



Cisco Security Cloud Control: Mesh Policy Engine

Cisco Security Cloud Control
Updated NaN,



© 2023–2025 Cisco Systems, Inc. All rights reserved.

Full Cisco Trademarks with Software License

Full Cisco Trademarks with Software License

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

All printed copies and duplicate soft copies of this document are considered uncontrolled. See the current online version for the latest version.

Cisco has more than 200 offices worldwide. Addresses and phone numbers are listed on the Cisco website at www.cisco.com/go/offices.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: <https://www.cisco.com/c/en/us/about/legal/trademarks.html>. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

Topics included

1 Mesh Policy Engine Overview.....	6
Introduction to Mesh Policy Engine.....	7
Benefits of Mesh Policy Engine.....	7
Capabilities of Mesh Policy Engine.....	7
Mesh Policy Engine API.....	8
Persisted Connection Management overview.....	9
Mesh Policy Engine UI and Persisted Connection Management API comparison.....	9
Getting Started with Mesh Policy Engine.....	10
Claim a Mesh Policy Engine subscription.....	10
Assign a Mesh Policy Engine admin role.....	15
Access the Mesh Policy Engine API.....	17
2 Install Targets.....	18
Prerequisites for install targets.....	19
Make secure connections to Mesh Policy Engine for third-party security devices.....	20
Manage credentials.....	20
Create an install target: Firewall Threat Defense.....	23
Create an install target: ASA.....	24
Create the ASA intermediary manager.....	24
Associate ASA install targets with the manager.....	26
Create an install target: Fortigate, Palo Alto, Juniper.....	28
Create an install target: AMD Pensando (API).....	30
View install targets.....	33
3 Domains, Dynamic Objects, and Topology.....	34
Configure domains.....	35
Manage dynamic objects (API).....	36
Create a topology.....	37
Construct your topology (API).....	38
View your topology.....	43
Sample Topology.....	46
4 Policies.....	50
About Policies.....	51
Priority rulesets.....	52
Core rulesets.....	53
Default rulesets.....	53
NAT rulesets.....	54

Guidelines for policies.....	58
Configure networks.....	59
Create a network.....	59
Create a network group.....	62
Configure services.....	64
Create a service.....	64
Create a service group.....	66
Configure rulesets.....	67
Create a ruleset.....	67
Create a rule.....	69
Filter Rules.....	72
Push a ruleset to an install target.....	74
Import policies.....	75
Imported policy mapping.....	75
Import a policy.....	86
View Previous Imports.....	91

5 Changesets.....94

About changesets.....	95
Changeset workflow.....	95
Manage changesets.....	99
Rebase a changeset.....	100
Validate and commit a changeset.....	102

1 Mesh Policy Engine Overview

Topics:

- [Introduction to Mesh Policy Engine](#)
- [Benefits of Mesh Policy Engine](#)
- [Capabilities of Mesh Policy Engine](#)
- [Mesh Policy Engine API](#)
- [Getting Started with Mesh Policy Engine](#)

An overview of Mesh Policy Engine for centrally managing, tracking, and deploying security policies across multivendor firewalls.

Mesh Policy Engine helps you orchestrate and manage security policies across security devices from multiple vendors in one dashboard. It simplifies policy updates, tracks ownership and version history, and ensures consistent policy deployment. This approach makes policy management easier and strengthens network security as your organization grows.

Introduction to Mesh Policy Engine

Learn how Mesh Policy Engine helps teams manage security policies consistently across complex multivendor and multicloud networks.

Managing network security policies can be challenging when your organization has complex, enterprise-level, multivendor networks and needs to move to large-scale multicloud architectures. Your network operators or application development team often creates or edits security policies to support increased connectivity and access for product teams and developers across your organization. These frequent changes can clutter policy management and make it hard to scale your enterprise network as your organization grows.

In addition, because there are a large number of rules in the policy, it becomes even harder to manage and keep track of the policies' versions, changes, and ownership history. If you do not manage these complex and changing security policies carefully, your organization's network security could weaken and become more vulnerable to malicious threats.

To address these needs, you can use Mesh Policy Engine in Security Cloud Control, which helps you orchestrate and manage security policies across security devices from multiple vendors. With Mesh Policy Engine, you can track policy ownership, policy change history, and version control. Mesh Policy Engine makes policy management easier and helps keep your network security strong.

Using Mesh Policy Engine, you can view all your security policies on one dashboard, create new policies, update existing ones, and deploy them to security devices. After you import or create a policy in Mesh Policy Engine, you can deploy it to any security device, irrespective of the security device vendor.

Benefits of Mesh Policy Engine

Find the operational and security benefits of using Mesh Policy Engine to manage enterprise firewall policies across vendors and environments.

Implementing Mesh Policy Engine in your enterprise environment delivers these key business outcomes:

- **Reduced operational complexity**—Eliminate the need for specialized expertise in each vendor's CLI or management console. Managing all security devices from a single location reduces the overhead of maintaining a multi-vendor environment.
- **Accelerated agility**—Policy updates that traditionally took weeks to coordinate across different teams and vendors can be implemented consistently across the entire infrastructure in minutes.
- **Enhanced device performance**—Placing only relevant rules on specific devices reduces the size of policy documents on enforcement points, which improves device performance and makes troubleshooting easier.
- **Improved security posture**—Using Mesh Policy Engine as the "source of truth" for all workload access strengthens your security posture. Mapping each policy rule to your original request or intent simplifies annual audits and ensures your security intent remains clear during translation.
- **Automated policy consistency**—Your defined security intent is maintained across the network. If someone makes a manual change directly on a device, Mesh Policy Engine restores your intended policy during the next deployment.
- **Scalability**—Built on an elastic cloud architecture, Mesh Policy Engine automatically scales to handle large enterprise environments with high request volumes and complex, dynamic policy requirements.

Capabilities of Mesh Policy Engine

Find details about Mesh Policy Engine capabilities for multivendor policy management, reusable policy objects, lifecycle tracking, and policy import and deployment.

Mesh Policy Engine is a centralized, intent-based policy automation feature within Security Cloud Control. Its primary capabilities include:

- Multi-vendor policy orchestration—Supports a diverse range of enforcement points, including Cisco Secure Firewall (ASA and Firewall Threat Defense), Palo Alto Networks, Fortinet, Juniper, and AMD Pensando DPUs. It uses vendor-supported APIs to write policies directly to third-party devices.
- Intent-based policy translation—Instead of manually writing complex firewall rules, you define your intent (e.g., "App A needs to talk to App B"). Mesh Policy Engine translates your intent into specific, outcome-based rules and places them correctly across your network.
- Comprehensive policy ingestion—Ingests existing policy and enforcement state from managed security devices, ensuring that legacy rules are maintained and integrated with new intent-based policies.
- Dynamic object and group management—Supports static IPv4 objects and dynamic objects. When you update dynamic objects using the APIs, Mesh Policy Engine updates the relevant policies automatically.
- Advanced rule ordering and optimization—Categorizes rules into Priority, Core, and Default to influence ordering, and removes overlapping or duplicate rules before deployment.
- Topology-aware policy deployment—You define the relationships between objects and enforcement points, and Mesh Policy Engine uses this topology to deploy policies only to devices relevant to the traffic path, avoiding unnecessary policies on other devices.
- Robust lifecycle management:
 - Every change is versioned, allowing for historical comparisons and rollbacks.
 - Detailed records track what was changed, when, and by whom.
 - RESTful APIs enable you to integrate Mesh Policy Engine into CI/CD workflows with tools such as Terraform or Ansible.

Mesh Policy Engine API

Learn how to use Mesh Policy Engine APIs to manage security policies, connectivity requirements, policy objects, and device deployments.

The Mesh Policy Engine API lets you create, update, and manage security policies for a device.

The Mesh Policy Engine consists of several key modules, each featuring its own API operations:

- Manual Policy Manager (MPM)—These API operations enable network administrators to configure and manage policies in the Mesh Policy Engine UI. The operations enable create, read, update, and delete (CRUD) on MPM changesets, manage MPM policies, handle related entities, track request status, and more. *These functions are supported in the Mesh Policy Engine UI.*
- Persisted Connection Management (PCM)—These API operations let users declare their connectivity requirements as permit-only rules. These rules are placed in the environment through their relationship to topology. These API operations support CRUD on PCM policies, network groups, service groups, related entities, and tracking of PCM request status.
- Topology—These API operations facilitate the configuration and management of gateway sets, zones, addresses, and more. *The only topology functions that the Mesh Policy Engine UI supports are viewing.* To create the topology, you must use the topology API. Refer to [Create a topology](#) on page 37.
- Policy Import—These API operations allow users to import policies into MPM and track the status of the request. *These functions are supported in the Mesh Policy Engine UI.*

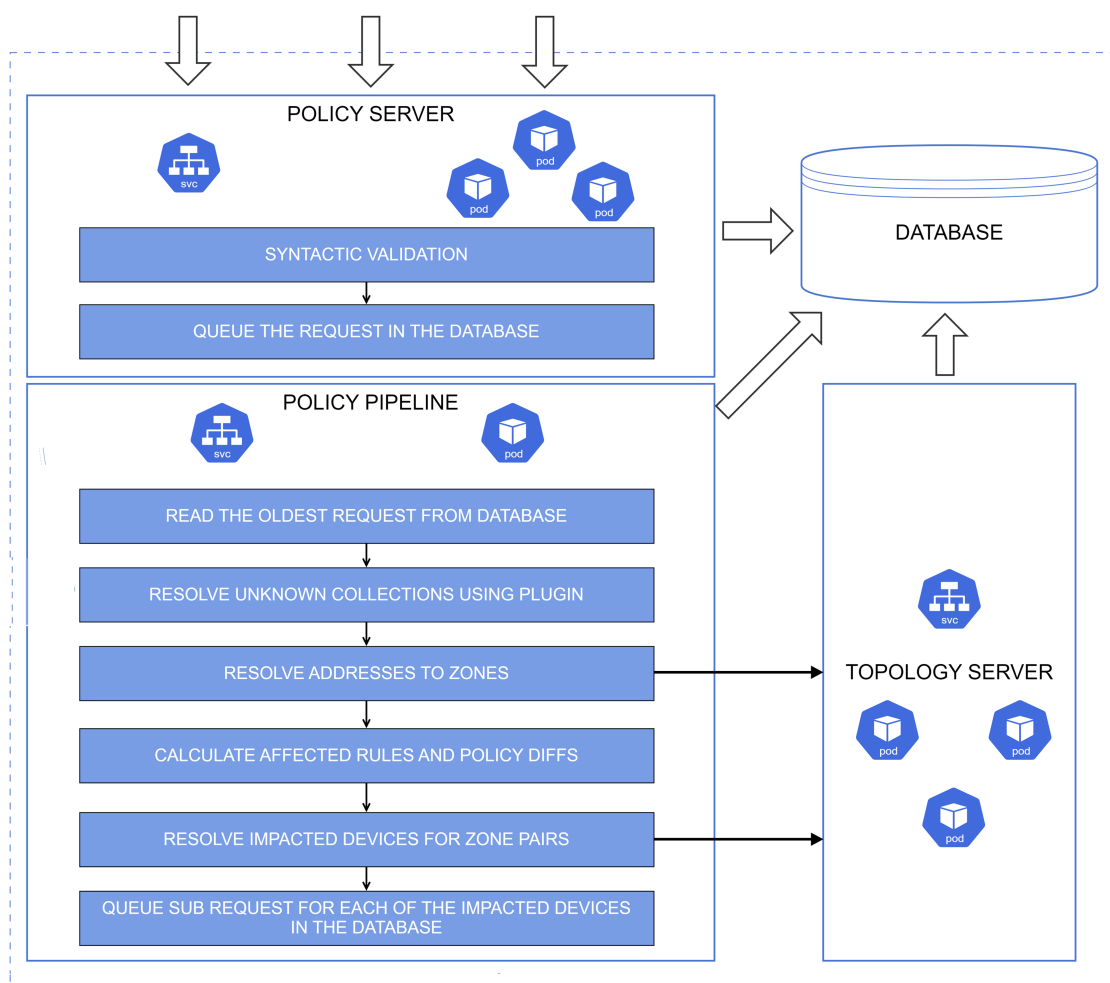
For more information, refer to the [Mesh Policy Engine API](#).

Persisted Connection Management overview

An overview of the Persisted Connectivity Manager API for creating, viewing, updating, and deleting connectivity requirements in Mesh Policy Engine.

The Persisted Connection Management (PCM) for create, read, update, and delete (CRUD) operations, helping you manage your connectivity requirements. The PCM comprises two subcomponents:

- **Policy server (pcm-server):** This component exposes the APIs that receive user connectivity requirements. These statements of connectivity intent are a set of *Allow* only rules. After the request is received, the server validates the request and queues the request in the database to be processed by the pipeline.
- **Policy pipeline (pcm-pipeline):** This is an upstream pipeline processor. It dequeues the request and processes it to determine which security devices need to be reprovisioned. For any change performed by the user, the impacted rules are identified and the policy pipeline resolves the rule's endpoint groups and app groups, if any, to leaf endpoints and apps. For every new endpoint, Mesh Policy Engine determines which zone it is placed in the zone topology. Each rule's terminal zone pair is determined, and all the enforcement points in the affected zone pairs are identified using the zone topology and enqueued to the fulfillment layer.



To learn about the PCM APIs, refer to the [Mesh Policy Engine API](#).

Mesh Policy Engine UI and Persisted Connection Management API comparison

Compare Manual Policy Manager and Persisted Connectivity Manager to choose between interface-based policy management and API automation.

Mesh Policy Engine uses the Manual Policy Manager (MPM) API component to serve policy management needs through the user interface, while the Persisted Connection Management (PCM) API component provides automated policy management flows. The other significant differences between Mesh Policy Engine and Persisted Connection Management are listed in this table.

Mesh Policy Engine UI	Persisted Connection Management API
In most cases, a network operator leverages the Manual Policy Manager capabilities.	In most cases, an application developer who uses network connectivity intent to automate policy flows leverages the API capabilities.
You can do all the policy management activities using the Mesh Policy Engine graphical user interface . You can also use the Manual Policy Manager API component directly, though we recommend that you manage policies using the graphical user interface.	PCM APIs perform all the policy management activities based on user-defined automation workflows .
Changesets track all the edits across policies, rules, endpoints, endpoint groups, applications, and application groups.	PCM APIs create or update policies, rules, endpoints, endpoint groups, and application groups.
Users can manually deploy policies, policy versions, or policy changes to install targets.	Users get to automate policy deployment using PCM APIs.
The Mesh Policy Engine allows the import of policies from supported firewalls in a vendor-agnostic policy format.	There are no import PCM APIs.
The policies defined by the Mesh Policy Engine can contain both allow and deny statements . You can specify which actions are allowed or denied.	The policies defined by the PCM API can contain only allow statements . Because these policies permit only explicit allow rules, any other actions are denied.
Mesh Policy Engine supports four types of policies—Core, Default, Priority, and NAT.	The PCM API supports only Core policies.



Note

- Any entity in Mesh Policy Engine, such as a policy, cannot be managed by both the Mesh Policy Engine and the PCM API simultaneously.
- Note that PCM API-managed **endpoints** can be referenced as Mesh Policy Engine **dynamic objects**. This allows automated deployment of changes to Mesh Policy Engine install targets without manual intervention.

Getting Started with Mesh Policy Engine

Set up Mesh Policy Engine in Cisco Security Cloud Control, activate a subscription, assign administrator access, and obtain an API key.

Set up Mesh Policy Engine in Cisco Security Cloud Control to manage policies across multi-vendor security devices.

Claim a Mesh Policy Engine subscription

Learn about included, trial, and paid Mesh Policy Engine subscriptions and the activation option for each type.

Security Cloud Control includes several types of subscriptions:

- **Paid subscription**—A paid subscription provides the full-functionality Mesh Policy Engine. This type of subscription requires a claim code that you can activate for a product instance. Refer to [Claim a paid or trial Mesh Policy Engine subscription](#) on page 11.
- **Trial subscription**—A trial subscription provides the full-functionality Mesh Policy Engine. This type of subscription requires a claim code that you can activate for a product instance. Trials can be stacked with paid subscriptions to extend the subscription period. Refer to [Claim a paid or trial Mesh Policy Engine subscription](#) on page 11.
- **Included subscription**—If you have a paid or trial subscription for another product, such as Firewall Management, it may have an included subscription, which supports a basic-functionality version of Mesh Policy Engine. You need to manually enable this type of subscription, but you don't need a separate claim code. Refer to [Enable an included Mesh Policy Engine subscription](#) on page 14.

 Note

The included version of Mesh Policy Engine does not include support for third-party devices.

If you have an included subscription of a product such as Firewall Management, it doesn't also have an included subscription for Mesh Policy Engine; only the paid or trial subscription of Firewall Management has an included subscription of Mesh Policy Engine.

For more information about subscriptions, see [Security Cloud Control Administration Guide](#).

Claim a paid or trial Mesh Policy Engine subscription

Claim a paid or trial Mesh Policy Engine subscription in Cisco Security Cloud Control with the claim code from your welcome email.

Using the claim code from your welcome email, claim the Mesh Policy Engine subscription within your organization.

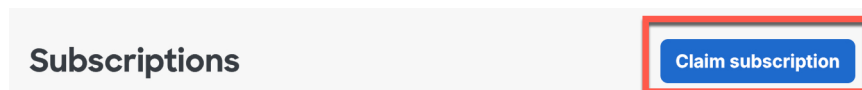
Before you begin

Set up your Security Cloud Sign On account and sign in. Use the [Account Sign Up](#) page to set up a Security Cloud Sign On account or to sign in to your existing account.

Procedure

1. From the Security Cloud Control Home page, choose **Product Management >Platform Management > Product Subscriptions > Subscriptions**.
2. Click **Claim subscription**.

Figure 1: Claim subscription



3. Enter the claim code and click **Next**.

Figure 2: Subscription claim code

Claim Subscription

1 Enter subscription claim code

2 Review products

3 Review subscription

Enter subscription claim code

To begin, enter your claim code below and click **Next**. For detailed instructions please read our [documentation](#).

Subscription claim code *

< Cancel Next

4. For Cisco Mesh Policy Engine Essential, choose **Create new instance** from the drop-down list.

Figure 3: Create new instance

Claim Subscription

✓ Enter subscription claim code

2 Review products

3 Review subscription

Review products

To use an existing instance instead of creating a new one, choose **Attach existing instance** from the dropdown menu for the product. Post-claim actions may also be required to apply licenses to some product instances. [Read documentation for more details](#).

Products	Create new instance or attach existing ⓘ
Cisco Mesh Policy Engine Essential	Create new instance ▼

< Cancel Back Next

For trial subscriptions only—if you already have a Mesh Policy Engine instance and are applying a paid subscription to replace an existing trial subscription or are stacking a trial subscription on top of an existing paid subscription, choose **Attach existing instance**.

5. If you have any other products on this claim code, make your selections, and then click **Claim**.

Figure 4: Claim

Claim Subscription

✓ Enter subscription claim code

✓ Review products

3 Review subscription

Review subscription

Subscription ID: PreviewSubPGM0004RB Organization region ⓘ: North America

Products	Deployments	Instance type
Cisco Mesh Policy Engine Essential	United States ✓	New

<

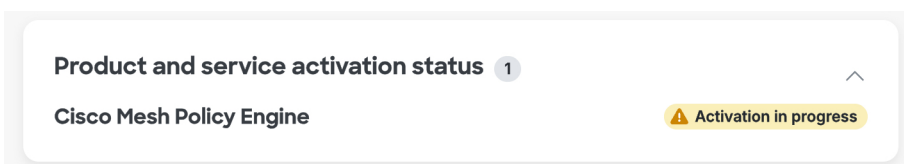
Cancel

Back

Claim

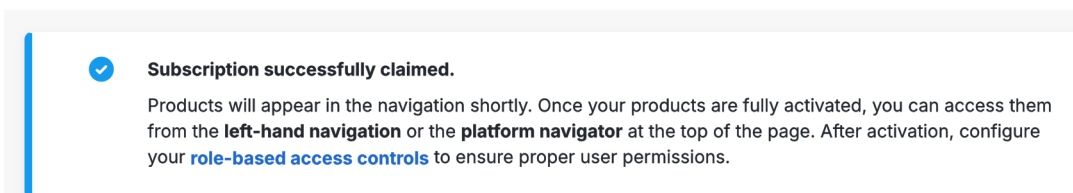
You may see this message during activation:

Figure 5: Activation in progress



You will see a message at the top of the page showing success.

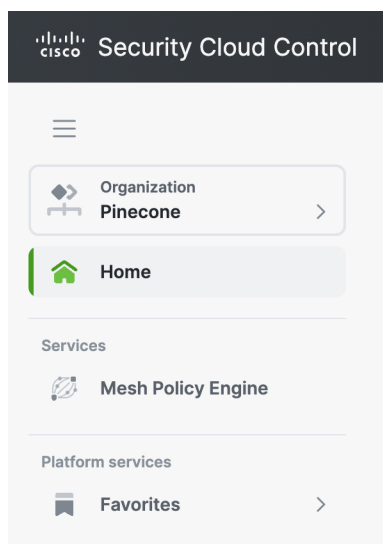
Figure 6: Product successfully claimed



6. Add a Mesh Policy Engine role for other users before they can use Mesh Policy Engine.

Mesh Policy Engine only appears in the Security Cloud Control menu under **Services** for users or groups that have an assigned Mesh Policy Engine role. As the organization administrator, you are probably a member of the All Products Administrator group, so you are assigned the Mesh Policy Engine admin role automatically. Refer to [Assign a Mesh Policy Engine admin role](#) on page 15.

Figure 7: Services menu



Enable an included Mesh Policy Engine subscription

Enable an included Mesh Policy Engine subscription from Cisco Security Cloud Control and open the activated service.

If you have an included subscription for Mesh Policy Engine, follow these steps.

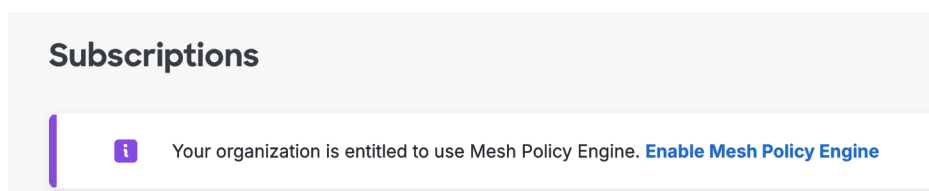
Before you begin

Set up your Security Cloud Sign On account and sign in. Use the [Account Sign Up](#) page to set up a Security Cloud Sign On account or to sign in to your existing account.

Procedure

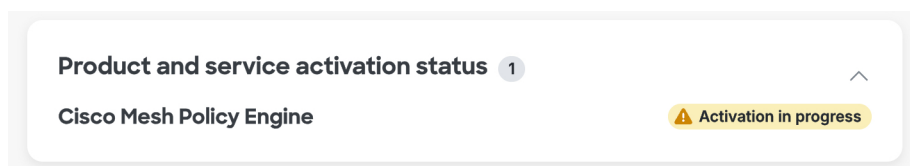
1. From the Security Cloud Control Home page, choose **Product Management > Platform Management > Product Subscriptions > Subscriptions**.
2. In the notifications banner that appears on the **Subscriptions** page, click **Enable Mesh Policy Engine**.

Figure 8: Enable the included subscription



You may see this message during activation:

Figure 9: Activation in progress



You will see a message at the top of the page showing success.

Figure 10: Product successfully claimed



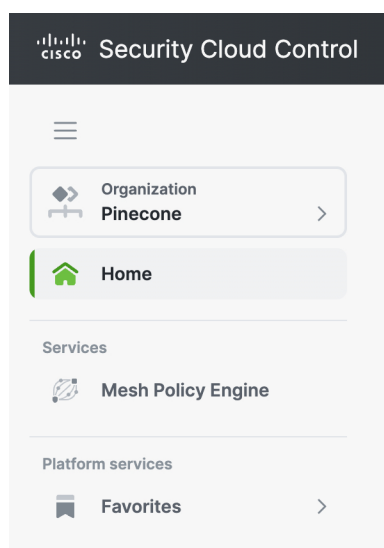
Subscription successfully claimed.

Products will appear in the navigation shortly. Once your products are fully activated, you can access them from the **left-hand navigation** or the **platform navigator** at the top of the page. After activation, configure your **role-based access controls** to ensure proper user permissions.

3. Add a Mesh Policy Engine role for other users before they can use Mesh Policy Engine.

Mesh Policy Engine only appears in the Security Cloud Control menu under **Services** for users or groups that have an assigned Mesh Policy Engine role. As the organization administrator, you are probably a member of the All Products Administrator group, so you are assigned the Mesh Policy Engine admin role automatically. Refer to [Assign a Mesh Policy Engine admin role](#) on page 15.

Figure 11: Services menu



Assign a Mesh Policy Engine admin role

Assign a Mesh Policy Engine administrator role so a user can access and manage the service.

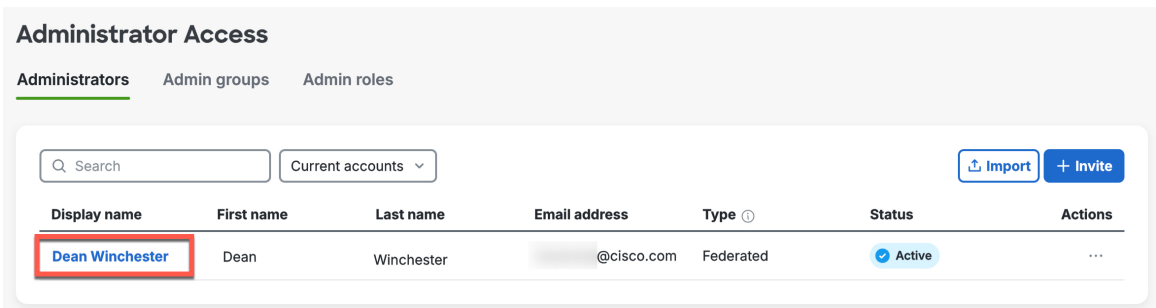
Users can't access Mesh Policy Engine until you assign a Mesh Policy Engine admin role. This procedure describes how to assign a role to each user separately.

You can alternatively choose the role and then assign users to the role ([Assign Roles to Existing Users](#)), or you can assign roles to groups ([Assign Roles to Groups](#)).

Procedure

1. From the Security Cloud Control Home page, choose **Platform Management > Access Management > Administrator Access**.
2. On the **Administrators** page, with **Current Accounts** selected in the drop-down list, click the **Display Name** of the user.

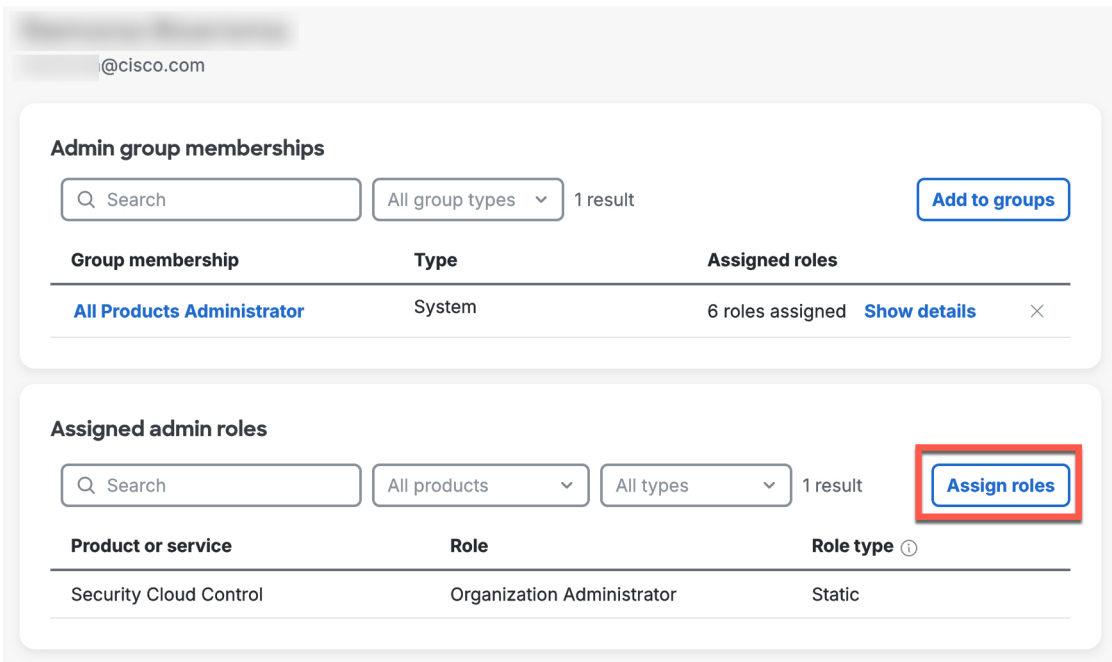
Figure 12: Display name



The resulting page displays user details such as assigned admin roles and admin groups.

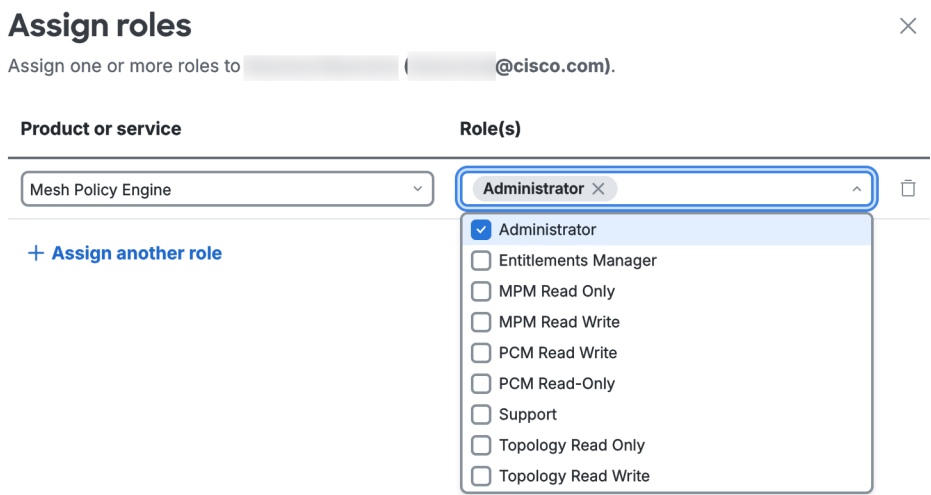
3. Click **Assign roles**.

Figure 13: Assign roles



4. In the **Assign Roles** slide-in pane, assign roles to products.

Figure 14: Assign roles per product



- a) From the **Product or service** drop-down list, choose a product.
- b) From the **Roles** drop-down list, choose one or more roles to assign.

Mesh Policy Engine supports static roles.

Here is a list of static roles and their level of access to APIs.

Table 1: Mesh Policy Engine Static Roles

Role	Description
Administrator	Provides full platform authority with access to all API operations.
Entitlements Manager	Provides limited authority for managing client credentials. Does not provide full platform access.
Support	Provides access to a broad set of API operations for support and operational tasks.
PCM Read-Only	Provides read-only access to Persisted Connection Management (PCM) API operations.
PCM Read Write	Provides read and write access to PCM API operations.
Topology Read Only	Provides read-only access to Topology API operations. (performed by network operators).
Topology Read Write	Provides read and write access to Topology API operations. (performed by network operators).
MPM Read Only	Provides read-only access to Manual Policy Manager (MPM) API operations.
MPM Read Write	Provides read and write access to MPM API operations.

- c) Click **+Assign another role** to add more products for the user.
- d) Click **Assign roles**.

The selected product roles are assigned to the user.

Access the Mesh Policy Engine API

Learn how to obtain the API key required to use Mesh Policy Engine APIs.

Many functions in Mesh Policy Engine must be performed in the API. To use the Mesh Policy Engine API, you must first obtain an API key. You can acquire this API key directly from the Security Cloud Control user interface. For more information, see [Mesh Policy Engine API](#).

2 Install Targets

Topics:

- [Prerequisites for install targets](#)
- [Make secure connections to Mesh Policy Engine for third-party security devices](#)
- [Manage credentials](#)
- [Create an install target: Firewall Threat Defense](#)
- [Create an install target: ASA](#)
- [Create an install target: Fortigate, Palo Alto, Juniper](#)
- [Create an install target: AMD Pensando \(API\)](#)
- [View install targets](#)

Learn how to prepare, add, and manage firewall install targets for security policy deployment from Mesh Policy Engine.

An install target is a firewall or group of firewalls, like a high-availability pair or cluster, that shares the same security policy.

Prerequisites for install targets

Find supported firewall vendors, software versions, policy capabilities, and operating modes required for Mesh Policy Engine install targets.

Supported devices

Device or Manager	Software Version	Security Policies Supported?	NAT Policies Supported?	Mode	Note
AMD Pensando Distributed Service Card (DSC) managed by Policy and Services Manager (PSM) and Policy Distribution Target (PDT)	1.64.5-E-1	Yes	No	Cluster	
AMD Pensando DSC managed by PSM and PDT	1.64.5-E-2	Yes	No	Cluster	Timeouts are not supported for non-RPC ALGs (DNS, FTP, TFTP, RTSP).
Cisco Secure Firewall ASA managed by Security Cloud Control Firewall Management	Any	Yes	No	As supported by Security Cloud Control Firewall Management	
Cisco Secure Firewall Threat Defense managed by Security Cloud Control Firewall Management	Any	Yes	No	As supported by Security Cloud Control Firewall Management	
Fortinet FortiGate	7.4.3	Yes	Yes	Standalone	
Fortinet FortiManager	7.4.4	Yes	No	Policy import	
Juniper	22.4R3.25	Yes	Yes	Multinode high-availability mode	
Juniper	22.4R3.25	Yes	Yes	Cluster mode	
Palo Alto Networks	PAN-OS 11.1.4-h7	Yes	No	Active/passive mode Active/active mode	

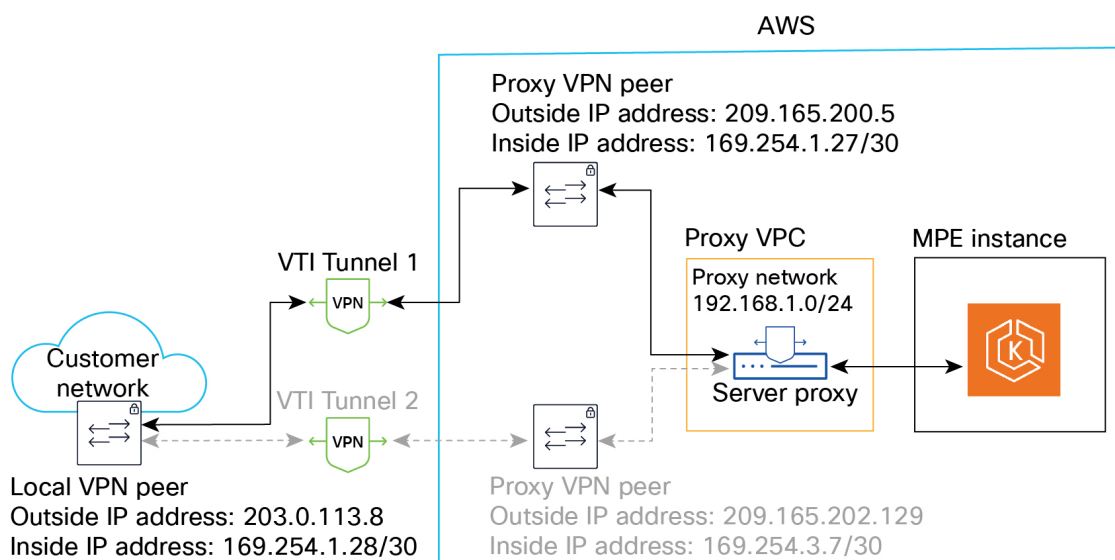
Make secure connections to Mesh Policy Engine for third-party security devices

Learn how to establish a secure proxy connection between Mesh Policy Engine and third-party security devices over an AWS VPN.

You need to work with the Cisco product team to establish a secure connection for third-party security devices. For Cisco security devices, a separate proxy connection is not required.

To access and update policies on your third-party security devices, Mesh Policy Engine must securely connect to your on-premises network through a proxy over an AWS VPN. The proxy must be set up in a dedicated virtual private cloud (VPC) with a user-defined subnet and must be protected by a username and password. With this setup, only your tenant can access the proxy.

Figure 15: Secure Connection to Mesh Policy Engine



To have a secure tunnel configured for your Mesh Policy Engine, you must provide these details to the Cisco product team:

- Private **/24** subnet. Ensure that this subnet does not overlap with the existing subnets on your network.
- DNS server details: Details of the DNS server that the proxy must use.
- VPN connection specifics:
 - Device IP address for VPN termination
 - Supported encryption algorithms
 - Asymmetric key ciphers
 - Hashes

Manage credentials

Manage credentials that Mesh Policy Engine uses to connect to install targets and intermediary managers through the interface or API.

To securely connect to the install targets, Mesh Policy Engine must authenticate the install targets using their credentials. Create a local copy of your credentials using JSON. Each credential is a reusable object that you can use for one or more

install targets by name. This procedure lets you upload all of your credentials in bulk. If you want to update a credential, you need to paste in the complete credential set; Mesh Policy Engine replaces any existing credentials with the ones you upload.



Note

You don't need to add Firewall Threat Defense credentials because they are imported directly from Security Cloud Control.

Procedure

1. For the Security Cloud Control Firewall Management intermediary manager for ASA, generate a TOKEN type credential.
 - a) From the Security Cloud Control Home page, click **Firewall**.
 - b) Choose **Administration > API User Management**, and click the Add icon (+).
 - c) Specify a **Username**, choose the **Deploy Only** Role, and click **Add**.

Figure 16: API username and role

Grant API Only Access to CDO_cisco-techdocs__te5ww0

Username
dwinchester

Role
Deploy Only

Close Add

- d) Click **Generate API Token**.

Figure 17: Generate API token

User	Last Login	Token	Roles	
dwinchester@CDO_cisco-techdocs__te5ww	-	☐ No API Token ☒ Generate API Token	Deploy Only	

- e) Click **Copy API Token** and save it for use to add to the credentials, and then click **OK**.

Figure 18: Copy API token



Use this token in your local JSON credentials file.

2. Create a local JSON file with your named credential objects.

Use the following JSON example showing the two credential types, USERNAME_PASSWORD and TOKEN:

```
{
  "integrationCredentials": {
    "Credential_1": {
      "type": "USERNAME_PASSWORD",
      "usernamePasswordCredentials": {
        "username": "username_1",
        "password": "password_1"
      }
    },
    "Credential_2": {
      "type": "TOKEN",
      "tokenCredentials": {
        "token": "token_1"
      }
    }
  }
}
```

3. From the Security Cloud Control Home page, click **Mesh Policy Engine.**

4. Choose **Install Targets > Credentials > Manage Credentials.**

5. Paste the JSON text for all credentials into the window and click **Submit.**

Figure 19: Client Credentials

Client Credentials

Enter a valid JSON blob to update client credentials.

[illegible]

Clear

Submit

When you add install targets and topology, you will refer to the credential name. In the above graphic, the name shown is "fw-mgr."

Create an install target: Firewall Threat Defense

Import a Firewall Threat Defense device from Cisco Defense Orchestrator as an install target in Mesh Policy Engine.

For Firewall Threat Defenses already present in Security Cloud Control and managed by Firewall Management, you need to import them into Mesh Policy Engine to make them install targets. When you deploy a Mesh Policy Engine policy, it uses the organization's Firewall Management instance. You don't need to manually add Firewall Management as an intermediate manager for Firewall Threat Defense.

Procedure

1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Install Targets > Import Devices**.
3. From the list of devices, select the Firewall Threat Defense that you want to import, and click **Import**.

Figure 20: Import devices

Import devices

Import FTD devices for centralized policy management.

[Create import](#)
[Previous import requests](#)
[Import devices](#)

i

Imported devices is limited to compatible devices already connected to Security Cloud Control.

To add more devices to the list go to [Security Devices](#) and onboard additional compatible devices.

Actions	Name	Import Status	UID
	FTD-Demo-Device	Imported	a26fecda-5cbe-474d-9ae7-48bd7d80a904

What to do next

Import policies from Firewall Management into Mesh Policy Engine. Alternatively, create new policies and manage them through Mesh Policy Engine.

Create an install target: ASA

Learn how to prepare ASA install targets for policy management with an intermediary manager in Mesh Policy Engine.

For the ASA, you need to associate the Firewall Management instance as an intermediary manager with Mesh Policy Engine and then add the ASA install targets to the intermediary manager. When Mesh Policy Engine deploys an ASA policy, it only deploys it to Firewall Management; you have to manually deploy the policy from within Firewall Management.

Create the ASA intermediary manager

Connect Cisco Defense Orchestrator as an intermediary manager to add and manage ASA install targets in Mesh Policy Engine.

To manage the ASA, Mesh Policy Engine pushes the policy to Security Cloud Control Firewall Management. You then have to manually deploy the policies from Firewall Management. To add ASAs as install targets, first connect Firewall Management to Mesh Policy Engine and then associate ASAs with the intermediary manager.

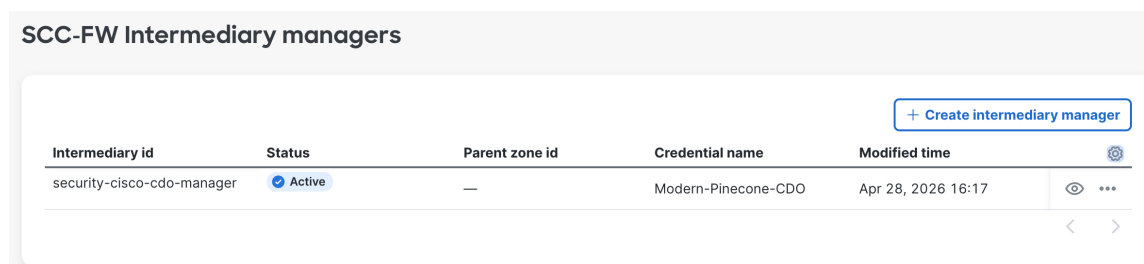
Before you begin

Add ASAs to Security Cloud Control Firewall Management.

Procedure

1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Install Targets > Install Targets > Intermediary Managers > SCC-FW**.

Figure 21: SCC-FW Intermediary managers



Intermediary id	Status	Parent zone id	Credential name	Modified time	
security-cisco-cdo-manager	Active	—	Modern-Pinecone-CDO	Apr 28, 2026 16:17	⋮

+ Create intermediary manager

3. Click **Create intermediary manager**. Click **Add host**, and set these values:

Figure 22: Create SCC-FW intermediary manager

Create SCC-FW intermediary manager

ID *

Status *

Hosts *

Hostname *

Port *

Priority

+ Add host

Credential name *

Parent zone id

[Cancel](#) [Save](#)

- **ID**—A name of your choice.
- **Status**—**ACTIVE**. Always choose this option; INACTIVE is not supported.
- **Hostname**—**security.cisco.com**. If you are in a staging environment, check for your specific hostname in the URL.
- **Port**—**443**.
- **Priority**—Leave blank. You can only have once instance of Firewall Management.
- **Credential name**—The name of the token credential you added using [Manage credentials](#) on page 20.
- (Optional) **Parent zone id**—If you have configured zone hierarchy in your topology, this setting links the manager to a parent zone.

You can only add one instance of Firewall Management to Mesh Policy Engine.

4. Click **Save**.
5. To edit the manager, click the Eye icon (👁).

Figure 23: Edit the intermediary manager

security-cisco-cdo-manager

ID

security-cisco-cdo-manager

Status

Active

Parent zone id

—

Credential name

Modern-Pinecone-CDO

Modified time

4/28/2026, 4:17:02 PM

Hosts

Host name	Port	Priority
us.manage.security.cisco.com	443	1

Intermediary manager actions

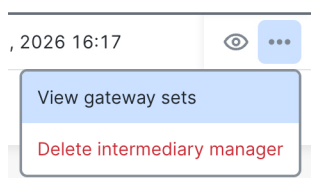
[View gateway sets](#)

[Delete](#)

Ok

6. To delete the manager, click the More icon (...), and choose **Delete intermediary manager**.

Figure 24: View gateway sets



Associate ASA install targets with the manager

Associate ASA gateway sets with an intermediary manager so you can manage them as install targets in Mesh Policy Engine.

After you add the Security Cloud Control Firewall Management as an intermediate manager, you need to associate the ASA install targets (also called gateway sets) with the manager.

Before you begin

[Create the ASA intermediary manager](#) on page 24.

Procedure

1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Install Targets > Install Targets > Intermediary Managers > SCC-FW**.

Figure 25: SCC-FW Intermediary managers

SCC-FW Intermediary managers

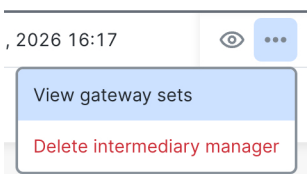
[+ Create intermediary manager](#)

Intermediary id	Status	Parent zone id	Credential name	Modified time	
security-cisco-cdo-manager	Active	—	Modern-Pinecone-CDO	Apr 28, 2026 16:17	👁️ ...

< >

- Click the More icon (...), and choose **View gateway sets**.

Figure 26: View gateway sets



- Click **Create gateway set** to add the ASA install target.

Figure 27: Create gateway set

SCC-FW Intermediary managers > Gateway sets (security-cisco-cdo-manager)

4 Status 10 Device type 0 Install target [+ Create gateway set](#)

GatewaySet Id	Name	Device Type	Status
 No matches found			

< >

- Set the following values for the install target.

Mesh Policy Engine requires you to add the ASA install target separately for each type of policy: global, interface (inbound), and interface (outbound).

Figure 28: Install target values

Create SCC-FW intermediary manager gateway set

☒ Global

Status *
ACTIVE

Direction *
INBOUND

Device name *
asa1

Interface name *
outside

[Cancel](#) [Save](#)

- Global**—For a policy that applies to all interfaces or zones, enable the **Global** toggle. Global policies are always inbound.

- **Status**—The status can be **ACTIVE**, **HELD**, **SKIPPED**, or **UNMANAGED**. The status of a device decides how Mesh Policy Engine interacts with it.
 - **ACTIVE**—The device is ready for policy management.
 - **HELD**—Any request to the device remains pending. Such pending requests are executed only after the state is changed to **ACTIVE**.
 - **SKIPPED**—Requests to the device are not executed.
 - **UNMANAGED**—Requests to the device are ignored for updates. All pending requests are closed.
- **Direction**—**INBOUND** or **OUTBOUND**.
- **Device name**—Specify the name of a device present in Security Cloud Control Firewall Management. The name must match exactly.
- **Interface Name**—For inbound or outbound policies, the interface name to which to attach the policy. The name must match exactly.

6. Click **Save**.

7. For a given ASA, repeat this procedure for each policy type: global, interface (inbound), and interface (outbound).

Create an install target: Fortigate, Palo Alto, Juniper

Add FortiGate, Palo Alto, or Juniper firewalls as install targets for direct policy deployment from Mesh Policy Engine.

Create an install target for security devices that support direct deployment. These security device types are supported for this procedure:

- Fortigate
- Palo Alto
- Juniper

Procedure

1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Install Targets > Install Target Creation**.
3. Click **Add network device**, and enter the options on the **Create install target** page.

Figure 29: Create install target

Create install target

☒ Use buckets

ID *

[Generate ID](#)

Name

Device type *

Status *

Network devices

ID *

IP address * ⓘ

Hostname * ⓘ

Port

Os version

ID *

IP address * ⓘ

Hostname * ⓘ

Port

Os version

[+ Add network device](#)

Credential name

Virtual path

[Cancel](#) [Save](#)

- **Use buckets**—Turn on the **Use Buckets** toggle to optimize policy creation. Bucketing is a policy optimization algorithm that uses different buckets to optimize flows. The policy composer uses four separate buckets to optimize flows, depending on the type of protocol and the timeout. Application protocols with timeout and without timeout are split into two buckets. Other internet protocols are also divided in two categories—with timeout and without. Flows in each bucket are optimized and sorted separately. All the flows from different buckets are combined, preserving their sorting order.
- **ID**—An identifier for the set of devices called an install target. Use the **Generate ID** option to create a random identifier.
- **Name**—A name of your choice.
- **Device type**—Specifies the make of the devices.
- **Status**—The status can be **ACTIVE**, **HELD**, **SKIPPED**, or **UNMANAGED**. The status of a device decides how Mesh Policy Engine interacts with it.
 - **ACTIVE**—The device is ready for policy management.
 - **HELD**—Any request to the device remains pending. Such pending requests are executed only after the state is changed to ACTIVE.
 - **SKIPPED**—Requests to the device are not executed.
 - **UNMANAGED**—Requests to the device are ignored for updates. All pending requests are closed.

- **Network devices**—Click **Add network device** for each node, for example, multiple nodes in a cluster.
 - **ID**—A unique name for the device. One is automatically generated, or you can enter a manual ID.
 - **IP address** or **Hostname**—Set one of these values for the network device.
 - **Port**—(Optional) The port for the network device.
 - **Os version**—(Optional) The OS for your network device.
- **Credential Name**—The credential name that you specified in [Manage credentials](#) on page 20. Mesh Policy Engine uses the credentials to connect to a device.
- **Virtual path**—Reserved for future use.

4. Click **Save**.

You see the install target on the **Install Targets** page.

Create an install target: AMD Pensando (API)

Create AMD Pensando install targets with Mesh Policy Engine APIs for application and infrastructure policies.

In an AMD Pensando device, the Policy and Services Manager (PSM) uses a Policy Distribution Target (PDT) to deploy network and security policies.

Policy Distribution Targets (PDT) allow granular policy application to specific Pensando Distributed Services Cards (DSC). By using Mesh Policy Engine to manage these targets, you can avoid over-provisioning the DSCs and ensure that policies are only applied to the relevant DSCs. Mesh Policy Engine manages the lifecycle of policies and their relationships to PDTs, while the underlying DSC-to-PDT mapping should be maintained by your network team.

For a given PDT, you can configure multiple infrastructure-related policies, but configure only one application-related policy.

Before you begin

Before you begin the configuration, review these prerequisites:

- Your network team must create a 1:1 relationship between each DSC and a PDT on the PSM.
- Mesh Policy Engine manages the policies created on the PSM and also manages the relationships between policies and PDTs. Mesh Policy Engine stores the records of the PSMs and PDTs but does not store DSC information directly.
- Ensure that you have the unique IDs for each of the PDTs.

Create install targets for application-related policy

For each PDT, Mesh Policy Engine creates a policy with all the relevant application flows.

To apply the policies on the relevant devices, Mesh Policy Engine requires your network topology.

1. To build the topology, first create the intermediary manager, which is the PSM, using this API:

```
POST /topology/manager/pensando-dsc
Payload:
{
  "id": "string",
  "hostDetails": [
    {
```

```

    "hostname": "string",
    "port": 0,
    "priority": 1
  }
],
"status": "INACTIVE",
"parentZoneId": "string",
"metadata": {
  "modifiedTimestamp": "2026-01-14T09:33:38.499Z",
  "monitorRetryCount": 0,
  "monitorMessage": "string",
  "lastMonitoredTimestamp": "2026-01-14T09:33:38.499Z"
},
"credentialName": "string"
}

```

2. Create an application gateway for the PDT, using this API:

```

POST /topology/manager/pensando-dsc-pdt{managerId}/pdt/{pdtId}/address/app

Payload:
{
  "parentZone": "string",
  "addresses": [
    {
      "value": "1.1.1.1/32",
      "exactMatch": true
    }
  ]
}

Response:
{
  "reqId": "3fa85f64-5717-4562-b3fc-2c963f66afa6", // to track the order to
  completion
  "requestingSystem": "PCM",
  "affectedEntities": {
    "gwSetId": "string", // logical representation of the policy
    destined to the pdt
    "enforcementPointId": "string", // logical representation of PDT
    "zoneId": "string"
  }
}

```

Using the elements that you provide in these APIs, Mesh Policy Engine builds the topology—creates the zones, gateway sets and enforcement points that are necessary to represent the PDTs and PSMs. It then pushes the policy to the PSM, which applies the policy to the PDT.

 Note

When you associate an address with a PDT, you can provide a `parent zone` that acts as a parent zone for the PDTs. If you do not provide a parent zone here, the `parent zone` for a PSM is used. If the PSM too doesn't provide a parent zone, Mesh Policy Engine uses a default parent zone. After you have associated an address and a parent zone with a PDT, you can't associate a different parent zone with it.

Create install targets for infrastructure-related policy

Each infrastructure subnet has a dedicated policy on the PSM. This policy is associated with multiple PDTs that require the subnet. To apply the policy on the PDTs, Mesh Policy Engine requires your network topology. To build your topology, first create the intermediary manager (PSM):

```
POST /topology/manager/pensando-dsc

Payload:
{
  "id": "string",
  "hostDetails": [
    {
      "hostname": "string",
      "port": 0,
      "priority": 1
    }
  ],
  "status": "INACTIVE",
  "parentZoneId": "string",
  "metadata": {
    "modifiedTimestamp": "2026-01-14T09:33:38.499Z",
    "monitorRetryCount": 0,
    "monitorMessage": "string",
    "lastMonitoredTimestamp": "2026-01-14T09:33:38.499Z"
  },
  "credentialName": "string"
}
```

Push the infrastructure-related policy to the PSM using this API:

```
POST /topology/manager/pensando-dsc-pdt{managerId}/pdt/{pdtId}/address/infra

Payload:
{
  "parentZone": "string",
  "address": {
    "value": "1.1.1.1/32",
    "exactMatch": true
  }
}

Response:
{
  "reqId": "3fa85f64-5717-4562-b3fc-2c963f66afa6", // request to track order completion
  "requestingSystem": "PCM",
  "affectedEntities": {
    "gwSetId": "string", // represents the subnet PSM combination
  }
}
```

```

    "enforcementPointId": "string",      // represents the PDT
    "zoneId": "string"                  // represents the subnet
  }
}

```

With this API call, Mesh Policy Engine creates the topology including the zones, gateway sets, enforcement points that represent the PDTs and PSMs, as well as zone paths and their relationships. Set the `exactMatch` parameter to 'true' to ensure that only rules specifically matching that subnet are applied. This setting prevents unintended flow filtering.

This API call returns a request ID (`reqId`) that allows you track the policy deployment to the PDTs.

View install targets

Review the firewalls, policy types, and deployment status associated with Mesh Policy Engine install targets.

You can view install targets and the policies associated with them.

1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.

2. Choose **Install Targets > Install Targets > Install Targets**.

For each install target in your organization, you can view details such as gateway identifier, type of device, status of the device, and its virtual path (if applicable).

3. To see the policies that are associated with an install target, locate the install target from the list, click the ellipsis (...) icon, and select **View associated policies**.

The **Associated policies** window displays details of each policy that is associated with the selected install target.

3 Domains, Dynamic Objects, and Topology

Topics:

- [Configure domains](#)
- [Manage dynamic objects \(API\)](#)
- [Create a topology](#)

Learn how to organize security policies with domains, manage dynamic objects, and build and view network topologies in Mesh Policy Engine.

Domains, dynamic objects, and topology help you streamline and visualize how to apply your security policies.

Configure domains

Create a domain to organize related security policies, rules, and network objects in Mesh Policy Engine.

A domain is a centralized entity that groups and manages related security policies, rules, and network objects. Grouping policies under a domain ensures streamlined management of the enterprise network policy and better security posture.

If your enterprise has many network policies that are designed to meet a variety of security requirements, you must group policies with the same intent under their respective domains.

In the **Policies** menu, select a domain from the drop-down menu at the top to view all policies that are grouped under that domain. Create a policy after selecting a domain. The policy is then associated with that domain.

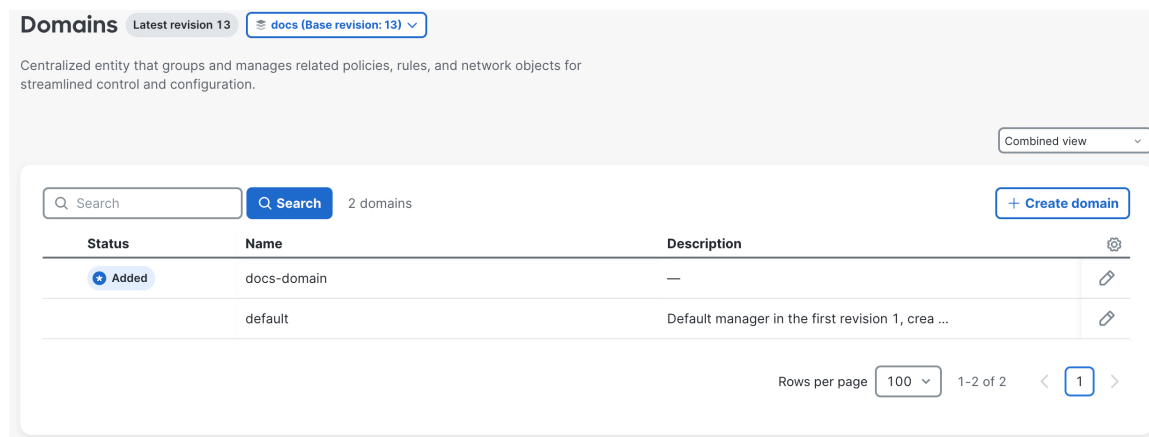
Before you begin

[Manage changesets](#) on page 99.

Procedure

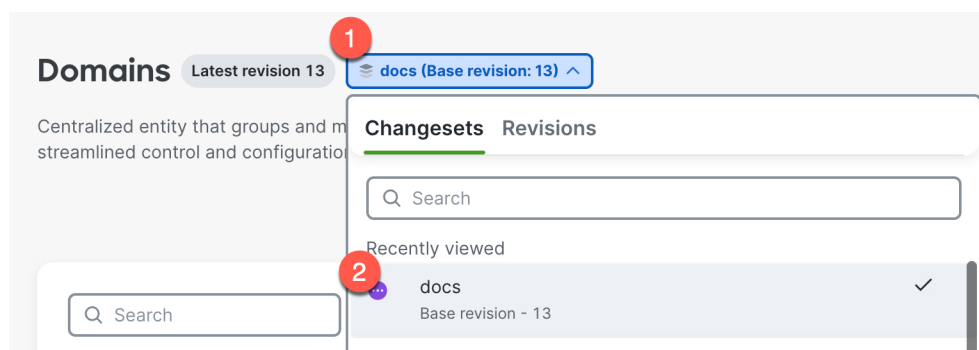
1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Domains**.

Figure 30: Domains



3. From the drop-down menu at the top, choose a changeset or create one and select it.

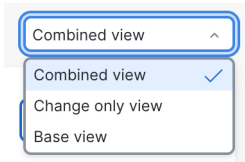
Figure 31: Choose changeset



If you don't have a changeset, create one. A newly created changeset is automatically checked out.

For the selected changeset, you can view all domains (**Combined view**), new or changed domains (**Change only view**), or existing domains (**Base view**).

Figure 32: Views



4. Click **Create Domain**, enter a name, and click **Create**.

Figure 33: Create domain

5. To edit or delete a domain, click the Edit icon (✎) and either change the domain name and click **Save**, or click the Delete icon (Delete 🗑).

Manage dynamic objects (API)

Create, update, and delete dynamic objects with Mesh Policy Engine APIs for use as sources and destinations in policy rules.

You can group IP addresses that change dynamically based on networking requirements, into dynamic objects in Mesh Policy Engine

You can include dynamic objects in the source and destination fields of a rule. Including dynamic objects in rules ensures that the latest configurations are automatically applied to the relevant gateway sets

You can view the list of dynamic objects on the GUI. However, you must use the APIs to create, update, or delete dynamic objects.



Note

Use dynamic objects only in fully pinned governance policies such as Default and Priority. Fully pinned flows do not use topology resolution. Instead, they are directly associated with the gateway sets.

From the Security Cloud Control Home page, click **Mesh Policy Engine..** Choose **Dynamic Objects** to see the list of existing dynamic objects.

Use the following Mesh Policy Engine APIs to create and manage dynamic objects:

- GET /pcm/endpointgroups/{endpointGroupId}

- POST /pcm/endpointgroups/{endpointGroupId}
- PUT /pcm/endpointgroups/{endpointGroupId}
- DELETE /pcm/endpointgroups/{endpointGroupId}

To add a dynamic object, follow these steps:

Before you begin

[Access the Mesh Policy Engine API](#) on page 17.

Procedure

1. Use this API: POST - /mpe/api/endpointgroups/{endpointGroupId}, where `endpointGroupId` is the ID of the network group that you want to add. Here is an example API schema:

```
{
  "description": "string",
  "memberEndpoints": [
    {
      "ip": "",
      "tags": [
        "key:value"
      ]
    }
  ],
  "memberGroups": [
    "memberGroup1"
  ]
}
```

Note that the Mesh Policy Engine API network accepts the API request only in this format.

2. Replace the parameter values in the request body to reflect your device's data, before making the request.
`tags` are key-value pairs that help you filter the networks. For information on tags, see [Topology Construction](#).
3. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
4. In the Mesh Policy Engine menu, click **Dynamic Objects** and ensure that the dynamic object is added.
See the [Mesh Policy Engine API](#) for more information.

Create a topology

Learn how network topologies represent zones, gateways, and allowed paths for security policy planning in Mesh Policy Engine.

Mesh Policy Engine uses the network topology to identify the placement of security policies and devices within your enterprise network. You can construct the network topology using the Persisted Connectivity Manager's APIs—populate the endpoints, zones, and determine the path between the zones. As a result, Mesh Policy Engine can identify which security devices in the topology require policy updates and determine when to update them. It also maintains an inventory of each path and the devices along those paths.

Construct your topology (API)

Find details about defining zones, gateway sets, and allowed paths to construct a network topology with Mesh Policy Engine APIs.

Before you begin

[Access the Mesh Policy Engine API](#) on page 17.

Topology components

The key components of a Mesh Policy Engine topology are:

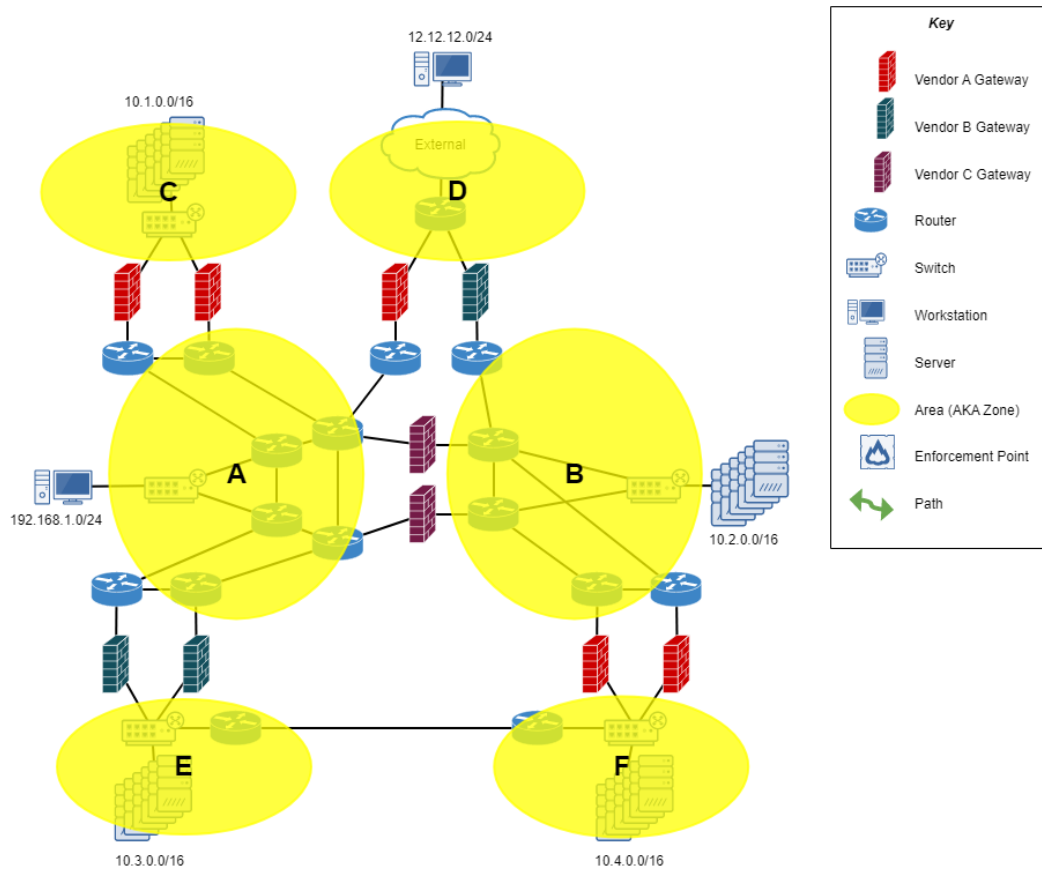
- **Zone:** Subset of the network that exists behind the security enforcement points.
- **Adjacency:** An adjacency is between a pair of zones. There may be one or more security devices in an adjacency, and the same device may be in more than one adjacency. These are typically firewalls with interfaces connected to two or more security zones.
- **Zone path:** A series of adjacent zones that connect a pair of terminal zones.
- **Tag :** Key-value pair that further restricts where a policy is placed in a network.
- **Terminal zones:** Pair of significant zones that represent where the source and destination addresses exist.
- **Adjacent zones:** Zone pairs that translate to a zone path.

When new policies or policy changes are committed, Mesh Policy Engine resolves all the networks to their zones, and uses the zone paths between terminal zones to determine which security devices must be updated.

Mesh Policy Engine maintains an inventory of every path and the position of every install target along each path in your topology. This allows Mesh Policy Engine to determine the install targets and the security policy requirements that relate to an install target.

Step1: Break down the network into zones

Figure 34: Zones



Use this API to create zones:

POST /topology/zone

Request body example:

```
{
  "zones": [
    {
      "id": "string",
      "type": "DEFAULT",
      "stubby": true,
      "aggregatedStubby": true,
      "createdTime": "2026-02-04T04:41:08.569Z",
      "child": true,
      "parentZone": "string"
    }
  ]
}
```

Associate an IP address with the zone.

When you use the API to associate an IP address with a zone, you specify the exactMatch parameter.

POST /topology/zone/{zoneid}/address

Request body:

```
{
  "addresses": [
    {
      "value": "1.1.1.1/32",

```

```

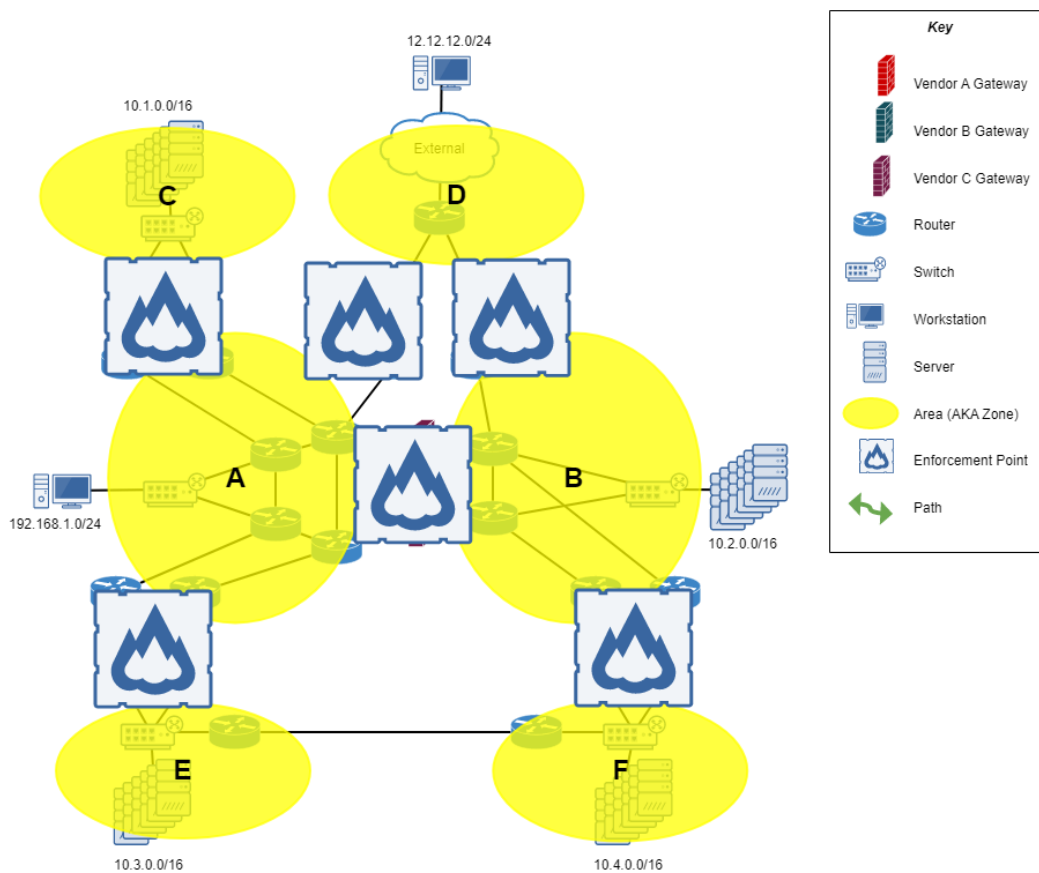
    "exactMatch": true
  }
]
}

```

If `exactMatch` is "true", only network objects with the same subnet base IP and netmask are associated to a zone. If `exactMatch` is "false", any network object whose IP and netmask matches or is a subset of the address range is associated to a zone.

Step2: Determine the install targets between adjacent zones

Figure 35: Install targets between adjacent zones



Use this API to create adjacencies:

```
POST /topology/zone-path/adjacent-zones
```

Request body:

```

{
  "adjacentZones": [
    {
      "zoneOneId": "string",
      "zoneTwoId": "string"
    }
  ]
}

```

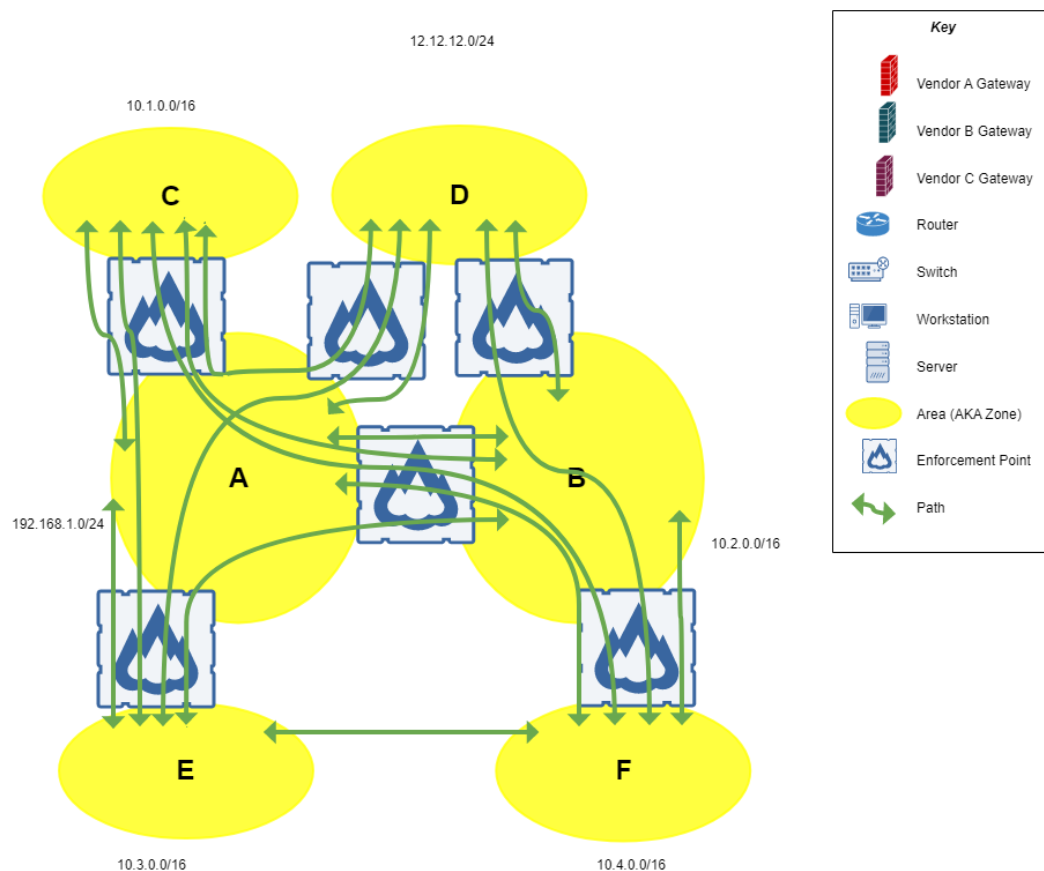
Add install targets to adjacencies:

```
POST /topology/zone-path/adj-zone-ds

Request body:
{
  "adjacentZonesToGatewaySets": [
    {
      "zoneOneId": "string",
      "zoneTwoId": "string",
      "gatewaySetId": "string"
    }
  ]
}
```

Step3: Make an inventory of the allowable paths between zones across adjacencies

Figure 36: Zone Paths



Use this API to create the zone paths:

```
POST /topology/zone-path

Request body:
{
  "zonePath": [
    {
      "zone": "string"
    }
  ]
}
```

```

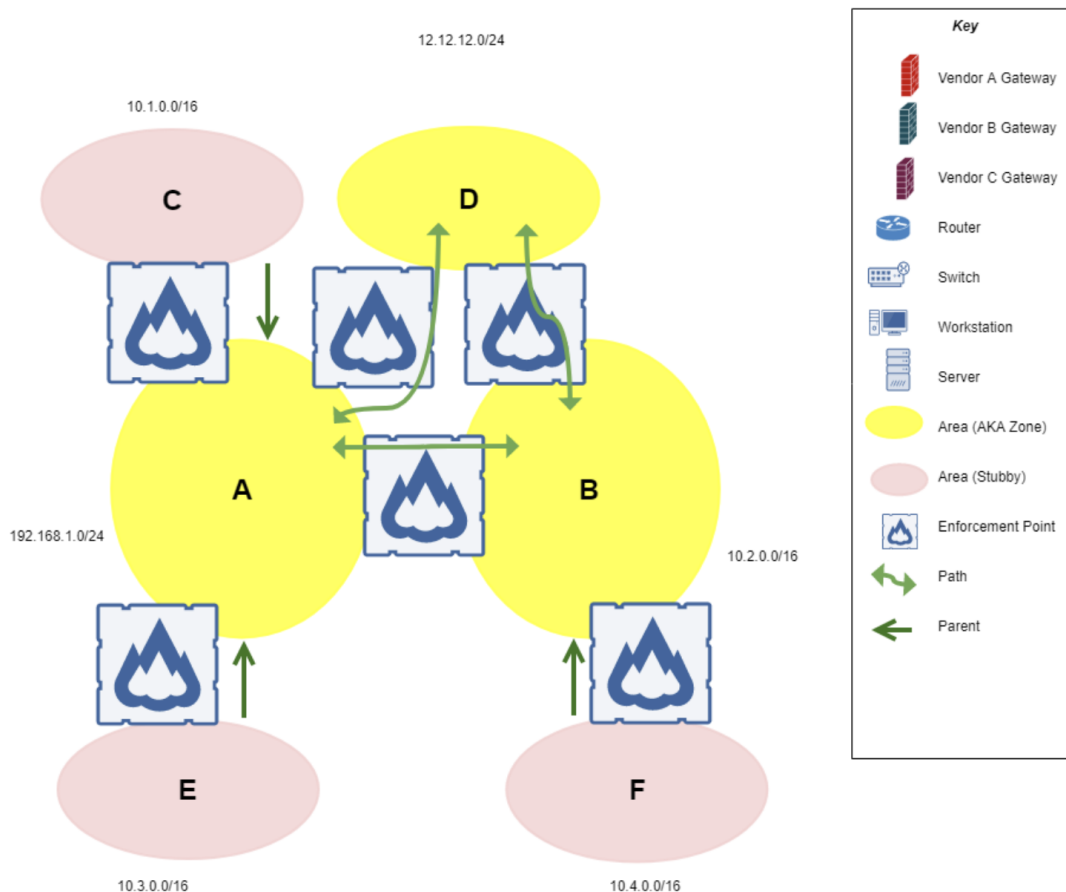
    ]
}

```

Step 4: (Optional) Define child zones

A child zone is a small, segmented part of the network that is fully contained under a parent zone. Child zones inherit the zone paths of their parent. This inheritance simplifies network design by reducing the number of paths and zones that need to be managed. It also optimizes paths and policy calculation. For example, you can configure a VLAN behind a pair of firewalls as a child zone. This configuration ensures that all traffic is automatically routed through the firewalls, which simplifies policy enforcement and network path management.

Figure 37: Child Zones



Child zones automatically inherit a zone path from their parent zones. All transit sets (that lie between the terminal zones) on the inherited zone path also receive the traffic flow, because the child zone is treated as a terminal of that path.

A child zone creates an implicit path to its parent. Even if no explicit zone path is defined, the system enforces all traffic to or from that child zone on the security device at that adjacency.

Tags

Use tags (sets of key-value pairs) on the policy objects to provide an extra filter with IP or prefix matching.

A flow matches a zone only if both the IP and the tags align with that zone's definition.

For example, in this rule, the flow from 10.10.1.1/32 to 10.99.9.1/32 would match all the gateway sets with that IP address in destinations. A flow defined from 10.10.1.1 to 10.99.9.3 with the tag “Color: Yellow” matches only that destination, and the policy applies only to that gateway set.

```
{
  "rules": [
    {
      "id": "rule1",
      "sources": [{"ip": "10.10.1.1/32"}],
      "destinations": [{"ip": "10.99.9.1/32"}],
      "tags": [{"Color: Yellow"}],
      "apps": ["6_ignore_80_80_null"]
    },
    {
      "id": "rule2",
      "sources": [{"ip": "10.99.9.2/32"}],
      "destinations": [{"ip": "10.99.9.3/32"}],
      "tags": [{"Color: Red"}],
      "apps": ["6_ignore_80_80_null"]
    },
    {
      "id": "rule3",
      "sources": [{"ip": "10.99.9.3/32"}],
      "destinations": [{"ip": "10.99.9.4/32"}],
      "tags": [{"Color: Purple"}],
      "apps": ["6_ignore_80_80_null"]
    }
  ]
}
```

Tags enable segmentation within shared IP spaces. When multiple zones share overlapping or identical IP prefixes, tags allow you to distinguish each zone.

Tags and topology together determine which devices in your network receive policies. Tags first narrow the zones in which an address participates. Then, topology determines which firewalls are located on the zone paths of those zones and, as a result, receive the policy.

View your topology

View a graphical network topology and inspect its zones, gateways, inventory, and discovered paths in Mesh Policy Engine.

After you have built your network topology, you can see a graphical representation of your topology zones and gateways on the Mesh Policy Engine interface. You can also browse device inventory, and analyze the paths between network entities.

Before you begin

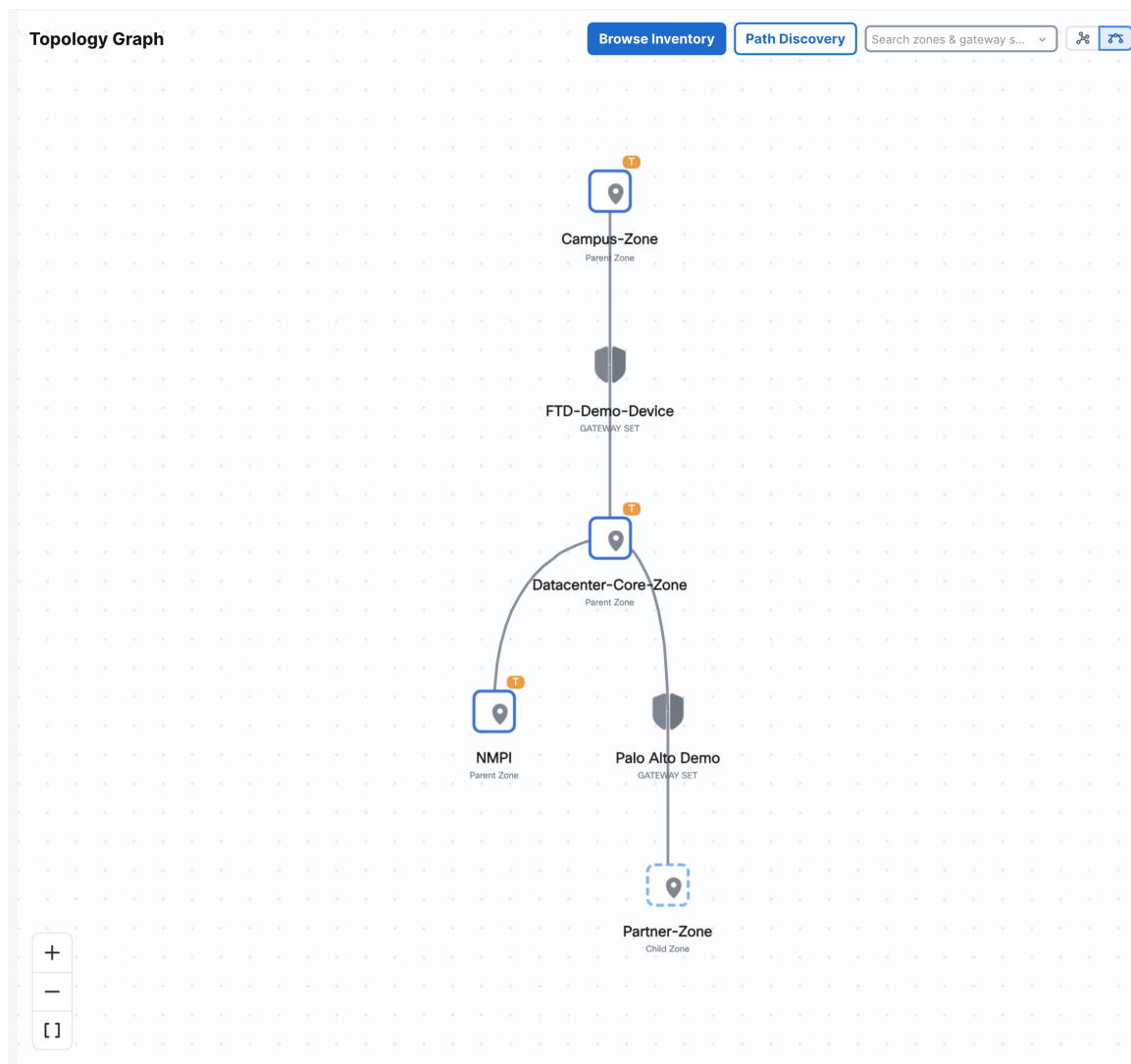
[Construct your topology \(API\)](#) on page 38.

Procedure

1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. In the Mesh Policy Engine menu, choose **Topology > Topology Graph**.

Mesh Policy Engine displays a graphical representation of your topology, including zones, install targets, and the paths between them.

Figure 38: Topology



Hover over a zone or a install target to see its details.

3. Optional: Click **Browse Inventory** to see a list of all items in your topology.

Figure 39: Inventory

The screenshot shows the 'Inventory' pane with a search bar and two expandable sections: 'Zones' and 'Gateway sets'.

Zones (4 items):

ID	Zone type	Is chi...	Parent Zone
Campus-Zone	DEFAULT	False	-
Datacenter-Core-Zone	DEFAULT	False	-
NMPI	DEFAULT	False	-
Partner-Zone	DEFAULT	True	Datacenter-Core-Zone

Gateway sets (2 items):

ID	Name	Device type	Status
a26fecda-5cbe-474d-	FTD-Demo-	FTD	ACTI'

4. Optional: Click **Path Discovery** to view the zone paths between any two entities in your network.

In the **Path Discovery** pane, you can select a zone, an install target, or a set of source and destination entities to view their zone paths.

Figure 40: Path discovery

Path Discovery

×

Discover zone paths between zones and gateway sets

Zones

Gateway sets

View zone paths for selected zone

Campus-Zone

▼

Show Paths

OR

View zone paths between terminal zones

Source zone

Select source zone

▼

Destination zone

Select destination zone

▼

Show Paths

Path results for Campus-Zone

< 1/2

ZONE

Campus-Zone

1

Path results for Campus-Zone

< 1/2

ZONE

Campus-Zone

↓

ZONE

Datacenter-Core-Zone

↓

ZONE

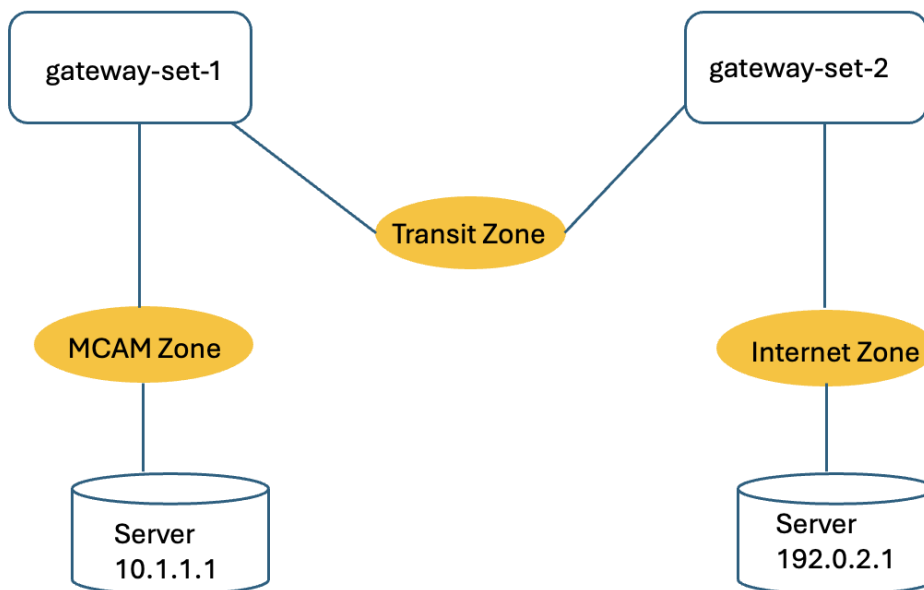
NMPI

Sample Topology

Find a sample workflow for building a network topology with Persisted Connectivity Manager APIs, including zones, adjacencies, gateway sets, and policies.

Use the PCM APIs to build a sample topology as described in this document. The sample topology includes two install targets (gateway-set-1 and gateway-set-2), each located in a different terminal zone. The zone that lies on the path between these terminal zones serves as a transit zone.

Figure 41: Sample Topology



1. Create zones called "MCAM," "Transit," and "Internet":

```

POST /topology/zones
{
  "zones": [
    {
      "id": "MCAM"
    },
    {
      "id": "Transit"
    },
    {
      "id": "Internet"
    }
  ]
}

```

2. Add addresses to the MCAM zone:

```

POST /topology/zone/MCAM/addresses
{
  "addresses": [
    {
      "value": "10.1.1.0/24",
      "exactMatch": "false"
    }
  ]
}

```

3. Add addresses to the Internet zone:

```
POST /topology/zone/Internet/addresses
{
  "addresses": [
    {
      "value": "192.0.2.1/24",
      "exactMatch": "false"
    }
  ]
}
```

4. Create adjacencies:

```
POST /topology/zone-path/adjacent-zones
{
  "adjacentZones": [
    {
      "zoneOneId": "MCAM",
      "zoneTwoId": "Transit"
    },
    {
      "zoneOneId": "Transit",
      "zoneTwoId": "Internet"
    }
  ]
}
```

5. Add install targets to adjacencies:

```
POST /topology/zone-path/adj-zone-ds
{
  "adjacentZonesToGatewaySets": [
    {
      "zoneOneId": "MCAM",
      "zoneTwoId": "Transit",
      "gatewaySetId": "gateway-set-1"
    },
    {
      "zoneOneId": "Transit",
      "zoneTwoId": "Internet",
      "gatewaySetId": "gateway-set-2"
    }
  ]
}
```

6. Create a zone path from the MCAM zone to the Internet zone:

```
POST /topology/zone-path
{
  "zonePath": [
    {
      "zone": "MCAM"
    },
    {
      "zone": "Transit"
    },
    {
      "zone": "Internet"
    }
  ]
}
```

```
    ]
  }
}
```

7. Create a zone path from the MCAM zone to the "Transit" zone:

```
POST /topology/zone-path
{
  "zonePath": [
    {
      "zone": "MCAM"
    },
    {
      "zone": "Transit"
    }
  ]
}
```

8. Create a zone path from Transit to Internet:

```
POST /topology/zone-path
{
  "zonePath": [
    {
      "zone": "Transit"
    },
    {
      "zone": "Internet"
    }
  ]
}
```

9. Create a policy:

Create a rule specifying the source IP address as 10.1.1.1/32 and destination IP address as 192.0.2.1/32 : tcp80

Expected Result

The rule is added to the policies on both the install targets (gateway-set-1 and gateway-set-2).

4 Policies

Topics:

- [About Policies](#)
- [Guidelines for policies](#)
- [Configure networks](#)
- [Configure services](#)
- [Configure rulesets](#)
- [Push a ruleset to an install target](#)
- [Import policies](#)

Learn how to create, update, validate, and deploy security policies and related objects with the Mesh Policy Engine interface.

Guides you through creating, updating, and managing security policies in Mesh Policy Engine using the GUI. It includes changeset-based workflows, policy types, NAT rule validation, install target management, and the administration of domains, networks, services, and dynamic objects.

About Policies

An overview of security policies and the rulesets, networks, services, domains, and changesets used to create and manage them in Mesh Policy Engine.

A policy is a list of ordered rules that define how the devices in a network can communicate. For most install targets, Mesh Policy Engine owns the entire security policy for your install target, and when you make changes to a ruleset and push it, Mesh Policy Engine compiles changes into the complete policy.

For Firewall Threat Defense, Mesh Policy Engine adds a new child policy with an auto-generated name, maintaining the existing assigned policy as the parent policy. All rules are put in the Default section in the generated Firewall Threat Defense policy.



Note

Before editing a policy, ensure that you have chosen a changeset and a domain. The edits you do to the policy are recorded as part of the changeset and the domain that you choose.

Policy components

You can create, read, update, and delete the components and their relationships using changesets. You can manage these policy components:

- **Domain**—All entities are scoped and grouped under a domain.
- **Ruleset**—Defines the rules associated with install targets. Rulesets are associated with the install targets either directly or through topology resolution.
- **Networks**—IPv4 addresses used in the rules.
- **Network Groups**—Collection of networks and network groups.
- **Services**—Application protocols and port ranges used in rules.
- **Service Groups**—Collection of services and service groups.

Access policy

The resulting policy on the install target is constructed from three categories of Mesh Policy Engine-managed rulesets. Each category has distinct ordering and optimization behaviors that affect how rules are enforced.

Mesh Policy Engine supports three types of rulesets for access policy:

- **Priority**—A ruleset that contains allow and deny rules, which will be prepended to the final policy that is constructed during policy execution without any changes (post optimization). Both allow and deny rules are preserved and pushed as permit and deny flows. If you push multiple priority rulesets, the internal rule order is preserved, but the order of the rulesets can't be guaranteed.
- **Core**—A ruleset that contains allow or deny rules. However, the deny rules are ignored, and only permit flows are pushed.
- **Default**—A ruleset that contains allow and deny rules, which will be appended to the final policy that is constructed during policy execution without any changes (post optimization). Both allow and deny rules are preserved and pushed as permit and deny flows. If you push multiple priority rulesets, the internal rule order is preserved, but the order of the rulesets can't be guaranteed.

To ensure the deployed rules align with your security intent, organize order-dependent rules within a single ruleset and avoid overlapping rules across multiple priority or default rulesets.

NAT policy

Mesh Policy Engine supports a policy that contains Network Address Translation (NAT) rules. See [NAT rulesets](#) on page 54.



Note

Only Juniper supports NAT policies.

Priority rulesets

Learn when to use priority rulesets for security policies that must take precedence, such as threat blocks, compliance rules, and exceptions.

Priority rulesets are intended for security policies that must be evaluated before any other rules on the device. Common use cases include explicit deny rules for known-bad traffic, regulatory compliance rules, and exceptions that must take precedence over general access policies.

How priority rulesets are ordered

- Position in the policy—Priority rules are at the top of the policy, ahead of core rules, and default rules.
- Internal rule order within a ruleset—Preserved as authored.
- Order across multiple priority rulesets—Not guaranteed. We recommend using only one priority ruleset. When multiple priority rulesets are assigned to the same install target, Mesh Policy Engine preserves the internal order of each ruleset, but the relative order between rulesets may vary at deployment.

Optimization behavior

Priority rulesets are pushed to the device as-is, without optimization:

- Allow rules are deployed as explicit permit flows.
- Deny rules are deployed as explicit deny flows.
- No deduplication, merging, or normalization is performed across rules.

Design guidance

To ensure predictable enforcement:

- Place rules with strict ordering dependencies (for example, a specific deny that must precede a broader allow) within a single priority ruleset.
- When using multiple priority rulesets for organizational purposes (for example, separating compliance rules from operational rules), ensure the rules do not overlap or conflict across rulesets.
- Use priority rules sparingly because they bypass Mesh Policy Engine's optimization engine and can increase the policy size on the device.

Core rulesets

Learn when to use core rulesets for general access control policies and how to design them for scalable policy management.

Core rulesets form the bulk of access control policy and are designed to express security intent at scale. Mesh Policy Engine applies sophisticated normalization and optimization to core rules to produce a compact, efficient ruleset on the install target.

How core rulesets are ordered

- Position in the policy—Core rulesets appear between the priority section and the default section.
- Internal rule order—Core rules do not maintain a strict authored order. Mesh Policy Engine normalizes and consolidates them based on intent, so the original order within or across core rulesets is largely irrelevant to the final deployed policy.
- Order across multiple core rulesets—All core rulesets assigned to an install target are merged into a single normalized intent set.

Optimization behavior

Core rules undergo the most significant transformation:

1. Intent normalization—Mesh Policy Engine extracts the intent of each rule (allow or deny) across all core rulesets and combines them into a unified permission set.
2. Deny rules are subtractive, not explicit—Deny rules in core rulesets do not appear as explicit deny statements on the install target. Instead, they carve out exceptions from the allow rules during compilation. The resulting ruleset contains only permit flows that already account for the denials.
3. Rule consolidation—Mesh Policy Engine merges, compresses, and deduplicates flows. Strategies include combining rules with the same source and destination but different services, collapsing adjacent subnets into supernets, and removing redundant entries.
4. Bucketing (optional)—When you enable bucketing on the install target, Mesh Policy Engine separates flows into distinct buckets (for example, by protocol type and timeout) and optimizes each bucket independently before combining them.

Design guidance

- Use core rulesets to express the bulk of your access control policy. They are designed to handle complex, overlapping intents efficiently.
- Organize core rulesets logically (by application, team, or use case) without worrying about rule order; Mesh Policy Engine handles consolidation.
- For explicit deny rules that must appear on the install target (for example, for logging or audit purposes), use priority or default rulesets instead, since core deny rules are subtractive and do not produce visible deny entries on the install target.

Default rulesets

Learn when to use default rulesets for catch-all, logging, deny, or permit behavior after other policy rules.

Default rulesets are intended for fallback or catch-all behavior that should apply only after priority and core rulesets have been evaluated. Common use cases include explicit deny-all rules at the end of the policy, default logging rules, and broad permit rules for traffic not covered by more specific core rulesets.

How default rulesets are ordered

- Position in the policy—Default rulesets are at the bottom of the policy.
- Internal rule order within a ruleset—Preserved as authored.
- Order across multiple priority rulesets—Not guaranteed. We recommend using only one default ruleset. When multiple default rulesets are assigned to the same install target, Mesh Policy Engine preserves the internal order of each ruleset, but the relative order between rulesets may vary at deployment.

Optimization behavior

Default rulesets are pushed to the device as-is, without optimization:

- Allow rules are deployed as explicit permit flows.
- Deny rules are deployed as explicit deny flows.
- No deduplication, merging, or normalization is performed.

Design guidance

- Use default rulesets for catch-all behavior that should apply only when no more specific rule matches. A common pattern is a final "deny all" rule to enforce default-deny semantics.
- As with priority rulesets, place rules with ordering dependencies in a single default ruleset, since order across multiple default rulesets is not guaranteed.
- Ensure rules across multiple default rulesets are non-overlapping to avoid non-deterministic enforcement.

NAT rulesets

Find supported NAT policy use cases, rule requirements, and valid network and service configurations for Juniper and Fortinet install targets.

Mesh Policy Engine supports a policy that contains Network Address Translation (NAT) rules.



Note

Only Juniper and Fortinet support NAT policies.

When you push a NAT ruleset, Mesh Policy Engine replaces the existing install target NAT policy entirely.

This section describes how Mesh Policy Engine validates the NAT rules.

Valid NAT use cases

Source	Destination	Service
MATCH	MATCH	MATCH
HIDE	MATCH	MATCH
HIDE	STATIC	MATCH
HIDE	MATCH	STATIC
HIDE	STATIC	STATIC

Source	Destination	Service
STATIC	MATCH	MATCH
STATIC	STATIC	MATCH
STATIC	MATCH	STATIC
STATIC	STATIC	STATIC

Validation of the Endpoints in a NAT Rule

Source Endpoints

1. If the `SrcNatType` is `MATCH`, the rule must adhere to these conditions:
 - Should contain at least one source `endpointMatch` or `endpointGroupMatch`
 - Cannot contain source `endpointTranslated`
2. If the `SrcNatType` is `STATIC`, the rule must adhere to these conditions:
 - Should contain exactly one source `endpointMatch` of /32 mask
 - Cannot contain source `endpointGroupMatch`
 - Should contain exactly one source `endpointTranslated` of /32 mask
3. If the `SrcNatType` is `HIDE`, the rule must adhere to these conditions:
 - Should contain at least one source `endpointMatch` or `endpointGroupMatch`
 - Should contain exactly one source `endpointTranslated` of /32 mask

Destination Endpoints

1. If the `DstNatType` is `MATCH`, the rule must adhere to these conditions:
 - Should contain exactly one destination `endpointMatch` of /32 mask
 - Cannot contain destination `endpointGroupMatch`
 - If `AppNatType` is `STATIC`, destination endpoint must follow `STATIC` rules
2. If the `DstNatType` is `STATIC`, the rule must adhere to these conditions:
 - Should contain exactly one destination `endpointMatch` of /32 mask
 - Cannot contain destination `endpointGroupMatch`
 - Should contain exactly one destination `endpointTranslated` of /32 mask

Validation of the App in a NAT Rule

1. If the `AppNatType` is `MATCH`, the rule must adhere to these conditions:
 - Cannot contain `translatedAppMatch`
 - App should always be `ANY`
2. If the `AppNatType` is `STATIC`, the rule must adhere to these conditions:

- OriginalApp should contain exactly one appMatch
- TranslatedApp should contain exactly one appTranslated
- appMatch and appTranslated are validated in this way:
 - a. Can't contain originalAppGroup
 - b. Protocol must always be the same
 - c. App protocol should always be **ignore**
 - d. Port range should be equal to 1 for both the original and translated app
 - e. Timeout should not be present
 - f. ANY is not allowed for Static

Examples of invalid NAT rules

SL No.	srcApp	dstApp	appType	sourceIp	srcEp	dstIp	dstEp	app	srcTranslated	dstTranslated	appTranslated	Why Invalid
1	MATCH	MATCH	MATCH	[1.1.1.0/24, 2.2.2.1/32]	epGroup1 - [1.1.2.0/24, 2.2.2.0/24]	[1.1.1.1/32, 2.2.2.1/32]	epGroup2 - [1.1.3.0/24, 2.2.3.0/24]	any	[1.1.1.2/32]	[2.2.2.2/32]	6.6.6.0/24	srcApp, dstApp and appType as MATCH cannot contain translated information
2	STATIC	STATIC	STATIC	[2.2.2.1/32]	epGroup1 - [1.1.2.0/24, 2.2.2.0/24]	[1.1.1.1/32]	epGroup2 - [1.1.3.0/24, 2.2.3.0/24]	6.6.6.0/24	[1.1.1.2/32]	[2.2.2.2/32]	6.6.6.0/24	srcApp, dstApp and appType as STATIC cannot contain epGroup or appGroup

SL No.	srcNtType	dstNtType	appNtType	sourceEp	srcEPGroup	dstEPGroup	dstNtType	app	srcPort	dstPort	appPort	Why Invalid
3	STATIC	STATIC	STATIC	[1.1.1.0/24]		[222.0/24]	null	60000-100000	[1.1.1.2/32]	[222.2/32]	60000-100000	srcNtType and dstNtType should contain endpoint of mask /32 and appNtType as STATIC should have port range as 1 and cannot contain timeout
4	HIDE	STATIC	STATIC	[1.1.1.0/24, 222.1/32]	epGroup1 - [1.1.20/24, 222.0/24]	[222.1/32]	null	any	[1.1.1.0/24]	[222.0/24]	any	srcNtType and dstNtType should contain translated endpoint of mask /32 and appNtType as STATIC cannot be any

Examples of valid NAT Rules

The invalid NAT rules listed earlier are corrected here:

SL No.	srcNtType	dstNtType	appNtType	sourceEp	srcEPGroup	dstEPGroup	dstNtType	app	srcPort	dstPort	appPort
1	MATCH	MATCH	MATCH	[1.1.1.0/24, 222.1/32]	epGroup1 - [1.1.20/24, 222.0/24]	[1.1.1.1/32, 222.1/32]	epGroup2 - [1.1.30/24, 222.3.0/24]	any			
2	STATIC	STATIC	STATIC	[222.1/32]		[1.1.1.1/32]		60000-100000	[1.1.1.2/32]	[222.2/32]	60000-100000
3	STATIC	STATIC	STATIC	[1.1.1.1/32]		[222.1/32]	null	60000-100000	[1.1.1.2/32]	[222.2/32]	60000-100000

SL No.	srcType	dstType	appType	sourceIp	srcPort	dstPort	dstPort	app	srcPort	dstPort	app	
4	HIDE	STATIC	STATIC	[1.1.1.0/24, 222.1/32]	epGroup1 - [1.1.20/24, 222.0/24]	[222.1/32]	null	6	6	[1.1.1.1/32]	[222.2/32]	6

Guidelines for policies

Find Mesh Policy Engine guidelines for deploying policies and configuring service objects, ICMP, timeouts, and NAT across supported firewalls.

Deployment guidelines

- ASA—The policy that is pushed gets staged in the **Device with Pending Changes** menu. You must review the policy and deploy it to the ASA manually.
- Fortinet—Mesh Policy Engine does not validate the staged configuration and only performs postcommit validation. Any inconsistencies discovered during this validation, however unlikely, will raise an error, but the device configuration will not be rolled back.

General service object guidelines

- All—Mesh Policy Engine doesn't support AppIDs in service objects; it always uses the protocol and port numbers. For example, FTP is represented by protocol 6 and port 21.
- AMD Pensando—Mesh Policy Engine does not create DNS app with TCP protocol because AMD Pensando security devices do not support it. Instead, it creates a TCP protocol app with port numbers and timeout configured.
- AMD Pensando—Mesh Policy Engine does not support SUNRPC and MSRPC apps because they need special fields for Pensando security devices. These fields are specific to AMD Pensando security devices.

ICMP in a service object guidelines

- Firewall Threat Defense—Doesn't support rules with ICMP type 30 in them.
- Palo Alto Networks—Doesn't support ICMP without any type (null type).
- Palo Alto Networks—Mesh Policy Engine lets you create ICMP services with the same type and code but a different timeout. Palo Alto Networks doesn't support ICMP services that only differ by timeout.

Timeouts in a service object guidelines

- Firewall Threat Defense—Doesn't support timeouts in service objects, so they are ignored.
- Fortinet—If you specify a timeout less than 300 seconds, the timeout in the rule will be changed to 300 seconds.
- Palo Alto Networks—Mesh Policy Engine lets you create ICMP services with the same type and code but a different timeout. Palo Alto Networks doesn't support ICMP services that only differ by timeout.

NAT policy guidelines

- Only Juniper and Fortinet supports NAT policies.

- Juniper–Mesh Policy Engine creates a source NAT rule, a destination NAT rule, and possibly an address set for each NAT rule. If a NAT rule contains a DNAT MATCH and a service NAT MATCH, the destination NAT rule creation is skipped. This affects how destination NAT rules preclude each other.
- Fortinet–
 - To manage network address translation (NAT) rules using Mesh Policy Engine, central source NAT (SNAT) must be enabled in Fortinet firewalls.
 - Mesh Policy Engine creates an SNAT rule, a virtual IP address, and possibly an IP pool for each NAT rule provided. If a NAT rule contains a DNAT MATCH and a Service NAT MATCH, the virtual IP address creation will be skipped. This affects how virtual IP addresses preclude each other.
 - DNAT MATCH must not overlap with any static routes present in a device.
 - If an SNAT translation of a particular type (static, hide) is created for some translation address, ensure that there are no SNAT translations of a different type created for the same translation address. For example, if a STATIC SNAT translation to address 10.10.10.10/32 is created, there can be no HIDE translations to the same address.

Configure networks

Learn how networks and network groups represent policy rule sources and destinations in Mesh Policy Engine.

A network is the source or destination of a connection. In a topology, a network is usually an IP address. A network group is a logical collection of networks.

Create a network

Create a network object with a CIDR address range for use as a source or destination in Mesh Policy Engine rules.

Create a network object for use in rules.



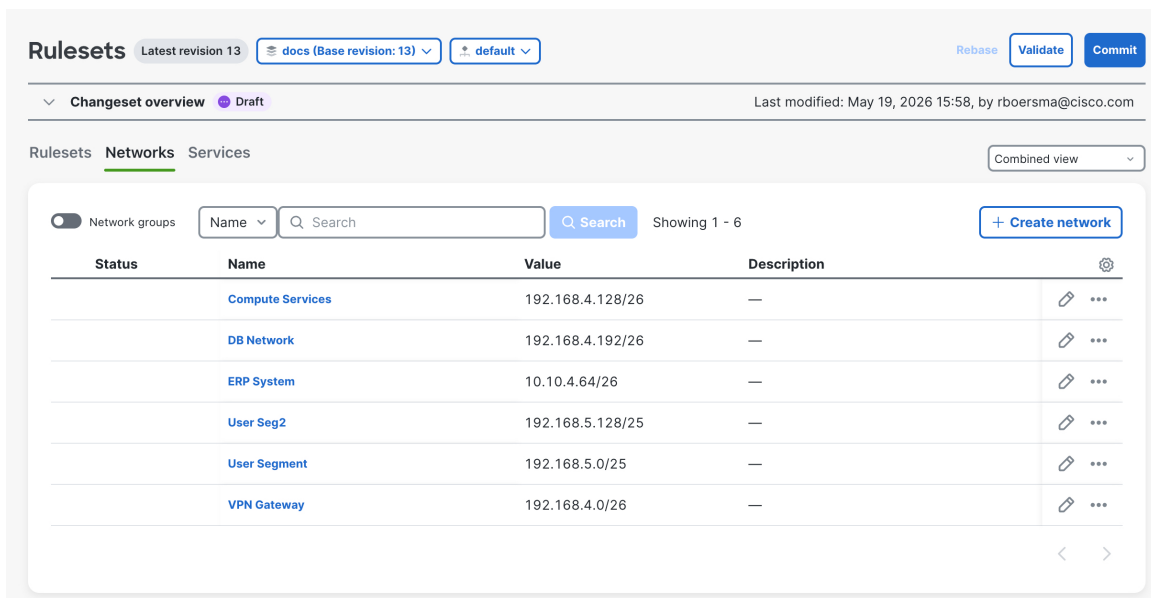
Note

IPv6 is not supported.

Procedure

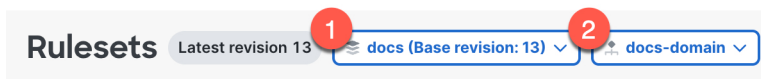
1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Policies > Networks**.

Figure 42: Networks



- From the drop-down menu at the top, choose a changeset and then choose a domain.

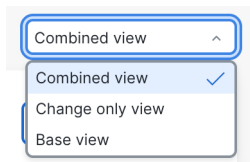
Figure 43: Choose a changeset and domain



If you don't have a changeset, create one. A newly created changeset is automatically checked out.

For the selected changeset, you can view all networks (**Combined view**), new or changed networks (**Change only view**), or existing networks (**Base view**).

Figure 44: Views



- Click **Create network**.

Figure 45: Create network

Create network

docs (Base revision: 13) docs-domain

☐ Create network group

Name *

Value * ⓘ

Description

- Ensure that **Create network group** toggle button is off.
- Enter a **Name** for the network.
- Enter a **Value** for the IPv4 network in slash (/) CIDR notation, for example, 192.168.1.0/24. You can specify a subnet or a host (/32). Ranges are not supported. IPv6 is not supported.
- Optional: Enter a **Description**.

5. Click **Create**.

The newly created network appears in the **Networks** page.

6. To edit a network, click the Edit icon (✎), change the settings, and click **Save**.

7. To view where a network is used, click the More icon (⋮) and choose **Usage**.

Figure 46: Usage

VPN Gateway usage ✕

List of entities impacted by the selected object.

Rulesets 1		
Name	Type	Install targets
Data Center Policy	Core	PALO_ALTO

8. To compare revisions, click the More icon (⋮) and choose **Compare revisions**.

Figure 47: Compare revisions

Compare Revisions for VPN Gateway

This view highlights modifications of rulesets, rules, and objects between revisions.

First revision

11

Second revision

13

Compare

Revision details

Revision

11

Created by

dkovenat@cisco.com

Committed on

Mar 20, 2026 11:43

Description

—

Revision

13

+ Create Changeset

Created by

dkovenat@cisco.com

Committed on

Apr 30, 2026 14:47

Description

—

Comparison overview

Understand how entities have evolved from Revision 11 to Revision 13

00

00

00

00

Added

Edited

Deleted

No Difference

9. To delete a network, click the More icon (...) and choose **Delete**.

You can also click the Edit icon (✎) and then click the Delete icon (Delete 🗑).

Create a network group

Create a network group to reuse related networks or network groups as a single object in policy rules.

Create a network group by grouping individual or grouped networks together.

Procedure

1. In the Mesh Policy Engine menu, choose **Policies > Network Creation**.
2. Choose **Policies > Networks**.

Figure 48: Networks

3. From the drop-down menu at the top, choose a changeset and then choose a domain.

Figure 49: Choose a changeset and domain

If you don't have a changeset, create one. A newly created changeset is automatically checked out.

For the selected changeset, you can view all networks (**Combined view**), new or changed networks (**Change only view**), or existing networks (**Base view**).

Figure 50: Views

4. Click **Create network**.
5. Turn the **Create network group** toggle on.
6. Enter a **Name** for the network group.
7. Click the **Search** text box under **Manage Members** and select network objects from the **Individual Networks** or **Grouped Networks** categories to combine under the network group.
8. Click **Create**.

The newly created network group appears in the list of network groups on the **Networks** page.

9. To edit a network group:
 - a) Locate the network group you want to edit, then click the edit icon next to it.
 - b) Make the required changes.
 - c) Click **Save**.

Configure services

Learn how services define protocols, ports, and timeout settings for use in Mesh Policy Engine policy rules.

In Mesh Policy Engine, a service is a collection of connectivity details such as protocols, port numbers, and timeout settings.

Create a service

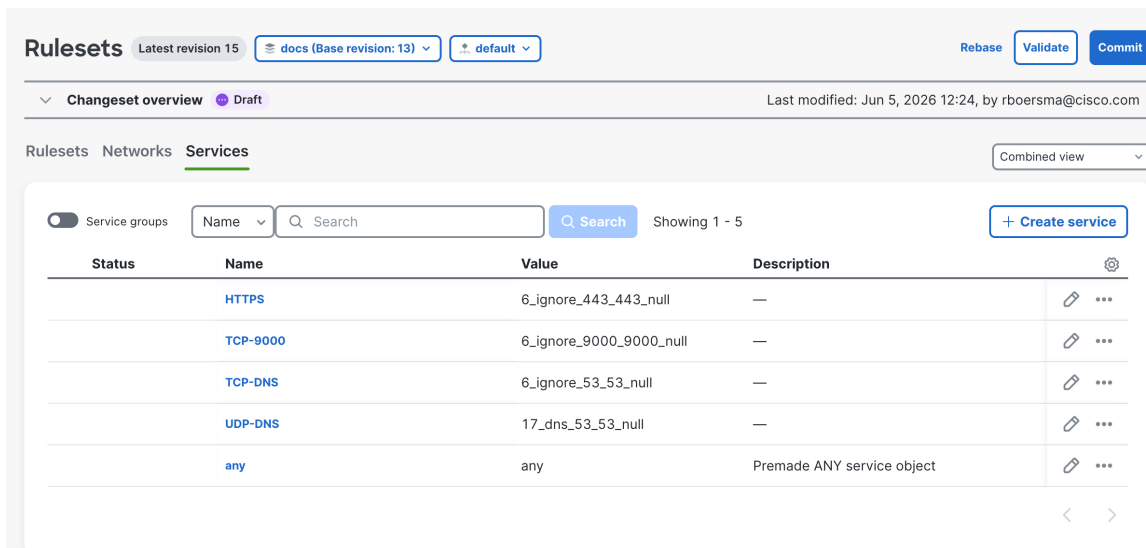
Create a service object with protocol, port, timeout, and description values for use in Mesh Policy Engine policy rules.

Create a service object for use in rules.

Procedure

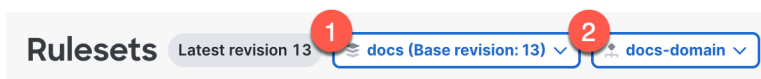
1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Policies > Services**.

Figure 51: Services



3. From the drop-down menu at the top, choose a changeset and then choose a domain.

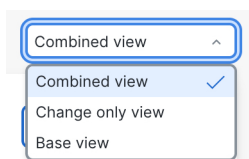
Figure 52: Choose a changeset and domain



If you don't have a changeset, create one. A newly created changeset is automatically checked out.

For the selected changeset, you can view all services (**Combined view**), new or changed services (**Change only view**), or existing services (**Base view**).

Figure 53: Views



4. Click **Create service**.

Figure 54: Create service

The screenshot shows the 'Create Service' form with the following fields and controls:

- Toggle buttons:** 'Create service group' (off) and 'Any service object' (off).
- Name:** A text input field containing 'ftp'.
- Service protocol:** A dropdown menu with 'ignore' selected.
- IANA protocol:** A dropdown menu with 'tcp(6)' selected.
- Port start:** A text input field containing '21'.
- Port end:** A text input field containing '21'.
- Idle timeout:** A slider control set to a low value, with a 'seconds' label and a 'Return to default' button.
- Description:** A large text area for entering a description.
- Buttons:** 'Cancel' and 'Create' buttons at the bottom.

- Ensure that **Create service group** and **Any service object** toggle buttons are off.
A premade **any** object already exists (with **Any service object** toggled on).
- Enter a **Name** for the service.
- Optional: Choose a **Service protocol** to filter the IANA protocols automatically, or choose **ignore** if you want to set the protocol manually.
For example, if you choose **dns**, then both **udp(17)** and **tcp(6)** are included in the IANA protocol list.
- Choose the **IANA protocol** type.
- Depending on the protocol, you can enter the following values:
 - udp(17)** or **tcp(6)**—**Port Start** and optional **Port End**, between 0 and 65535.
 - icmp(1)**—**Icmp type** and **Icmp code**, between 0 and 255.
- Optional: Specify an **Idle timeout** for the service.
Some install target types don't support timeouts in objects, in which case, this setting will be ignored.
- Optional: Enter a **Description**.
- Click **Create**.

After the service is created, you can see it on the **Services** page.

5. To edit the service:

- Locate the service you want to edit, then click the Edit icon (✎) next to it.
- Make the necessary changes to the service data, and click **Save**.

6. To delete the service:

- Locate the service you want to edit, then click the More icon (⋮) next to it, and choose **Delete**.
- You're prompted to confirm; click **Delete**.

7.

Create a service group

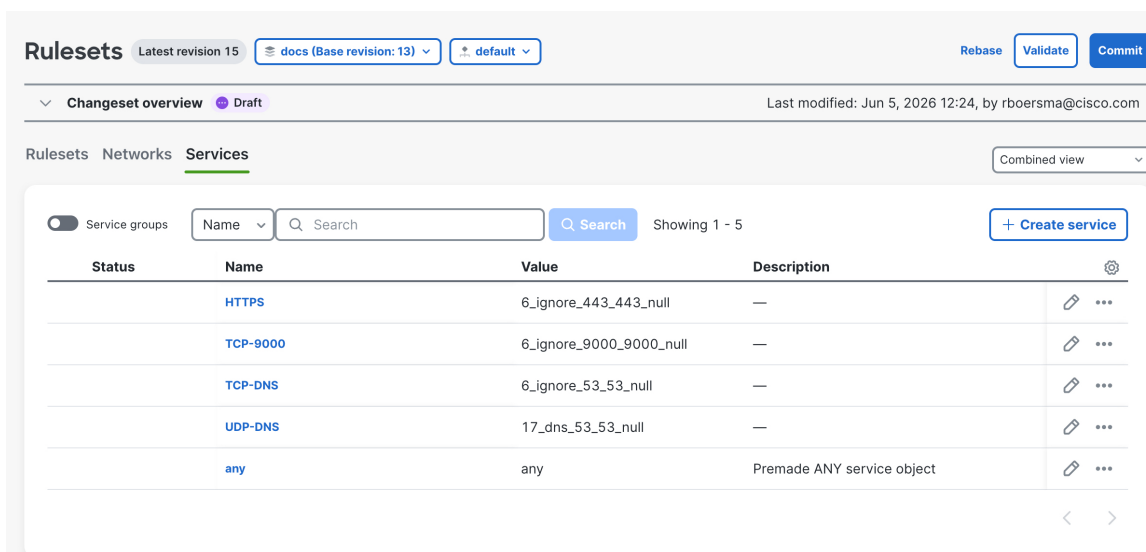
Create a service group to reuse related services as a single object in Mesh Policy Engine policy rules.

Create a service group by selecting and grouping individual or grouped services.

Procedure

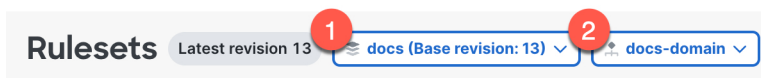
1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Policies > Services**.

Figure 55: Services



3. From the drop-down menu at the top, choose a changeset and then choose a domain.

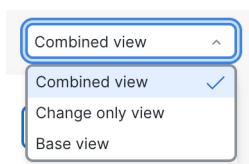
Figure 56: Choose a changeset and domain



If you don't have a changeset, create one. A newly created changeset is automatically checked out.

For the selected changeset, you can view all services (**Combined view**), new or changed services (**Change only view**), or existing services (**Base view**).

Figure 57: Views



4. Choose **Policies > Services**.
5. Click **Create service**.
6. Turn the **Create service group** toggle on.

7. Enter a **Name** for the service group.
8. Click the **Search** text box under **Manage Members** and select services from the **Individual Services** or **Grouped Services** categories to combine under the service group.
9. Click **Create**.

The newly created service group appears in the list of service groups on the **Services** page.

10. To edit a group:
 - a) Locate the group that you want to edit, then click the edit icon next to it.
 - b) Make the required changes to the service group.
 - c) Click **Save**.

Configure rulesets

Learn how rulesets organize ordered access or NAT rules for policy deployment to install targets in Mesh Policy Engine.

A ruleset is an ordered list of rules governing behavior between source and destination networks. Rules contain parameters such as source and destination, communication protocols, and other options to allow or deny traffic.

Create a ruleset

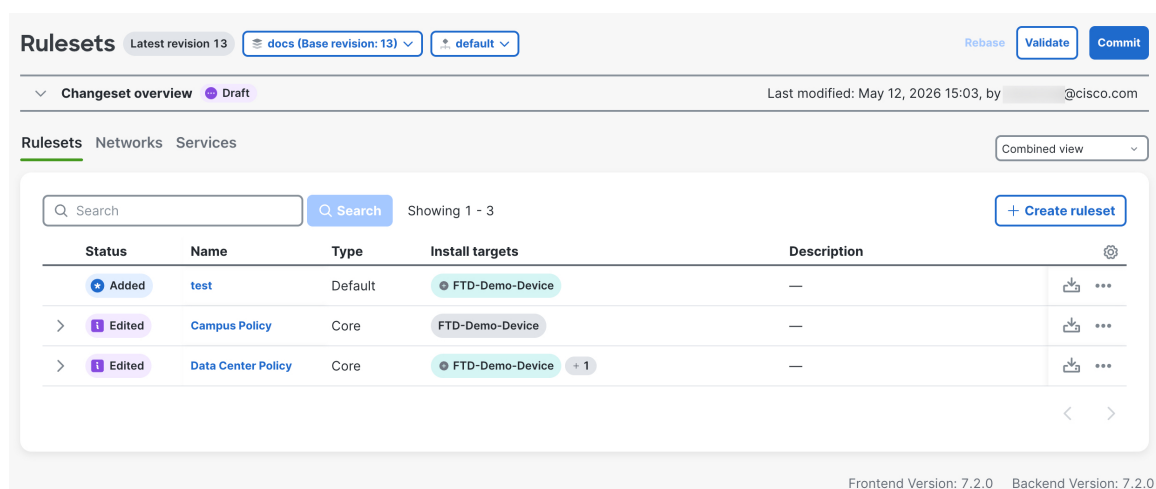
Create a ruleset, associate install targets, and add it to a changeset for policy deployment from Mesh Policy Engine.

Mesh Policy Engine lets you create a policy out of multiple rulesets, which are a collection of several rules. Note that the Mesh Policy Engine system sometimes refers to rulesets as policies. Follow the steps below to create a ruleset.

Procedure

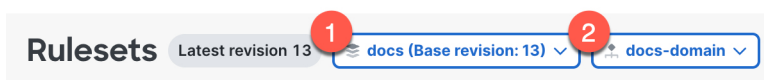
1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Policies > Rulesets**.

Figure 58: Rulesets



3. From the drop-down menu at the top, choose a changeset and then choose a domain.

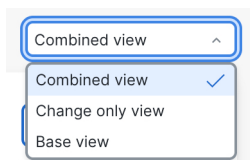
Figure 59: Choose a changeset and domain



If you don't have a changeset, create one. A newly created changeset is automatically checked out.

For the selected changeset, you can view all rulesets (**Combined view**), new or changed rulesets (**Change only view**), or existing rulesets (**Base view**).

Figure 60: Views



4. Click **Create ruleset** and set these fields.

Figure 61: Create ruleset

- **Name**—The name for the ruleset.
- **Type**—One these ruleset types:
 - **Priority**—A ruleset that contains allow and deny rules, which will be prepended to the final policy that is constructed during policy execution without any changes (post optimization). Both allow and deny rules are preserved and pushed as permit and deny flows. If you push multiple priority rulesets, the internal rule order is preserved, but the order of the rulesets can't be guaranteed.
 - **Core**—A ruleset that contains allow or deny rules. However, the deny rules are ignored, and only permit flows are pushed.
 - **Default**—A ruleset that contains allow and deny rules, which will be appended to the final policy that is constructed during policy execution without any changes (post optimization). Both allow and deny rules are preserved and pushed as permit and deny flows. If you push multiple priority rulesets, the internal rule order is preserved, but the order of the rulesets can't be guaranteed.
 - **NAT**—A ruleset of Network Address Translation (NAT) rules.

**Note**

Only Juniper and Fortinet support NAT policies.

- **Install targets**—Choose one or more install targets. For NAT rulesets, an install target can only be associated with one ruleset.
- **Description**—An optional description.

5. Click **Create**.

The newly created policy appears on the **Rulesets** tab.

6. To edit a ruleset:

- Click the More icon (...) corresponding to the ruleset that you want to edit, and choose **Edit policy**
- Make changes to the **Name**, **Install Targets**, or **Description** columns.
- Click **Save**.
- Click **Validate** to let Mesh Policy Engine compare your recent changes with the base revision and then submit the changes.
- Click **Commit** to update the ruleset.

Note that committing a change does not automatically update the ruleset. You must push the changes to the install target.

Create a rule

Create and position an access control rule in a Mesh Policy Engine ruleset.

You can add rules to existing rulesets. Depending on the type, rule order affects how and when policies are enforced. Place the new rule in the desired position within the ruleset's execution sequence.

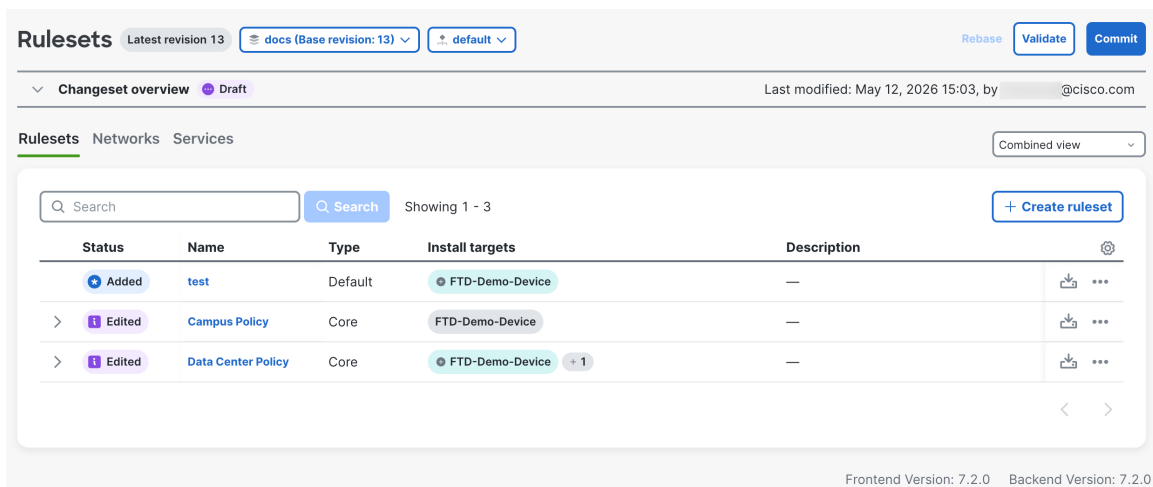
Before you begin

- [Create a network](#) on page 59 or [Create a network group](#) on page 62.
- [Create a service](#) on page 64 or [Create a service group](#) on page 66.

Procedure

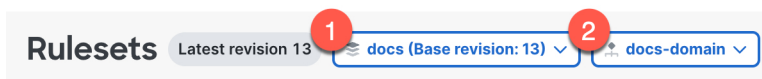
1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Policies > Rulesets**.

Figure 62: Rulesets



3. From the drop-down menu at the top, choose a changeset and then choose a domain.

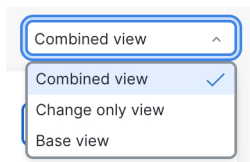
Figure 63: Choose a changeset and domain



If you don't have a changeset, create one. A newly created changeset is automatically checked out.

For the selected changeset, you can view all rulesets (**Combined view**), new or changed rulesets (**Change only view**), or existing rulesets (**Base view**).

Figure 64: Views



4. Click the ruleset name in which you want to create a rule.
5. Click **Create Rule**.

Figure 65: Create rule

Create rule docs (Base revision: 13) docs-domain Ruleset core1

Section: allowed-traffic

☒ Rule is enabled

Order * 1

Description

Specify access
Specify which users and networks can access which resources.

Action

☒ **Allow**
Specified traffic is allowed if security requirements met.

☐ **Block**
Specified traffic is blocked or forbidden for security reasons.

From *
Specify one or more sources
10.0.1.0 (1 Network)

To *
Specify one or more destinations
internet (1 Network)

Services *
Specify one or more communications protocol
any (1 Service)

Advanced settings

Rule targets

☒ Only apply to policy targets **Recommended**

☐ Add discovered targets from topology

Cancel Create

a) Optional: Choose a **Section** header.

Section headers can help you organize your rules.

b) Optional: Toggle off **Rule is enabled** if you don't want to use this rule yet.

c) Choose the **Order** number from the drop-down list.

If you choose the number of an existing rule, that rule and any following rules will be moved below the new rule. For example, if you already have 10 rules, you can add a rule to the end by choosing 11; or you can choose 5, and rules 5 through 10 will be renumbered 6 through 11.

d) Optional: Enter a **Description**.

e) Under **Specify access**, choose the **Action**, **Allow** or **Deny**.

f) In the **From** drop-down list, check one or more networks or network groups.

You can choose from existing networks or create one.

g) In the **To** drop-down list, check one or more networks or network groups.

You can choose from existing networks or create one.

h) In the **Services** drop-down list, check one or more services or service groups.

You can choose from existing services or create one.

i) Under **Advanced Settings**, choose the **Rule targets**. whether the rule should apply only to the targets you specify or also to those discovered from the topology.

- **Only apply to policy targets**—This rule will only be applied to selected install targets for the ruleset.
- **Add discovered targets from topology**—For core rulesets only, you can automatically apply the rule to appropriate install targets based on the matching characteristics in the topology.

j) Click **Create**.

The new rule is added to the ruleset.

Figure 66: View rules

Status	Order	Section	Source networks	Destination networks	Services	Action	Description	Rule targets
Added	1	test	Compute Services (1 network)	DB Network (1 network)	HTTPS (1 service)	Permit	-	Topology
	2	No section header	User Segment User Seg2 (2 networks)	Compute Services (1 network)	TCP-DNS UDP-DNS (2 services)	Permit	-	Apply
	3	No section header	User Segment (1 network)	ERP System (1 network)	HTTPS TCP-9000 (2 services)	Permit	-	Topology

- To edit or delete a rule, click the Edit icon (✎) and either change the settings and click **Save**, or click the Delete icon (Delete 🗑️).

Filter Rules

Filter Mesh Policy Engine rules by source, destination, or service to locate specific policy entries.

Use the search interface to narrow down the list of rules based on specific criteria for Source, Destination, or Service.

Procedure

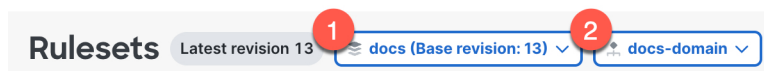
- From the Security Cloud Control Home page, click **Mesh Policy Engine**.
- Choose **Policies > Rulesets**.

Figure 67: Rulesets

Status	Name	Type	Install targets	Description
Added	test	Default	FTD-Demo-Device	—
Edited	Campus Policy	Core	FTD-Demo-Device	—
Edited	Data Center Policy	Core	FTD-Demo-Device + 1	—

3. Optional: From the drop-down menu at the top, choose a changeset and then choose a domain.

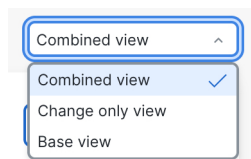
Figure 68: Choose a changeset and domain



If you only want to view the committed rules, then leave the selection at the latest revision.

If you selected a changeset, you can view all rulesets (**Combined view**), new or changed rulesets (**Change only view**), or existing rulesets (**Base view**).

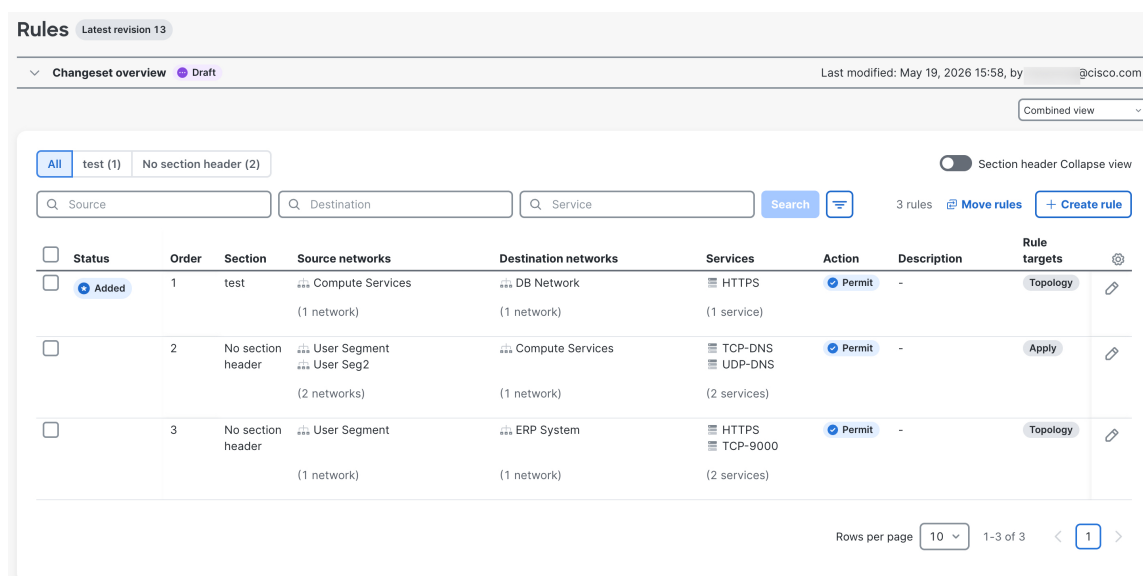
Figure 69: Views



4. Click the ruleset that contains the rules you wish to view.

You see a table containing all rules in the selected ruleset.

Figure 70: View rules



5. Locate the search bars. Three distinct search fields, **Source**, **Destination**, and **Service**, are displayed above the table of rules.
6. Select a filter category. For example, click the **Source** field to filter by the source network.
7. Choose a filter operator.

The drop-down menu lists the available filter methods. Select the operator that matches your search requirements:

- Name equals (=): Finds entries where the name matches your input exactly.
- Name prefix (^): Finds all entries where the name begins with your input characters.
- Value equals (=): Finds entries where the specific value such as IP address or network identifier matches your input exactly.
- Value includes shorter prefix (<): Searches for broader network ranges or services that include this shorter prefix.

- Value includes longer prefix (>): Searches for more specific, granular subnets or ranges within a broader network or service that you input.

8. Enter the search criteria by typing the specific name or value into the selected search field.

9. To apply the filter, click **Search**.

The display is refreshed and the rules that match your search criteria appear.

Push a ruleset to an install target

Push a ruleset from a committed changeset to one or all associated install targets.

After you are done with all the changes in the changeset and have committed it, you can push the changes to the install targets for each ruleset. When you push a new or changed ruleset, Mesh Policy Engine:

1. Gathers all rulesets assigned to the install target.
2. Normalizes and optimizes core rules together.
3. Preserves priority and default rules in their respective sections.
4. Validates NAT rules (if any).
5. Generates a single, complete device policy.
6. Replaces the install target's existing policy with the result, or in the case of Firewall Threat Defense, adds or replaces the child policy, maintaining the existing assigned policy as the parent policy. All rules are put in the Default section in the Firewall Threat Defense policy.

Procedure

1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Policies > Rulesets**
3. Click the **Push policy** icon next to the ruleset that you want to push.
4. In the **Push Policy** page, verify the install target, hostname, and the gateway set ID of the install target.
5. Click **Push to Device** to push the ruleset to an install target.

To push the ruleset to all the associated install targets, click **Select all** and click **Push to all devices** in the upper right corner.



Note

- For an ASA, the ruleset that is pushed gets staged in the **Device with Pending Changes** menu. You must review the ruleset and deploy it to the ASA manually.
- For all other active install targets, the ruleset gets pushed to the install target directly.

Import policies

Learn how to import existing firewall policies into Mesh Policy Engine and review supported mappings, unsupported elements, and previous import requests.

In addition to creating new policies, you can import existing policies from an install target. When a policy is imported, Mesh Policy Engine creates a changeset automatically to account for the new policy and prompts the user to accept or reject the changeset.

Imported policy mapping

Learn how Mesh Policy Engine maps imported firewall policies to supported policy objects and handles unsupported parameters.

This section describes how Mesh Policy Engine maps imported policies to Mesh Policy Engine components, including how unsupported parameters are handled.

ASA policy mapping

Find ASA policy import schema requirements and mappings for access groups, network objects, services, access lists, and ICMP names.

This topic describes how Mesh Policy Engine ingests an imported ASA policy.

ASA and equivalent Mesh Policy Engine entities

This table shows the entities in ASA and their equivalents in Mesh Policy Engine.

ASA	Mesh Policy Engine
Access Group	Policy
Access List	Rule
Service group	Service group
Service object	Service
Network group	Network group
Network object	Network

Access group mapping

When you import a policy, Mesh Policy Engine retrieves the device configuration, parses it, and imports access groups.

After extracting the access group and its associated objects, Mesh Policy Engine locates the access group that matches your selected policy name. Then, it aligns the other components of the access group with their counterparts in Mesh Policy Engine. Each access group can contain multiple access lists, which reference various network and service objects.

Network objects mapping

ASA network objects are categorized into three distinct types: HOST, SUBNET, and ANY. Each type maps to the Mesh Policy Engine network objects as described here:

HOST—Represents a single IP address without a subnet. It is mapped to CIDR format with a /32 suffix (example: 192.168.1.10/32).

SUBNET—Represents a network defined by an IP address and a subnet mask. Mesh Policy Engine converts the IP address and subnet mask to CIDR notation (example: IP: 192.168.1.0, Subnet: 255.255.255.0 → 192.168.1.0/24).

ANY—Represents all possible IP addresses. It is mapped to the CIDR 0.0.0.0/0.

Network groups can contain multiple network objects or nested groups, and they are mapped to network groups in Mesh Policy Engine.

Service objects mapping

A service, also called as 'app' in Mesh Policy Engine is defined in a canonical string format. It encapsulates the information required to uniquely identify the service. ASA service objects and groups are aligned with Mesh Policy Engine 'apps' and 'app-groups', which are represented by these canonical strings.

The structure of a canonical string is illustrated here, using railroad diagrams:

Figure 71: Canonical String Structure

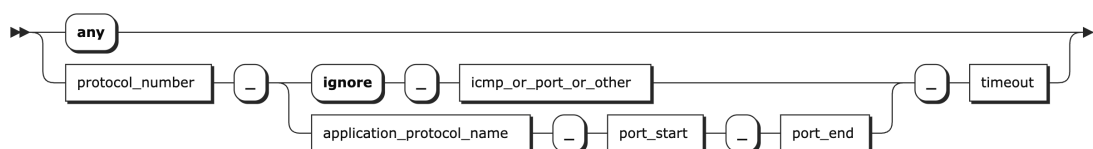
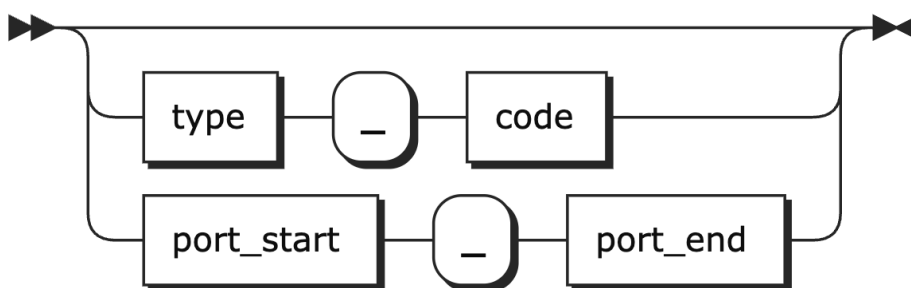


Figure 72: CMP or Port Specification Structure



An ASA service object contains several components:

- Protocol: Represented by either the name of the protocol or its corresponding numerical value.
- Port Selector: If applicable, its values are EQ, NEQ, LT, GT, or RANGE.
- Starting Port: Indicates the destination port, which can be specified as a literal or a number.
- Ending Port: Indicates the concluding port for the range and is specified only when the port selector is set to RANGE.
- Type Code: Includes type and code literals that define an ICMP service.

Mesh Policy Engine maps the ASA service as described here:

- Translates the ASA protocol (either name or number) to the corresponding Mesh Policy Engine protocol, which includes a list of supported protocols: `pptp`, `are`, `ipse`, `esp`, `ospf`, `icmp`, `igmp`, `vrrp`, `skip`, `pin`, `ah`, `tcp`, and `udp`.
- Converts the literal port values to their default known values; for instance, `https` corresponds to port 443. Refer the [table](#) for a comprehensive list of supported port literals.
- Associates the ICMP type literals with their corresponding type and code numbers. Refer the [table](#) for a comprehensive list of supported ICMP names.
- Translates the services `any`, `ip`, and `nos` to `any` service in Mesh Policy Engine. (The literal `any` represents any service.)

- Adds the literal port value corresponding to a Mesh Policy Engine-supported application protocol to the service string. Supported protocols include `ftp`, `sqlnetv2`, `tftp`, `sip`, `rasp`, `sunrpc`, `msrpc`, and `dns`.
- For services with protocols that support ports, if ports are not provided, Mesh Policy Engine assumes the entire range of 0-65535.
- If the ICMP type and code are not specified, Mesh Policy Engine assumes "Echo request", type: 8, code: 0.
- If Mesh Policy Engine does not directly support the NEQ port selector, it uses the service for ports less than the specified value and another service for ports greater than the value. It then combines these two services into a single service group to use them together.
- ASA employs a protocol-prefixed notation for implicit objects, allowing for their use without prior creation. Mesh Policy Engine supports the following services using this notation: `tcp-echo`, `tcp-sqlnet`, `udp-echo`, and `tcp-rtsp`. Mesh Policy Engine creates these services and maps them with default ports values if applicable.
- ASA doesn't support service timeouts. When creating a canonical string, Mesh Policy Engine replaces `timeout` with `null`.

Mesh Policy Engine creates an 'app' using a canonical string that encodes these details. For instance, an ASA object with the TCP protocol destination equal to port 80 is translated to: `6_ignore_80_80_null`. This service string uniquely defines an 'app' in Mesh Policy Engine.

Access list mapping

Access lists in ASA function similarly to rules in Mesh Policy Engine. Access lists can either reference an existing network and service objects, or declare them inline. If an existing object is referenced, Mesh Policy Engine first ensures that the object was created prior to the access list and the preceding rules, and then maps it to the equivalents. In cases where objects are declared inline, Mesh Policy Engine creates these objects before mapping. When parsing access lists, Mesh Policy Engine supports all the features of network and service objects that are mentioned earlier. ASA access lists can be either active or inactive. After you import them into Mesh Policy Engine, they appear as enabled or disabled.

Mesh Policy Engine does not support certain ASA features in access lists. For example, you cannot use these elements in access lists:

- Source ports
- Time ranges
- Interfaces instead of IP addresses
- Logging details

Table 2: Port Literals and Their Associated Ports

Port literal	Default port	Application Protocol Type
aol	5190	SIP
bgp	179	
biff	512	
bootpc	68	
bootps	67	
chargen	19	
cifs	3020	
citrix-ica	1494	

Port literal	Default port	Application Protocol Type
cmd	514	
ctiqbe	2748	
daytime	13	
discard	9	
dnsix	195	
domain	53	DNS
dns	53	DNS
echo	7	
exec	512	
finger	79	
ftp	21	FTP
ftp-data	20	FTP
gopher	70	
h323	1720	
hostname	101	
http	80	
https	443	
ident	113	
imap4	143	
irc	194	
isakmp	500	
klogin	543	
kshell	544	
kerberos	749	
ldap	389	
ldaps	636	
login	513	
lotusnotes	1352	
lpd	515	
mobile-ip	434	
msrpc	135	MSRPC
nameserver	42	
netbios-dgm	138	
netbios-ns	137	

Port literal	Default port	Application Protocol Type
netbios-ssn	139	
nfs	2049	
nntp	119	
ntp	123	
pcanywhere-data	5631	
pcanywhere-status	5632	
pim-auto-rp	496	
pop2	109	
pop3	110	
pptp	1723	
radius	1645	
radius-acct	1646	
rip	520	
rpc	111	SUNRPC
rsh	514	
rtsp	554	RTSP
secureid-udp	5510	
sip	5060	SIP
smtp	25	
snmp	161	
snmptrap	162	
sqlnet	1521	SQLNETV2
ssh	22	
sunrpc	111	SUNRPC
syslog	514 for udp, 6514 for tcp	
tacacs	49	
talk	517	
telnet	23	
tftp	69	TFTP
time	37	
uucp	540	
vxlan	4789	
who	513	
whois	43	

Port literal	Default port	Application Protocol Type
www	80	
xmcp	177	

Table 3: Supported ICMP names

Service Name	Type	Service Name	Type
echo-reply	0	unreachable	3
source-quench	4	redirect	5
alternate-address	6	echo	8
router-advertisement	9	router-solicitation	10
time-exceeded	11	parameter-problem	12
timestamp-request	13	timestamp-reply	14
information-request	15	information-reply	16
mask-request	17	mask-reply	18
traceroute	30	conversion-error	31
mobile-redirect	32		

**Note**

ICMP code is always zero.

Firewall Threat Defense policy mapping

Find Firewall Threat Defense policy import requirements, supported object mappings, and error handling details for Mesh Policy Engine.

You can import Firewall Threat Defense policies from the Cloud-Delivered Firewall Management Center. Mesh Policy Engine translates device-specific configurations into its vendor-neutral policy model. During the import, Mesh Policy Engine preserves as much of the original policy intent as possible and clearly indicates any elements that it cannot fully translate.

This section describes how Mesh Policy Engine ingests the policy.

Firewall Threat Defense and equivalent Mesh Policy Engine entities

Elements of the Firewall Threat Defense policy that are supported by Mesh Policy Engine are directly mapped to their equivalents. Elements that are not supported by Mesh Policy Engine are replaced with empty place-holders to preserve the rule structure.

This table shows the entities in a Firewall Threat Defense policy and their equivalents in Mesh Policy Engine.

Firewall Threat Defense	Mesh Policy Engine
Access Control Policy	Policy
Security Policy	

Firewall Threat Defense	Mesh Policy Engine
Access Rule	Rule
Network Object	Network
Network Group	Network Group
Port Object	Service
Port Object Group	Service Group

Handling import errors

If your Threat Defense policy contains unsupported elements, Mesh Policy Engine manages them appropriately. You can see what part of the policy couldn't be imported and take actions accordingly. See the list for details.

- Import process is a success: The policy is imported even with unsupported elements.
- Changeset marked for review: The changeset status is set to `ERROR`. These errors appear as conflicts that must be resolved before you can proceed.
- Errors Visible in UI: You see an import error in the UI when an unsupported element is flagged.
- Resolution Required: You must resolve these errors in the UI before you can validate and commit the changeset.

After you resolve the errors, the changeset status changes from `ERROR` to `DRAFT`. You can validate and commit the changeset.

Network objects mapping

Host values are automatically converted to CIDR notation (/32) during import.

Table 4: Network objects configuration mapping

Threat Defense Policy Element	Mesh Policy Engine Mapping	Example
Host	Network	10.0.0.1 → 10.0.0.1/32
Network	Network	10.0.0.0/24 → 10.0.0.0/24
Range	Network	10.0.0.1-10.0.0.100 → 10.0.0.1-10.0.0.100
FQDN	Creates empty network object	Shows an import error: Network object 'server-dns' was dropped because type FQDN is not supported in MPE. MPE supports Host, Network, and Range network object types only.

Threat Defense rules can contain inline network literals that are not defined as named objects. These are automatically converted to named networks during the import.

Network groups mapping

Network groups are supported and can contain:

- Individual network object references (Host, Network, Range)
- Inline network literals (converted to auto-generated endpoints)
- Nested network group references
- Mixed combinations of all the above



Note

If a network group contains FQDN objects, those objects become nested empty groups within the parent group, and you can see the corresponding import errors.

Table 5: Network group configuration mapping

Threat Defense Configuration	Mesh Policy Engine mapping
Objects only	Network group with object references
Literals only	Network group + auto-generated networks
Objects and literals	Network group with both
Nested groups	Network group with nested group references

Services mapping

Table 6: Supported service objects configurations

Threat Defense Configuration	Mesh Policy Engine Mapping	Example
TCP with single port	Service	TCP/443 → TCP app on port 443
TCP with port range	Service	TCP/8080-8090 → TCP app with range
UDP with port range	Service	UDP/1000-2000 → UDP app with range
UDP with single port	Service	UDP/53 → UDP app on port 53
TCP any port	Service	TCP (no port) → TCP any
UDP any port	Service	UDP (no port) → UDP any
IP Protocol only	Service	See supported protocols below
ICMP with type AND code	Service	ICMP 8/0 → ICMP echo request

Table 7: Supported IP protocols

Protocol	Number
ICMP	1
IGMP	2
GRE	47
ESP	50
AH	51
SKIP	57
OSPFv2	89
PIM	103
VRRP	112

Unsupported service object configurations results in creating an empty service group. Here are the unsupported configurations and the corresponding errors.

Table 8: Unsupported service objects

Configuration	Import Error
ICMPv6	ICMP object 'ping6' was dropped because ICMPv6 is not supported in MPE. MPE supports ICMPv4 only.
ICMP without code	ICMP object 'ping-type-only' (type 8) was dropped because it lacks an ICMP code. MPE requires both ICMP type and code to be specified.
All protocols on specific port	Port object 'all-8080' was dropped because all protocols on a specific port (8080) are not supported in MPE. MPE supports 'any' only without a port.
Unknown/unsupported protocol	Port object 'custom-proto' was dropped because protocol 'XYZ' is not supported in MPE. MPE supports TCP, UDP, ICMP, ESP, GRE, and AH protocols.

Threat Defense rules can contain inline port literals that specify protocol and port directly. These are automatically converted to named services during import.

Service groups mapping

Service groups in Threat Defense policy are mapped with the following behavior:

Scenario	Mesh Policy Engine Mapping
All members supported	Service group with all members
Some members unsupported	Service group with supported members + nested empty groups
All members unsupported	Empty service group

Mesh Policy Engine does not support applications and inline application filters of Threat Defense access control rules.

- If an access control rule references applications, Mesh Policy Engine creates an empty app group named after the referenced application (such as FTP) during policy import and attaches the import error to that group. The imported rule references the app groups created by Mesh Policy Engine.
- If the access control rule contains inline application filters, during the import of the rule, Mesh Policy Engine displays an import error message with the details of the filter criteria that couldn't be imported. However, the imported rule remains functional without the application filter criteria.

Threat Defense Configuration	Mesh Policy Engine Behavior	Import Error
Application reference	Creates empty app group	Application '<appName>' is not supported by MPE.
Multiple applications	Each creates empty group	One import error per application (one per empty app group)
Inline Application Filters	Rule-level import error	MPE does not support application filters. Rule contains inline application filters: [filter details]

Security policy mapping

Threat Defense Element	Mesh Policy Engine Mapping	Notes
ALLOW	PERMIT	Standard allow rule
BLOCK	DENY	Standard deny rule
TRUST	PERMIT	Trust action mapped to permit Trust action bypasses deep inspection
MONITOR	N/A	Not imported (rule is skipped)
BLOCK_INTERACTIVE		
BLOCK_RESET		
Empty source networks	"any" network	Matches all source addresses
Empty destination networks	"any" network	Matches all destination addresses

Threat Defense Element	Mesh Policy Engine Mapping	Notes
0.0.0.0/0 literal	Network 0.0.0.0/0	Auto-generated "any address" endpoint
Empty source ports	"any" service	Matches all ports/protocols

FortiGate policy mapping

Find FortiGate policy import schema requirements and mappings for addresses, services, schedules, and security rules in Mesh Policy Engine.

FortiGate and equivalent Mesh Policy Engine entities

Fortinet entity model differs from the Mesh Policy Engine model.

Table 9: Comparison of entity models

FortiGate	Mesh Policy Engine
ADOM (Administrative Domains)	Manager
Policy package	Policy
Policy	Rule
Service	Service
Address	Network

Entities and their support in Mesh Policy Engine

The Address entity can be of two types: IP Range, and IP Mask.

Mesh Policy Engine supports [CIDR](#) address ranges. When Mesh Policy Engine finds an unsupported address type, it throws an exception.

The Service entity consists of Protocols, Timeouts, Service Types, and Session Helpers. Here is a list of service configurations that Mesh Policy Engine supports.

Protocol Types	Session Helpers	Timeouts	Service Types	IP Protocols (number - name)
• udp • tcp • icmp • ip	• auto • disable • dns-tcp • dns-udp • ftp • tftp • sip • rtsp	session-ttl timeout	firewall	1-ICMP, 2-IGMP, 89-OSPF,IGP, 112-VRRP, 57-SKIP, 103-PIM, 47-GRE, 50-ESP, 51-AH, 6-TCP, 17-UDP

 Note

Mesh Policy Engine does not support IPv6 addresses.

Mesh Policy Engine throws an exception in these cases:

- Unsupported address type
- Service type is 'proxy'
- Unsupported protocol type
- An address group with a non-empty list of excluded members.

Import a policy

Import firewall policies from FortiManager, Cisco Defense Orchestrator, or Panorama into Mesh Policy Engine.

To import a policies from a security manager, complete these steps. Once imported, you can use these policies across all your install targets. This procedure supports the following managers:

- FortiManager–FortiGate policies
- Security Cloud Control Firewall Management–ASA policies
- Cloud-Delivered Firewall Management Center–Firewall Threat Defense policies
- Panorama–Palo Alto Networks policies

See the [Mesh Policy Engine API](#) for details about importing policies directly using the payload method.

When you import a policy, Mesh Policy Engine creates a changeset automatically.

Before you begin

- Add your credentials for FortiManager, Security Cloud Control Firewall Management, or Panorama according to [Manage credentials](#) on page 20.
- Review how policies are mapped in Mesh Policy Engine, including unsupported parameters. See [Imported policy mapping](#) on page 75.

Procedure

1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Policies > Import > Import Policy**.

Figure 73: Create import

Import Policies

Import device rules for centralized policy management.

Create import Previous import requests Import devices

1 Device changes

2 Summary

Domain name *

Policy name *

Target domain name

Target policy name

Import details *

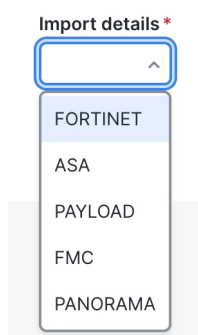
Reset form

Configure these parameters.

- **Domain Name**—If your manager supports domains, enter the name. If your manager doesn't support domains, such as for ASA or Firewall Threat Defense, enter the name of the Mesh Policy Engine domain in which you want to add this policy.
- **Policy Name**—Enter the name of the policy in the manager.
- **Target Domain Name**—Enter the name of the domain in Mesh Policy Engine where you want to add this policy. If you leave this name blank, the **Domain Name** is used. For install targets that don't support domains, leave this field blank.
- **Target Policy Name**—If you want to change the name from the original, enter a new name for use in Mesh Policy Engine.

- **Import Details**—Choose the manager type.

Figure 74: Install target type



Import details *

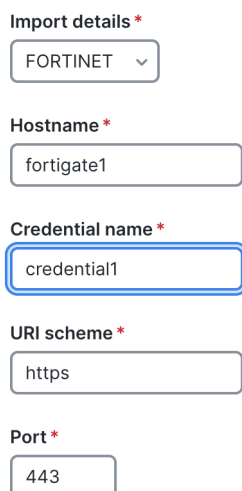
- FORTINET
- ASA
- PAYLOAD
- FMC
- PANORAMA

 Note

For **PAYLOAD**, see the [Mesh Policy Engine API](#).

3. For **FORTINET**, identify the connection settings for FortiManager.

Figure 75: Fortigate connection details



Import details *

FORTINET

Hostname *

fortigate1

Credential name *

credential1

URI scheme *

https

Port *

443

- **Hostname**—The hostname or IP address for FortiManager.
- **Credential Name**—The credential name that you specified in [Manage credentials](#) on page 20
- **URI scheme**—https or http
- **Port**—443

4. For **ASA**, identify the connection settings for Security Cloud Control Firewall Management.

Figure 76: ASA connection details

Import details *

ASA

Hostname *

security.cisco.com

Credential name *

fw-mgmt

Device name *

asa1

URI scheme *

https

Port *

443

- **Hostname**—security.cisco.com
- **Credential name**—The credential name for Security Cloud Control Firewall Management that you specified in [Manage credentials](#) on page 20.
- **Device name**—Specify the name of the ASA present in Security Cloud Control Firewall Management. The name must match exactly.
- **URI scheme**—https
- **Port**—443

5. For **FMC**, you don't need to specify any Cloud-Delivered Firewall Management Center connection settings.

6. For **Panorama**, identify the connection settings for Panorama.

Figure 77: Panorama connection details

Import details *

PANORAMA ▾

Hostname *

panorama1

Credential name *

pw3

URI scheme *

https

Port *

443

Rule type *

Pre rules ▾

Device group

- **Hostname**—The hostname or IP address for Panorama.
- **Credential Name**—The credential name that you specified in [Manage credentials](#) on page 20
- **URI scheme**—https or http
- **Port**—443
- **Rule type**—Pre rules or Post rules
- **Device group**—The name of the device group.

7. Click **Submit** to start the import.

Figure 78: Policy import in progress

Import Policies

Import device rules for centralized policy management.

Create import Previous import requests Import devices

✓ Device changes

2 Summary

Policy import in progress
The import job has been initiated and is being processed by the pipeline

Go back Close

The import process takes a few seconds. After the import is successful, you see a success message and the imported policy is available in a new changeset.

If the connection fails, you see the following error:

Figure 79: The import job failed

The import job failed

Processing policy import id d6d725c8-a6c8-46e7-bccd-77b7f6b500a6 for policy new-policy, manager docs ended with error: Unable to find entity credential 'asa1'. Please ensure the credential has been added and that the credential name matches exactly

- Click **Checkout** to see the policy under the changeset.

When you import a policy, Mesh Policy Engine creates a changeset automatically.

- Review the policy, and click **Commit**.

You can see the imported policy under **Policies > Rulesets** when you select the new changeset and domain.

View Previous Imports

View previous Mesh Policy Engine policy import requests and inspect their status, source, target, and error details.

You can view information about your previous requests to import policies into Mesh Policy Engine. Complete these steps to see the details of past import requests.

Procedure

- From the Security Cloud Control Home page, click **Mesh Policy Engine**.
- Choose **Policies > Import > Previous Imports**.

The **Previous import requests** page displays all your earlier import requests in a tabular format, showing the request ID, status, user email address, date, and more.

Figure 80: Previous import requests

Previous import requests						
Import device rules for centralized policy management.						
Create import Previous import requests Import devices						
Search user... Search Start time → End time Calendar Status						
Request ID	Status	Request system	User	Created	Error code	
d6d725c8-a6c8-46e7-bccd-77b7f6b500a6	Rejected	MPM_POLICY_IMPORT		Jun 17, 2026 9:23	-	
7a4b461b-385c-4605-8c96-d3ce9b206f59	Rejected	MPM_POLICY_IMPORT		Mar 25, 2026 20:49	-	

- To see the details of an import, click its **Request ID**.

Figure 81: Import request details

7a4b461b-385c-4605-8c96-d3ce9b206f59

×

Status	REJECTED
Request System	MPM_POLICY_IMPORT
Import Type	FMC
Domain Name	test-fmc-domain
Policy Name	test-fmc-policy
Target Domain Name	test-fmc-domain
Target Policy Name	test-fmc-policy
Created Time	Mar 25, 2026 20:49
Modified Time	Mar 25, 2026 20:49
Changeset IDs	—
Error Code	—
Error Message	Processing policy import id 7a4b461b-385c-4605-8c96-d3ce9b206f59 for policy test-fmc-policy, manager test-fmc-domain ended with error: Invalid export request for policy 'test-fmc-policy': { "errorCode": "BAD_REQUEST", "errorMsg": "Invalid policy name", "details": { "errorCode": "ERR_NO_RETRY_EXPORT_FAILED", "lastUpdateTime": "2026-03-26T02:49:12.371537418Z", "status": "EXPORT_FAILED" } }}

Import Details

{ }

5 Changesets

Topics:

- [About changesets](#)
- [Changeset workflow](#)
- [Manage changesets](#)
- [Rebase a changeset](#)
- [Validate and commit a changeset](#)

Learn how to use changesets to organize, validate, commit, and resolve conflicts in updates to security policies and related objects.

A changeset is a formal collection of changes or change commits made to a policy.

About changesets

An overview of changesets and how they help you organize, review, and track policy updates before you validate and commit them.

A changeset is a formal collection of changes or change commits made to a policy. Policy changes could span across several components such as policies, networks, or services. Mesh Policy Engine tracks any change done to the components using a changeset. Therefore, it is mandatory to select a changeset before making changes to any of these components.

The **Changesets** page lists all the changesets along with data such as status, name, and timestamps. To filter changesets based on their statuses, choose an option from the **Status** drop-down list. You could also choose what data you want to see using the **Gear** icon on the right. Click **Reset All** to clear all the filters.

Here are some terms to understand when working with changesets in Mesh Policy Engine:

- **Revision:** Each revision represents the state of the data at a certain point in its history. Every time a changeset is committed, a new revision is created.
- **Latest revision:** The latest revision is the current version of all data that exists in the Mesh Policy Engine system.
- **Base revision:** The revision from which a changeset is created.
- **Submit:** Applies all changes to the entities, creating a new revision after successful processing.
- **Error detection:** A changeset is validated by passing it through a series of syntactic and semantic validations to locate possible configuration errors. You must resolve these errors for managing or committing a policy successfully.
- **Rebase:** The changing of the base revision of a changeset to the latest revision.
- **Conflict:** A conflict occurs if the latest revision of the entities in the changeset are different when performing a rebase.

Changeset workflow

Find details about changeset states and actions for creating, validating, rebasing, resolving conflicts, and committing policy updates.

You must use a changeset to collect one or more changes you want to make to a policy or its objects.

1. Choose or create a changeset.

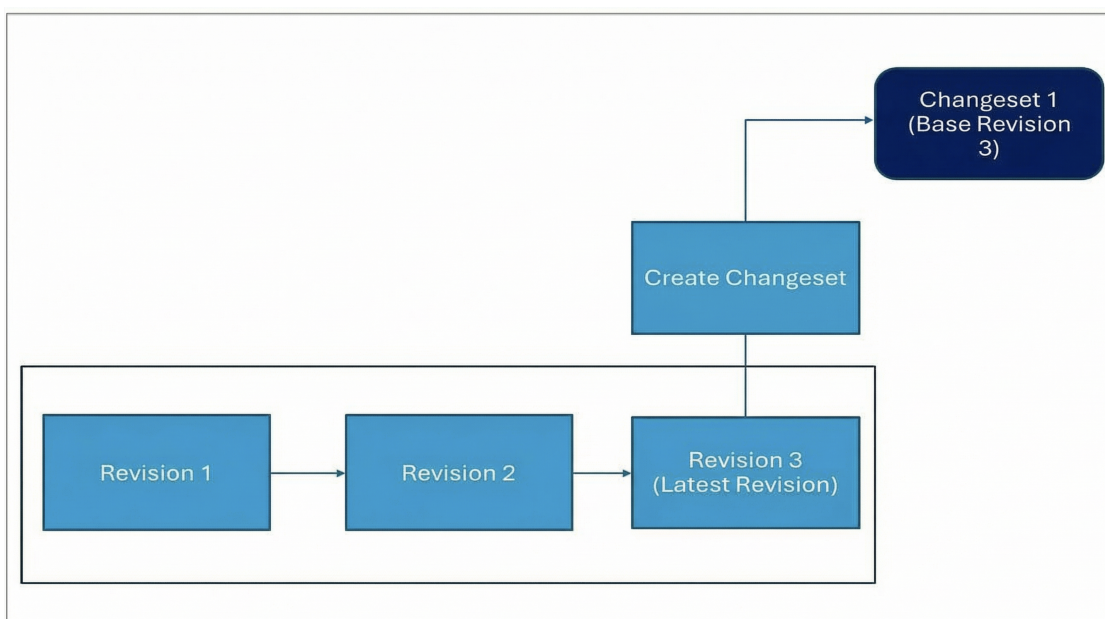
You will add all policy changes to the changeset.

a. Create a new changeset.

You can only create a changeset from the latest revision.

As you can see in this sample figure, changeset 1 is created from revision 3, which is the latest revision.

Figure 82: Create a changeset

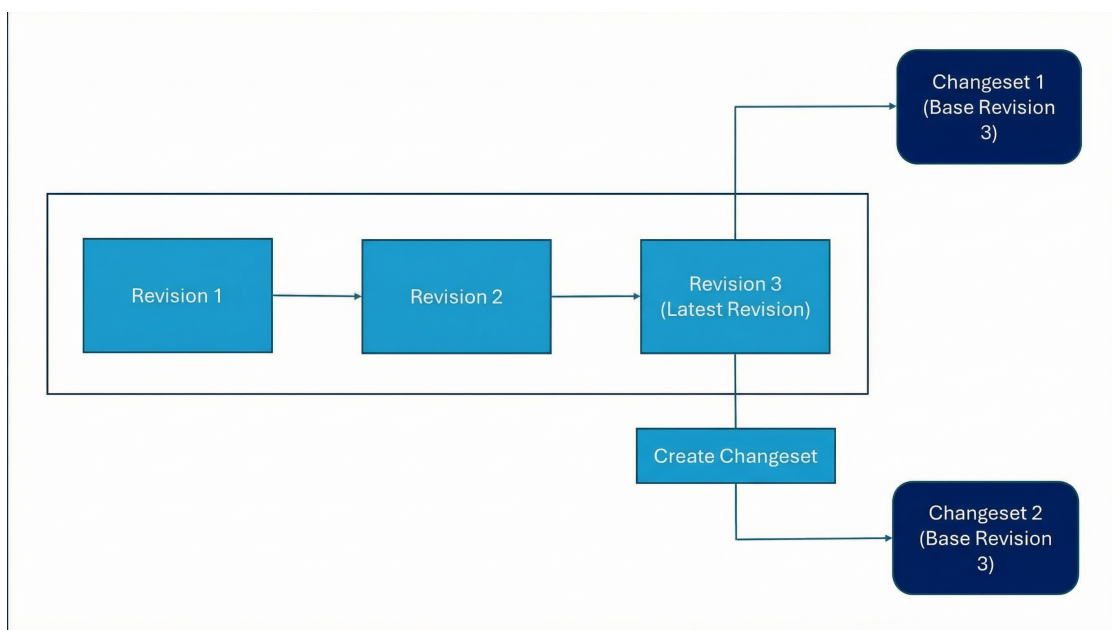


b. Create additional changesets.

You can create multiple changesets from the same latest revision. However, only the first committed changeset can be applied to that revision. For any in-progress changesets that you commit later, you will need to rebase it to a new revision before you can commit it.

In this sample figure, changeset 2 is also created from revision 3. Changeset 2 does not contain the changes made in changeset 1. If there are conflicts between the changes on these changesets, you can resolve them after validation.

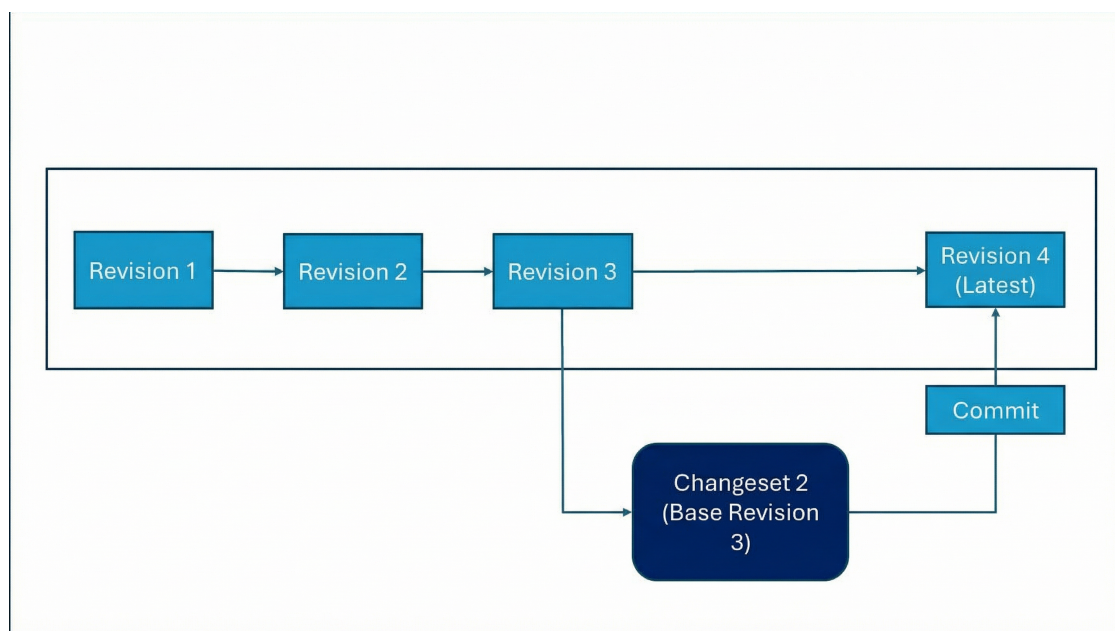
Figure 83: Multiple changesets



2. Rebase if needed.

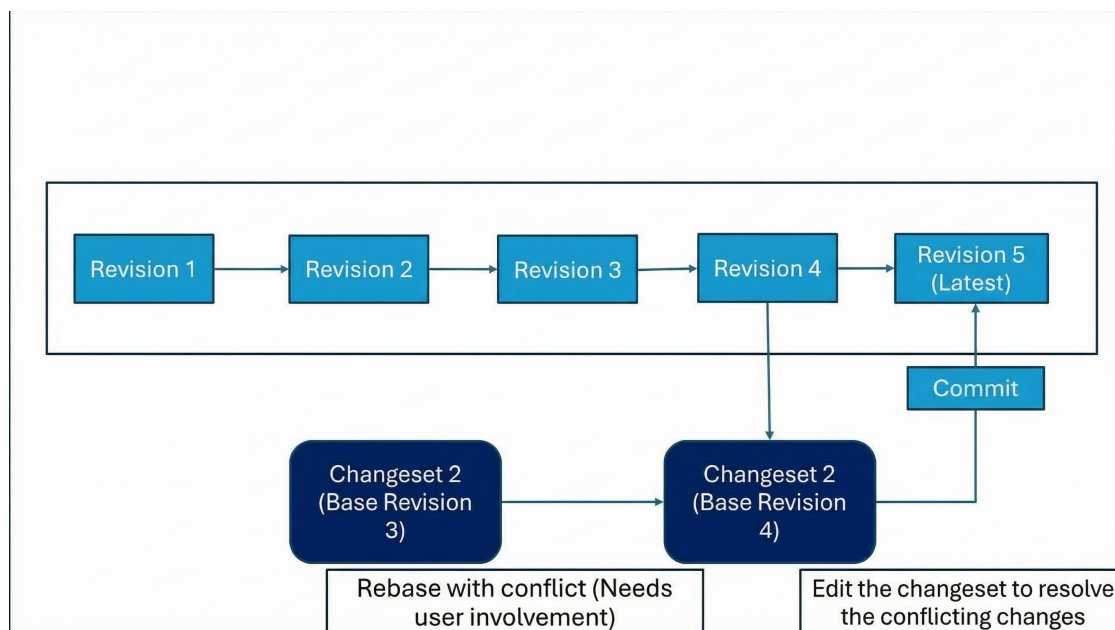
If changeset 1 is committed before changeset 2, and a new revision 4 is created. The current changeset 2 tries to commit to the new revision 4, which is not the same as revision 3 from which it was created. Changes in changeset 2 are not committed because its base revision is different from the current latest revision.

Figure 84: Revision error



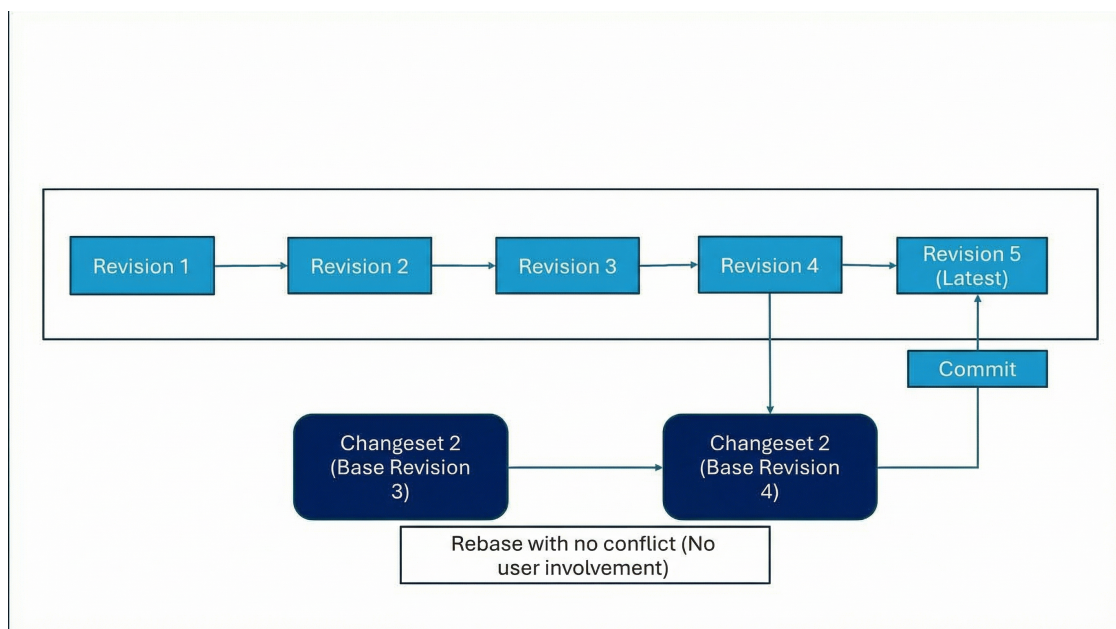
Submit the changeset for rebasing to change the base revision to current latest revision and detect conflicts, if any.

Figure 85: Rebase and resolve conflicts in changeset



If changeset 2 does not have any conflicts, the base revision is updated to the current latest revision (revision 4) and committed without any user intervention. The sample figure here shows the rebase without conflicts. Committing changes from changeset 2 into revision 4 creates a new revision, revision 5. Revision 5 now becomes the new latest.

Figure 86: Rebase changeset without conflicts



3. Validate changes and resolve conflicts.

Submit the changeset for validation and detect errors, if any. This action does not yet commit the changes. Resolve the conflicts detected in the conflict detection step, and reset the status of the changeset based on the original action.

The non-conflicting changes are carried over, and you can view and resolve the conflicting changes.

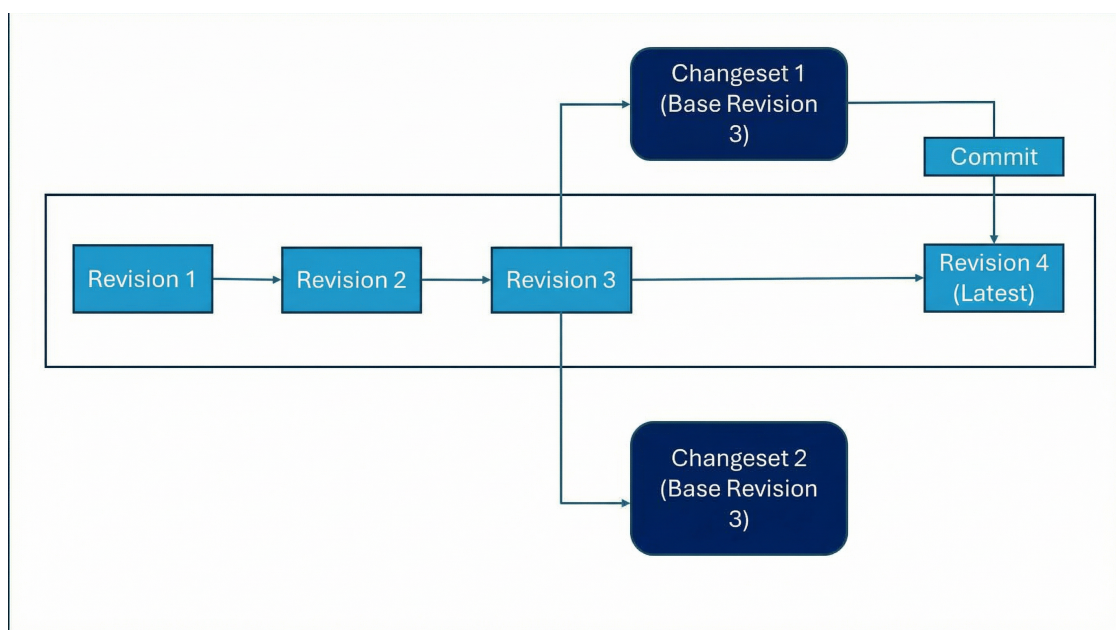
a. Rebase if needed.

4. Commit the changeset.

If a changeset's base revision matches the current latest, the system commits all changes in the changeset without conflicts.

In this sample figure, committing changes from changeset 1 creates a new revision, revision 4. Now revision 4 becomes the new latest revision.

Figure 87: Commit changeset



Manage changesets

Create, select, and review changesets to organize updates to rulesets, networks, and services before validation and commit.

To update a ruleset, network, or service, you need a changeset. Mesh Policy Engine requires you to choose a changeset when making changes. Follow these steps to create a changeset, choose a changeset for policy changes, view changeset details, or delete a changeset.

Procedure

1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Changesets**.
3. Create a new changeset.
 - a) Click **Create Changeset**.

Figure 88: Create changeset

Create new changeset

Name *
docs2

Description
Description

Cancel Create

- b) Enter a **Name** for the changeset.
- c) Optional: Enter a description about the changes that you plan to make as part of the changeset. You can choose to display the descriptions for all the changesets in the **Changesets** page.
- d) Click **Create**.

The changeset that you just created is automatically selected (checked out) on the **Policies** page so you can start making policy changes on this ruleset.

4. Choose (check out) a changeset for policy changes by clicking the **View** (👁) button.

Figure 89: Choose a changeset

Changesets Latest revision 22 docs2 (Base revision: 22) v

A changeset is a collection of modifications made to a system that includes policy updates, rule changes, and configuration updates. Use changesets to track, validate, and apply changes.

Status v Q Search Q Search Showing 1 - 29 + Create changeset

Status	Name	Created by	Base revision	Description	Created on	Last modified	
👁	docs2		22	—	Jul 8, 2026 13:51	Jul 8, 2026 13:51	👁 ...

5. To view a changeset's details, click the changeset name.

Figure 90: View a changeset

Changesets Latest revision 22 docs2 (Base revision: 22) ▾

A changeset is a collection of modifications made to a system that includes policy updates, rule changes, and configuration updates. Use changesets to track, validate, and apply changes.

Status ▾ Q Search Search Showing 1 - 29 + Create changeset

Status	Name	Created by	Base revision	Description	Created on	Last modified	
	docs2		22	—	Jul 8, 2026 13:51	Jul 8, 2026 13:51	 ...

Figure 91: Changeset details

docs2 

Status  DRAFT

Created by

Base revision 22

Description —

Created on Jul 8, 2026 13:51

Last modified Jul 8, 2026 13:51

Changeset actions

 Checkout

 Delete

See these **Status** values:

- **DRAFT**—The status of a changeset that was just created, until it is submitted or committed.
- **SUBMITTED**—The status of a changeset that was just submitted. Note that this is a temporary status and might change to one of the other statuses.
- **ERROR_DETECTED**—The status of a changeset that contains errors.
- **CONFLICT_DETECTED**—The status of a changeset that is yet to be committed while another changeset with changes to the same components was committed. Because the current changeset contains changes that are conflicting with the base revision, the conflicts must be resolved before the current changeset can be committed.
- **COMMITTED**—The status of a changeset that was just committed and updated to the current latest revision because it had no conflicts with the base revision.

6. To delete a changeset, click the More icon (...) corresponding to the changeset and click **Delete**.

Rebase a changeset

Resolve conflicting edits by rebasing a changeset so you can validate and commit its policy updates.

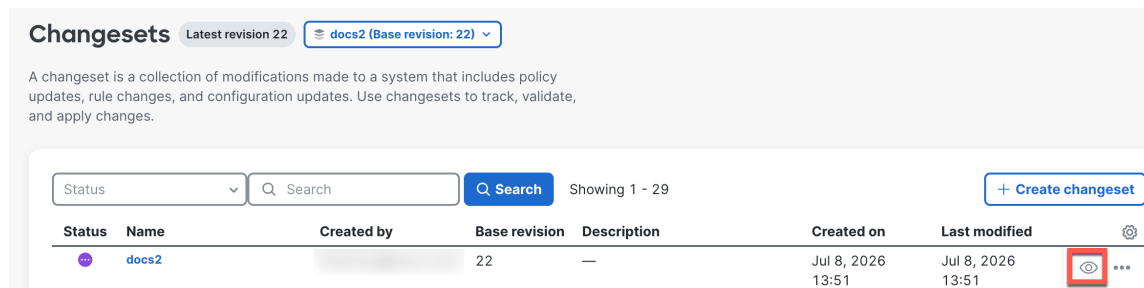
A conflict can occur when the same entity is edited by two different users and they both commit their edits. You must first resolve the conflict before you commit your updates.

Here is a generic process that outlines how to resolve changeset conflicts.

Procedure

1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Changesets**.
3. Choose (check out) a changeset by clicking the **View** (👁) button.

Figure 92: Choose a changeset



4. Click **Rebase**.
5. A conflict in an entity such as domain, ruleset, networks, or services, is identified by a yellow triangle icon (⚠). Click the ellipsis (...) icon for the entity and select **Resolve conflict**.

⚠	10.60.50.10-10.70.80.90-rrrr	10.60.50.10-10.70.80.92	descsrrrrr	⋮
	172.31.45.160/32	172.31.45.160/32	-test	Resolve conflict
	19.19.19.19	19.19.19.19/32	—	Link ticket
	ANY	0.0.0.0/0	—	Usage
	Blue Dev	10.55.55.128/30	nk_cs_2	Compare revisions
	Blue Server 2	10.55.55.129/32	—	Delete

6. In the **Resolve Conflict** page, from the **Metadata Conflicts** pane, select the change that you want to resolve.

The Metadata Conflicts pane gives you three choices to resolve the conflict:

- The base revision for the changeset.
- The latest revision for the changeset.
- The revision that you have made changes to.

Mesh Policy Engine resolves the conflict based on your choice. If you choose base revision, all your changes are reverted to the base revision data when you click Resolve. Similarly, the latest revision data is retained when you choose it. If you choose the revision in which you have made changes, your changes will be updated as the latest revision.

7. If the entity with a conflict is a group, such as a network group or service group, conflicts may occur within the group members. In these cases, the **Members Conflicts** pane displays the group members under each revision—base revision, latest revision, and your changeset revision.

Members Conflicts

Choose which members version to use for this network group

Base Version
Original network group

Latest
Previously Committed

Name	Value	Status	Description	Accept/Rejec
salesweb	192.168.5.80/32	Added	—	✓ ✕
nk_ep_101	1.1.1.1/32	Added	—	✓ ✕
Blue Server 8	10.55.55.135/32	Deleted	—	✓ ✕
Blue Server 6	10.55.55.133/32	Deleted	—	✓ ✕

Reject All
Accept All

Changeset
Changeset

Name	Value	Status	Description	Accept/Rejec
new-ep9	9.9.9.9/32	Added	—	✓ ✕
new-ep8	8.8.8.8/32	Added	—	✓ ✕

Reject All
Accept All

Accept or reject changes under each member in the revision of your choice.

When you finish, a **Members Result** pane summarizes the changes that you made to the selected revision.

Blue App1

Description
test

Members Result

Name	Value	Status	Description
salesweb	192.168.5.80/32	Added	—
nk_ep_101	1.1.1.1/32	Added	—
Blue Server 8	10.55.55.135/32	Deleted	—
Blue Server 6	10.55.55.133/32	Deleted	—
new-ep9	9.9.9.9/32	Added	—
new-ep8	8.8.8.8/32	Added	—

Reset
Resolve

8. Click **Resolve**.

The conflict is resolved. The entity appears as **Edited** on the corresponding entity page.

Validate and commit a changeset

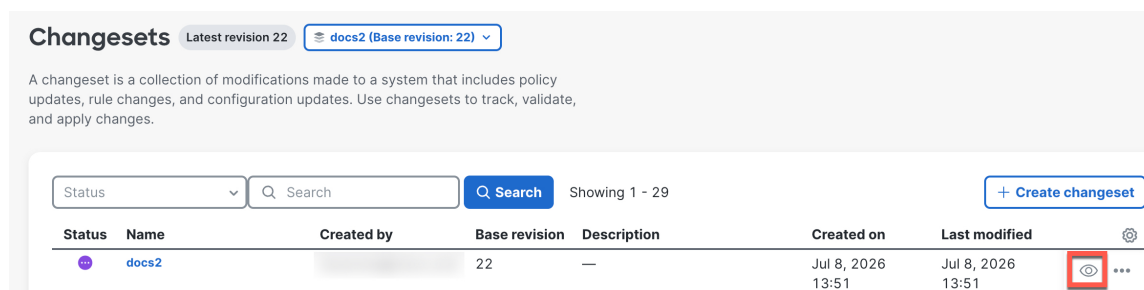
Review a draft changeset, validate its policy updates, and commit the completed changes.

You can make changes to a changeset only if it is in a draft state. You can't edit a changeset that has already been committed.

Procedure

1. From the Security Cloud Control Home page, click **Mesh Policy Engine**.
2. Choose **Changesets**.
3. Choose (check out) a changeset by clicking the **View** (👁) button.

Figure 93: Choose a changeset



4. Click **Validate**.

Ensure that the changeset you are trying to validate contains changes. You cannot commit a changeset without changes in it.

5. Resolve any errors.

This table lists the types of errors you might encounter in a changeset.

Error Type	Description	Trigger Conditions
Duplicate Name	An entity with the same name already exists.	You attempt to create an entity with a name that is already in use, or update an entity to a name that belongs to a different entity.
Empty Name	The entity name is blank or empty.	You attempt to create or update an entity without providing a valid name.
Missing	The entity does not exist in the system.	You attempt to update or delete an entity that does not exist.
Ticket Error	External ticket validation failed or required ticket is missing.	You attempt to create, update, or delete an entity, but the external ticket is either missing when required or fails validation.
Undeleted Device Sets	The entity has active gateway set associations.	You attempt to delete a domain or policy that is still associated with gateway sets.
Undeleted Relationship Errors	The entity is being referenced by other entities.	You attempt to delete an entity that is still being referenced by groups (network groups or service groups), security rules, or NAT rules.

Error Type	Description	Trigger Conditions
Rule Order Error	The rule ordering is invalid.	You attempt to update a policy's rule order with a new order that conflicts with rules being added, removed, or reordered.
Target Relationship Errors	Invalid gateway set (target) associations.	You attempt to update a policy to add gateway sets that do not exist, or to remove gateway set associations that do not exist.
Child Relationship Errors	Invalid child entity relationships	You attempt to create or update a group to add or remove child entities (networks, network groups, services, or service groups) that do not exist or have invalid relationships.
Dynamic Objects in Unpinned Rule	Dynamic objects cannot be added to unpinned rules	You attempt to add or update a rule to add dynamic objects to a rule that is not marked as fully pinned.
Rule Relationship Errors	Invalid relationships among the rule entities	You attempt to create or update a rule to add or remove source or destination networks, network groups, services, service groups, or dynamic objects that do not exist or have invalid relationships.
NAT Rule Match Errors	Invalid match criteria configuration in NAT rule	You attempt to create or update a NAT rule with invalid source or destination networks, network groups, services, or service group match criteria.
NAT Rule Translated Errors	Invalid translated configuration in NAT rule	You attempt to create or update a NAT rule with invalid source or destination networks or services translated configuration.

6. Click **Commit**.
