a Virtual Server object.

```
POST /orgs/1/sec_policy/draft/virtual_servers

{
    "name" : "Common/chris-VIP1",
    "service" : {
        "href" : "/orgs/1/sec_policy/draft/services/1"
    },
    "labels" : [],
    "providers" : [],
    "mode" : "unmanaged",  # enforced
    "discovered_virtual_server" : {
        "href" : "/orgs/1/discovered_virtual_servers/23338ceb-7580-466a-
bbcf-a645b82ce97b"
    }
}
```

Step 5. Modify the enforcement mode, labels, and backend/provider labels of the Virtual Server.

```
PUT /orgs/1/sec_policy/draft/virtual_servers/84bae9dd-f1f6-4322-bffc-
f07354b0622a

{
    "mode" : "enforced",
    "labels" : [{"href" : "/orgs/1/labels/448"}, {"href" : "/orgs/1/labels/
444"}],
        # any RAEL tuple
```

```
    "providers” : [{"label":{"href":"/orgs/1/labels/449"}}]
        # note: providers may have different labels
}
```

Step 6. Provision the Virtual Server into an active policy.

```
POST /orgs/1/sec_policy
```

```
{
    "update_description” : "Provision my first VS",
    "change_subset” : {
        "virtual_servers” : [{"href” : "/orgs/1/sec_policy/draft/
virtual_servers/
                 84bae9dd-f1f6-4322-bffc-f07354b0622a"}]
    }
}
/orgs/1/sec_policy/draft/virtual_servers/84bae9dd-f1f6-4322-bffc-
f07354b0622a
/orgs/1/sec_policy/active/virtual_servers/84bae9dd-f1f6-4322-bffc-
f07354b0622a
```

Step 7. Write rules that apply to the Virtual Server.

```
POST /orgs/1/sec_policy/draft/rule_sets/1480/sec_rules
```

```
{
    "enabled" : true,
    "providers" : [
        {"label" : {"href" : "/orgs/1/labels/444"}},
        {"label" : {"href" : "/orgs/1/labels/448"}}
    ],
    "resolve_labels_as" : {
        "consumers" : ["workloads"],
        "providers" : ["virtual_services"]    # NOTE: Must be
virtual_services
    },
    "consumers" : [
        {"actors" : "ams"}  # All Workloads
    ]
}
{
    "consumers" : [
        {"label" : {"href" : "/orgs/1/labels/444"}}
    ],
    "providers" : [
    {
        "virtual_server" :
        {"href" : "/orgs/1/sec_policy/draft/virtual_servers/
                84bae9dd-f1f6-4322-bffc-f07354b0622a"}
    }
],
"enabled" : true,
"resolve_labels_as" : {
"consumers" : ["workloads"],
"providers" : ["virtual_services"]
    }
```

Remove Filtering

Some types of virtual servers are not visible, such as those without default server pools. From the NEN 2.1.0 release onwards, you can do filtering related to such virtual servers. You can see VIPs that do not have a pool associated with them or are not SNAT/Auto-SNAT.

To view all types of virtual servers configured on the F5 load balancers, you must enter specific commands during the NEN installation (on a NEN by NEN basis). These commands disable (enabled by default) the built-in filter running on the NEN on the Leader PCE cluster.

1. Enter the following command:

   ```
   sudo -su ilo-nen /opt/illumio-nen/illumio-nen-ctl slb-enable
   --virtual-server-filtering disabled
   ```
2. Restart the NEN on both db0 and db1 nodes:

   ```
   sudo -u ilo-nen /opt/illumio-nen/illumio-pce-ctl restart
   ```

NEN Integration with Switches

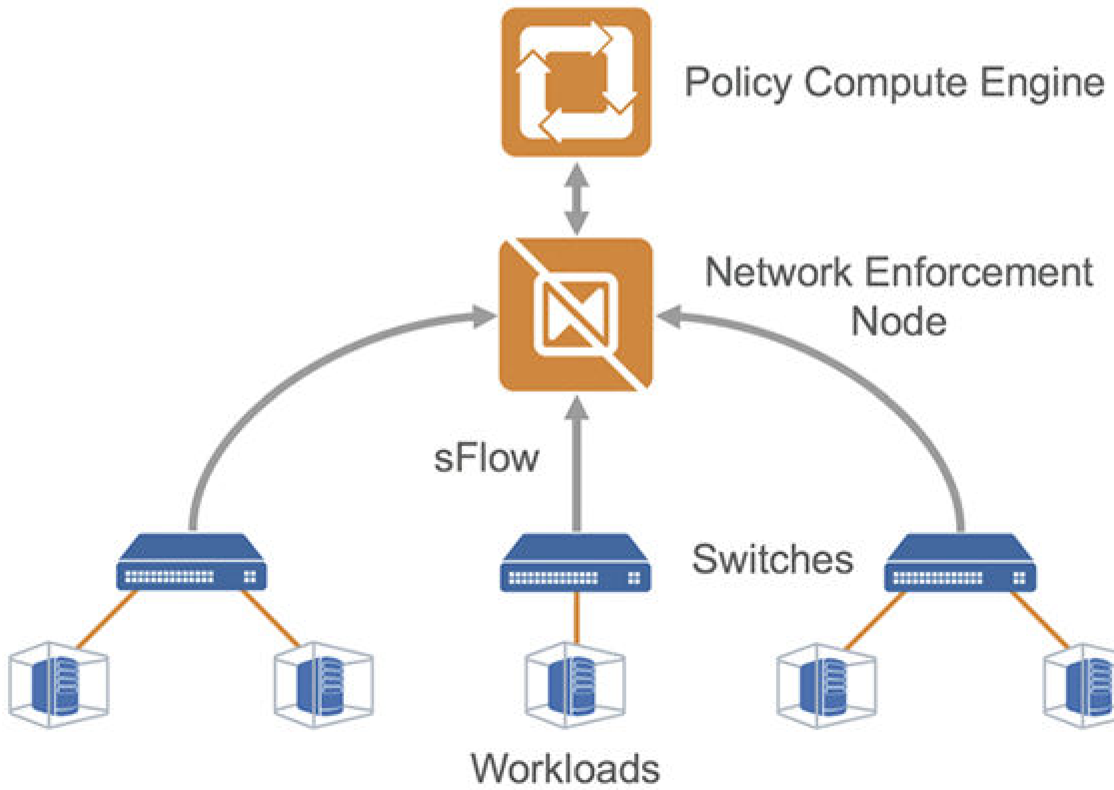
This section describes how to create security policy and apply those policies on the switches for use with the NEN.

Overview of Switch Integration

The Illumio Network Enforcement Node (NEN) is the Illumio Core switch interface, which allows you to get visibility and enforcement on switches. Using the NEN, you can secure workloads that are attached to network switches. You can use the NEN to generate access control lists (ACLs) and load those on your switches to protect the ports to which your workloads are attached.

How the NEN Receives Switch Data

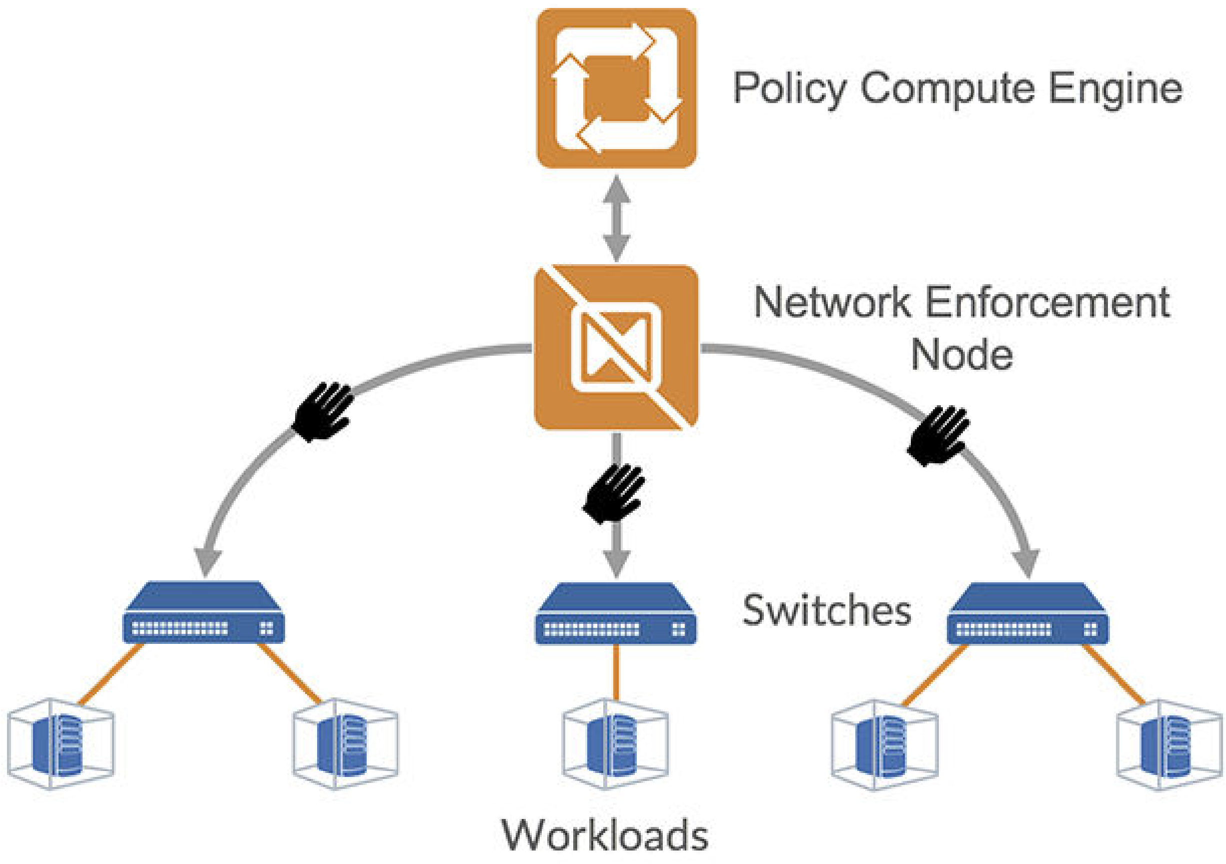
With the NEN, network administrators can configure their switches to send sFlow data to an sFlow collector, such as the NEN. An Illumio Core administrator can configure the NEN to listen for sFlow data from switches and associate workloads to those switches. The NEN receives sFlow data directly from the switches, summarizes it, and uploads it to the PCE. You can view this traffic flow in the Illumination® map and stream it out of the PCE through UDP in Splunk, CEF, or LEEF formats.



Extended Policy Model

The Illumio policy model encompasses workloads with native stateful firewalls built-in, such as Linux iptables or Windows Filtering Platform. Although all systems might not have a firewall built in, they still have segmentation requirements. To solve this use case, Illumio has extended its policy model to switches.

Illumio administrators can use the NEN to convert natural language policies into ACLs, which the switches understand natively. Your organization's teams that use Illumio Core can download ACLs from the PCE and provide them to the networking team for review before applying new policies to the switches.



Limitations for Switch Integration

This release is subject to the following limitations:

- You must provide a switch IP address and an interface traffic flow ID for interfaces that need to be monitored for sFlow data.
- The NEN discards sFlow data from an interface that it does not monitor.
- The Illumio Core generates only IPv4 ACLs that can be applied to either the L3/Routed interfaces or Switch Virtual Interface (SVI) for L2 interfaces when they are a member of a VLAN. Whenever ACLs are applied to the SVI, workloads within the same VLAN can freely communicate regardless of policy.

  This is a limitation of IPv4 ACLs on switches. Inter-VLAN or routed traffic will still be filtered by ACLs.

Requirements for Switch Integration

- Illumio-provided PCE 20.2.0 or later and NEN 2.1.0 (includes NFC) software packages.
- Cisco Nexus 9200 or 9300 or Arista 7000 series switch.
- Workloads that are directly attached to the switch on L2 or L3 ports or on port channels.

NOTE

The NEN targets top-of-rack (TOR) switches that are directly attached to the workload and not the core switches. For example, Cisco Nexus 9200 and 9300 (TOR) switches are supported, but the Cisco 9500 series switches are not supported.

Workflow for Setting up NEN Switch Integration

This following is an overview of the steps required for working with the NEN for switch integration:

1. In the PCE web console:
   a. Define the switches.
   b. Create unmanaged workloads.
   c. Assign those unmanaged workloads to switch interfaces.
   d. Create security policy rules to protect the workloads attached to the switches.
2. Use the PCE REST API or the PCE web console to generate switch ACLs based on your organization's security policies.
3. Copy and paste the generated ACLs to configure the switch via the switch's command line.
4. Using the PCE REST API or the PCE web console, inform the PCE that the ACLs have been loaded.

Result: The PCE-generated ACLs on the switch will protect the target workloads.

Supported Switches and Configurations

The following switches are supported in this release:

- Cisco Nexus 9200 and 9300 series
- Arista 7000 series

Switch Configuration

The following ACL and interface configurations are supported for the Illumio NEN integration:

ACL Implementation	Switch Interfaces	ACL Type
Router ACL (RACL) RACLs support both inbound and outbound enforcement.	• VLAN interface (SVI) • Layer 3 physical interface • Layer 3 port-channel interface	IPv4

IMPORTANT

Unsupported interface and ACL configurations

The NEN does not support:

- VLAN ACL (VACL) or Virtual Teletype (VTY) ACL as the ACL implementation
- VLAN trunk port (switchport mode trunk) or sub-interface as the switch interface
- MAC ACL type
- IPv6 ACL type
- PACLs for Layer 2 interfaces.

Administrative Access to the Switch

You or your network administrators need administrative access to your switches to configure them and load the NEN-generated ACLs.

NOTE

The PCE and the NEN do not send any communication to the switch and never log into the switch. The PCE and the NEN do not require root or admin privileges on the switch.

Sufficient TCAM

Your switch's ternary content-addressable memory (TCAM) must be sufficient to store the IPv4 RACLs generated by the NEN.

NOTE

Illumio does not provide a mechanism to check the TCAM depth or available memory for each platform. Your network or security administrators need to check whether the generated IP ACLs can be handled by the switch.

Enable sFlow

The NEN relies on sFlow to provide network traffic flow data for Illumination. Your switch must be configured with sFlow. See your vendor documentation for information.

Configure sFlow Output

The output of sFlow from the switch must be sent to the PCE so it can be monitored. The well-known port for sFlow is port UDP 6343. See Configure Switches for NEN [425] for information.

Network Connectivity between Switches and NEN

The NEN listens for sFlow from the switches.

> **IMPORTANT**
>
> Ensure that your network is configured to allow communication between your switches and the NEN.

Switch Information

You need to provide switch-related information in the PCE web console. See the table listed in Add Unmanaged Workloads and Switch Definitions in the PCE Web Console [428] for information.

Configure Switches for the NEN

sFlow on the switch must be configured to send its output to the NEN. In addition, the sFlow-monitored interfaces on the switch must be configured in the NEN service via the PCE web console. If the NEN service receives sFlow information from an unrecognized or undefined network endpoint (or interface), it will reject that information. The NEN service continually aggregates the sFlow traffic and sends the aggregated information to the PCE traffic collector every 10 minutes.

> **NOTE**
>
> sFlow is only a sampling protocol, so all the flows might not be recorded. If the default sampling rate is not sufficient for your use case, see your vendor documentation.

Configure sFlow on Cisco Switch

Use the following (config)# commands to configure sFlow on a Cisco 9000 series switch:

1. Enable sFlow:

   ```
   (config)# feature sflow
   ```
2. In the following command, the NEN_ip_address variable is the IP address of the NEN primary node:

   ```
   (config)# sflow collector-ip NEN_ip_address vrf default
   ```
3. In the following command, the switch_IP_address variable is the IP address of the switch, which you will also use in the PCE web console. switch_IP is a management IP address.

   ```
   (config)# sflow agent-ip switch_IP_address
   ```
4. In the following command, the interface_name_to_monitor variable is a mnemonic name that you have already defined on the switch for the interface, which you will also use in the PCE web console.

```
(config)# sflow data-source interface interface_name_to_monitor
```

5. Repeat the above sflow data-source interface command for all interfaces on the switch that you want to secure.

See Add Unmanaged Workloads and Switch Definitions in the PCE Web Console [428] for information.

Example of sFlow Configuration for Cisco

```
nexus9000(config)# show run sflow

!Command: show running-config sflow

feature sflow

sflow collector-ip 10.10.10.1 vrf default
sflow agent-ip 10.20.20.1

sflow data-source interface Ethernet1/7
```

In this example:

- The IP address on the switch that can communicate with the PCE is 10.20.20.1.
- The PCE/NEN IP address (sFlow collector) is 10.10.10.1.
- A workload is directly attached to interface Ethernet 1/7.

Collect SNMP ifIndex Value for Cisco

When the switch reports sFlow to the NEN, it includes interface index details in the flow records. When the NEN receives sFlow, it parses the records and retains the records only for the interfaces you specify in the NEN configuration. You need to collect the ifindex IDs and add them to the NEN configuration later. You can determine your switches' SNMP ifIndex values using the following command:

```
# show interface snmp-ifindex
```

Manufacturer/ Model	Command	Notes
Cisco 9000	In privileged mode: `show interface snmp-ifindex`	This command outputs the IFMIB (decimal) and the ifIndex (hex) values. You need the IFMIB (decimal) value later. This value is required to configure **Monitor Traffic** for the NEN.

Example of Command Output

```
nexus9000# show interface snmp-ifindex
----------------------------------------------------------------------------
----
Port   IFMIB      Ifindex (hex)
----------------------------------------------------------------------------
----
mgmt0  83886080  (0x5000000)
Eth1/1 436207616 (0x1a000000)
```

```
Eth1/2 436208128 (0x1a000200)
Eth1/3 436208640 (0x1a000400)
Eth1/4 436209152 (0x1a000600)
Eth1/5 436209664 (0x1a000800)
Eth1/6 436210176 (0x1a000a00)
Eth1/7 436210688 (0x1a000c00)
Eth1/8 436211200 (0x1a000e00)
```

This example uses Ethernet 1/7 interface as an sFlow source interface. To enter the interface information in the PCE, collect the decimal value of the ifIndex. In case of the Cisco Nexus 9000, this value is in the **IFMIB** column of the command output. The command output above shows 436210688 as the IFMIB value for Ethernet 1/7 interface. This value is required to configure the **Monitor Traffic** field in the PCE configuration page.

Configure sFlow on Arista Switch

Use the following commands to configure sFlow on an Arista 7000 series switch:

1. Run sFlow (this command is similar to enabling sFlow on a Cisco switch):

   ```
   sflow run
   ```
2. In the following command, the IP address is the destination PCE IP to which the sFlow information should be sent:

   ```
   sflow destination 10.6.1.158
   ```
3. In the following command, the IP address is the source IP from where the sFlow information is sent:

   ```
   sflow source 10.21.6.1
   ```

On an Arista switch, the list of sFlow command options are:

Command	Description
`destination`	Set sFlow collector destination.
`extension`	Configure sFlow extension settings.
`polling-interval`	Set polling interval (secs) for sFlow.
`qos`	Configure QoS parameters.
`run`	Run sFlow globally.
`sample`	Set sample characteristics for sFlow.
`source`	Set the source IP address.
`source-interface`	Configure the source interface for sFlow datagrams.
`vrf`	Configure VRFs.

Collect SNMP ifIndex Value for Arista

You can determine your Arista switches' SNMP ifIndex values using the following command:

```
arista7000# show snmp mib ifmib ifindex
Ethernet1: Ifindex = 1
Ethernet2: Ifindex = 2
Ethernet3: Ifindex = 3
Ethernet4: Ifindex = 4
Ethernet5: Ifindex = 5
Ethernet6: Ifindex = 6
Ethernet7: Ifindex = 7
Ethernet8: Ifindex = 8
```

Add Unmanaged Workloads and Switch Definitions in the PCE Web Console

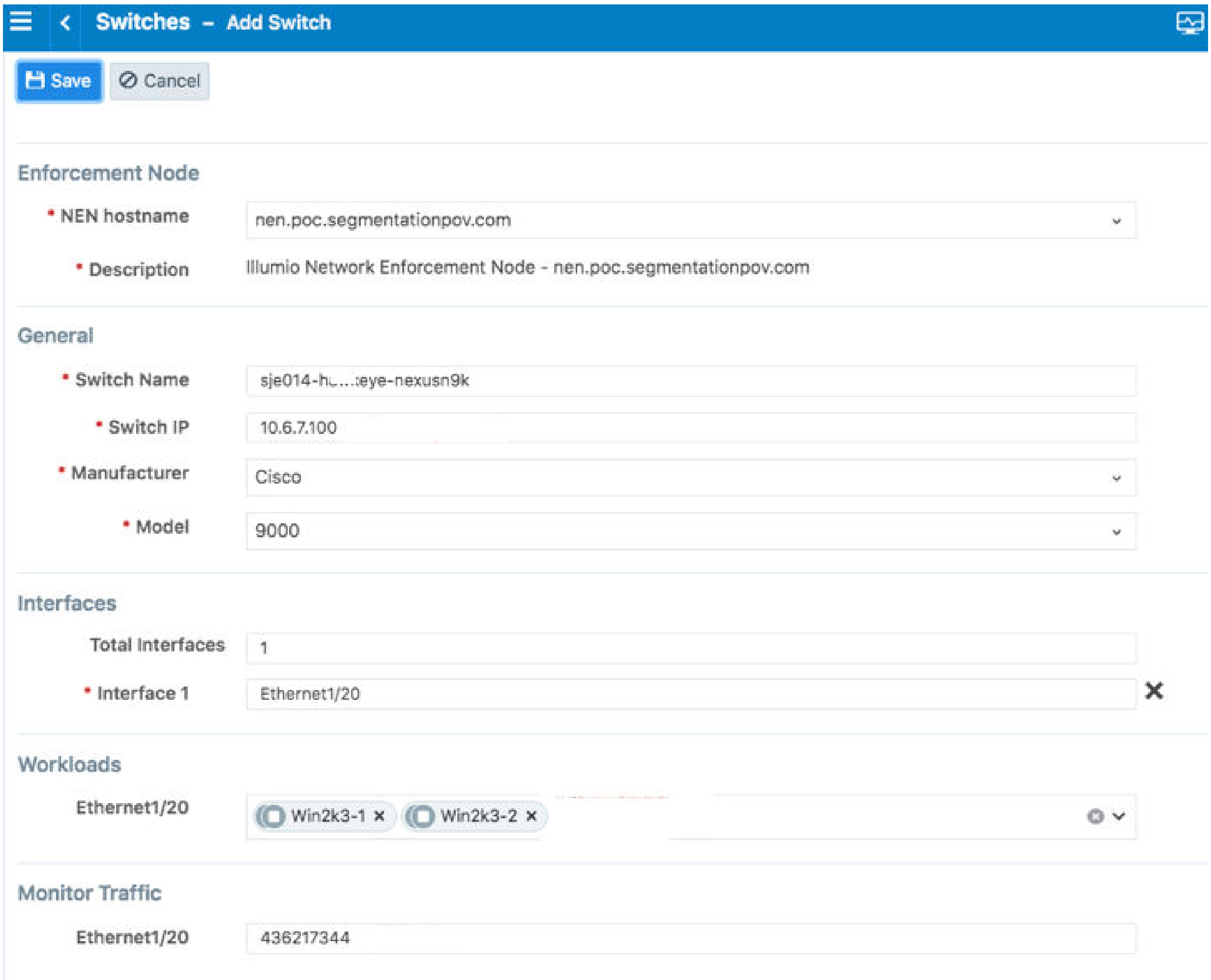
To create a security policy, the switches and the workloads attached to them should be defined in the PCE web console as follows:

1. Log into the PCE web console.
2. Define the unmanaged workloads that are attached to the switch by selecting **Workloads and VENs > Workloads > Add > Add Unmanaged Workload**. You will associate these unmanaged workloads with their switches later.

   See the Security Policy Guide for information on adding unmanaged workloads.
3. Define the switches and associated workloads, by selecting **Infrastructure > Switches**.
4. Click **Add**.
5. Enter the details in the displayed fields as described in the table below.
6. After entering or selecting values for all the required fields, click **Save**.

Fields in the **PCE web console > Infrastructure > Switches > Add Switch** page:

Field Name	Description	Required	Notes
NEN hostname	FQDN of the NEN that runs the NEN service	Yes	This field is populated with the FQDN of your NEN. You cannot edit this field.
Description	Description of the NEN service	Yes	This field is populated with "Illumio Network Enforcement Node" and the FQDN of your NEN. You cannot edit this field.
Switch Name	A free-form, mnemonic name of your choice for the switch	Yes	Make this name easy to remember and distinguishable from other switch names.
Switch IP	IP address of the switch	Yes	Corresponds to switch_IP_address that you defined in Configure sFlow on Cisco Switch [425]. It is also known as sflow agent-ip in Cisco switches.
Manufacturer	Name of the switch manufacturer	Yes	Select Cisco.
Model	Model number of the switch	Yes	Select 9000.
Interfaces	Defined interfaces on the switch	No	Corresponds to interface_name_to_monitor you defined on the switch and configured in Configure sFlow on Cisco Switch [425]. This can be a custom string. You can also add interfaces that are not monitored by sFlow.
Workloads	Names of workloads connected to the switch's defined interfaces	No	Only those workloads assigned to the switch interfaces are secured. You can attach one or more workloads to an interface.
Monitor Traffic	SNMP ifIndex of the switch interface See Collect SNMP ifIndex Value for Cisco [426] and Collect SNMP ifIndex Value for Arista [427].	Yes/No	If the interface is monitored by sFlow, the **Monitor Traffic** field is required.



NEN Switch Configuration Using REST API

To manage network switches reporting data flows to the NEN and to get the generated ACLs to enforce policies based on what's been defined in the PCE, you need to complete these tasks:

1. Get the list of switches and their details.
2. Generate the ACLs for one or all switches.
3. Print the ACLs in the desired format.

The sections below describe the manual steps, which can be inserted in any script to automate this process.

Get List of Switches and Details

To get the list of all the network switches registered against the NEN, run the following curl command:

```
curl -u api_xxx:xxx -H "Accept: application/json" -X GET
https://mypce.domain.io:8443/api/v2/orgs/1/network_devices
```

Result: Returns a list of switches with all the reported endpoints (ports, workloads) to the NEN.

Curl Command of Get List of Switches

```
curl -u api_1853ebfcb1187acb4:9c2a381773a44e3a609448109278c02c4ec1fe597
f9643af71a832c0a8b0c0d0
-H "Accept: application/json" -X GET https://mypce.domain.io:8443/api/v2/
orgs
/1/network_devices
```

Response


```
[
   {
      "network_enforcement_node" : {
         "href" : "/orgs/1/network_enforcement_nodes/
f64e78b7-2917-409f-9093-
         9d6ddaa35799"
      },
      "href" : "/orgs/1/network_devices/f07a077a-70ad-4b57-b82a-
f1d204fcfd99",
      "configure" : false,
      "network_endpoints" : [
         {
            "href" : "/orgs/1/network_devices/f07a077a-70ad-4b57-b82a-
            f1d204fcfd99/network_endpoints/1ff6f037-d644-438e-
ab32-019a45a7d8d5"
         },
         {
            "href" : "/orgs/1/network_devices/f07a077a-70ad-4b57-b82a-
f1d204fcfd99/
            network_endpoints/dd687e16-6998-4a39-8bde-a7fb445f18d9"
         },
         {
            "href" : "/orgs/1/network_devices/f07a077a-70ad-4b57-b82a-
f1d204fcfd99/
            network_endpoints/7345aed3-1fbd-4596-ada9-f6cbfb361dfe"
         },
         {
            "href" : "/orgs/1/network_devices/f07a077a-70ad-4b57-b82a-
f1d204fcfd99/
            network_endpoints/be58f614-7cc7-4132-a409-97ea8334dfeb"
         }
      ],
      "enforcement_instructions_data_timestamp" : "2019-05-06T15:45:02Z",
      "enforcement_instructions_data_href" : "/orgs/1/datafiles/
      49b11cf6-d6f9-4efc-8cb2-c1a444cb9c02",
      "supported_endpoint_type" : "switch_port",
      "config" : {
         "model" : "9000",
         "name" : "cisco-n9k",
         "rules_format" : "cli",
         "ip_address" : "10.1.2.3",
         "device_type" : "switch",
         "manufacturer" : "Cisco"
      },
```

```
      "status" : "unmonitored"
   }
]
```

Generate ACLs for Switches

To generate ACLs for a specific switch registered against the NEN, run the following curl command:

```
curl -u api_xxx:xxx -H "Content-Type: application/json" -d {}
-X POST https://mypce.domain.io:8443/api/v2/orgs/1/network_devices/
xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxx/enforcement_instructions_request
```

NOTE

Replace the xxx-...-xxx value with the value of the switch for which you intend to generate ACLs.

Curl Command Using Generate ACLs

```
curl -u api_1853ebfcb1187acb4:9c2a381773a44e3a609448109278c02c4
ec1fe597f9643af71a832c0a8b0c0d0
-H "Content-Type: application/json" -d {} -X
POST https://mypce.domain.io:8443 -d {}
-X POST https://mypce.domain.io:8443/api/v2/orgs/1/network_devices/
f07a077a-70ad-4b57-b82a-f1d204fcfd99/enforcement_instructions_request
```

Result: Response with a 202 status code = Accepted.

The ACLs are generated on the NEN and are ready for use in a few minutes.

IMPORTANT

API POST Requirements

While sending a POST request, you must include the header (-H) flag and the data (-d) flag. Even if you do not have any data to send, you must insert an empty data flag, as shown in the above example.

NOTE

Illumio recommends that you insert a pause in any script to allow the NEN to generate the new ACLs for the specific switch. It takes approximately 30 seconds to generate all the ACLs.

The PCE will not send any update or acknowledgment to the REST client once it is finished generating the new ACLs for the switch.

Alternatively, you might want to generate ACLs for all the switches in your inventory to deliver them to your network team, by using the all_devices: true key-value pair in your JSON payload while sending the POST request.

To generate ACLs for all the switches registered against the NEN, run the following curl command:

```
curl -u api_xxx:xxx -H "Content-Type: application/json" -d
'{"all_devices": true}'
-X POST https://mypce.domain.io:8443/api/v2/orgs/1/
network_devices/multi_enforcement_instructions_request
```

Get List of ACLs

To download ACLs for a specific switch registered against the NEN, get the updated value of the enforcement_instructions_data_href key. This value keeps changing because each time the NEN generates new ACLs for a switch, it is considered to be a new datafile.

1. To get the updated enforcement_instructions_data_href value for a network switch, run the following curl command:

   ```
   curl -u api_xxx:xxx -H "Accept: application/json" -X
   GET https://mypce.domain.io:8443/api/v2
   /orgs/1/network_devices/enforcement_instructions_data_href'
   ```

   The above command returns a list of switches. You have to then parse the JSON output and filter on the enforcement_instructions_data_href key to get the updated value. You can use the JQ tool to filter outputs on any JSON file.
2. After you retrieve the updated value, use it in the following curl command to get the generated ACLs:

   ```
   curl -u api_xxx:xxx -H "Accept: application/json" -X GET
   https://mypce.domain.io:8443/api/v2
   /orgs/1/datafiles/xxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx
   ```

   Replace the xxx-...-xxx value with the value of the enforcement_instructions_data_href key that you got by running the previous GET request.

Example of Get List of ACLs

```
curl -u api_1853ebfcb1187acb4:9c2a381773a44e3a609448109278
c02c4ec1fe597f9643af71a832c0a8b0c0d0
```

```
-H "Accept: application/json" -X GET https://mypce.domain.io:8443/
api/v2/orgs/1/network_devices/
enforcement_instructions_data_href' "/orgs/1/datafiles/
d1bdbb23-60c4-439e-bd74-ca0d03d959a7"
```

Output:

```
no ip access-list ILLUMIO_ACLS-Ethernet1-21-Inbound
p access-list ILLUMIO_ACLS-Ethernet1-21-Inbound
!---Inbound ACL Rules ---
    permit ip host 10.10.100.201 host 10.10.100.202
    permit ip host 10.10.100.201 host 10.10.100.203
    permit ip host 10.10.100.201 host 10.10.100.204
    permit tcp any any established
    permit udp any eq 68 any eq 67
    permit udp any range 1024 65535 any eq 53
    exit
            ...
no ip access-list ILLUMIO_ACLS-VLAN-20-Outbound
ip access-list ILLUMIO_ACLS-VLAN-20-Outbound
!---Outbound ACL Rules ---
    permit ip host 10.10.100.201 host 10.10.100.204
    permit ip host 10.10.100.202 host 10.10.100.204
    permit ip host 10.10.100.203 host 10.10.100.204
    permit tcp any any established
    permit udp any eq 67 any eq 68
    permit udp any eq 53 any range 1024 65535
    exit
```

Integrate the Illumio PCE with IBM i running Precisely Assure Security

> **NOTE**
> For the most up-to-date information about configuring Precisely Assure Security for this integration, see the Precisely document Commands for the Assure SAM integration with Illumio.

This topic describes how to integrate **IBM i** computers running Precisely Assure Security with your Illumio PCE. This integration differs from the typical switch integration in the following ways:

- Although the **IBM i** is not a switch, this integration uses the PCE switch integration user interface to perform the integration.
- Instead of generating ACLs as you would do when integrating a switch, you'll generate a Precisely-formatted CSV file to configure relevant policy on your **IBM i** computer that is running Precisely.
- Flow information is collected by Assure Security and sent to the Illumio Flowlink server (see Illumio Flowlink Configuration and Usage Guide).

Add Unmanaged Workloads and IBM i Definitions

To create a security policy, add unmanaged workloads representing each **IBM i** server included in the PCE policy. A set of csv data is generated for each configured **IBM i** unmanaged workload. To define the **IBM i** servers and the workloads attached to them as unmanaged workloads in the PCE web console, complete the following steps:

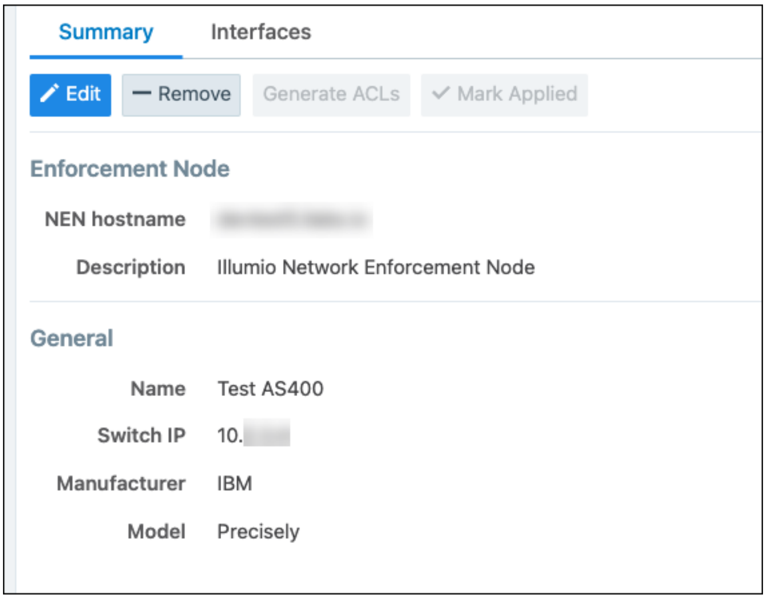
1. Log into the PCE web console.
2. Define the **IBM i** servers as unmanaged workloads by adding an unmanaged workload through the PCE. You will associate these unmanaged workloads with their **IBM i** Precisely integration later. (For more about adding unmanaged workloads, see Adding Unmanaged Workloads in the Security Policy Guide.

   > **NOTE**
   > The workload name must be the IBM i system name in upper-case.

   - Illumio Core **release 22.5** and earlier: Go to **Workloads and VENs > Workloads** . . . click **Add > Add Unmanaged Workload**
   - Illumio Core **release 23.2** and later: Go to **Servers & Endpoints > Workloads**. . . click **Add > Add Unmanaged Workload**
3. Define the **IBM i** integration and associated workloads in the PCE by going to **Infrastructure > Switches**.
4. Click **+Add**.
5. Enter details:
   - **NEN hostname:** This field is populated with the FQDNs of the NENs paired with your organization's PCE. Select the appropriate NEN.
   - **Description:** This field is populated with "Illumio Network Enforcement Node" and the FQDN of the NEN. You cannot edit this field.
   - **Switch Name:** Enter a unique name that's easy to remember.
   - **Switch IP:** IP address of the **IBM i** computer.
   - **Manufacturer:** Select IBM.
   - **Model:** Select Precisely.
6. Click **Save**.
7. Click **Interfaces**.
8. Click **Edit** and then enter details:
   - **Total Interfaces:** 1
   - **Interface 1:** Enter a name, such as interface 1.
   - **Workloads:** Select the unmanaged workload representing the appropriate **IBM i** computer. Only workloads assigned to the **IBM i** server interfaces are secured. You can attach one or more workloads to an interface.
   - **Monitor Traffic:** Ignore this setting. It doesn't apply to this integration.
9. Click **Save**.

> **NOTE**
> If your unmanaged **IBM i** computer has two or more network interfaces, the generated ACL file will include duplicate entries for Inbound Rules, one pair of entries for each interface. This is expected behavior.

Fields in the **PCE web console > Infrastructure > Switches > Add Switch** page:



Apply Policy for Switches

To apply the security policy, you need to:

1. Create the policy and generate ACLs for loading onto the switch.
2. Load the generated ACLs onto the switch.
3. Inform the PCE that the ACLs have been loaded onto the switch.

Create Security Policy

In the PCE web console, create label-based policies for the workloads that are bound to the switch ports. For information on how to create policies, see the Security Policy Guide.

> **NOTE**
> Make sure you provision the policies before generating the ACLs.

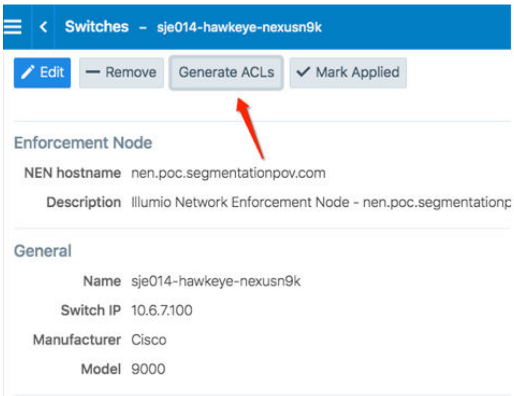
Generate and Download ACLs

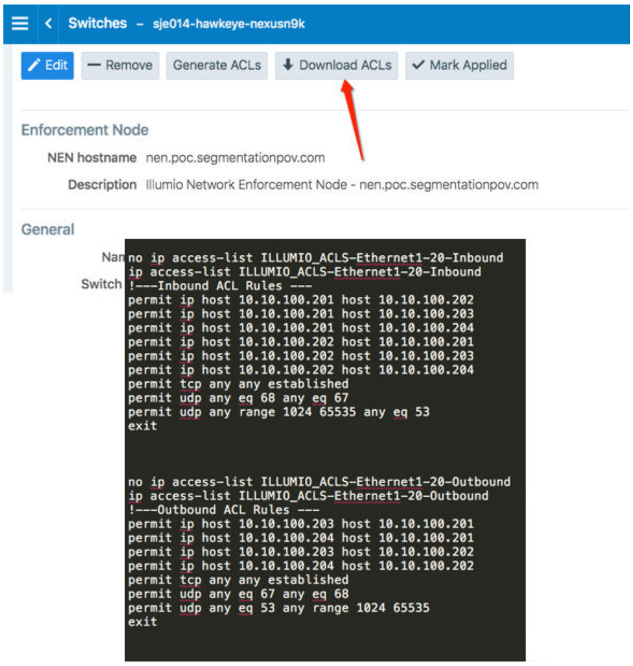
After you have created new policies for the NEN-managed workloads and provisioned them in the PCE, you can generate the ACLs. The PCE writes those ACLs to its local files.

Create the associated switch ACLs as follows:

1. Log into the PCE web console.
2. From the PCE web console menu, choose **Infrastructure** > **Switches**.
3. Select the switch.
4. On the Switches page, click **Generate ACLs**. It takes a moment for the ACL generation to complete.

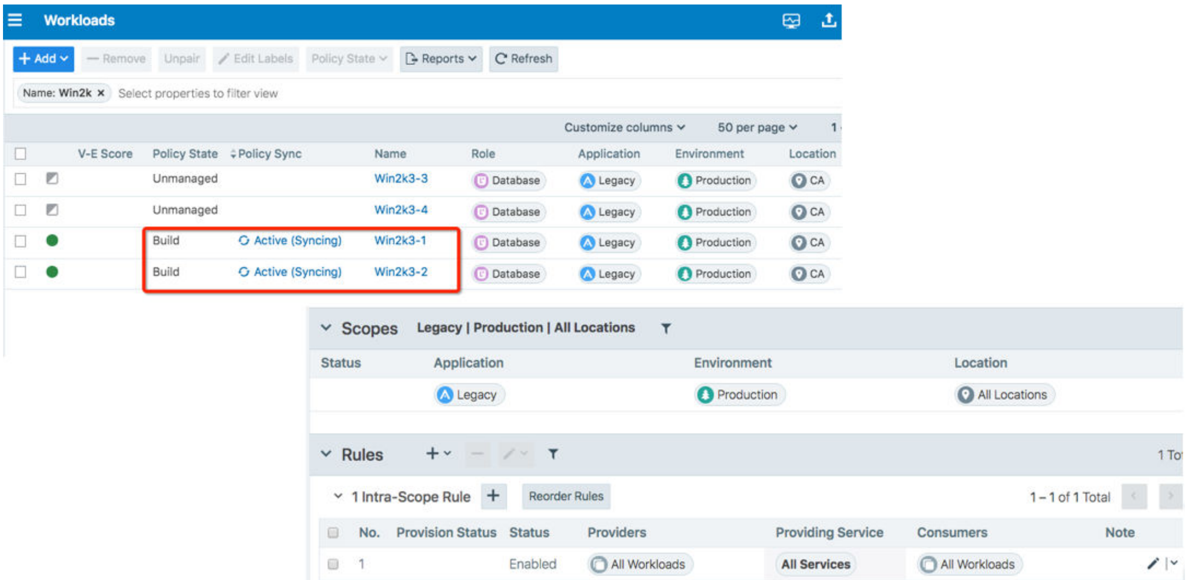
5. After the ACLs have been generated, click **Download ACLs** to download a .txt file of the updated ACLs from your web browser.
6. Go to the Downloads folder on your computer and open the .txt file. This file contains a list of inbound and outbound ACLs.





The workloads associated with the switch remain in an **Active (Syncing)** policy sync state until you click the **Mark Applied** button on the Switches page.

> **NOTE**
> You might see a "Syncing" notification appear in the PCE web console until you mark the ACLs as **Applied**.



Apply ACLs on the Switch

After generating the ACLs from the NEN, you can copy the text of the ACLs from the PCE-generated files and paste them into the switch's command line to configure the ACLs on the switch. Each interface per direction requires a separate text file.

Example of Inbound and Outbound ACLs

The following ACLs are generated for only Ethernet 1/7 interface of a Cisco Nexus 9000:

```
no ip access-list ILLUMIO_ACLS-Eth1-7-Inbound
ip access-list ILLUMIO_ACLS-Eth1-7-Inbound
!---Inbound ACL Rules ---
permit tcp any any established
permit udp any eq 68 any eq 67
permit udp any range 1024 65535 any eq 53
exit


no ip access-list ILLUMIO_ACLS-Eth1-7-Outbound
ip access-list ILLUMIO_ACLS-Eth1-7-Outbound
!---Outbound ACL Rules ---
permit tcp host 10.6.4.94 host 10.0.17.17 eq 5432
permit tcp any any established
permit udp any eq 67 any eq 68
permit udp any eq 53 any range 1024 65535
exit
```

NOTE

By default, the NEN generates ACLs to allow basic infrastructure services such as DHCP and DNS from all IP addresses in addition to the policies defined on the PCE. You cannot prevent DNS and DHCP ACLs from being generated by the NEN. If your network administrator does not want DHCP or DNS rules added to the switch, you can remove those ACL lines while copying the ACLs over to the switch.

To configure the ACLs on to the switch, the network administrator must log into the Cisco Nexus 9000 command line and go into configuration mode. Once in configuration mode, the ACLs can be copied from the NEN in to the switch command line as shown in this example:

```
nexus9000# conf
Enter configuration commands, one per line. End with CNTL/Z.
nexus9000(config)# no ip access-list ILLUMIO_ACLS-Eth1-7-Inbound
nexus9000(config)# ip access-list ILLUMIO_ACLS-Eth1-7-Inbound
nexus9000(config-acl)# !---Inbound ACL Rules ---
nexus9000(config-acl)# permit icmp host 10.0.17.17 0.0.0.0/0
nexus9000(config-acl)# permit tcp any any established
nexus9000(config-acl)# permit udp any eq 68 any eq 67
nexus9000(config-acl)# permit udp any range 1024 65535 any eq 53
nexus9000(config-acl)# exit
nexus9000(config)#
nexus9000(config)#
nexus9000(config)#
nexus9000(config)# no ip access-list ILLUMIO_ACLS-Eth1-7-Outbound
nexus9000(config)# ip access-list ILLUMIO_ACLS-Eth1-7-Outbound
nexus9000(config-acl)# !---Outbound ACL Rules ---
nexus9000(config-acl)# permit tcp host 10.6.4.94 host 10.0.17.17 eq 5432
```

```
nexus9000(config-acl)# permit tcp 0.0.0.0/0 host 10.0.17.17 eq 22
nexus9000(config-acl)# permit tcp any any established
nexus9000(config-acl)# permit udp any eq 67 any eq 68
nexus9000(config-acl)# permit udp any eq 53 any range 1024 65535
nexus9000(config-acl)# exit
```

After the ACLs have been added to the switch, they must be applied to an interface on the switch as either a PACL or a RACL. To take advantage of both inbound and outbound ACLs, this example uses RACLs. If the workload is directly attached to a Layer 2 interface (switch-port mode access), you must apply the RACL to the VLAN/SVI interface. If the workload is directly attached to a Layer 3 interface (no switchport), the ACL can be directly applied to the physical interface (or port).

Optional Command to Verify the Interface Configuration

```
# show run interface ethernet x/y
```

Example of Command Usage for Ethernet 1/7

```
nexus9000(config)# show run int eth1/7

!Command: show running-config interface Ethernet1/7
!Time: Tue Oct 16 17:32:52 2018

version 7.0(3)I5(1)

interface Ethernet1/7
switchport
switchport access vlan 17
no shutdown
```

This interface is a Layer 2 interface and is part of VLAN 17. You must apply this to VLAN 17 interface (SVI).

> **NOTE**
> For L2 interfaces, the VLAN must have a Switch Virtual Interface (SVI).

```
nexus9000(config)# interface vlan 17
nexus9000(config-if)# ip access-group ILLUMIO_ACLS-Eth1-7-Inbound in
nexus9000(config-if)# ip access-group ILLUMIO_ACLS-Eth1-7-Outbound out
```

Result: The ACLs are now configured on the switch and any communication through VLAN 17 will be denied if it is not permitted in the ACLs.

Mark ACLs as Applied

You have to inform the PCE after you have loaded the ACLs because the NEN service does not configure the generated ACLs on the switch.

1. Log into the PCE web console.

2. From the PCE web console menu, choose **Infrastructure** > **Switches**.
3. Select the switch.
4. On the Switch page, click **Mark Applied**.

Flowlink Configuration and Usage

About Flowlink

The Flowlink application normalizes and aggregates the network flow data that it collects from different types of network sources into a format that can be ingested by the PCE for use by traffic data applications. It does not resolve any flow data source and destination IP addresses in to the PCE workloads. The PCE displays the flow in Illumination and marks the policy decision as 'unknown'. Flowlink is supported on standard PCE clusters and also on Supercluster.

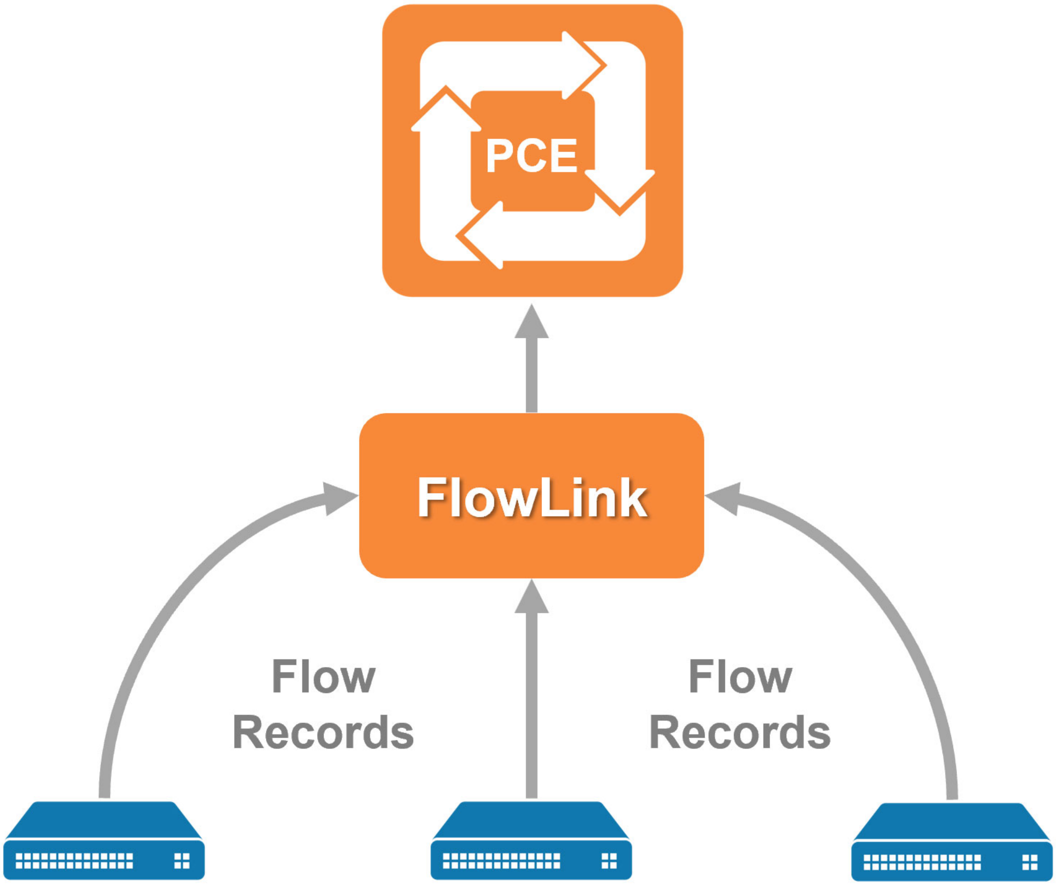
This section describes the Flowlink application, the types of flow records it supports, its scale, and limitations.

Overview

This section provides an overview of how Flowlink works and the supported flow records.

How Flowlink Works

Flowlink can receive the flow data by connecting to a data source provided by you and adheres to your organizations' data format. It may consume flows at a rate that is slower than the source speed. Therefore, the flow sender caches the flow data for 48 hours or more. If the PCE is unable to accept flow data because of the rate of flow or availability issues, Flowlink caches the data locally to a disk for a configurable period of time or disk space and retries periodically (user-configurable number of minutes). It aggregates data flows and sends them to the PCE once every configurable number of minutes. It does not have access to the PCE data and therefore no knowledge of workloads, virtual services, and other objects.



Supported Flow Record Formats

The following types of flow records are supported:

- AWS VPC flows
- IPFIX v10
- NetFlow v5, v7, v9, and v10
- sFlow v5
- Text (customizable parser configured by the user. For example, Syslog or Kafka).

Scale and Limitations

This section lists the supported scale and known limitations to be considered while using Flowlink.

PCE

- The PCE processes up to 10K unique flows/second. This is the total number of Flowlink and VEN flows received by the PCE.

- The PCE handles up to 20 concurrent POSTs.
- The PCE allows a maximum file size of 100MB per POST.
- For each IP address that exists in your data flows, you need to create corresponding unmanaged workloads in the PCE if you want to see those traffic flows in Illumination. Else, those flows will not be displayed.

Flowlink

- Flowlink supports multiple flow data sources.
- The maximum number of sources per Flowlink is not reported. As a best practice, consider one source per Flowlink.
- Flows with Class D addresses are ignored.
- Flowlink is not installed as a service, nor does it support a High Availability (HA) configuration. As such, it doesn't restart automatically if the host fails or is rebooted. In those cases, you need to restart Flowlink manually.
- The following two limitations are generic traffic limitations with Illumination and are not specific to Flowlink:
  - At least one IP address in the reported flow must match an IP address of a workload object (managed or unmanaged).
  - If a virtual service object and workload object have the same IP address, flow lines will always be drawn to the virtual service.

Flowlink Configuration

This section describes how to configure and run Flowlink.

Configure Flowlink

This section provides requirements and steps you need to follow to configure Flowlink.

Requirements

- CentOS or RHEL server
- Root privileges to the server
- Flowlink RPM downloaded from the Illumio Support site
- PCE with Service Account API Key and Secret

> **IMPORTANT**
> You must have Global Owner privileges to configure Flowlink.

CPU, Memory, and Storage Requirements

To install Flowlink, your hardware must meet the capacity requirements detailed in this section.

Machine Type	Cores/Clock Speed[1]	RAM per Node[2]	Storage Device Size[3] and IOPS[4]
Flowlink 2500 workloads	• 2 cores • Intel® Xeon(R) CPU E5-2695 v4 at 2.10GHz or equivalent	8 GB	• 1 x 20 GB • 100 IOPS per device

Footnotes:

[1] CPUs:

- The recommended number of cores is based only on physical cores from allocated CPUs, irrespective of hyper-threading or virtual cores. For example, in AWS one vCPU is only a single hyper-thread running on a physical core, which is half a core. 16 physical cores equates to 32 vCPUs in AWS.
- Full reservations for vCPU. No overcommit.

[2] Full reservations for vRAM. No overcommit.

[3] Additional disk notes:

- Storage requirements for network traffic data can increase rapidly as the amount of network traffic increases. Allocating a separate, large storage device for traffic data can accommodate these rapid changes without potentially interrupting the service.
- Network File Systems (NFS) is not supported.

[4] Input/output operations per second (IOPS) are based on 8K random write operations. IOPS specified for an average of 300 flow summaries (80% unique `src_ip, dest_ip`, `dest_port`, `proto`) per workload every 10 minutes. Different traffic profiles might require higher IOPS.

Flowlink Storage Partitioning

Storage Device	Partition mount point	Size to Allocate	Notes
Device 1, Partition A	/	20 GB	Logrotate must be configured to limit the disk consumption of Flow & System Logs.

Install Flowlink RPM

1. Login as a root user.
2. Install the RPM.

   The default install location is: `/usr/local/bin/`

   Standard installation:

   ```
   sudo su
   rpm -ivh illumio-flowlink-x.x.x-yy.x86_64.rpm
   ```

   For FIPS compliance (applies only to Flowlink version 1.2 and later; see [444] for more information):

```
sudo rpm -ivh --nodigest illumio-flowlink-1.2.0-104.x86_64.rpm
```

> **IMPORTANT**
> Only the Install Flowlink RPM [444] step needs root user login.
>
> The Create a Service Account API Key [445], Create YAML Configuration File [447], and Run Flowlink [447] steps can be run by logging in as any user.

In the following sections, `/home/employee` directory is used as an example. The `api_info` file should be in a directory writable by the user, for example in the `/home/employee` directory.

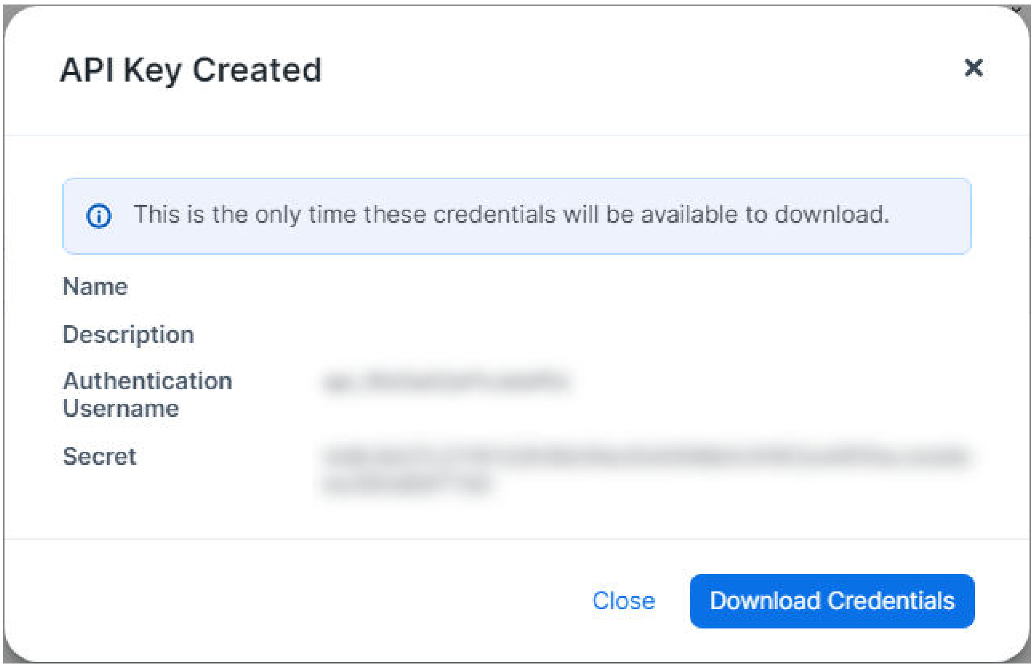
Create a Service Account API Key

> **NOTE**
> This procedure requires Global Owner privileges.

Keep in mind

- There are two ways to create a Service Account API key for Flowlink:
  - Through the API. See API Keys.
  - Through the PCE Web Console (described in the procedure below).
- The Org ID value is not shown when you create a Service Account API key.
- Service accounts are always organization-based and specific to a PCE. While creating a service account, users create their permissions and an `api_key` is created implicitly. Deleting a service account removes its permissions and all associated API keys.

1. In the PCE UI, go to **Access > Service Accounts**.
2. Click **Add** and configure settings.
   - **Name**
   - **Description** (optional)
   - **Access Restriction**: None.
   - **API Key expiration**: Keep the default or choose a different option.
   - **Roles and Scopes**: Select Global Administrator. The **All** is chosen automatically and cannot be changed.
3. Click **Save**.
4. When the *API Key Created* dialog appears, preserve the credentials (make a note or download them).



5. Copy the values of the **Authentication Username** and **Secret** into to a text file on the Flowlink server.
   Use a space to separate the key and secret. For example:
   `api_xxxxxxxxxxxxxx yyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyy`
6. Copy the absolute path of the file PCE API file `/home/employee/api_info`. You will need it in the Flowlink configuration file.

Configure HTTP/HTTPS Proxy

> **NOTE**
> Applies only to version 1.3.0 and later.

When Flowlink is running behind a proxy or in a corporate network and the PCE is in the cloud, Flowlink can access the PCE via HTTP/HTTPS proxy configurations.

The following configuration parameters are available to define an HTTP/HTTPS proxy:

```
proxy_config:
  https_proxy: <HTTPS_PROXY>
  http_proxy: {} <HTTPS_PROXY>{}
```

The following is an example of a Flowlink YAML configuration file:

```
proxy_config:
  https_proxy: http://proxy.corporate.com:3128
  http_proxy: http://proxy.corporate.com:3128
```

In the example above, the HTTP/HTTPS proxy is running on FQDN `proxy.corporate.com{{ port: 3128}}`.

Create YAML Configuration File

1. In the `/home/employee` directory, create a YAML configuration file. You can find an example yml file at `/usr/local/illumio/config.yml.example`.
2. Enter the parameters.
   Example of Flowlink configuration:

```
pce_addr: mypce.example.com:8443
api_key: $cat /home/employee/api_info
data_directory: /home/employee
aggregation_minutes: 10
consumers:
  - name: netflow
    parser:
      type: netflow
    connectors:
      - type: udp
        properties:
          ports: '2055'
```

   The above configuration listens for NetFlow on UDP 2055 from any data source. The absolute path is: `/home/employee/config.yaml.netflow`

Run Flowlink

1. To manage Flowlink, use the following commands:

```
illumio-flowlink-ctl start --config <path to config file> [--log-file
<path to log file>]
illumio-flowlink-ctl stop
illumio-flowlink-ctl status
```

   The default path for the log file is <data_directory specified in config file>/`flowlink.log`
2. To start Flowlink, use the `illumio-flowlink-ctl start` command. Make sure that you include the `--config` option in the start command, which will begin running the program in the background.
   Example with expected output:

```
illumio-flowlink-ctl start --config /home/employee/config.yaml.netflow

OUTPUT TO CONSOLE
Checking Flowlink started successfully.
OK.
Output logs can be found at: /home/employee/flowlink.log

OUTPUT IN LOG FILE (/home/employee/flowlink.log)
2020-03-11T09:58:51.173203-07:00 Waiting for signal
2020-03-11T09:58:51.330757-07:00 Starting Data Consumer: netflow
2020-03-11T09:58:51.331162-07:00 Listening for netflow messages on udp
port: 2055
2020-03-11T09:58:51.332929-07:00 Reporting flows every 10 minutes
```
3. To stop Flowlink, use the `illumio-flowlink-ctl stop` command.
   Example with expected output:

```
illumio-flowlink-ctl stop
```

```
OUTPUT ON CONSOLE
/illumio-flowlink-ctl stop
Stopping Flowlink: ......... Stopped.

OUTPUT IN LOG FILE (/home/employee/flowlink.log)
2020-03-11T09:58:57.097817-07:00 Got signal
2020-03-11T09:58:57.097835-07:00 Telling connectors to stop
2020-03-11T09:58:57.097856-07:00 Allowing parsers to drain
2020-03-11T09:58:57.098766-07:00 udp exiting
2020-03-11T09:58:57.098800-07:00 udp exiting
2020-03-11T09:58:57.101361-07:00 udp exiting
2020-03-11T09:58:57.101400-07:00 udp exiting
2020-03-11T09:58:57.103881-07:00 udp exiting
2020-03-11T09:58:57.103905-07:00 udp exiting
2020-03-11T09:58:57.106527-07:00 udp exiting
2020-03-11T09:58:57.106579-07:00 udp exiting
2020-03-11T09:58:57.109120-07:00 udp exiting
2020-03-11T09:58:57.109145-07:00 udp exiting
2020-03-11T09:58:57.111790-07:00 udp exiting
2020-03-11T09:58:57.111837-07:00 udp exiting
2020-03-11T09:58:57.113853-07:00 udp exiting
2020-03-11T09:58:57.113912-07:00 udp exiting
2020-03-11T09:58:57.116262-07:00 udp exiting
2020-03-11T09:58:57.116397-07:00 udp exiting
2020-03-11T09:58:57.118365-07:00 udp exiting
2020-03-11T09:58:57.119002-07:00 udp exiting
2020-03-11T09:58:57.120865-07:00 udp exiting
2020-03-11T09:58:57.121108-07:00 udp exiting
2020-03-11T09:58:57.123517-07:00 udp exiting
2020-03-11T09:58:57.123552-07:00 udp exiting
2020-03-11T09:58:57.126043-07:00 udp exiting
2020-03-11T09:58:57.126079-07:00 udp exiting
2020-03-11T09:59:02.100923-07:00 Writing flows
2020-03-11T09:59:02.100969-07:00 Flow count: 48468
2020-03-11T09:59:02.417261-07:00 Waiting for file senders to drain
2020-03-11T09:59:02.418564-07:00 Sending file: /home/employee/
traffic_flows_1583945942416835.pb.gz
2020-03-11T09:59:07.390307-07:00 Response Code 204
```

4. To check the status of Flowlink, use the `illumio-flowlink-ctl status` command. Example with expected output:

```
illumio-flowlink-ctl status

OUTPUT ON CONSOLE
/illumio-flowlink-ctl status
Flowlink: RUNNING
```

Configure YAML

Flowlink requires configurable parameters using a YAML file.

NOTE

Refer to the `/usr/local/illumio/flowlink_config_schema.json` file provided with the Flowlink RPM for definitions of all the fields supported by the Flowlink configuration file.

Key Value Parameters

This table describes the YAML file key-value parameters.

Parameter	Required/ Optional	Description
`aggregation_minutes`	Optional	The interval (in minutes) in which flows are aggregated and sent to the PCE. Default interval: 10 Minimum allowed interval: 5 Maximum allowed interval: 60 For example: `aggregation_minutes: 10`
`api_key`	Required	API key and secret of the PCE. This allows Flowlink to POST flows to the PCE. The API key and secret can be copied into a file. You can run a script to *cat* the contents of that file. In the example below, a file called *api_info* is created which contains the PCE API key and secret. For example: `api_key: $cat /home/employee/api_info`
`consumers`	Required	A list of dictionaries. It requires a name, parser, and connector. Flowlink configuration supports one or many consumers (flow types). For more details about configuring the ingested flow types, see Ingested Flow Types [451].
`data_directory`	Required	The pathname of a directory where Flowlink can store any unsent data flow files or any restart information. For example: `data_directory: /home/employee/`
`data_directory_size_mb`	Optional	The maximum size (in Megabytes) of data that can be stored in the data directory before being pruned. Default: 500 Minimum value: 100 For example: `data_directory_size_mb: 200`
`file_retention_hours`	Optional	The maximum number of hours unsent data flow files will be stored before being pruned. Default: 24 Minimum: 4 For example:

Parameter	Required/ Optional	Description
		`file_retention_hours: 8`
`metrics_print_seconds`	Optional	The frequency (in seconds) at which the metrics information is printed. Default: 60 Minimum: 15 For example: `metrics_print_seconds: 60`
`org_id`	Required for SaaS Optional for on-premises	The org id to which the flow data will be posted. The default id is 1. For example: `org_id: 1`
`pce_addr`	Required	FQDN of the PCE and port. For example: `pce_addr: https://mypce.example.com:8443`

Ingested Flow Types

This section provides the Consumer Syntax when using various supported parsers and connectors.

IPFIX, NetFlow, and sFlow Parsers

```
consumers:
  - name:  # Required. An array of properties defining the data consumers
 configured for Flowlink. For example: netflow
    parser:
      type: #Required. Information describing the parser associated with
      the data consumer. List of supported values: 'netflow', 'ipfix',
      'sflow', 'aws', or 'text'
    connectors:
      - type: #Required. Information describing the data source connector
      associated with the data consumer. Supported values: 'udp', 'tcp',
     'kafka', or 'aws'
        properties:
          ports: #Required parameter to describe tcp or udp port.
          For example: '2055'
          remote_addrs: #Optional parameter. String or list of IP
          address(es) to listen for as trusted data sources. Default is
          allow all IPs. CIDRs are not supported. For example:
          '192.168.1.10,192.168.1.15'.
```

AWS Parser and Connector

```
consumers:
  - name:  # Required. An array of properties defining the data consumers
    configured for Flowlink. For example: aws
    parser:
      type: #Required. Information describing the parser associated with
      the data consumer. Supported value: aws
    connectors:
      - type: #Required. Information describing the data source connector
       associated with the data consumer. Supported value: aws
        properties:
          region: #Required. Configures the AWS region of where the VPC
          flow logs are stored. Value not wrapped in quotes.
          Examples: us-west-2 or us-east-1
          credentials: #Required. This is the AWS Access Key ID and
          AWS Access Key Secret created by IAM. The IAM user must have
          privileges to read Cloud Watch logs. You can put the contents
          into a file and run a script to cat the file.
          Value not wrapped in quotes. For example:
          $cat /home/employee/aws_info
          log_groupname: #Required. The name of the AWS Log Group.
          Value not wrapped in quotes. For example: myVPCFlowLogs
```

NOTE

The Access Key ID and Key Secret format should be the same as defined in YAML Configuration.

Text Parser with TCP or UDP Connector

```
consumers:
  - name: # Required. An array of properties defining the data consumers
    configured for Flowlink. For example: syslog
    parser:
      type: #Required. Information describing the parser associated with
      the data consumer. Supported value: 'text'
      properties:
        src_ip: #Required. Attribute tag or field number (starting at 1)
        used to extract source IP. For example: sip
        dst_ip: #Required. Attribute tag or field number (starting at 1)
        used to extract destination IP. For example: dip
        dst_port: #Required. Attribute tag or field number (starting at 1)
        used to extract destination port. For example: dport
        protocol: #Required. Attribute tag or field number (starting at 1)
        used to extract protocol. For example: prot
        icmp_type: #Optional. Attribute tag or field number (starting at 1)
        used to extract icmp type. For example: type
        icmp_code: #Optional. Attribute tag or field number (starting at 1)
        used to extract icmp code. For example: code
        timestamp: #Optional. Attribute tag or field number (starting at 1)
        used to extract timestamp. Default: 1. For example: "date_time, 1"
```

```
      timestamp_format: #Optional. A string used to describe the
      timestamp format field(s) in a record. The following values can be
      used year: yy[yy], month(Jan[uary] etc): mmm[mmm], dayOfMonth:
      dd or _d, dayOfWeek(Mon[day], etc):
      ddd[ddd], hour: HH, minutes: MM, seconds(with optional precision):
      SS[.0{1 or more}], timeZone: ZZZ, -HH[:MM], -HHMM, ZHH[:MM],
      ZHHMM, unix timestamp: unix. For example: "mm dd yyyy HH:MM:SS"
  connectors:
    - type: #Required. Information describing the data source connector
     associated with the data consumer. List of supported values: 'tcp',
     'udp', or 'sctp'
      properties:
        ports: #Required parameter to describe tcp or udp port.
        For example: '514'
        remote_addrs: #Optional. A comma separated list of remote host
        addresses from which to accept flows.
        For example: '192.168.200.13'
```

Text Parser with Kafka Connector

```
consumers:
  - name: # Required. An array of properties defining the data consumers
  configured for Flowlink. For example: syslog
    parser:
      type: #Required. Information describing the parser associated with
      the data consumer. Supported value: 'text'
      properties:
        src_ip: #Required. Attribute tag or field number used to extract
        source IP. For example: sip
        dst_ip: #Required. Attribute tag or field number used to extract
        destination IP. For example: dip
        dst_port: #Required. Attribute tag or field number used to extract
        destination port. For example: dport
        protocol: #Required. Attribute tag or field number used to extract
        protocol. For example: prot
        icmp_type: #Optional. Attribute tag or field number used to extract
        icmp type. For example: type
        icmp_code: #Optional. Attribute tag or field number used to extract
        icmp code. For example: code
        timestamp: #Optional. Attribute tag or field number used to extract
        timestamp. For example: "date_time, 1"
        timestamp_format: #Optional. A string used to describe the
        timestamp format field(s) in a record. The following values can be
        used year: yy[yy], month(Jan[uary] etc): mmm[mmm], dayOfMonth:
        dd or _d, dayOfWeek(Mon[day], etc):
        ddd[ddd], hour: HH, minutes: MM, seconds(with optional precision):
        SS[.0{1 or more}], timeZone: ZZZ, -HH[:MM], -HHMM, ZHH[:MM], ZHHMM,
        unix timestamp: unix. For example: "mm dd yyyy HH:MM:SS"
    connectors:
      - type: kafka
        properties:
          version: #Required. The version of the kafka broker(s).
          For example: 1.2.0
          brokers: #Required. A comma separated list of kafka brokers
          using FQDN and port. For example: example.com:9092
          group: test
```

```
          topics: test
          client_id: flowlink
```

Ingested Flow Examples

This section provides flow examples while using the supported parsers and connectors.

IPFIX

The below example shows a consumer that listens for IPFIX on UDP 4739 coming only from an IPFIX exporter whose IP address is 192.168.11.5. The flows from other IPFIX exporters will be discarded.

```
consumers:
  - name: ipfix
    parser:
      type: ipfix
    connectors:
      - type: udp
        properties:
          ports: '4739'
          remote_addrs: '192.168.11.5'
```

NetFlow

The below example is using NetFlow in which Flowlink will parse NetFlow records via UDP 6500 and listen for any data source IP address.

```
consumers:
  - name: netflow
    parser:
      type: netflow
    connectors:
      - type: udp
        properties:
          ports: '6500'
```

AWS

The below example is of an AWS consumer in which the CloudWatch Log Group name is myVPCFlowLogs and is configured in the AWS Oregon region.

```
consumers:
  - name: aws
    parser:
      type: aws
    connectors:
      - type: aws
        properties:
          region: us-west-2
          credentials: $cat /home/employee/aws_info
          log_groupname: myVPCFlowLogs
```

Text

The below example is of a text consumer using Syslog and listening on UDP 6514. The syslog format uses sip attribute to extract the source IP of the flow.

```
consumers:
  - name: syslog
    parser:
      type: text
      properties:
        src_ip: sip
        dst_ip: dip
        dst_port: dport
        protocol: prot
        timestamp: "date_time, 1"
        timestamp_format: "mmm dd yyyy HH:MM:SS"
    connectors:
      - type: udp
        properties:
          ports: "6514"
```

YAML

```
pce_addr: 2x2mypce.example.com:8443
api_key: $cat api_info
data_directory: /home/employee/
aggregation_minutes: 5
consumers:
  - name: netflow
    parser:
      type: netflow
    connectors:
      - type: udp
        properties:
          ports: '6500'
  - name: ipfix
    parser:
      type: ipfix
    connectors:
      - type: udp
        properties:
          ports: '6514'
```

FIPS Compliance for Flowlink

This section describes the operational requirements for compliance with Federal Information Processing Standard (FIPS) 140-2 for Illumio Flowlink.

The Federal Information Processing Standard Publication (FIPS PUB) 140-2 is a U.S. government computer security standard used to approve cryptographic modules. An authorized cryptographic equipment assessment laboratory has tested and verified that Flowlink faithfully incorporates the use of cryptographic functions provided by the FIPS 140-2 validated modules as it applies to data in transit.

FIPS Prerequisites

The server on which Flowlink is installed must be running a FIPS-validated version of RHEL 8 -- such as RHEL 8.2 -- in FIPS mode and satisfy the Security Policy as stated in Red Hat Enterprise Linux 8 OpenSSL Cryptographic Module version rhel8.20200305.1

Enable Flowlink FIPS Compliance

1. After installing RHEL8.x, follow the required steps in Section 9.1, Crypto Officer Guidance, Red Hat Enterprise Linux 8 OpenSSL Cryptographic Module NIST Security Policy.
2. Reboot the system.
3. After the system starts, check that FIPS mode is enabled:

```
$ fips-mode-setup --check
FIPS mode is enabled.
```

4. Install the Flowlink RPM using this command:

```
sudo rpm -ivh --nodigest illumio-flowlink-1.2.0-104.x86_64.rpm
```

5. To configure Flowlink, see Configure Flowlink [443].

When you've completed this procedure, Flowlink is FIPS compliant.

Flowlink Usage

This section describes how to export IPFIX or NetFlow v9 flow records from F5 BIG-IP to an external flow collector and some solutions while troubleshooting.

Collect Flow Records from F5

The example listed in the following steps uses a virtual edition of the F5 BIG-IP appliance in AWS and the Illumio Flowlink application to gather and parse flow data.

IMPORTANT

IPFIX and NetFlow have slightly different configuration steps depending on which flow record standard you choose.

Requirements

- Flowlink (flow collector)
- F5 BIG-IP system with LTM
- A virtual server configured on F5 box

NOTE

F5 must have a self-IP interface. The flows are sent out of this interface. When Flowlink is not in the same subnet as the self-IP, you must know the default gateway IP of the self-IP interface.

Create a Pool for Flow Collector

To create a pool of flow collectors to receive the flow record messages from the F5 system:

1. In the F5 UI, click **Main > Local Traffic > Pools > Pool Lists > Create**.
2. Enter a unique name in the **Name** field, which represents the flow collector.
3. A *Health Monitor* is not required. If you want to see if the F5 system can reach the flow collector, select `gateway_icmp` and move it to the Active box.
4. In the **New Member** section, configure the collector IP address.
5. Click **Add**.
   If you are using **IPFIX**, use the following configuration:

Field	Value
Node Name	Enter the Collector IP address
Service Port	4739

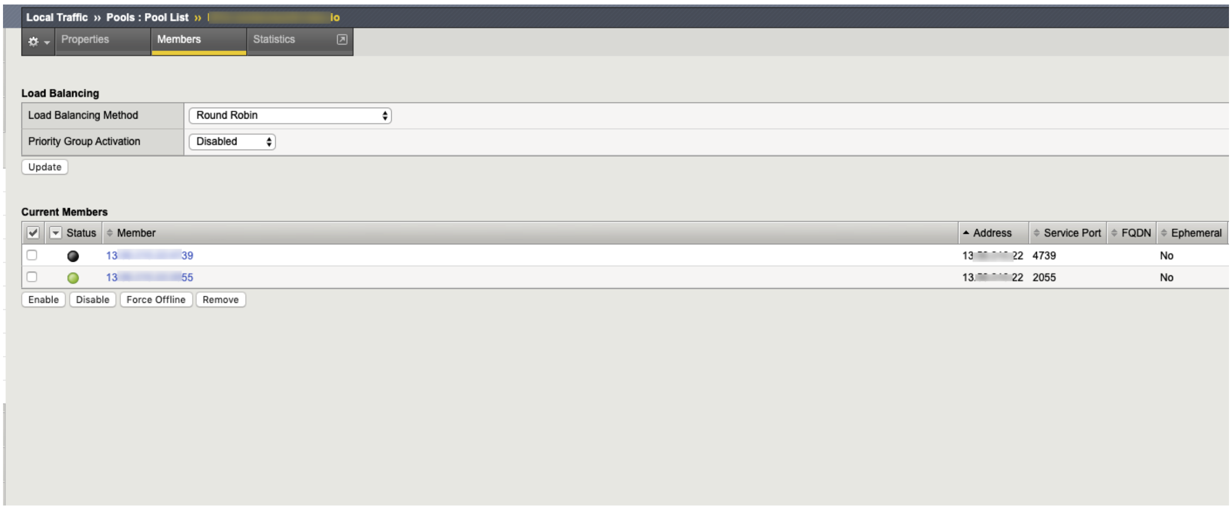
   If you are using **NetFlow**, use the following configuration:

Field	Value
Node Name	Enter the Collector IP address
Service Port	2055

6. Click **Finished**.

The below example shows two (2) different nodes configured in one pool. Both nodes have the IP address. However, one is for IPFIX and one is for NetFlow. Even though F5 allows two nodes in the pool, it is recommended to only have one node enabled (either 2055 or 4739).

Example with NetFlow enabled and IPFIX disabled:



Create a Log Destination

To create a log destination to stream the logs in either IPFIX or NetFlow V9 format to the Pool:

1. In the F5 UI, click **Main > System > Logs > Configuration > Log Destinations > Create**.
2. Enter a unique name in the **Name** field, which represents the flow collector.
3. In the **Type** field, select IPFIX.
4. Configure the IPFIX Settings.
   If you are using **IPFIX**, use the following configuration:

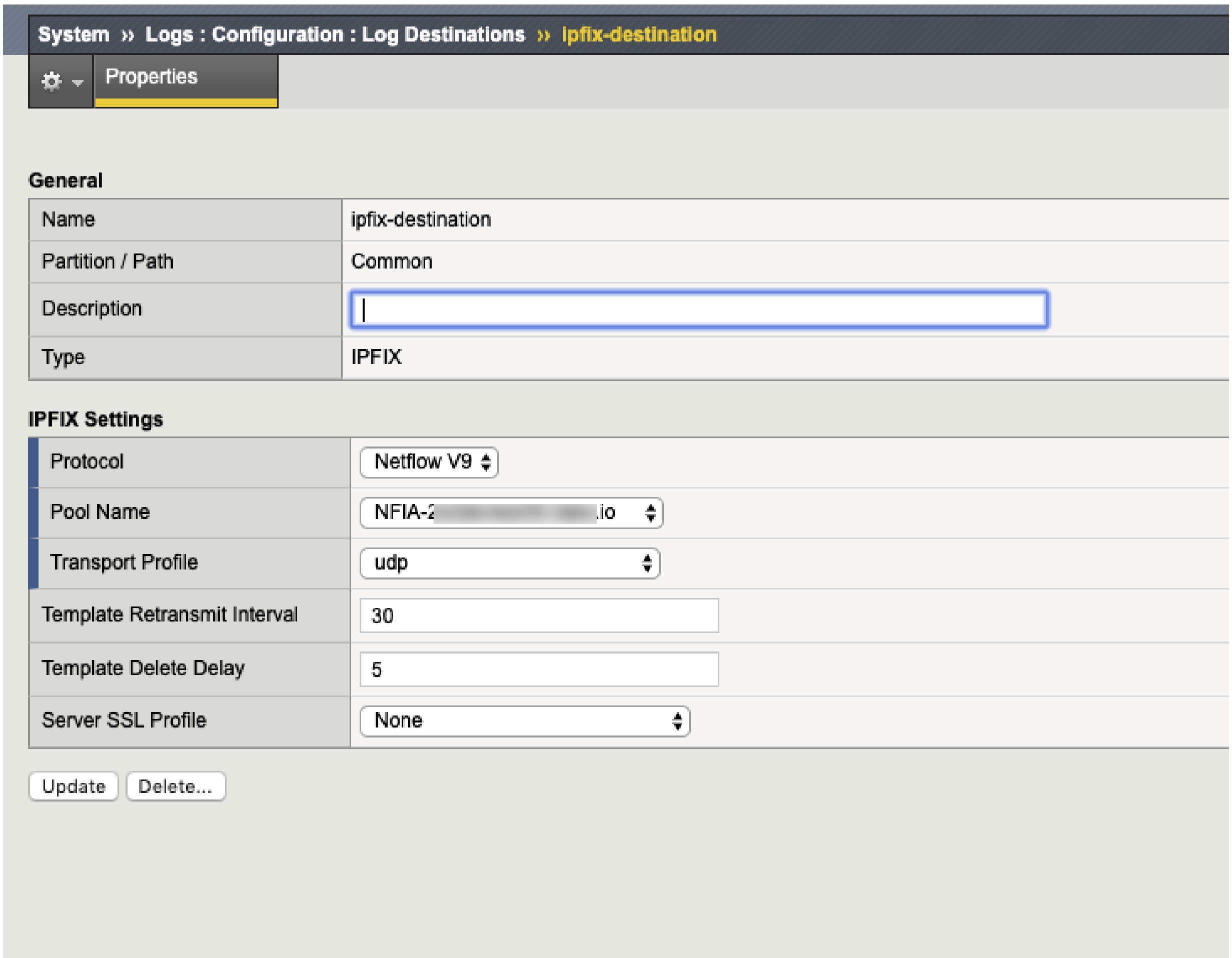
Field	Value
Protocol	Select IPFIX
Pool Name	Select the pool created earlier
Transport Profile	UDP

   If you are using **NetFlow**, use the following configuration:

Field	Value
Protocol	Select NetFlow V9
Pool Name	Select the pool created earlier
Transport Profile	UDP

5. Click **Finished**.

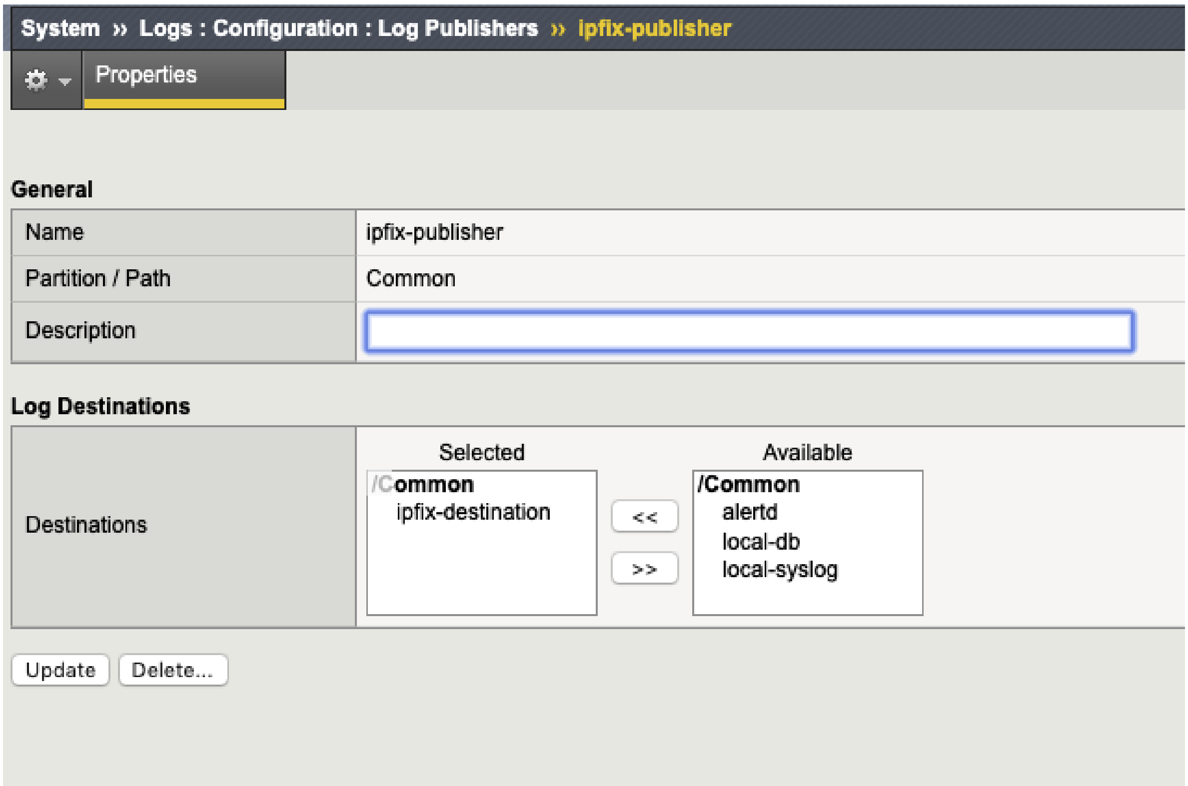
Example of a Log Destination configuration with NetFlow:



Create a Log Publisher

To create a log publisher to send logs to the specified log destination:

1. In the F5 UI, click **Main > System > Logs > Configuration > Log Publishers > Create**.
2. Enter a unique name in the **Name** field, which represents the flow collector.
3. In the **Destination** field, move your log destination from *Available* to *Selected*.
4. Click **Finished**.



Create an iRule

To create an iRule to which it parses network traffic and sends flow records to the specified log publisher:

1. Go to **Main > iRules > iRule List > Create**.
2. Enter a unique name in the **Name** field, which represents the flow collector.
3. In the **Definition** text field, enter the rules for parsing traffic. Ensure the iRule points to the *log publisher* created earlier.
4. Click **Finished**.

> **IMPORTANT**
> In the iRule example shown below, replace *<insert_log_publisher_name_here>* with the name of the log publisher.

```
when RULE_INIT {
  set static::http_rule1_dest ""
  set static::http_rule1_tmplt ""
}


# CLIENT_ACCEPTED event to initiate IPFIX destination and template
when CLIENT_ACCEPTED {
  set start [clock clicks -milliseconds]
  if { $static::http_rule1_dest == ""} {
    # open the logging destination if it has not been opened yet
```

```
    set static::http_rule1_dest [IPFIX::destination open -publisher /Common/
    <insert_log_publisher_name_here>]
  }
  if { $static::http_rule1_tmplt == ""} {
    # if the template has not been created yet, create the template
    set static::illumio_flowlink_POC_tmplt [IPFIX::template create
"flowStartMilliseconds
    sourceIPv4Address sourceIPv6Address destinationIPv4Address
destinationIPv6Address
    sourceTransportPort destinationTransportPort protocolIdentifier
octetTotalCount
    packetTotalCount octetDeltaCount packetDeltaCount
postNATSourceIPv4Address
    postNATSourceIPv6Address postNATDestinationIPv4Address
postNATDestinationIPv6Address
    postNAPTSourceTransportPort postNAPTDestinationTransportPort
postOctetTotalCount
    postPacketTotalCount postOctetDeltaCount postPacketDeltaCount
flowEndMilliseconds "]

  }
  set rule1_msg1 [IPFIX::msg create $static::http_rule1_tmplt]
}

# SERVER_CONNECTED event to initiate flow data to specified log publisher
and populate 5
tuples
when SERVER_CONNECTED {
  set client_closed_flag 0
  set server_closed_flag 0
  IPFIX::msg set $rule1_msg1 flowStartMilliseconds $start
  IPFIX::msg set $rule1_msg1 protocolIdentifier [IP::protocol]

  # Clientside
  if { [clientside {IP::version}] equals "4" } {
    # Client IPv4 address
    IPFIX::msg set $rule1_msg1 sourceIPv4Address [IP::client_addr]
    # BIG-IP IPv4 VIP address
    IPFIX::msg set $rule1_msg1 destinationIPv4Address [clientside
{IP::local_addr}]
  } else {
    # Client IPv6 address
    IPFIX::msg set $rule1_msg1 sourceIPv6Address [IP::client_addr]
    # BIG-IP IPv6 VIP address
    IPFIX::msg set $rule1_msg1 destinationIPv6Address [clientside
{IP::local_addr}]
  }
  # Client port
  IPFIX::msg set $rule1_msg1 sourceTransportPort [TCP::client_port]
  # BIG-IP VIP port
  IPFIX::msg set $rule1_msg1 destinationTransportPort [clientside
{TCP::local_port}]

  # Serverside
  if { [serverside {IP::version}] equals "4" } {
```

```
    # BIG-IP IPv4 self IP address
    IPFIX::msg set $rule1_msg1 postNATSourceIPv4Address [IP::local_addr]
    # Server IPv4 IP address
    IPFIX::msg set $rule1_msg1 postNATDestinationIPv4Address
[IP::server_addr]
  } else {
    # BIG-IP IPv6 self IP address
    IPFIX::msg set $rule1_msg1 postNATSourceIPv6Address [IP::local_addr]
    # Server IPv6 IP address
    IPFIX::msg set $rule1_msg1 postNATDestinationIPv6Address
[IP::server_addr]
  }
  # BIG-IP self IP port
  IPFIX::msg set $rule1_msg1 postNAPTSourceTransportPort [TCP::local_port]
  # Server port
  IPFIX::msg set $rule1_msg1 postNAPTDestinationTransportPort
[TCP::server_port]
}

# SERVER_CLOSED event to collect IP pkts and bytes count on serverside
when SERVER_CLOSED {
  set server_closed_flag 1
  # when flow is completed, BIG-IP to server REQUEST pkts and bytes count
  IPFIX::msg set $rule1_msg1 octetTotalCount [IP::stats bytes out]
  IPFIX::msg set $rule1_msg1 packetTotalCount [IP::stats pkts out]
  # when flow is completed, server to BIG-IP RESPONSE pkts and bytes count
  IPFIX::msg set $rule1_msg1 octetDeltaCount [IP::stats bytes in]
  IPFIX::msg set $rule1_msg1 packetDeltaCount [IP::stats pkts in]
    IPFIX::destination send $static::http_rule1_dest $rule1_msg1
}

# CLIENT_CLOSED event to collect IP pkts and bytes count on clientside
when CLIENT_CLOSED {
  set client_closed_flag 1
  # when flow is completed, client to BIG-IP REQUEST pkts and bytes
octetDeltaCount
  IPFIX::msg set $rule1_msg1 postOctetTotalCount [IP::stats bytes in]
  IPFIX::msg set $rule1_msg1 postPacketTotalCount [IP::stats pkts in]
  # when flow is completed, BIG-IP to client RESPONSE pkts and bytes count
  IPFIX::msg set $rule1_msg1 postOctetDeltaCount [IP::stats bytes out]
  IPFIX::msg set $rule1_msg1 postPacketDeltaCount [IP::stats pkts out]
  # record the client closed time in ms
  IPFIX::msg set $rule1_msg1 flowEndMilliseconds [clock click -milliseconds]
    # send the IPFIX log
    IPFIX::destination send $static::http_rule1_dest $rule1_msg1
}
```

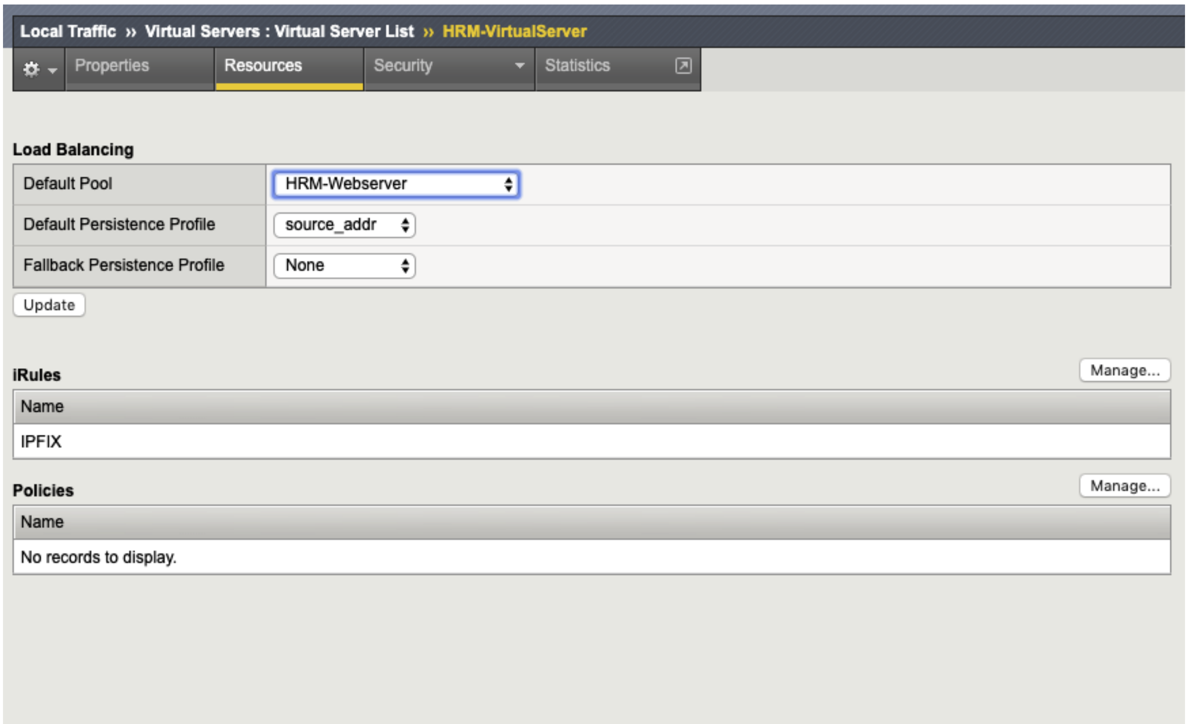
Apply the iRule to a Virtual Server

To apply the iRule to a virtual server whose traffic you want to parse:

1. Go to **Main > Virtual Server > Virtual Server List**.
2. Select the virtual server you want to monitor.
3. Click the **Resources** tab. In the iRule section, click **Manage**.
4. Select the **iRule** that you previously created and move the iRule from *Available* to *Enable*.

5. Click **Finished**.

   Example of a Virtual Server Resources page with the new iRule applied:



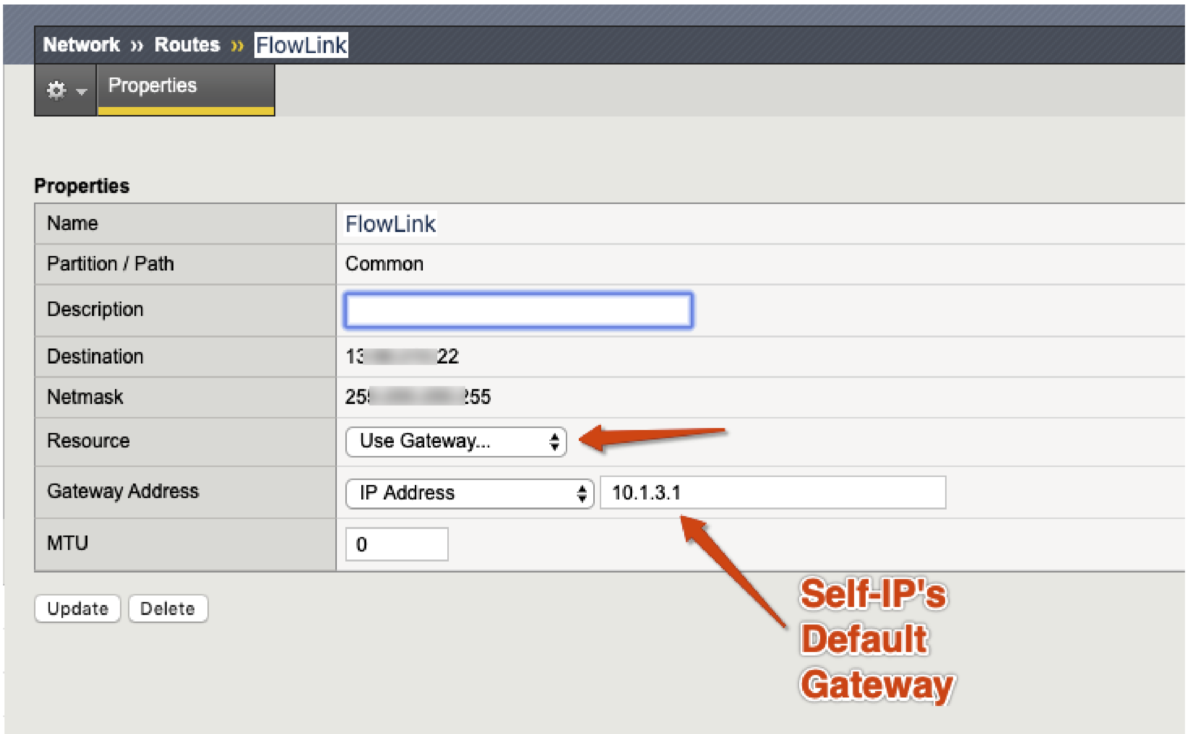
Create a Route Entry

By default, all traffic is sent out of the management interface. However, F5 does not support flow exports via the management NIC. You must add a route to force traffic, which is destined to the flow collector to leave a self-IP interface.

To create a route entry, if the F5 self-IP is unable to reach the flow collector:

1. In the F5 UI, click **Main > Network > Routes > Add**.
2. In the **Properties** section, create a route entry to send the flow records from F5 to the external flow collector IP address.

   For Resource, select the *Use Gateway* option.



Troubleshooting

This section describes how to troubleshoot some issues when configuring or using Flowlink.

Flowlink not Receiving Data

1. Make sure iptables is turned *Off* on Flowlink, or make sure iptables is not blocking the ports that Flowlink is listening on.
2. Use `netstat -a` to make sure Flowlink is listening on the correct ports.

> **NOTE**
> netstat has a bug, which shows that applications are only listening with IPv6 on listed ports, when they are actually listening on those ports with IPv4.

Unable to Ping or TCPdump on the F5 Self-IP Interface

1. SSH to F5 as an administrator.
2. List the interfaces to see the interface names.

```
admin@(ip-10-1-1-197)(cfg-sync Standalone)(Active)(/Common)(tmos)#
show net interface

----------------------------------------------------------------
Net::Interface
Name  Status     Bits     Bits     Pkts     Pkts  Drops  Errs  Media
                   In      Out       In      Out
```

```
----------------------------------------------------------------
1.1       up    1.3G    1.1G    2.6M    2.6M      0     0   none
1.2       up  177.7M  301.4M  298.9K  310.4K      0     0   none
mgmt      up  310.9G  876.6G  298.8M  325.5M      0     0   none
```

3. Run TCPdump to listen for traffic between Self-IP interface and flow collector IP.
4. Generate traffic while the TCPdump is running by either opening another SSH session and doing PING test or by sending normal traffic through the virtual server. If you turned on **health monitoring** with gateway_icmp enabled from the Create a Pool for Flow Collector [457] section, then F5 should already generate ICMP traffic.

   The example shown below uses interface name *1.2* with flow collector *IP 13.56.210.22*. Health monitoring with *gateway_icmp* is enabled.

```
admin@(ip-10-1-1-197)(cfg-sync Standalone)(Active)(/Common)(tmos)#
tcpdump -
ni 1.2 host 13.56.210.22
tcpdump: verbose output suppressed, use -v or -vv for full protocol
decode
listening on 1.2, link-type EN10MB (Ethernet), capture size 65535 bytes
09:08:47.855318 IP 10.1.3.223 > 13.56.210.22: ICMP echo request, id
54351,
seq 37906, length 20 out slot1/tmm3 lis=
09:08:47.857694 IP 13.56.210.22 > 10.1.3.223: ICMP echo reply, id 54351,
seq 37906, length 20 in slot1/tmm3 lis=
09:08:52.864852 IP 10.1.3.223 > 13.56.210.22: ICMP echo request, id
54354,
seq 37906, length 20 out slot1/tmm2 lis=
09:08:52.867091 IP 13.56.210.22 > 10.1.3.223: ICMP echo reply, id 54354,
seq 37906, length 20 in slot1/tmm2 lis=
```

Network Connectivity

The flow to test network connectivity is:

- Network device > Flowlink
- Flowlink > PCE

TCPdump

To use TCPdump:

- Run on a network device to verify flow records are sent out.
- Run on Flowlink to verify flow records are coming in.

Debug Option

Flowlink has a debug option that displays:

- Incoming flow records
- IP, port, and protocol recorded for flow records
- Each time flows are aggregated and uploaded to the PCE
- PCE response code to `POST`

To debug Flowlink in the session, add the `--debug` flag to your Flowlink command.

Example with the debug option enabled:

```
CONFIG_FILE=/home/employee/config.yaml.netflow /usr/local/bin/illumio/
flowlink --debug
```

IMPORTANT

Using the debug flag, generates a large amount of data to the console. Enable this option only if needed.

Legal Notice

Resources

- Legal information
- Trademarks statements
- Patent statements
- License statements

Contact Information

- Contact Illumio
- Contact Illumio Legal
- Contact Illumio Documentation