# Manage Policies Lifecycle in Secure Workload

## Segmentation Policy Basics

The purpose of segmentation and microsegmentation policies is to allow only the traffic your organization needs to conduct business, and block all other traffic. The goal is to reduce your network's attack surface without disrupting business operations.

Secure Workload segmentation policies allow or block traffic based on its source, destination, port, protocol, and a few other attributes that are typically platform-specific.

You can create some policies manually, and use Secure Workload's powerful automatic policy discovery feature to generate other policies based on existing network traffic.

You can review, refine, and analyze your policies, then enforce them when you are confident that they allow only the traffic that your organization needs.

**Important** Microsegmentation essentially creates a firewall around each workload.

Therefore, for traffic to pass between each consumer-provider pair, both ends of the conversation must allow the conversation to happen: The consumer and the provider must each have a policy that allows the traffic.

**Note** The terms *firewall rule*, *edge*, and *cluster edge* are sometimes used to mean "policy."

## Use Workspaces to Manage Policies

Workspaces (formerly called “Application workspaces” or “Applications”) are where you work with and manage policies.

You can perform all policy-related activities for a particular scope, such as creating, analyzing, and enforcing policies, in the workspace or workspaces associated with that scope.

Each workspace provides an isolated environment, allowing experimentation with no effect on other workspaces.

### Controlling User Access to Workspaces

Workspaces are meant to be used by multiple users from the same team as shared documents.

To control access to a workspace, assign user roles for the scope associated with the workspace. For more information, see the Roles section.

## Working with Policies: Navigating to the Workspaces Page

- **To work with policies, or to view existing application workspaces or create new ones:**

  Choose **Defend** > **Segmentation** from the navigation bar at the left side of the window.

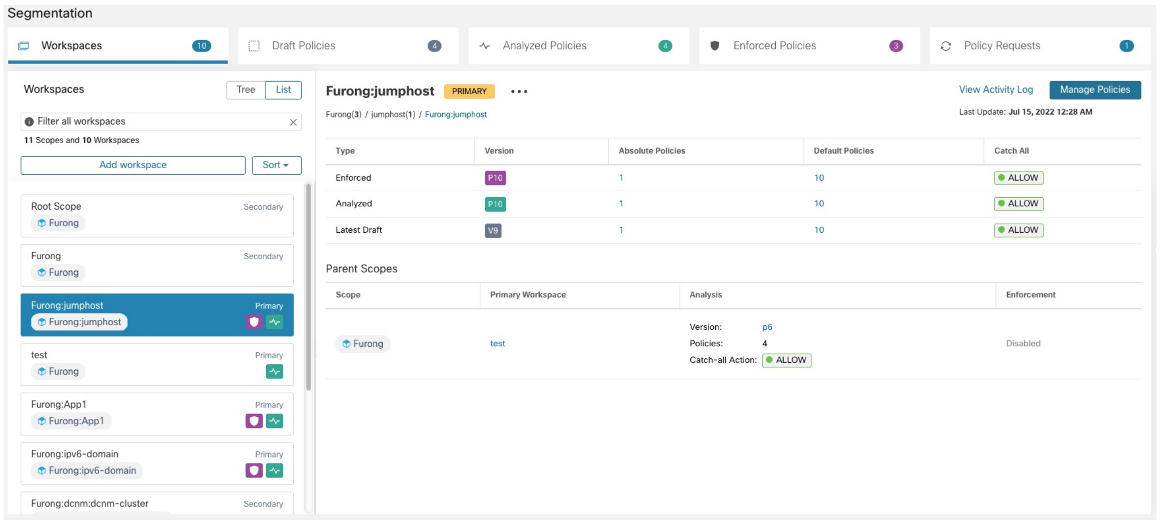
- **To view a particular workspace:**

  In the list of scopes at the left side of the Workspaces page, navigate to the scope associated with the workspace, then click the workspace. The current active workspace is highlighted in the list.

- **If you are looking at a workspace and want to return to the list of workspaces:**

  Click the **Workspaces** link near the left side of the page you are looking at.

*Figure 1: Workspace Management Page*



11 Scopes and 10 Workspaces

| | | |
|---|---|---|
| Root Scope | Furong | Secondary |
| Furong | Furong | Secondary |
| Furong:jumphost | Furong:jumphost | Primary |
| test | Furong | Primary |
| Furong:App1 | Furong:App1 | Primary |
| Furong:ipv6-domain | Furong:ipv6-domain | Primary |
| Furong:dcnm:dcnm-cluster | | Secondary |

Furong:jumphost PRIMARY

Furong(3) / jumphost(1) / Furong:jumphost

Last Update: Jul 15, 2022 12:28 AM

| Type | Version | Absolute Policies | Default Policies | Catch All |
|---|---|---|---|---|
| Enforced | P10 | 1 | 10 | ALLOW |
| Analyzed | P10 | 1 | 10 | ALLOW |
| Latest Draft | V9 | 1 | 10 | ALLOW |

Parent Scopes

| Scope | Primary Workspace | Analysis | Enforcement |
|---|---|---|---|
| Furong | test | Version: p6<br>Policies: 4<br>Catch-all Action: ALLOW | Disabled |

## Create a Workspace

To create policies for a scope, first create a workspace for that scope.

To create a workspace:

1. From the navigation menu on the left side of the window, choose **Defend** > **Segmentation**.
2. In the scope listing on the left side of the page, search for or scroll to the scope for which you want to create policies.
3. Hover over the scope until you see a blue plus sign, then click it.
4. Complete the form and click **Create** when done.

If a workspace exists for the scope, any additional workspaces is created as a secondary workspaces.

## Primary and Secondary Workspaces

For each scope, you can create one Primary workspace and multiple secondary workspaces.

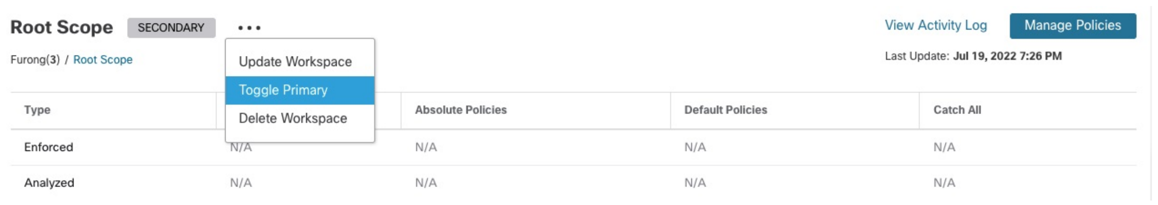
Only a primary workspace can be enforced. Other features that are available only for primary workspaces include the ability to manage policies in which consumer and provider reside in different scopes; live policy analysis; compliance reporting; and collaborative security policy definition.

Use secondary workspaces to experiment with policies when you want to preserve the existing policies in the primary workspace.

**To change a workspace to primary or secondary:**

You can switch a workspace from primary to secondary and conversely at any time by clicking the menu icon next to the workspace name at the top of the page and selecting **Toggle Primary**.

*Figure 2: Switching a Workspace Between Primary and Secondary*



| Type | | Absolute Policies | Default Policies | Catch All |
|---|---|---|---|---|
| Enforced | N/A | N/A | N/A | N/A |
| Analyzed | N/A | N/A | N/A | N/A |

## Rename a Workspace

To rename a workspace:

Click ••• beside the workspace type (Primary or Secondary) shown near the top of the page and choose **Update Workspace**.

## View Workloads in a Scope

In any workspace, click the **Matching Inventories** tab.

## Search Within a Workspace

To search within a workspace for workloads, clusters, or policies:

1. Select **Defend** > **Segmentation**.
2. From the list of scopes on the left, click the scope and workspace of interest.
3. Click **Manage Policies**.
4. Click the magnifying glass.
5. Enter search criteria.

### Search Criteria

Multiple criteria are treated as logical AND.

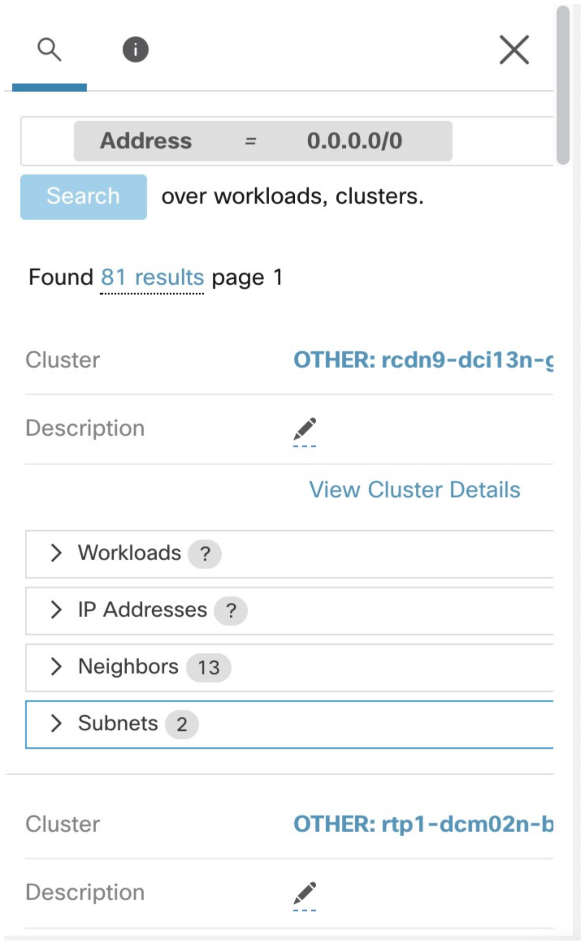
For IP addresses and numeric values:

- Indicate logical OR using a comma: ‘port: 80,443’.
- Range queries are also supported for number values: ‘port: 3000-3999’.

| Filters | Description |
|---|---|
| **Name** | Enter a cluster or workload name. Performs case-sensitive substring search. |
| **Description** | Searches cluster descriptions. |
| **Approved** | Matches approved clusters using the values ‘true’ or ‘false’. |

| Filters | Description |
|---|---|
| **Address** | Enter a subnet or IP address using CIDR notation (for example, 10.11.12.0/24). Matches workloads or clusters which overlap this subnet. |
| **Supernet** | Enter a subnet using CIDR notation (for example, 10.11.12.0/24) to match clusters whose workloads are fully contained in this subnet. |
| **Process** | Searches workload processes using case-sensitive substring search. |
| **Process UID** | Searches workload process usernames. |
| **Port** | Searches both workload provider port and policy port. |
| **Protocol** | Searches both workload provider protocol and policy protocol. |
| **Consumer Name** | Matches a policy's consumer cluster name. Performs a case-sensitive substring match. |
| **Provider Name** | Matches a policy's provider cluster name. Performs a case-sensitive substring match. |
| **Consumer Address** | Matches policies whose consumer address overlaps with the provided IP or subnet. |
| **Provider Address** | Matches policies whose provider address overlaps with the provided IP or subnet. |

### Search Example

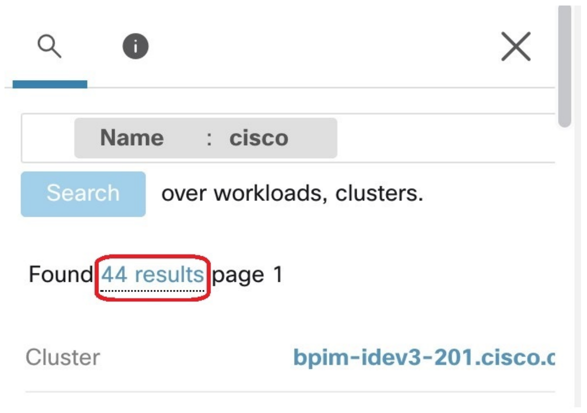


### Filtering Search Results by a Specific Type

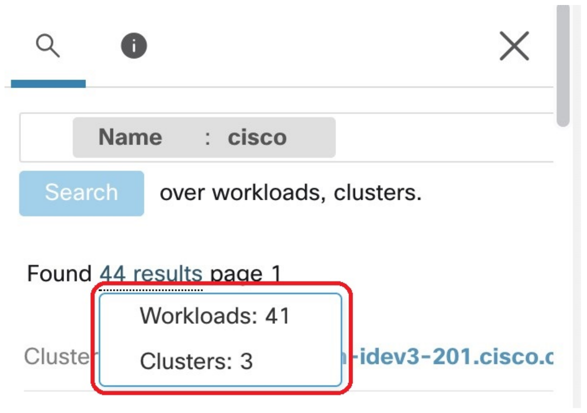
Search results may include multiple types of objects, for example workloads and clusters.

To filter search results by a specific type:

1. Click the result total:



2. Select the type from the dropdown:



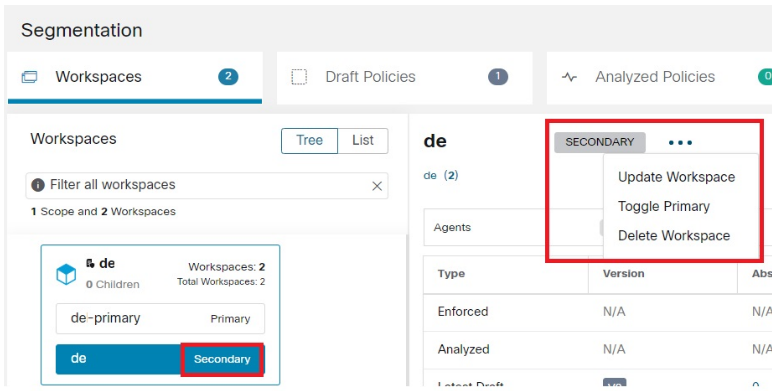
3. A type filter will be added and the search will be rerun.

## Deleting Workspaces

Only secondary (nonprimary) workspaces can be deleted. To switch a workspace to secondary, see Primary and Secondary Workspaces, on page 3.

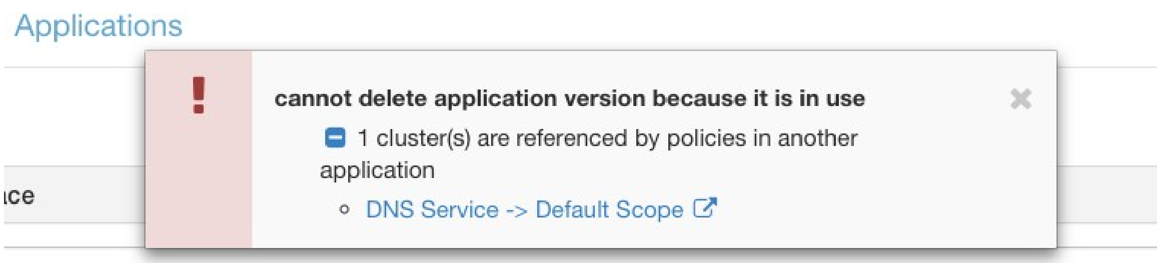
To delete a workspace:

1. Choose **Defend** > **Segmentation**.
2. In the list of scopes at the left side of the page, navigate to the scope containing the workspace to delete and click it.
3. Click the workspace to delete.
4. Click ••• beside **Secondary** and choose **Delete Workspace**.



If a workload or cluster in a workspace is referenced by a policy in another workspace as a result of a Provided Service, the dependent workspace cannot be deleted, and a list of the dependencies will be returned. This information can be used to fix the dependency.

***Figure 3: List of Items Preventing the Deletion of the Workspace***



cannot delete application version because it is in use

1 cluster(s) are referenced by policies in another application

- DNS Service -> Default Scope

In rare conditions there may be a cross dependency where Workspace A depends on a cluster in Workspace B and a Workspace B depends on a cluster in Workspace A. In this case, the individual policies or published policy versions (p*) must be deleted. The "delete restrictions" error provides links to all the policies so this can be accomplished.

To delete p* versions, see View, Compare, and Manage Analyzed Policy Versions, on page 120 or View, Compare, and Manage Enforced Policy Versions, on page 134.

# About Policies

## Policy Attributes

*Table 1: Policy Properties*

| Security Policy Property | Description |
|---|---|
| Scope for which the policy is defined | A policy generally affects only workloads that are members of the scope associated with the workspace in which the policy is defined.<br>(However, see also the topics under Address Policy Complexities, on page 82.)<br>For more information, see Policy Example, on page 11. |
| Consumer | The client of a service or the initiator of a connection.<br>Any scope, cluster, or inventory filter can be used as the consumer in a policy.<br>See important information in About Consumer and Provider in Policies, on page 10. |
| Provider | The server or the recipient of a connection.<br>Any scope, cluster, or inventory filter can be used as the provider in a policy.<br>See important information in About Consumer and Provider in Policies, on page 10. |
| Protocols and Ports | The server (listening) port and IP protocol of the service made available by the provider that should be permitted or blocked. |
| Action | ALLOW or DENY: Whether to allow or drop traffic from consumer to provider on the given service port/protocol. |
| Rank and Priority | For more information the rank and priority of policies in a workspace, see Policy Rank: Absolute, Default, and Catch-All, on page 9. |

## Policy Rank: Absolute, Default, and Catch-All

Policy rank determines whether a policy is overridden by a more specific policy lower in the priority list (or in a scope lower in the scope tree). The lowest priority policy in every scope is always the Catch-all rule.

| Policy Rank | Description |
|---|---|
| **Absolute** | Absolute policies take effect even if they contradict application-specific policies lower in the policy list (and thus, lower priority) or in scopes lower in the scope tree. Generally, use Absolute policies to enforce best practices, protect different zones, or quarantine-specific workloads. For example, use absolute policies to control traffic to DNS or NTP servers, or to meet regulatory requirements.<br>Absolute policies are listed above default policies in the policy priority list. |

| Policy Rank | Description |
|---|---|
| **Default** | Default policies can be overridden by policies lower in the policy list or in scopes lower in the scope tree. Generally, fine-grained policies are Default policies.<br><br>Default policies are listed below absolute policies in the policy priority list. |
| **Catch-All** | Each workspace has a catch-all policy that handles traffic in each direction that does not match any explicitly specified policies in the workspace. The catch-all action can be Allow or Deny.<br><br>In general, set the Catch-All policy as follows:<br>• **Allow** traffic in scopes higher in the scope tree, so that policies in scopes lower in the tree can evaluate the traffic.<br>• **Deny** traffic at the most specific leaf at the bottom of the scope tree.<br><br>This gives policies in all scopes in the tree the opportunity to match the traffic, while blocking traffic that does not match any policy in any scope.<br><br>The catch-all rule is applied to all interfaces on each workload in the workspace. |

## Policy Inheritance and the Scope Tree

Because your workloads are organized into a hierarchical scope tree, you can create general policies once in a scope at or near the top of the tree, and the policies can optionally apply to all workloads in all scopes below that scope in the tree.

You specify whether the general policies can be overridden by more specific policies lower in the tree.

See Policy Rank: Absolute, Default, and Catch-All, on page 9.

## About Consumer and Provider in Policies

The consumer and provider that is specified in a policy serve the following purposes:

- They specify the workloads or Secure Workload agents that receive policy or firewall rules.
- They specify the set of IP addresses to which the firewall rules that are installed on the workloads apply.

If a host has multiple interfaces (IP addresses), policies apply to all interfaces.

**Important** The above is the default behavior of how firewall rules are programmed on the workloads. If the IP addresses specified in the firewall rules differ from the IP addresses of the workloads that the policy is installed to, you need to separate the two purposes of consumers and providers in a policy. See Effective Consumer or Effective Provider, on page 101.

## Policy Example

The following example policy illustrates the importance of the scope in which a policy is defined, the impact of policy inheritance, and the use of inventory filters to create precise policies or policies that apply to workloads in multiple scopes.

Consider the following example involving three scopes:

- **`Apps`**

  and its child scopes

  - **`Apps:HR`** and
  - **`Apps:Commerce`**

In addition, the inventory filters PRODUCTION and NON-PRODUCTION specify production and nonproduction hosts, respectively. (You can define an inventory filter to apply to hosts within a scope or across scopes.)

Assume that the following policy is defined in the **`Apps`** scope:

```
DENY PRODUCTION -> NON-PRODUCTION on TCP port 8000 (Absolute)
```

Since this policy is an absolute policy that is defined in the primary workspace under the **`Apps`** scope, it affects all PRODUCTION/NONPRODUCTION hosts that are members of the **`Apps`** scope, including members of its descendant scopes (hosts that belong to the **`Apps:HR`** and **`Apps:Commerce`** scopes).

Now consider the case where the exact same policy is defined under the workspace that is associated with the **`Apps:HR`** scope. In this scenario, the policy can only affect PRODUCTION/NONPRODUCTION hosts that are members of the **`Apps:HR`** scope. More precisely, this policy results in inbound rules on NONPRODUCTION HR hosts (if any) denying connections on TCP port 8000 from **any** PRODUCTION host, and outbound rules on PRODUCTION HR hosts (if any) dropping connection requests to **any** NONPRODUCTION host.

# Create and Discover Policies

## Best Practices for Creating Policies

- For an overview of the entire segmentation process, see Get Started with Segmentation and Microsegmentation and subtopics.
- Manually create policies that apply broadly across your network.

  For example, block unwanted traffic to your workloads from outside your network, or quarantine vulnerable hosts.

  - Create manual policies in scopes at or near the top of your scope tree.

    For example, to block all traffic from outside your network to every host in your network, put the policy into the scope at the top of the tree.

  - If you want to be able to override the general policy for some workloads (for example, following the example above, you want to block general access from outside your network but you want some workloads to be accessible from outside the network), create the high-level policies as Default policies. Then create specific policies for the applicable workloads.

- Consider using templates to speed policy creation.
- See Manually Create Policies, on page 12, Policies for Specific Purposes, on page 14, and Policy Templates, on page 16.

- (Optional) Initially, automatically discover policies at a scope near the top of your tree, for all scopes in a branch of the tree, to create coarse policies that allow all existing traffic and limit future unwanted traffic. You can then build granular policies that protects your network from unnecessary or unwanted traffic.

  See Discover Policies for One Scope or for a Branch of the Scope Tree, on page 23 and Discover Policies Automatically, on page 20 for information.

- When you are ready to discover more granular policies, automatically discover policies for scopes at or near the bottom of your scope tree, especially in the scopes for individual applications.

  See Discover Policies for One Scope or for a Branch of the Scope Tree, on page 23 and Discover Policies Automatically, on page 20 for information.

- Be sure you have policies that address uncommon or infrequent activities and scenarios, such as failover, restoration from backup, once-yearly activities, and so on.
- After you have identified and allowed the traffic that your applications require, then look for any traffic that shouldn't be occurring and block such instances.

  Look first at traffic to and from your most sensitive applications.

  For example, if you see traffic from your customer-facing web app to your top-secret research and development app's database, you want to investigate.

- Work with your colleagues to ensure that the correct policies are applied to the correct workloads.
- Initially, when you enforce policies, consider setting the catch-all to Allow. Then, monitor traffic to see what matches the catch-all rule. When no necessary traffic is matching the catch-all rule, you can set the catch-all to Deny.

## Manually Create Policies

Typically, you can manually create policies that apply broadly across your network.

For example, you can manually create policies to:

- Allow access from all internal workloads to your NTP, DNS, Active Directory, or vulnerability scanning servers.
- Deny access from all hosts outside your organization to hosts inside your network unless explicitly permitted.
- Quarantine vulnerable workloads.

You can create absolute policies that cannot be overridden by more granularly applied policies, and default policies that can be overridden if a more specific policy exists.

You can create manual policies for scopes nearer the top of your tree.

### Before you begin

- (Optional) Consider using one of the templates available from **Defend** > **Policy Templates**.
- (Optional) If you know you have a set of workloads that receive the same policies, use an inventory filter to group them so you can easily apply policies to the set. The inventory filter can apply to only one scope, or to workloads in any scope. See Create an Inventory Filter.
- Make sure that the workloads in this scope are the workloads that you expect to be in this scope. See View Workloads in a Scope, on page 4.

### Procedure

**Step 1** Click **Defend** > **Segmentation**.

**Step 2** In the list on the left, search for or navigate to the scope in which you want to create the policy.

**Step 3** Click the scope and workspace in which you want to create the policy.

If you haven't yet created the workspace for this scope, see Create a Workspace, on page 3.

**Step 4** Click **Manage Policies.**

**Step 5** Click the **Policies** tab if it is not already selected.

**Step 6** Click **Add Policy**.

If you don't see an Add Policy button, see If the Add Policy Button Is Not Available, on page 13.

**Step 7** Enter information.

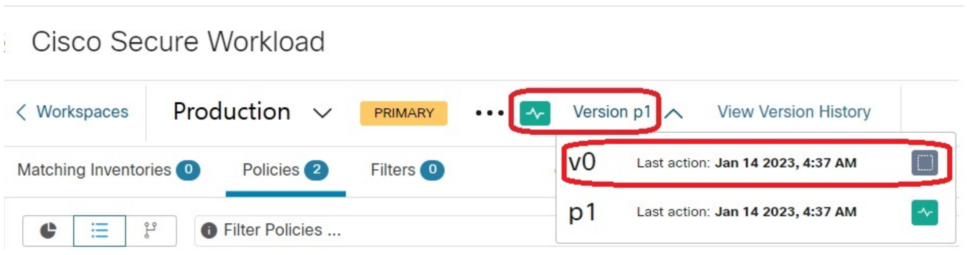
- For information about the **Absolute** checkbox, see Policy Rank: Absolute, Default, and Catch-All, on page 9. Generally, if you are creating policies that you don't expect exceptions for, enable this checkbox.
- **Priority** sets the order of the policy in the list. For more information about setting policy order, see Policy Priorities, on page 82 and subtopics. (You can set policy order later.)
- Consumer and provider can be entire scope, or, if you have created groups of workloads using inventory filters (or less optimally, clusters in the same workspace), you can choose those.

### What to do next

Make sure the **Catch-all** action is appropriate for the workspace. See Policy Rank: Absolute, Default, and Catch-All, on page 9.

## If the Add Policy Button Is Not Available

If you are trying to create a policy and the **Add Policy** button is not available, click the version showing at the top of the page and choose the latest "v" version, which is indicated with a gray square:



## Policies for Specific Purposes

### Create InfoSec Policies to Block Traffic from Outside Your Network

Use this procedure to quickly create a complete set of policies to control traffic entering your network from outside the network. The default set of policies allows only traffic using common ports and protocols and denies all other traffic. You can modify the default policy set to meet your needs.

#### Before you begin

Use this procedure if the following criteria are met:

- Your scope tree has a scope that is named **Internal** immediately below the root scope.

  This scope's members include, or will include, subnets encompassing all workloads on your internal network.

- The Internal scope does not yet have any policies defined in it.

**Note** Alternatively, you can use the **InfoSec** template available from **Defend** > **Policy Templates** to accomplish this with a few additional steps.

#### Procedure

**Step 1** Choose **Defend** > **Segmentation**,

**Step 2** Click the **Internal** scope and click the primary workspace.

If the Primary workspace does not yet exist, click the + button to create it.

**Step 3** Click **Manage Policies**.

**Step 4** Click **Add InfoSec Policies**.

**Step 5** Verify that all the policies in the list, including protocols and ports, are policies you want and delete and modify policies as desired.

**Step 6** Click **Create**.

### What to do next

(Optional) Add any additional policies to your Internal scope, such as policies that allow certain external traffic to specific workloads.

Place any specific policies below the more general policies in the list.

## Create Policies to Address Immediate Threats

If you must address an immediate threat, you can manually add a narrowly focused Absolute policy to a scope at or near the top of your scope tree, then enforce the primary workspace for that scope.

After you remediate the threat, you can remove that policy and reenforce the workspace.

## Create a Policy to Quarantine Vulnerable Workloads

You can:

- Create policies in advance, to automatically quarantine workloads with specific known vulnerabilities or a vulnerability severity threshold you specify.
- Create policies, to immediately quarantine workloads with detected known vulnerabilities that you deem sufficiently problematic.

This topic outlines the process for doing either.

### Before you begin

Look at the View Vulnerability Dashboard to see what policies are required.

### Procedure

**Step 1** Create an inventory filter that defines the vulnerabilities or the vulnerability severity threshold that you want to quarantine:

a) From the navigation bar at the left of the window, choose **Organize > Inventory Filters**.
b) Click **Create Inventory Filter**
c) Click the **(i)** button beside **Query** and enter `CVE` to see the relevant filter options.
d) Enter filter criteria that determine which workloads you wish to quarantine.
e) Be sure **Restrict query to ownership scope** is NOT selected.

**Step 2** Create a policy to quarantine affected workloads:

For general instructions, see Manually Create Policies, on page 12.

Recommendations:

- Create the policy in your **Internal** or other scope near the top of your scope tree.
- The policy should be an Absolute policy unless you want to allow exceptions. Be sure to create policies to address any exceptions.
- Create separate policies for consumer and provider.
- Set the priority of each policy to a low number so it will be hit before other policies in the list.
- Set the action to **Deny**.

**Step 3** Review, analyze, and enforce the policy or policies.

#### What to do next

Create an alert so you are notified when traffic hits this policy so you can remediate the problem and restore traffic to the vulnerable workload. See Configure Alerts.

## Policy Templates

Policy Templates are used to apply similar sets of policies to multiple workspaces.

Secure Workload includes some predefined templates, and you can create your own templates.

Policy templates require the scope owner capability on the root scope.

### System-Defined Policy Templates

To view available policy templates, navigate to **Defend** > **Policy Templates**.

To use a policy template, see Applying a Template, on page 19.

To modify a system-defined template, download the JSON file, edit it, then upload it.

### Create Custom Policy Templates

#### JSON Schema for Policy Templates

The policy template JSON schema is designed to mimic the schema of Export a Workspace. You can create a set of policies in a workspace, export it as JSON, modify the JSON, then import as a policy template.

| Attribute | Type | Description |
|---|---|---|
| name | string | (optional) Used as the name of the template during import. |
| description | string | (optional) Template description that is displayed during the apply process. |
| parameters | parameters object | Template parameters, see below. |
| absolute_policies | array of policy objects | (optional) Array of absolute policies. |
| default_policies | array of policy objects | (required) Array of default policies, can be empty. |

##### Parameters Object

The parameters object is optional but can be used to dynamically define filters as parameters to the template. The parameters are referenced using the `consumer_filter_ref or provider_filter_ref` policy attributes.

The keys of the parameters object are the reference names. The values are an object with a required `"type": "Filter"` and an optional description. An example Parameters object is shown below:

```
{
  "parameters": {
    "HTTP Consumer": {
      "type": "Filter",
      "description": "Consumer of the HTTP and HTTPS service"
    },
    "HTTP Provider": {
      "type": "Filter",
      "description": "Provider of the HTTP and HTTPS service"
    }
  }
}
```

The parameters can be referenced in the policy objects, for example: `"consumer_filter_ref": "HTTP Consumer"` or `"provider_filter_ref": "HTTP Provider"`.

### Special Parameter References

A few special references automatically map to a filter and do not need to be defined as parameters.

| Ref | Description |
|---|---|
| _workspaceScope | Resolves to the scope of the workspace to which the template is being applied. |
| _rootScope | Resolves to the root/top level scope. |

### Policy Object

To maintain compatibility with the workspace export JSON, the policy object contains multiple keys for consumers and providers. They are resolved as follows:

```
if *_filter_ref is defined
  use the filter resolved by that parameter
else if *_filter_id is defined
  use the filter referenced by that id
else if *_filter_name is defined
  use the filter that has that name
else
  use the workspace scope.
```

If a filter cannot be resolved as defined above, an error is returned both at the time of application and at the time of upload.

| Attribute | Type | Description |
|---|---|---|
| action | string | (optional) Action of the policy, ALLOW, or DENY (default ALLOW). |
| priority | integer | (optional) The priority of the policy (default 100). |
| consumer_filter_ref | string | Reference to a parameter. |

| Attribute | Type | Description |
|---|---|---|
| consumer_filter_name | string | Reference to a filter by name. |
| consumer_filter_id | string | ID of a defined Scope or Inventory Filter. |
| provider_filter_ref | string | Reference to a parameter. |
| provider_filter_name | string | Reference to a filter by name. |
| provider_filter_id | string | ID of a defined Scope or Inventory Filter. |
| l4_params | array of l4params | List of allowed ports and protocols. |
| Attribute | Type | Description |
| proto | integer | Protocol integer value (NULL means all protocols). |
| port | integer | Inclusive range of ports, for example, [80, 80] or [5000, 6000] (NULL means all ports). |

### L4param object

| Attribute | Type | Description |
|---|---|---|
| proto | integer | Protocol integer value (NULL means all protocols). |
| port | integer | Inclusive range of ports, for example, [80, 80] or [5000, 6000] (NULL means all ports). |

## Template Sample

```
{
  "name": "Allow HTTP/HTTPS and SSH",
  "parameters": {
    "HTTP Consumer": {
      "type": "Filter",
      "description": "Consumer of the HTTP and HTTPS service"
    },
    "HTTP Provider": {
      "type": "Filter",
      "description": "Provider of the HTTP and HTTPS service"
    }
  },
  "default_policies": [
    {
      "action": "ALLOW",
      "priority": 100,
      "consumer_filter_ref": "__rootScope",
      "provider_filter_ref": "__workspaceScope",
          "l4_params": [
            { "proto": 6, "port": [22, 22] },
          ]
        },
        {
          "action": "ALLOW",
          "priority": 100,
          "consumer_filter_ref": "HTTP Consumer",
          "provider_filter_ref": "HTTP Provider",
          "l4_params": [
            { "proto": 6, "port": [80, 80] },
            { "proto": 6, "port": [443, 443] }
          ]
        }
      ]
    }
```

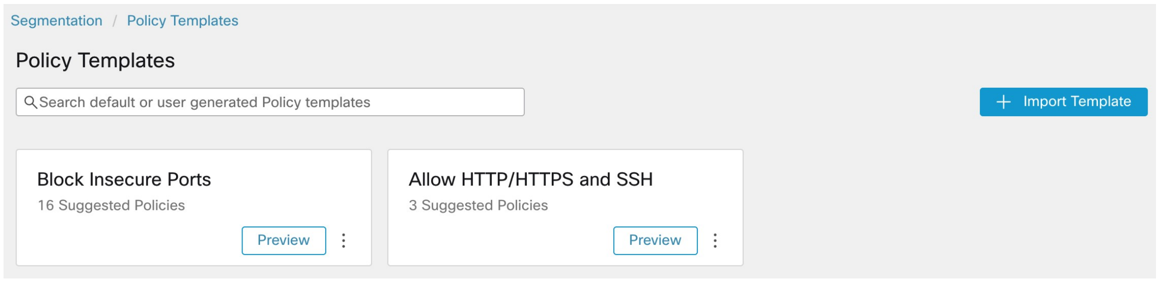
### Template Import

Policy Templates shown on the Policy Templates page that can be accessed from the main Segmentation page. This is where templates can be imported/uploaded using the “Import Template” button.

Templates are validated for correctness when they are uploaded. A helpful list of errors is provided to debug any issues.

Once a template is uploaded, it can be applied, downloaded, or have its name and description updated.

***Figure 4: Display of Available Templates***



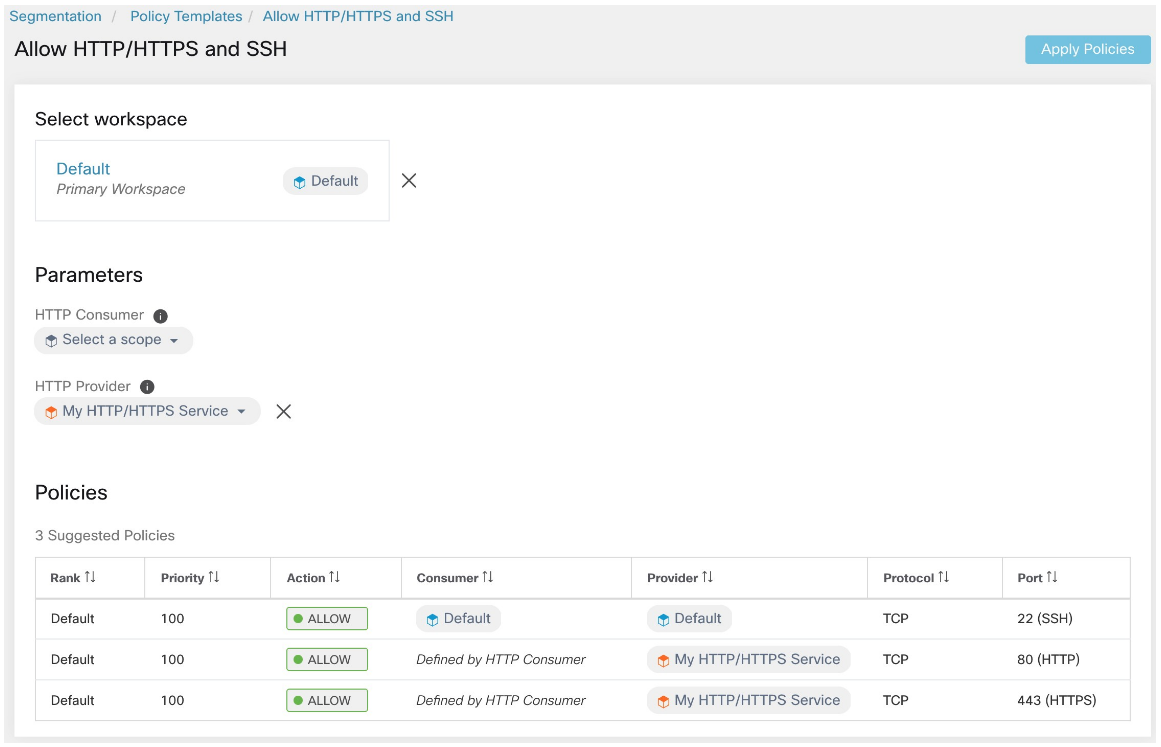
## Applying a Template

Applying a template to a workspace takes several steps:

1. Select a template to preview.
2. Select a workspace to apply the template to.
3. Fill in parameters, if necessary.
4. Review the policies.
5. Apply the policies.

The policies will be added to the latest version of the selected workspace. Policies created via a template can be filtered using the `From Template? = true` filter.

*Figure 5: Applying a Policy Template*



| Rank | Priority | Action | Consumer | Provider | Protocol | Port |
|---|---|---|---|---|---|---|
| Default | 100 | ALLOW | Default | Default | TCP | 22 (SSH) |
| Default | 100 | ALLOW | *Defined by HTTP Consumer* | My HTTP/HTTPS Service | TCP | 80 (HTTP) |
| Default | 100 | ALLOW | *Defined by HTTP Consumer* | My HTTP/HTTPS Service | TCP | 443 (HTTPS) |

## Discover Policies Automatically

Automatic policy discovery, sometimes referred to as policy discovery, and formerly known as Application Dependency Mapping (ADM), uses existing traffic flows and other data to do the following:

- Suggest a set of “allow” policies based on existing successful network activity.

  The goal of these policies is to identify the traffic that your organization needs, and block all other traffic.

- Group workloads into clusters based on similarity of their computing behavior

  For example, if an application includes multiple web servers, those might be clustered together.

  For more information, see Clusters, on page 72.

You can discover policies for each scope. Typically, you discover policies for scopes at or near the bottom of your scope tree, for example at the application level. However, for initial deployment, you might want to discover policies at a higher-level scope, so you have general, temporary policies in place while you create more refined policies.

You can discover policies as often as desired, to refine the suggested policies based on additional information.

You can manually modify suggested policies and clusters, and/or approve any of them so they are carried forward and not modified by subsequent discovery runs.

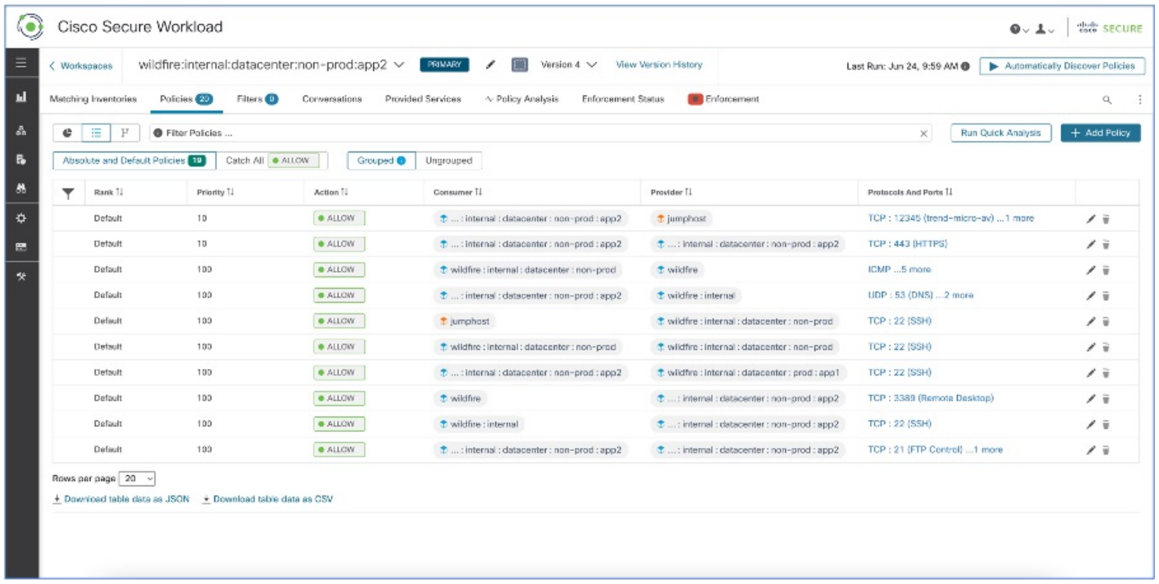
You can include both manually created policies and discovered policies in a workspace.

After you discover policies, you will review and analyze them before enforcing them.

To get started discovering policies, see How to Automatically Discover Policies, on page 21.

For more information, see Policy Discovery Details, on page 21.

***Figure 6: Example: Automatically Discovered Policies***



| Rank | Priority | Action | Consumer | Provider | Protocols And Ports |
|---|---|---|---|---|---|
| Default | 10 | ALLOW | ... : internal : datacenter : non-prod : app2 | jumphost | TCP : 12345 (trend-micro-av) ...1 more |
| Default | 10 | ALLOW | ... : internal : datacenter : non-prod : app2 | ... : internal : datacenter : non-prod : app2 | TCP : 443 (HTTPS) |
| Default | 100 | ALLOW | wildfire : internal : datacenter : non-prod | wildfire | ICMP ...5 more |
| Default | 100 | ALLOW | ... : internal : datacenter : non-prod : app2 | wildfire : internal | UDP : 53 (DNS) ...2 more |
| Default | 100 | ALLOW | jumphost | wildfire : internal : datacenter : non-prod | TCP : 22 (SSH) |
| Default | 100 | ALLOW | wildfire : internal : datacenter : non-prod | wildfire : internal : datacenter : non-prod | TCP : 22 (SSH) |
| Default | 100 | ALLOW | ... : internal : datacenter : non-prod : app2 | wildfire : internal : datacenter : prod : app1 | TCP : 22 (SSH) |
| Default | 100 | ALLOW | wildfire | ... : internal : datacenter : non-prod : app2 | TCP : 3389 (Remote Desktop) |
| Default | 100 | ALLOW | wildfire : internal | ... : internal : datacenter : non-prod : app2 | TCP : 22 (SSH) |
| Default | 100 | ALLOW | ... : internal : datacenter : non-prod : app2 | ... : internal : datacenter : non-prod : app2 | TCP : 21 (FTP Control) ...1 more |

## Policy Discovery Details

Additional information about automatic policy discovery:

- Automatic policy discovery considers conversations in which at least one end is a member workload of the scope within the time range selected. Membership in the scope is based only on the most current scope definition; former membership is not considered.
- By default, policy discovery produces policies and clusters by analyzing communication flows ("conversations"), but optionally can consider other information such as processes running on workloads or load balancer configurations.

  See Include Data From Load Balancers and Routers When Discovering Policies, on page 36.
- You can discover policies in any workspace within the scope. Discovery results in each workspace are independent of the results in other workspaces in the scope.
- For detailed discussions of complex concepts that are related to automatic policy discovery, see Advanced Features of Automatic Policy Discovery, on page 28 and Address Policy Complexities, on page 82.

## How to Automatically Discover Policies

Perform the following steps. At any point, you can decide to discover policies again.

Work with colleagues as needed to complete these steps.

| Step | Do This | More Information |
|---|---|---|
| 1 | Upload and label your workload inventory, and gather flow data that inform policy discovery. | See Get Started with Segmentation and Microsegmentation and subtopics. |
| 2 | Choose whether you discover policies for:<br>• Workloads in a single scope<br>• Workloads in all of the scopes in a branch of the scope tree | See Discover Policies for One Scope or for a Branch of the Scope Tree, on page 23.<br>(You can always discover policies again at any time.) |
| 3 | Choose the scope in which you discover policies. | This depends in part on whether you discover policies for a single scope or for a branch of the scope tree. |
| 4 | Choose the workspace in which you discover policies. | Generally, you will discover policies in the scope's primary workspace, because you can only analyze policies in a primary workspace. (However, you can always change a workspace to primary later.)<br>If your chosen scope does not yet have a workspace, see Create a Workspace, on page 3. |
| 5 | Confirm the inventory that you expect to include in policy discovery. | Verify the Workloads That Policy Discovery Will Apply To, on page 25 |
| 6 | (Optional) Create inventory filters to group workloads that you want to treat as a group. | See Create an Inventory Filter. |
| 7 | Set the **Catch-all** action for the workspace. | See Policy Rank: Absolute, Default, and Catch-All, on page 9 |
| 8 | Discover Policies | Discover Policies Automatically, on page 20<br>Be sure to complete the prerequisites in the "Before You Begin" section. |
| 9 | View and manage the clusters (groups of workloads) that policy discovery creates.<br>(This step applies only when you discover policies for a single scope; clusters are not generated when you discover policies for a branch of the tree.) | See Clusters, on page 72 and subtopics.<br>Evaluate the suggested clusters, optionally edit cluster membership as needed, and approve (or better, convert to inventory filters) any clusters that you want to make permanent. |
| 10 | Consider complexities such as policy inheritance and cross-scope policies. | See Address Policy Complexities, on page 82. |
| 11 | Review generated policies. | See Review Automatically Discovered Policies, on page 104 and subtopics |

| Step | Do This | More Information |
|---|---|---|
| 12 | Approve policies that you want to keep. | Approve Policies, on page 47 |
| 13 | Discover policies again as desired, to reflect additional flow data, changes in scope membership, or other changes. | Important: Before You Re-run Automatic Policy Discovery, on page 50<br><br>You can rerun policy discovery at any time.<br><br>Review and approve policies and clusters each time you discover policies. |
| 14 | Run live analysis to see how your policies affect your actual traffic. | When you believe that your policies do what you expect them to do, start Live Policy Analysis, on page 111.<br><br>If you change policies or rediscover policies, restart policy analysis (to analyze the current policies). |
| 15 | If you re-discover policies or make other changes, restart live analysis. | See After Changing Policies, Analyze Latest Policies, on page 119. |
| 16 | When you are confident that the policies will not block essential traffic, enforce the workspace. | See Enforce Policies and subtopics. |
| 17 | Verify that enforcement is working as expected. | See Verify That Enforcement Is Working as Expected, on page 130 |
| 18 | (Optional) Configure default policy discovery settings that optionally apply when discovering policies in any workspace. | See Default Policy Discovery Config, on page 45 and linked topics.<br><br>Because these are advanced settings, we recommend that you change them only if you have a specific need to change them. You can change them at any time during your process as you realize a need. |

## Discover Policies for One Scope or for a Branch of the Scope Tree

If either option is not possible when you discover policies for a particular scope, the selection is made for you and you will not see a choice of options.

***Table 2: Discovering Policies For***

| A Branch of the Scope Tree | A Single Scope |
|---|---|
| Use this method as a starting point, when you are beginning to use Secure Workload, to quickly generate a temporary set of coarse policies that allow existing traffic while helping to protect your network from future threats. | Use this method to fine-tune segmentation policies and ensure that all allowed flows are expected; the smaller number of policies makes it easier to see any existing anomalies that require investigation. |

| A Branch of the Scope Tree | A Single Scope |
|---|---|
| Typically, you use this method for scopes nearer the top of your scope tree.<br>The top of the branch can be any scope in the tree. | Typically, you use this method for scopes at or near the bottom of your scope tree, for example for scopes dedicated to a single application. |
| Discover policies only in one scope – the scope at the top of the branch that you choose. | Discover policies for each scope in the branch as needed. |
| All workloads in the chosen scope and all child and descendant scopes are included in discovery. | Workloads that are also members of any child scope are not included in discovery for this scope.<br>Policies are generated only for workloads that appear in the **Uncategorized Inventory** tab for that scope on the **Organize** > **Scopes and Inventory** page.<br>You can discover policies for workloads in child and descendant scopes separately. |
| All policies for workloads in all scopes in the branch reside in the scope at the top of the branch. | Assuming you also create policies for workloads in child and descendant scopes, policies reside in multiple scopes. |
| This method typically generates a large number of policies. | This method generates fewer policies in any individual scope. |
| Discovered policies apply to entire scopes; this option cannot create policies specific to subsets of workloads within scopes. | This option can generate policies that apply to subsets of workloads within the consumer and/or provider scope. (Workloads can be grouped by generated clusters and/or by configured inventory filters, and policies applied just to these subsets.) |
| All policies are created in a single scope at the top of the branch, so extra steps are not required when a policy's consumer and provider are in different scopes. | Allowing traffic between consumers and providers in different scopes requires extra steps.<br>See When Consumer and Provider Are in Different Scopes: Policy Options, on page 88. |
| Discovery can run even if a scope does not have any member workloads with installed agents, as long as descendant scopes have agents or external orchestrators or connectors that gather flow data. | The scope must have member workloads with installed agents or external orchestrators or connectors that gather flow data. |
| This option is available to root scope owners and site admins only. | You must have privileges to create policies for this scope. |
| The maximum number of agents and conversations is different for each option. See Limits Related to Policies. | |
| This option was formerly the Deep Policy Generation advanced configuration option for automatic policy discovery. The behavior has not changed. | This was formerly the default behavior for automatic policy discovery. |
| For additional details, see Discovering Policies for a Branch of the Scope Tree: Additional Information, on page 25. | -- |

### Discovering Policies for a Branch of the Scope Tree: Additional Information

- All workloads that are conversation endpoints, whether or not they are members of the scope in which policy discovery is run, are assigned the highest matching scope label according to the top-down order given in the external dependencies list.
- For advanced configuration options available when you generate policies for a branch of the scope tree, see:
  - Enable redundant policy removal, on page 42
  - Policy Compression, on page 38 and related subtopic, Hierarchical policy compression, on page 38
- Currently, the count of workloads shown for automatic policy discovery includes only those that are not also members of a subscope.

## Verify the Workloads That Policy Discovery Will Apply To

Before you automatically discover policies, verify that the workloads on which policy discovery will be based are in fact the set of workloads you expect. Discovered policies will be generated from flow data captured by agents on these workloads.

### Before you begin

Decide which of the options in Discover Policies for One Scope or for a Branch of the Scope Tree, on page 23 you can use.

### Procedure

**Step 1** From the navigation menu on the left, choose **Defend** > **Segmentation**.

**Step 2** Click the scope for which you want to discover policies.

**Step 3** Click the workspace in which you want to discover policies.

**Step 4** Click **Manage Policies**.

**Step 5** Click **Matching Inventories**.

**Step 6** If you discover policies for a single scope:

a) Click **Uncategorized Inventory**

This page shows workloads that are not also members of child scopes. (In standard automatic policy discovery, policies and clusters are generated in this scope only for workloads that are not also members of child scopes.)

b) Click **IP addresses**.

IP addresses on this page do not have Secure Workload agents installed.

Because they do not have agents that are installed, these IP addresses are not considered during automatic policy discovery for this scope UNLESS:

- Policy is being managed via a cloud connector
- The IP addresses are container-based inventory, in which case individual workloads appear on the **Pods** tab, or

- The workloads happen to communicate with a workload in this scope that is considered during policy discovery.

Before discovering policies, consider installing agents on workloads that need them and allowing some time to pass for flow data to accumulate.

c) Click **Workloads**.

Policies and clusters are generated only for workloads on this page and for IP addresses on the IP addresses tab that meet the criteria specified above for consideration.

d) If you have Kubernetes or OpenShift inventory, you will see a **Services** tab and a **Pods** tab.

If you have installed agents on your Kubernetes/OpenShift workloads, check the inventory on those tabs as well.

e) If you have load-balancer inventory, that inventory appears on the **Services** tab.

**Step 7** If you discover policies for a branch of the tree:

a) Click **All Inventory**

This process generates policies (but not clusters) for all workloads in this scope, whether they are also members of child scopes.

b) Click **IP addresses**.

IP addresses on this page do not have Secure Workload agents installed.

Because they do not have agents installed, these IP addresses will not be considered during automatic policy discovery for this scope unless:

- Policy is managed via a cloud connector
- The IP addresses are container-based inventory, in which case individual workloads appear on the **Pods** tab, or
- The workloads happen to communicate with a workload in this scope that is considered during policy discovery.

Before discovering policies, consider installing agents on these workloads and allowing some time to pass for flow data to accumulate.

c) Click **Workloads**.

Policies are generated only for workloads on this page and for IP addresses on the IP addresses tab that meet the criteria specified above for consideration.

d) If you have Kubernetes or OpenShift inventory, you will see a **Services** tab and a **Pods** tab.

If you have installed agents on your Kubernetes/OpenShift workloads, check the inventory on those tabs as well.

e) If you have load-balancer inventory, that inventory appears on the **Services** tab.

**Step 8** Verify that the workloads are the set you expect.

## Automatically Discover Policies

Use this procedure to generate suggested Allow policies based on existing traffic on your network.

You can rediscover policies at any time.

### Before you begin

- Gather flow data before you can effectively automatically discover policies.

  Typically, this means you have installed agents on the workloads in the scope, or have configured and gathered data using a cloud connector or external orchestrator.

  Flow summary data that is used by automatic policy discovery is computed every 6 hours. Thus, upon initial deployment of Secure Workload, automatic policy discovery is not possible until such data is available.

  More flow data generally produces more accurate results.

  Before you enforce a policy, you should gather enough data to include traffic that occurs only periodically (monthly, quarterly, annually, and so on.) For example, if an application generates a quarterly report that gathers information from sources that the application does not access at other times, ensure that the flow data includes at least one instance of that report-generation process.

- Complete the steps up to this point in How to Automatically Discover Policies, on page 21.

- Meet the policy discovery-related Limits Related to Policies.

  If necessary, break larger scopes into smaller child scopes.

- Commit any scope changes before discovering policies, or any configured exclusion filters may not match (exclude) flows as expected. See Commit Changes.

**Important** If you are rerunning policy discovery, see the important considerations first: Important: Before You Re-run Automatic Policy Discovery, on page 50.

### Procedure

**Step 1** Choose **Defend** > **Segmentation**.

**Step 2** In the scope tree or list of scopes in the pane on the left, scroll to or search for the scope for which you want to generate policies.

**Step 3** Click a workspace (primary or secondary) in the scope.

**Step 4** Click **Manage Policies**.

**Step 5** Click **Automatically Discover Policies**.

**Step 6** If you see an option to discover policies for a branch or an entire scope, choose an option.

If you don't see an option, only one option is possible for the scope for which you are discovering policies.

For more information, see Discover Policies for One Scope or for a Branch of the Scope Tree, on page 23.

**Step 7** Choose the time range for the flow data that you want to include.

Experiment to find the right time range; you can generate policies as often as needed to get optimal results.

A shorter time range generates results faster, and may generate fewer results.

In general, a longer time range produces more accurate policies. However, if the scope definition has changed, do not include dates before the change is made.

Your time range should include traffic that occurs only periodically (monthly, quarterly, annually, and so on.) if applicable. For example, if an application generates a quarterly report that gathers information from sources that it does not access at other times, be sure that the time range includes at least one instance of that report-generation process.

To configure a time range beyond the last 30 days, select the **custom** range, and fill the required start and end times under the drop-down time selection widget.

**Step 8** (Optional) Specify advanced settings.

Generally, we suggest that you don't change advanced settings for initial discovery runs, then make changes only as needed to address specific issues.

For details, see Advanced Configurations for Automatic Policy Discovery, on page 35.

**Step 9** Click **Discover Policies**. Generated policies appear on this page.

---

### What to do next

- View Stop Automatic Policy Discovery in Progress, on page 28.
- Return to How to Automatically Discover Policies, on page 21 and continue with the next step in the table.
- You can rediscover policies at any time. For actions you should take first, see Important: Before You Re-run Automatic Policy Discovery, on page 50.

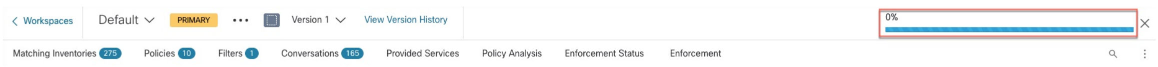
## Stop Automatic Policy Discovery in Progress

Progress of automatic policy discovery is always visible in the header. Navigating to other workspaces does not affect the progress.

To stop the run while it is in progress, click the **abort** button.

Once the run is complete, a message is displayed. If successful, **Click to see results** navigates to a different view showing the changes before and after the run. If automatic policy discovery fails, it is indicated with a different message and a reason.

*Figure 7: Automatic Policy Discovery Progress*



## Advanced Features of Automatic Policy Discovery

You must specify a time range for the discovery run. If necessary, you can configure advanced options.

You can configure advanced options for each workspace, or set defaults for all workspaces (the entire root scope), then modify the settings for individual workspaces as needed.

*Table 3: Configure Advanced Options for Automatic Policy Discovery*

| For a Workspace | For All Workspaces |
|---|---|
| Option descriptions for individual workspaces (in column 1) apply also for all workspaces (column 2) | |
| External Dependencies, on page 31 | Default Policy Discovery Config, on page 45 |
| Advanced Configurations for Automatic Policy Discovery, on page 35 | Default Policy Discovery Config, on page 45 |
| Exclusion Filters, on page 29 | Default Exclusion Filters, on page 45 |

## Exclusion Filters

If certain flows are generating unwanted policies, you can exclude those flows from automatic policy discovery using exclusion filters.

For example, to disallow certain protocols like ICMP in the final allow list model, you can create an exclusion filter with a protocol field set to ICMP.

**Note**

- Conversations that match exclusion filters are excluded for the purposes of policy generation and clustering, but remain in the Conversations View with red ‘excluded’ icon (see the Table View in Conversations). Likewise, workloads of the workspace incident on such conversations remain viewable as well.
- An exclusion filter that uses a cluster or a filter definition from a workspace is effective only in primary workspaces (otherwise, its cluster definitions are not visible to the label system, and any matching conversations are not excluded).
- Exclusion filters are versioned; to track modifications, see Activity Logs and Version History.
- For limits on the number of exclusion filters, see Limits Related to Policies.

You can create one or both of the following, then enable either or both when discovering policies:

- A list of exclusion filters for each workspace.
- A list of default exclusion filters that is available to all workspaces in your tenant.

You can also enable or disable either or both lists for the Default Policy Discovery Config.

For instructions, see Configure, Edit, or Delete Exclusion Filters, on page 29 and Enable or Disable Exclusion Filters, on page 31.

### *Configure, Edit, or Delete Exclusion Filters*

You can use this procedure to create a list of exclusion filters for a single workspace, or a list of default exclusion filters that are available to all workspaces.

#### Procedure

**Step 1** Do one of the following:

| To | Do This |
|---|---|
| Configure exclusion filters for a specific workspace | Navigate to the workspace, then do one of the following:<br>• Click **Manage Policies**, then click ⋮ near the top right of the page and select **Exclusion Filters**.<br>• From the automatic policy discovery configuration page, click the **Exclusion filters** link in the Advanced Configurations section.<br>• Delete a discovered policy; you will see an option to create an exclusion filter. |
| Configure default exclusion filters that are available to any workspace | **a.** Choose **Defend** > **Segmentation**,<br>**b.** Click the caret at the right side of the page to expand the Tools menu, then choose **Default Policy Discovery Config**.<br>**c.** Scroll to the bottom of the page.<br>**d.** Click **Default Exclusion Filters**. |

**Step 2** To create an exclusion filter, click **Add Exclusion Filter**.

**Step 3** Specify parameters for the flows to exclude from consideration during policy discovery:

You do not need to enter values for all of the fields. Any empty field is treated as a wildcard for matching flows.

Any conversation that matches all the fields of any exclusion filter is ignored for the purposes of policy creation and clustering.

| Option | Description |
|---|---|
| **Consumer** | Matches conversations where the consumer address is a member of the selected scope, inventory filter, or (for workspace-specific exclusion filters only, cluster). You can specify any arbitrary address space by creating a new custom filter. |
| **Provider** | Matches conversations where the provider address is a member of the selected scope, inventory filter, or (for workspace-specific exclusion filters only, cluster). You can specify any arbitrary address space by creating a new custom filter. |
| **Protocol** | Matches conversations with specified protocol. |
| **Port** | Matches conversations with provider (server) port matching the specified port, or port range. Enter port ranges using a dash separator, for example, “100-200” |

**Step 4** To edit or delete an exclusion filter, hover over the applicable row to see the **Edit** and **Delete** buttons.

**Step 5** If you are configuring default exclusion filters:

When the configured filters are ready to use, return to the **Default Policy Discovery Config** page, and click **Save** to make the changes available to individual workspaces.

#### What to do next

**Important** Exclusion filters are enabled by default in the workspace in which they are configured.

Default exclusion filters are enabled by default in all workspaces.

Both types of exclusion filters are enabled by default in the Default Policy Discovery Config.

Before discovering policies:

- Enable or disable exclusion filters and default exclusion filters.
    - In each workspace
    - On the Default Policy Discovery Config page

  For instructions, see Enable or Disable Exclusion Filters, on page 31.
- Commit any scope changes, or the filters may not match (and therefore exclude) the expected flows. See Commit Changes.

### *Enable or Disable Exclusion Filters*

You can create exclusion filters in each workspace and/or create a set of default exclusion filters that you can apply to all workspaces.

By default, both types of exclusion filters are enabled.

To make changes

- To enable or disable exclusion filters for a single workspace:

  In the workspace, click **Manage Policies**, then click **Automatically Discover Policies**, then click **Advanced Configurations**. You can enable exclusion filters and/or default exclusion filters for this workspace.
- To enable or disable exclusion filters in the Default Policy Discovery Config:

  Choose **Defend** > **Segmentation**, then click the caret at the right side of the page to expand the Tools menu. Then choose **Default Policy Discovery Config**. Scroll to or click **Advanced Configurations**. You can enable exclusion filters and/or default exclusion filters.

## External Dependencies

External dependencies are relevant only when you use the process that is described in (Advanced) Create Cross-Scope Policies, on page 89.

External Dependencies settings apply to automatically discovered policies involving communications to and from workloads that are members of a scope other than the scope in which policies are discovered. (That is, communications involving "external workloads.")

A workload that is not a member of the scope in which the policy exists is an *external workload*. Such workloads are the other end of a conversation with a *target workload* (which *is* a member of the scope in which the policy exists).

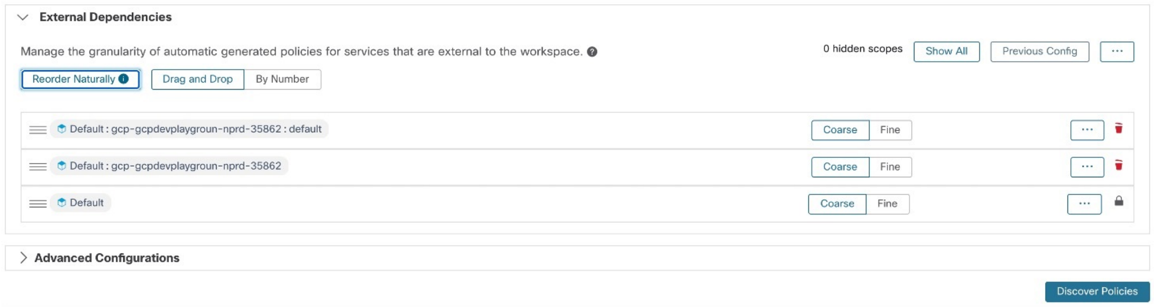
The External Dependencies list is an ordered list of all scopes in your hierarchy. Each scope in the list is set to one of the following:

- Generate specific or refined policies (more secure), OR
- Generate coarse policies in higher scopes, which may generalize better (that is, be more likely to allow legitimate flows that were not seen in the time range that is specified when discovering policies).

During policy discovery, the first scope (or cluster, or inventory filter – see below) that matches the workload will be used to generate the “allow” policy, where the matching order (and consequent granularity level) is determined by the top-down ranking that is displayed in the External Dependencies section.

A default scope order is configured for you, with all scopes set to "Coarse" by default.

***Figure 8: Default External Dependencies***



Manage the granularity of automatic generated policies for services that are external to the workspace.

| | | |
|---|---|---|
| Default : gcp-gcpdevplaygroun-nprd-35862 : default | Coarse | Fine |
| Default : gcp-gcpdevplaygroun-nprd-35862 | Coarse | Fine |
| Default | Coarse | Fine |

| To | Do This |
|---|---|
| View or fine-tune external dependencies for a workspace: | Navigate to the workspace and click **Automatically Discover Policies**, then click **External Dependencies**.<br><br>To reorder the scopes and choose granular options for each, see Fine-Tune External Dependencies for a Workspace, on page 33. |
| Configure default external dependencies for an entire root scope: | See Default Policy Discovery Config, on page 45. |

### External Dependencies: Granular Policies Involving Subsets of Scopes

You can optionally discover policies at a more granular level than scope-to-scope, to control traffic to a specified subset of the workloads in a scope.

For example, you may want to create policies specific to a certain type of host within an application, such as API servers; you can group those workloads into a subset within the application scope.

To generate policies specific to a subset of workloads within a scope, see Fine-Tune External Dependencies for a Workspace, on page 33.

## *Tips for Exploring External Dependencies*

Use the following tips to explore the behavior of automatic policy discovery for policies involving workspaces that are not members of the scope that is associated with the workspace in which the policies reside.

**Tip**

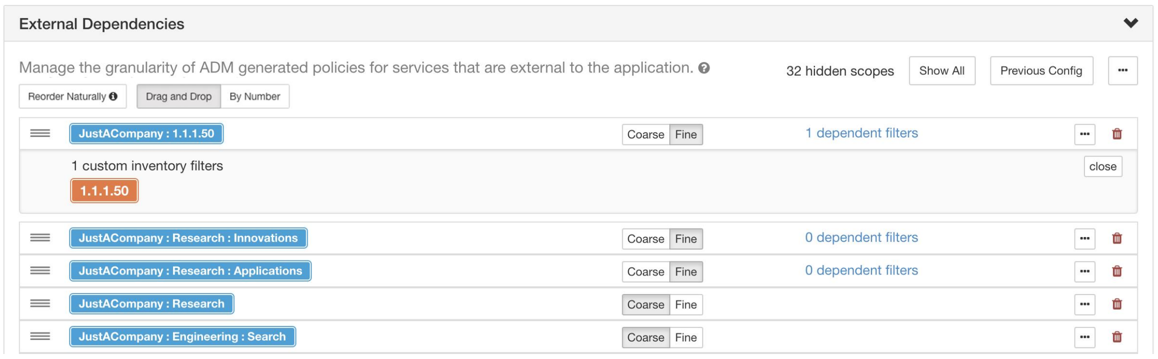
- You can remove and rearrange the list to generate policies at a desired granularity. For example, removing all Company: RTP subscopes will help generate wide policies to the whole Company:RTP scope, but not its individual components, while maintaining the higher granularity for Company: SJC scope. Furthermore, you can click on the **Fine** button next to any scope and see if there are finer grain candidates defined under that scope.
- By default, the root scope is configured as the lowest entry in the External Dependencies list, so that automatic policy discovery always generates policies to more specific scopes whenever possible. Initially, to view relatively few coarse-grained policies, you can temporarily place the root scope on the top of the external dependencies. This way, after automatic policy discovery, you will see all external policies of the workspace connecting to only one scope, the root scope (as every external workload maps to the root scope). The resulting number of generated policies are smaller and easier to examine and comprehend.
- You can also temporarily bundle all workloads that are members of the scope associated with the workspace ("internal workloads") into one cluster, approve the cluster, and then discover policies. Again, this results in a reduced set of policies, as no clustering (subpartitioning of the workspace/scope) takes place, so you can view policies that are either internal (connect to internal workloads), or external (connect an internal to an external workload). Later, you can view progressively more refined policies by unbundling internal workloads and/or placing one or a few external scopes of interest above the root.
- **Important** Always carefully examine policies involving the root scope, since these policies allow all traffic to and from the entire network. This is especially important when the root scope is placed low in the External Dependencies list and it is not your intention to generate coarse policies. Such policies may not have resulted from network-wide traffic in or out of the workspace scope. Rather they can be triggered by a few external endpoints which failed to receive finer scopes or inventory filter assignments beyond simply the root scope.

  While auditing these policies, you should examine the associated conversations (See Conversations) to identify these endpoints and subsequently categorize them into finer scopes or inventory filters, to avoid less-secure policies at the root-scope level.

## *Fine-Tune External Dependencies for a Workspace*

Use this procedure to create policies between specified subsets of workloads within scopes (rather than between entire scopes) during automatic policy discovery, when the provider of a policy belongs to a different scope than the scope in which policies are being discovered.

***Figure 9: Fine-tuning External Dependencies***



External Dependencies

Manage the granularity of ADM generated policies for services that are external to the application.

32 hidden scopes

| | | | |
|---|---|---|---|
| JustACompany : 1.1.1.50 | Coarse Fine | 1 dependent filters | |
| 1 custom inventory filters 1.1.1.50 | | | close |
| JustACompany : Research : Innovations | Coarse Fine | 0 dependent filters | |
| JustACompany : Research : Applications | Coarse Fine | 0 dependent filters | |
| JustACompany : Research | Coarse Fine | | |
| JustACompany : Engineering : Search | Coarse Fine | | |

### Before you begin

- Configure an inventory filter for each subset of workloads for which you want to generate specific policies. You can create any number of inventory filters, in any scope.

  There are several ways to create inventory filters:

  - Convert clusters of interest to inventory filters.

    (See Convert a Cluster to an Inventory Filter, on page 77),

    and/or

  - Create new inventory filters.

    See Create an Inventory Filter.

  These filters must have the following options enabled:

  - **Restrict query to ownership scope**

    **Provides a service external of its scope**

- See also Tips for Exploring External Dependencies, on page 32.

### Procedure

**Step 1** Navigate to the workspace in which you will discover policies.

**Step 2** Click **Automatically Discover Policies**.

**Step 3** Click **External Dependencies**.

**Step 4** If necessary, click **Show All** scopes.

**Step 5** (Optional) Leverage previous configurations:

- To reuse the changes you made to the list the last time you discovered policies, click **Previous Config**.
- If you have set up external dependencies in the global “Default Policy Discovery Config”, you can use the global list by clicking **Default Config**. Or, after obtaining the default list, you can modify it as desired (for that workspace only), and then use the customized version on subsequent runs by clicking **Previous Config** once.

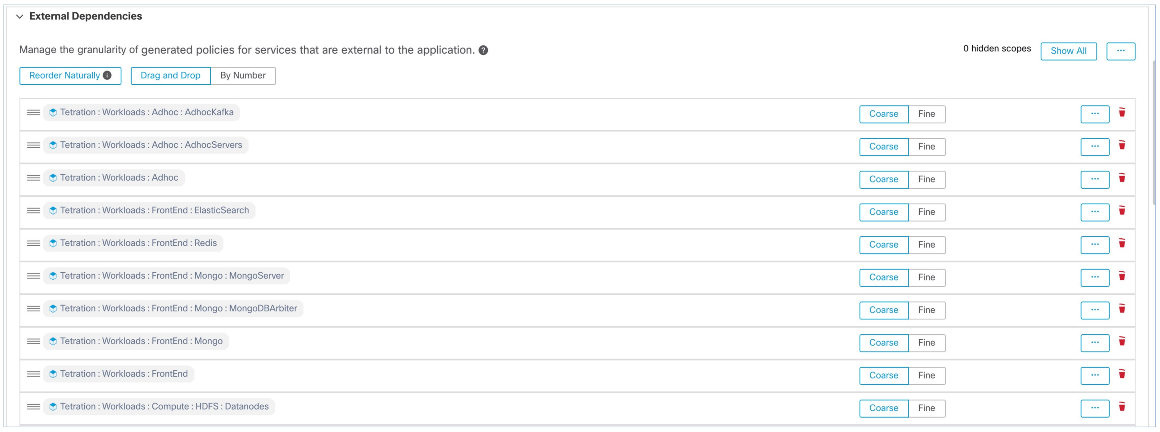
**Step 6** Reorder scopes (and inventory filters, if applicable) as needed.

Policy is applied based on the first scope or inventory filter in the list (starting from the top) that matches the traffic. For this purpose, you generally want to apply the most specific policy that matches traffic, so you want child scopes (more specific) above their parents (less specific).

- If you have recently created new child scopes, which by default are added to the bottom of the list, reorder the entire list to place child scopes above their parents:

  (Recommended) Click **Reorder Naturally**.

*Figure 10: Reorder naturally*



External Dependencies

Manage the granularity of generated policies for services that are external to the application.

0 hidden scopes

| | | |
|---|---|---|
| Tetration : Workloads : Adhoc : AdhocKafka | Coarse | Fine |
| Tetration : Workloads : Adhoc : AdhocServers | Coarse | Fine |
| Tetration : Workloads : Adhoc | Coarse | Fine |
| Tetration : Workloads : FrontEnd : ElasticSearch | Coarse | Fine |
| Tetration : Workloads : FrontEnd : Redis | Coarse | Fine |
| Tetration : Workloads : FrontEnd : Mongo : MongoServer | Coarse | Fine |
| Tetration : Workloads : FrontEnd : Mongo : MongoDBArbiter | Coarse | Fine |
| Tetration : Workloads : FrontEnd : Mongo | Coarse | Fine |
| Tetration : Workloads : FrontEnd | Coarse | Fine |
| Tetration : Workloads : Compute : HDFS : Datanodes | Coarse | Fine |

- (If you have a specific reason) To reorder the list manually:
  - Click **Drag and Drop**.
  - Click **By Number**:

    The external dependencies will be assigned priority values in multiples of 10. Change the values to change the order.

    Once numbers are modified, click **View** to update the list order and reassign multiples of 10 to each of the priorities.

**Step 7** Specify granularity for each row:

- Click **Fine** for each row for which you want to generate policies specific to configured inventory filters or clusters.

  Click **Coarse** to generate policies that apply to the entire scope.

- To apply granularity to all subscopes of a scope: Click the ··· button at the end of the scope's row.

## Advanced Configurations for Automatic Policy Discovery

Use advanced settings to include additional information when discovering policies or to adapt to a particular environment.

- To access these settings for a specific workspace, click **Automatically Discover Policies** in the applicable workspace.
- To change the defaults for all workspaces, see Default Policy Discovery Config, on page 45.

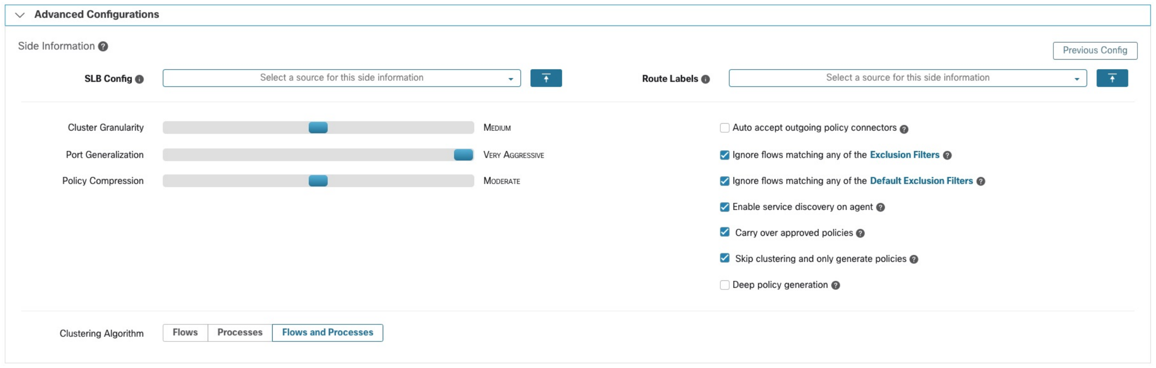


*Figure 11: Advanced Automatic Policy Discovery Configurations*

Advanced Configurations

Side Information

SLB Config: Select a source for this side information

Route Labels: Select a source for this side information

Previous Config

Cluster Granularity: Medium

Port Generalization: Very Aggressive

Policy Compression: Moderate

- [ ] Auto accept outgoing policy connectors
- [x] Ignore flows matching any of the Exclusion Filters
- [x] Ignore flows matching any of the Default Exclusion Filters
- [x] Enable service discovery on agent
- [x] Carry over approved policies
- [x] Skip clustering and only generate policies
- [ ] Deep policy generation

Clustering Algorithm: Flows | Processes | [✓ Flows and Processes]

## *Include Data From Load Balancers and Routers When Discovering Policies*

You can upload data from load balancers and routers to inform automatic policy discovery.

To access the following options, click **Advanced Configurations** in the automatic policy discovery settings and look at the "Side Informaton" or "sideinfo" section.

| Option | Description |
|---|---|
| **SLB Config**<br><br>(Upload load balancer configurations) | To download data from your load balancer in the correct format, see Retrieving LoadBalancer Configurations for Advanced Policy Discovery Configuration.<br><br>Supported formats for uploading loadbalancer configs:<br><br>• **F5 BIG-IP**<br>• **Citrix Netscaler**<br>• **HAProxy**<br>• Others:<br><br>Use the **Normalized JSON** schema.<br><br>You must convert any unsupported load balancer config into this schema.<br><br>This simple schema includes basic information on Virtual IPs (VIPs) and backend IPs.<br><br>To download a sample JSON file, click the info button beside **SLB Config**. |
| Upload **Route Labels** | You can upload a list of provisioned subnets/routes from the routers to help partition hosts based on pre-provisioned set of subnets. The clustering results generated by automatic policy discovery never span the subnet boundaries as defined by the uploaded data. You can modify the results after automatic policy discovery is complete.<br><br>To download a sample JSON file, click the info button beside **Route Labels**. |

**Note** Clusters do not span partition boundaries, meaning a cluster computed by automatic policy discovery does not contain target workloads from two different partitions. Partitions are computed from the uploaded load balancer or router data. However, you can freely move workloads from one cluster to another, for example by changing cluster query definitions (manual cluster editing), or disable the upload of any side info.

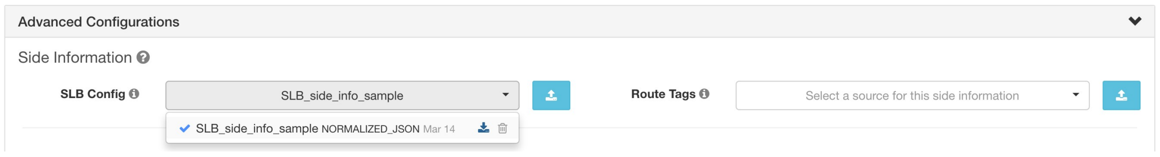
**To view or delete a previously uploaded Load Balancer (SLB Config) or Route Labels file:**

1. Click into the respective box labeled **Select a source for this side information**.

   A list of uploaded files will appear.

2. Click the download or trash icon beside the file to view or delete.

   *Figure 12: Uploaded Side Information*


Advanced Configurations
Side Information
SLB Config
SLB_side_info_sample
SLB_side_info_sample NORMALIZED_JSON Mar 14
Route Tags
Select a source for this side information

## *Cluster Granularity*

Clustering Granularity allows you to control the size of the clusters generated by automatic policy discovery.

- **Fine** results in more but smaller clusters
- **Coarse** results in fewer but larger clusters

**Note** You may not observe a significant change in the results due to many other signals that our algorithms take into account. For example, if there is a very high confidence in the generated clusters, changing this control will make little change in the results.

## *Port Generalization*

The **Port Generalization** option in **Advanced Configurations** for automatic policy discovery controls the level of statistical significance required when performing port generalization, i.e., replacing numerous ports being used as server ports on a single workload, with a port interval.

This setting can affect accuracy, number, and compactness of policies and the time required to generate them.

To disable port generalization, move the slider to the extreme left. Note that if disabled, automatic policy discovery and/or automatic policy discovery UI rendering time may be slowed substantially, in case many server ports are used by the workloads.

As the slider is moved to the right toward more aggressive generalization, less evidence is required to create port-intervals and also the criterion for replacing original policies (involving single ports) with port-intervals is relaxed.

### Background

Some applications such as Hadoop use and change many server ports in some interval, for instance in 32000 to 61000. Automatic policy discovery attempts to detect such behavior for each workload, using the workload's server port usages in the observed flows: by observing only a fraction of total possible ports (but numerous ports, eg 100s), automatic policy discovery may 'generalize' that any port in, say 32000 to 61000, could be used as a server port by the workload. Ports that fall within intervals are replaced with such intervals (when certain criteria on minimum observed counts are met). This results in fewer, more compact policies. Interval estimation is important for computing accurate policies: without sufficient generalization many legitimate future flows would be dropped if the policy is enforced. By merging numerous ports into one or a few intervals, the rendering time of the UI is sped up significantly as well.

You can control the degree of port generalization including disabling it.

*Policy Compression*

When policy compression is enabled, if policies in multiple clusters in the workspace are similar, then those policies can be replaced with one or more policies applicable to the entire parent scope. For example, if all or almost all clusters in the workspace provide the same port to the same consumer, then all of those cluster-specific policies are replaced with one policy in the parent scope. This reduces the number of policies significantly, minimizes clutter, and may also allow legitimate future flows that would have been dropped (accurate generalization).

The more aggressive the compression setting, the smaller is the required threshold on policy frequency in order to replace cluster-specific policies with a policy applicable to the entire parent.

**When generating policies for a branch of the scope tree:**

This knob can be used to alter the level of aggressiveness in Hierarchical policy compression.

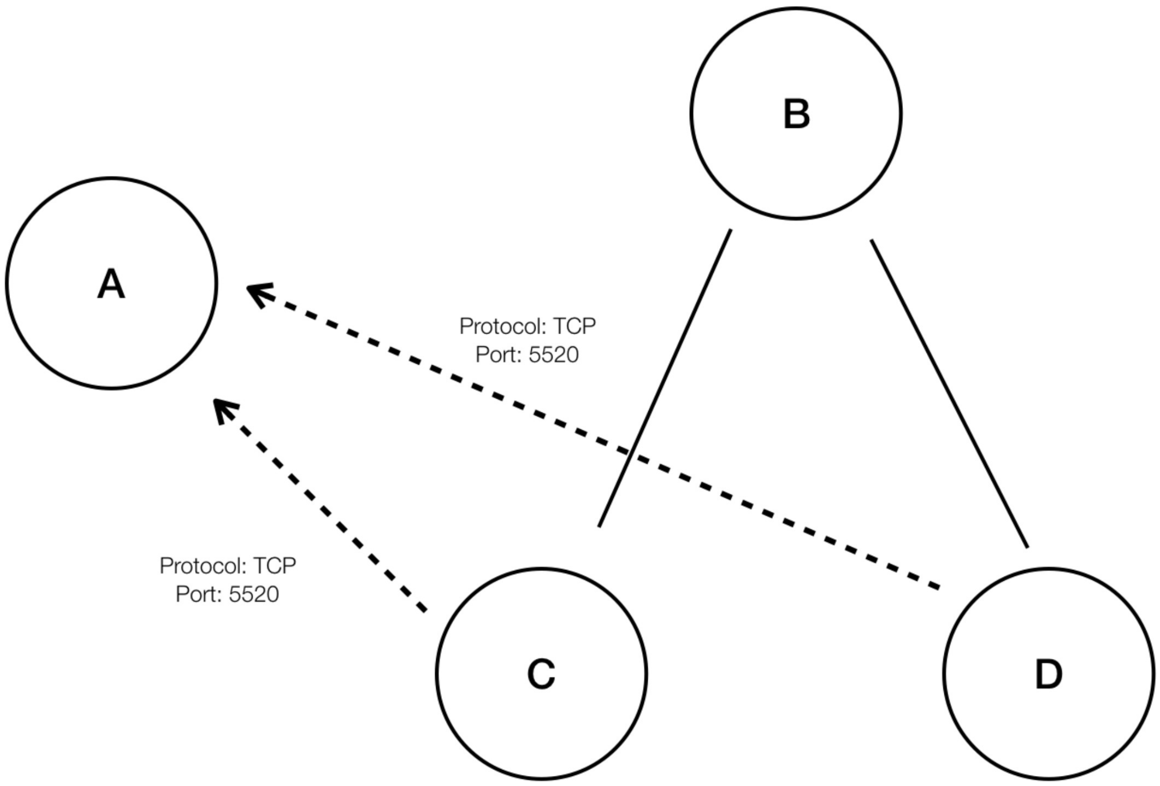
**Note** Currently, the automatic policy discovery conversations page does not support showing the conversations that led to a compressed policy (you may need to disable compression or use flow search).

Hierarchical policy compression

Policy compression can also be done when generating policies for a branch of the scope tree. The Policy Compression knob can be used to alter the level of aggressiveness in hierarchical policy compression. An example of hierarchical policy compression is illustrated below.

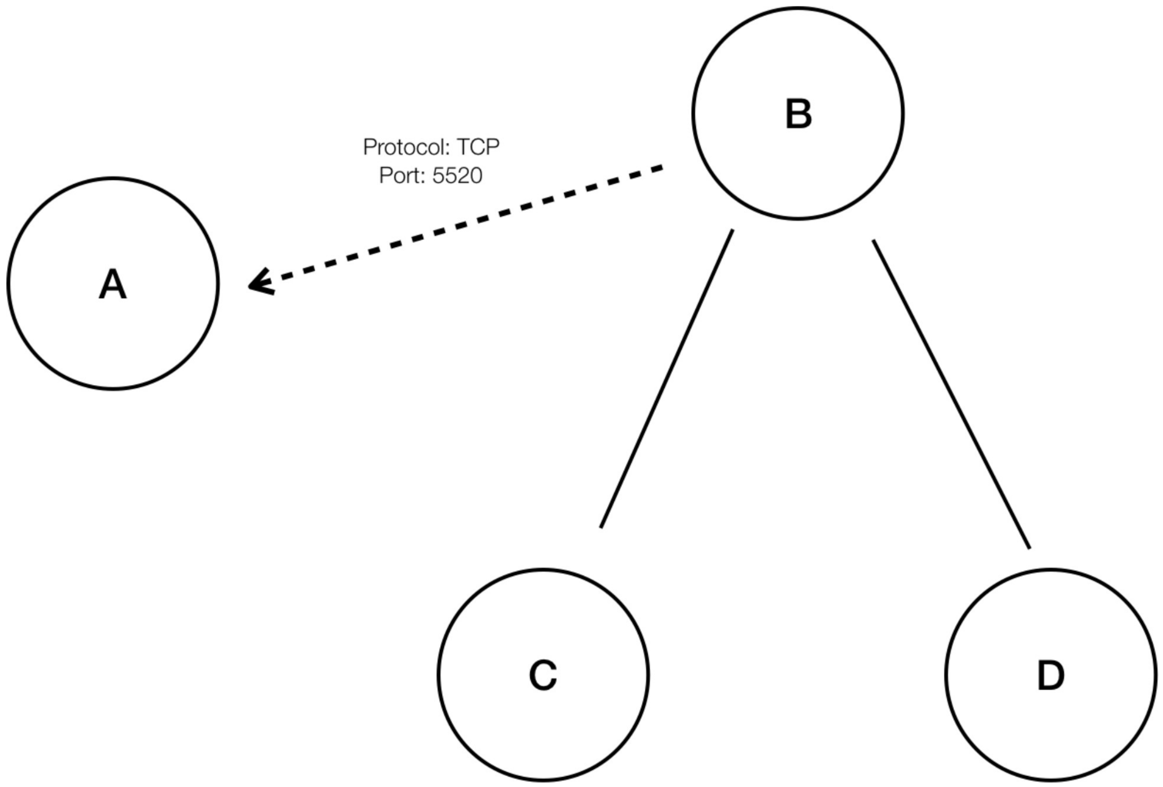
- Let A, B, C and D be scopes part of a scope tree, where "C" and "D" are the child scopes of "B". Let "C" → "A" be a TCP "ALLOW" policy on port 5520 and "D" → "A" be TCP "ALLOW" policy on port 5520.

*Figure 13: Before hierarchical policy compression*



- With hierarchical policy compression if a sufficiently large group child scopes involves in policies sharing the same port, protocol and destination or source, these policies will be replaced by a generalized policy that connects the parent scope to the common source or destination. In the above mentioned case "C" and "D" are child scopes of "B" and the policies "C" → "A" and "D" → "A" share the same destination, port and protocol. Since 100% of child scopes of "B" contain the similar policy the policy will be promoted to be "B" → "A", resulting in the following. Furthermore, hierarchical compression can be repeated so a generalized policy can go all the way to the root of the subtree (branch of the scope tree) .

*Figure 14: After hierarchical policy compression*



- The policy compression knob allows you to tune the aggressiveness of such compression, by changing the minimum required proportion of the policy-sharing child scopes (usually measured as the fraction of total number of child scopes) to trigger the compression. When disabled, each policy is generated between highest priority scopes based on the External Dependencies list. Subsequently, if you choose to impose the naturally ordered External Dependencies list, the policies generated will be the most granular policies among scopes.

### *Clustering Algorithm (Input to Clustering)*

Advanced users can choose the main source of data for clustering algorithms, that is, live network flows, or running processes, or both.

### *Auto accept outgoing policy connectors*

This option is applicable only when you use automatic policy discovery to create cross-scope policies using the method described in (Advanced) Create Cross-Scope Policies, on page 89.

Any outgoing policy requests created during automatic policy discovery will be automatically accepted.

For complete information, see Auto Accept Policy Connectors, on page 98 and Policy Requests.

**Note** This option is only available for root scope owners and site admins.

### *Auto Approve Generated Policies*

This option is applicable if you want to approve all policies generated by policy discovery.

**Note** Be aware that if you choose this option, and later on if you need to modify or undo any changes, you can only do so manually.

For more information, see Auto Accept Policy Connectors, on page 98 and Policy Requests.

**Note** This option is available for root scope owners and site administrators.

### *Ignore Flows Matching Exclusion Filters*

To ignore conversation flows that you specify, enable the applicable option. To view or modify either filters list, click the applicable **Exclusion Filters** link. For more information, see Exclusion Filters, Default Exclusion Filters, on page 45, and Configure, Edit, or Delete Exclusion Filters, on page 29.

### *Enable service discovery on agent*

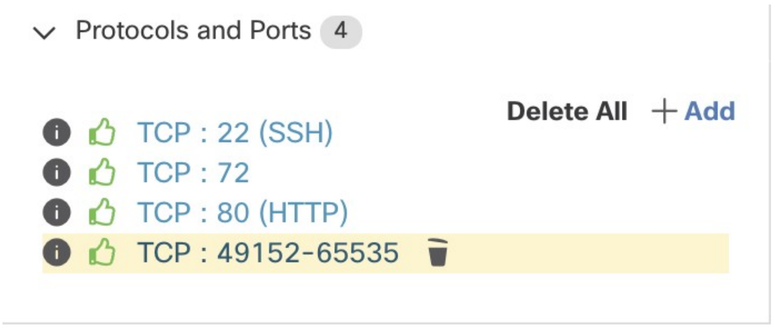
In certain applications, a large range of ports might be designated for use, but actual traffic might use only a subset of those ports during the time period included in policy discovery. This option allows the entire designated pool of ports for these applications to be included in policies for those applications, rather than just the ports seen in actual traffic.

Enabling this option allows ephemeral port-range information regarding services present on the agent node to be gathered. Policies are then generated based on this port-range information.

**Example:**

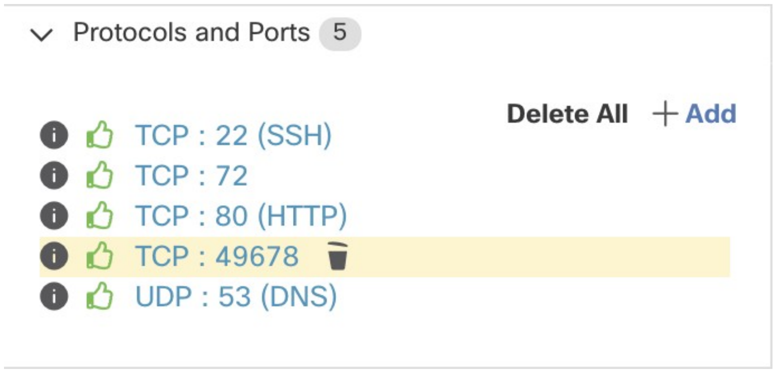
- Windows Active Directory Domain Server uses default Windows ephemeral port-range **49152-65535** to serve requests. When this flag is set this port range information is reported by the agent and policies are generated based on this information.

***Figure 15: Service discovery enabled on the agent***



TCP : 22 (SSH)
TCP : 72
TCP : 80 (HTTP)
TCP : 49152-65535

*Figure 16: Service discovery not enabled on the agent*



TCP : 22 (SSH)
TCP : 72
TCP : 80 (HTTP)
TCP : 49678
UDP : 53 (DNS)

### *Carry over Approved Policies*

This option is enabled by default.

When this flag is set, all the policies that you have marked as approved (including those approved using OpenAPI) will be preserved. This helps you to not have to re-define a particular broad DENY rule that should take effect regardless of the “allow” policies that are discovered by automatic policy discovery.

For details, see Approved Policies, on page 47.

### *Skip clustering and only generate policies*

If this option is selected, no new clusters are generated, and policies are generated from any existing approved clusters or inventory filters and otherwise involve the entire scope associated with the workspace (in effect, treating the entire scope as a single cluster). This option can result in substantially fewer (but coarser) policies.

### *Enable redundant policy removal*

This option is only available when generating policies for a branch of the scope tree.

This option enables/disables removal of redundant granular policies.

**Example:**

- Let Root, A, B, C, A1 and A2 be scopes part of a scope tree. Let the following be the policies:

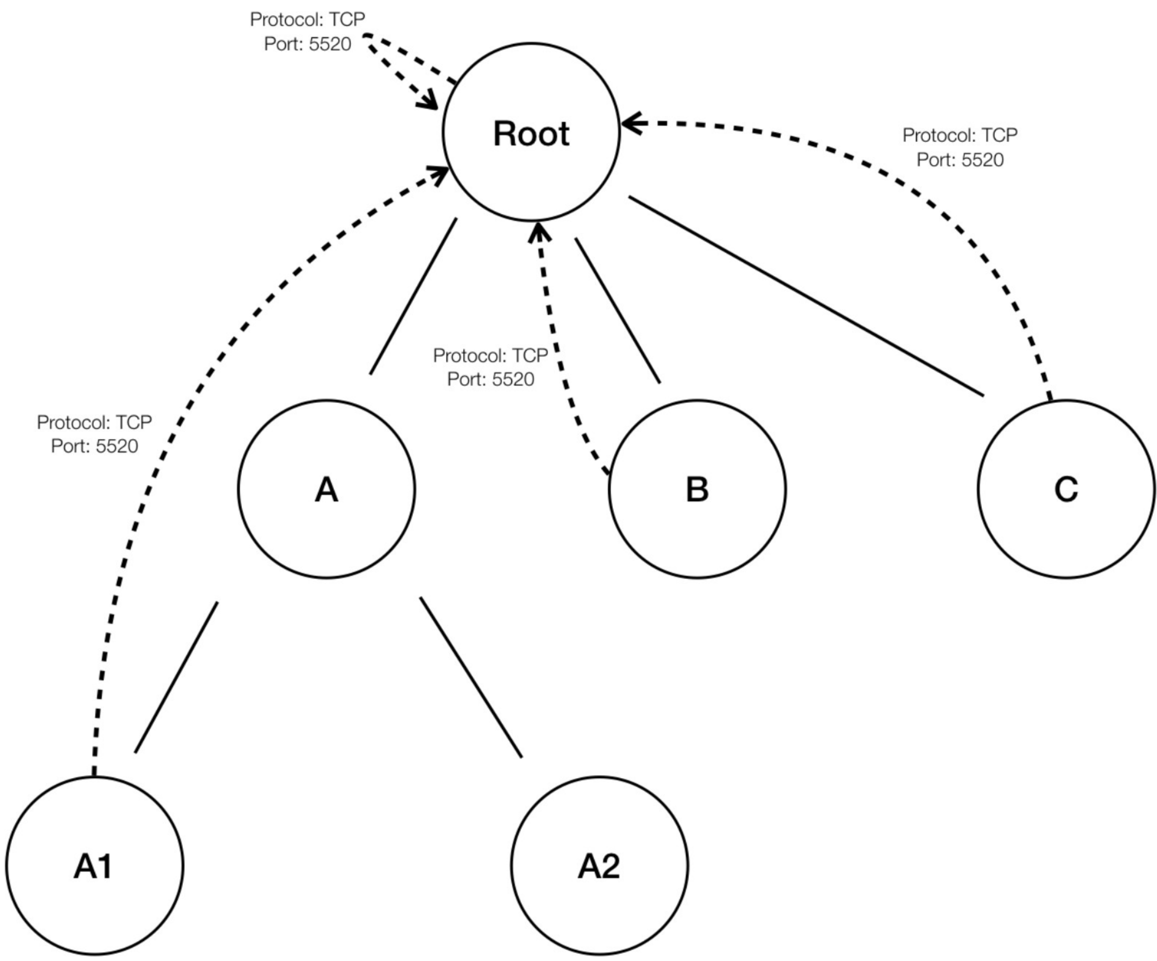
  **1.** “Root” → “Root”

  **2.** “B” → “Root”

  **3.** “C” → “Root”

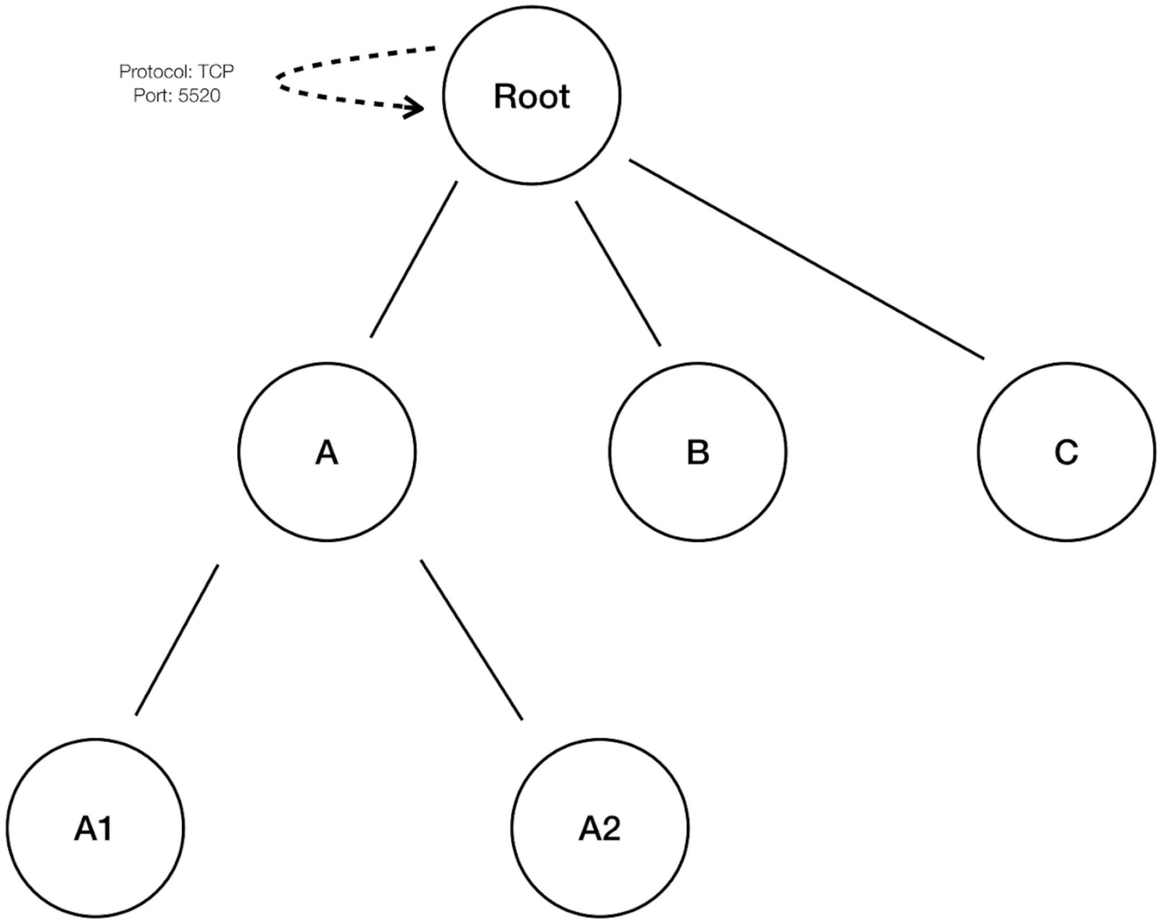
  4. “A1” → “Root”

*Figure 17: Before removal of redundant policies*



- The policies “B” →“Root”, “C” →“Root” and “A1”→ “Root” are redundant as the policy “Root” “Root” covers these policies. The remove redundant policies feature will check and remove such policies resulting in only one policy “Root” → “Root” as follows.

*Figure 18: After removal of redundant policies*



Redundant policy removal can be very useful in maintaining a succinct set of interpretable policies. The reduced policy set contains the minimal number of policies at the chosen compression level to cover all the workload traffic. However, you should always audit the policy through policy analysis and examine the corresponding conversations to evaluate the tightness of the resulting policies. This is especially important when there exists traffic to or from endpoints that are not categorized into finer scopes or inventory filters. Such endpoints may trigger the generation of coarser policies than intended, such as policies involving the root scope. If at the same time, redundant policy removal is enabled, more granular policies will be removed and will not be presented to you. To diagnose the source of (compressed) policies and to view finer level policies, turn off policy compression and redundant policy removal. Also note that currently, the automatic policy discovery conversations page may fail to show the conversations that lead to a compressed/generalized policy; so to get around this, you can turn off compression and redundant policy removal, so it is easier to find the conversations that lead to the generated policies.

**Tip** Since discovering policies for a branch of the scope tree discovers all policies for the scope subtree rooted at the workspace scope, these policies will cover all the legal traffic seen by automatic policy discovery for all the workloads under the subtree. When analyzing these policies using tools such as Policy Analysis (See Policies), you should turn off Policy Analysis in all the workspaces associated with the subscopes. This way, the policies (if any) residing in the subscope workspaces (usually receive a high priority due to more specific scope definition) will not take priority and interfere with the results. However, exceptions apply when the policies in the subscope workspaces are configured to cover different sets of traffic that usually involve finer inventory filters or clusters specific to the subscopes.

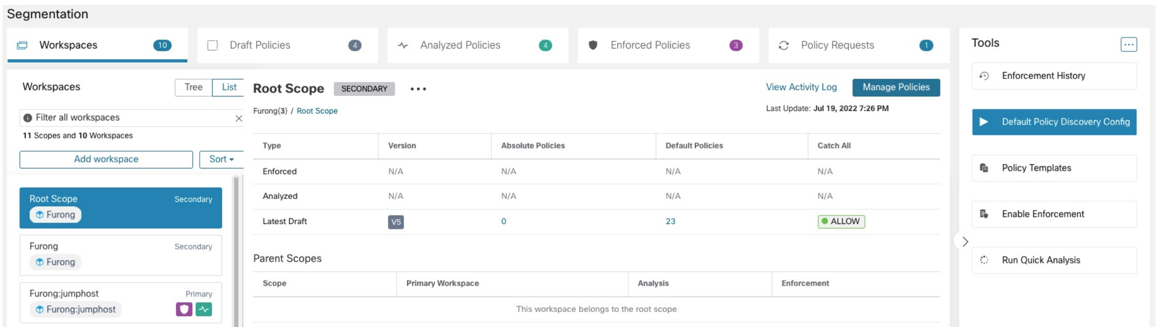
## Default Policy Discovery Config

You can configure default automatic policy discovery settings that can optionally be used in any workspace in the entire root scope.

To configure default options for policy discovery:

Choose **Defend** > **Segmentation**, then click the caret at the right side of the page to expand the Tools menu. Then choose **Default Policy Discovery Config**.

***Figure 19: Navigating to the Default Policy Discovery Config page***



Last Update: Jul 19, 2022 7:26 PM

| Type | Version | Absolute Policies | Default Policies | Catch All |
|---|---|---|---|---|
| Enforced | N/A | N/A | N/A | N/A |
| Analyzed | N/A | N/A | N/A | N/A |
| Latest Draft | V5 | 0 | 23 | ALLOW |

Parent Scopes

| Scope | Primary Workspace | Analysis | Enforcement |
|---|---|---|---|

This workspace belongs to the root scope

For information about options on the Default Policy Discovery Config page, see:

- External Dependencies, on page 31 and subtopics
- Advanced Configurations for Automatic Policy Discovery, on page 35 and subtopics
- Default Exclusion Filters, on page 45

**Important** When your default configurations are complete and ready to use in individual workspaces, click **Save**.

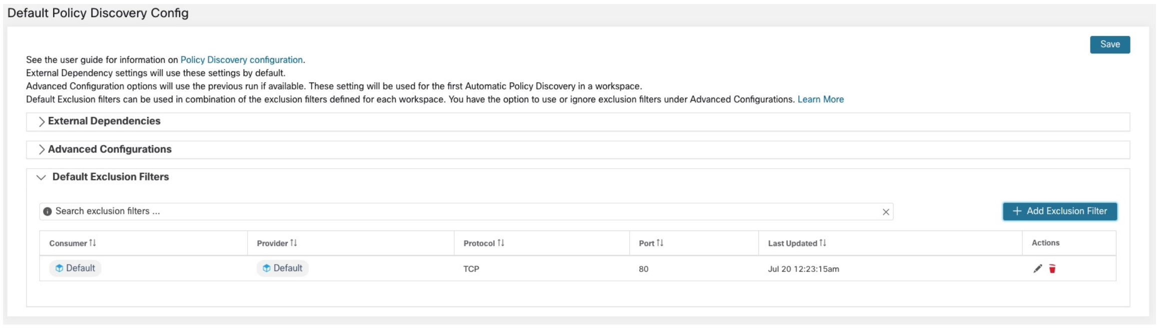
### *Default Exclusion Filters*

Exclusion Filters help you fine-tune policies and clusters suggested by automatic policy discovery by specifying traffic flows to exclude from discovery input.

For details, see Exclusion Filters.

You can make a global Default Exclusion Filters list that is available to all workspaces in your tenant, then specify for each workspace whether or not to use this default list when discovering policies.

*Figure 20: Default Exclusion Filters*



Default Policy Discovery Config

Save

See the user guide for information on Policy Discovery configuration.
External Dependency settings will use these settings by default.
Advanced Configuration options will use the previous run if available. These setting will be used for the first Automatic Policy Discovery in a workspace.
Default Exclusion filters can be used in combination of the exclusion filters defined for each workspace. You have the option to use or ignore exclusion filters under Advanced Configurations. Learn More

External Dependencies

Advanced Configurations

Default Exclusion Filters

Search exclusion filters ...

Add Exclusion Filter

| Consumer | Provider | Protocol | Port | Last Updated | Actions |
|---|---|---|---|---|---|
| Default | Default | TCP | 80 | Jul 20 12:23:15am | |

To configure default exclusion filters, see Configure, Edit, or Delete Exclusion Filters, on page 29.

To enable or disable default exclusion filters, see Enable or Disable Exclusion Filters, on page 31.

## Retrieving LoadBalancer Configurations for Advanced Policy Discovery Configuration

Below are the instructions for retrieving supported load balancer configuration files in a format that can be directly uploaded to Secure Workload for use in policy discovery. For more information, see Advanced Configurations for Automatic Policy Discovery and Include Data From Load Balancers and Routers When Discovering Policies, on page 36.

Note that all files must be encoded as ASCII.

### *Citrix Netscaler*

Concatenate the output of `show run` in your console and upload the file.

See Sample config file

### *F5 BIG-IP*

Upload the `bigip.conf` file.

**Note** If you have a file with a .UCS extension, unzip the archive folder and upload only the `bigip.conf file` within the configuration dump. If there are multiple `bigip.conf files`, concatenate and then upload the files.

See Sample config file

### *HAProxy*

Upload your `haproxy.cfg` file. The path is typically `/etc/haproxy/haproxy.cfg`.

See Sample config file

### *Normalized JSON*

If you find the above options limiting, convert your configs to the following JSON schema and upload them directly. The example JSON file can be directly downloaded by clicking the **i** icon next to SLB Config in Advanced Run Configurations for automatic policy discovery.

See Sample config file

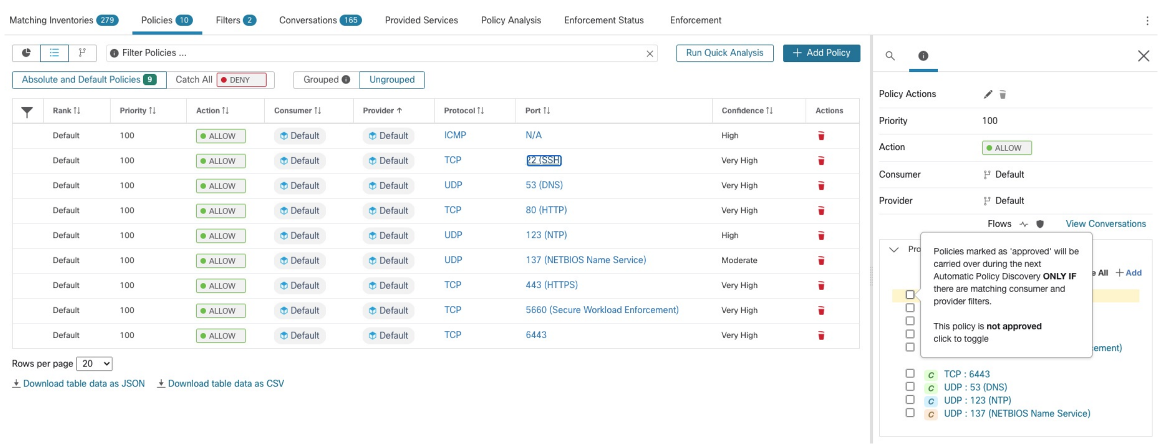
## Approve Policies

As you review policy discovery results, approve discovered policies that you want to keep, to carry them forward intact when you discover policies in future. For complete details, see Approved Policies, on page 47.

To approve a policy:

1. On the Policies page, for the policy you want to protect, click the value in the **Protocols and Ports** column.
2. In the panel that opens on the right, select the checkbox to the left of each protocol-and-port for which you want to retain the policy during future policy discovery.

***Figure 21: Approve Policies***



| | Rank | Priority | Action | Consumer | Provider | Protocol | Port | Confidence | Actions |
|---|---|---|---|---|---|---|---|---|---|
| | Default | 100 | ALLOW | Default | Default | ICMP | N/A | High | |
| | Default | 100 | ALLOW | Default | Default | TCP | 22 (SSH) | Very High | |
| | Default | 100 | ALLOW | Default | Default | UDP | 53 (DNS) | Very High | |
| | Default | 100 | ALLOW | Default | Default | TCP | 80 (HTTP) | Very High | |
| | Default | 100 | ALLOW | Default | Default | UDP | 123 (NTP) | High | |
| | Default | 100 | ALLOW | Default | Default | UDP | 137 (NETBIOS Name Service) | Moderate | |
| | Default | 100 | ALLOW | Default | Default | TCP | 443 (HTTPS) | Very High | |
| | Default | 100 | ALLOW | Default | Default | TCP | 5660 (Secure Workload Enforcement) | Very High | |
| | Default | 100 | ALLOW | Default | Default | TCP | 6443 | Very High | |

Rows per page 20

Download table data as JSON   Download table data as CSV

| | |
|---|---|
| Policy Actions | |
| Priority | 100 |
| Action | ALLOW |
| Consumer | Default |
| Provider | Default |

Flows   View Conversations

Policies marked as 'approved' will be carried over during the next Automatic Policy Discovery **ONLY IF** there are matching consumer and provider filters.

This policy is **not approved**
click to toggle

- [ ] TCP : 6443
- [ ] UDP : 53 (DNS)
- [ ] UDP : 123 (NTP)
- [ ] UDP : 137 (NETBIOS Name Service)

You can also use this procedure to remove approval from a policy.

### Approved Policies

In general, approved policies are not changed during automatic policy discovery, and automatic policy discovery does not suggest policies that would duplicate or overlap the effects of approved policies.

The following are approved policies:

- Manually created policies.
- Discovered policies that are manually approved.

  (When you are satisfied that a policy behaves as intended, you approve it to protect it from changes during future automatic policy discovery. See Approve Policies, on page 47.)
- Uploaded policies, unless explicitly marked as `approved: false`.
- Approved policies that are defined in parent and ancestor scopes (specifically, from the latest versions of their primary workspaces) that apply to workloads in this scope.
- Policies created when policy requests are accepted from another workspace when cross-scope policies are handled using the advanced method that is described in When Consumer and Provider Are in Different Scopes: Policy Options, on page 88. For example, this includes policies that are included from the Provided Services, on page 100 tab.

Approved policies are shown with a thumbs-up icon next to the protocol type when you click a policy's ports or protocols link and view details in the panel at the right side of the page.

#### Exceptions to Approved Policy Protections

Approved policies are preserved during future automatic policy discovery if *both* ends of the policy are any of: approved cluster; inventory filter; accepted policy request (for cross-scope policies); or a cluster that doesn't significantly change membership. (However, the cluster membership may have changed in the last case.)

Approved policies may not be protected during future automatic policy discovery runs if either end of the policy is a cluster that is not approved, and if, upon automatic policy discovery, no newly generated cluster has sufficiently high overlap with such cluster.

To protect a policy that involves an unapproved cluster, you should explicitly approve the clusters at each end of the policy.

There is also an advanced configuration for automatic policy discovery that is enabled by default. If you do not want to protect approved policies from changes, you can deselect this option for a workspace or for the global default policy discovery configuration:. See Carry over Approved Policies, on page 42.

### Troubleshoot Approved Policies

#### Approved policies are not being carried forward

If approved policies are not being carried forward as expected, make sure the **Carry over approved policies** option is selected in the advanced and/or default configuration settings for automatic policy discovery.

#### Finding conversations that are excluded from policy generation

During automatic policy discovery, any conversations that match the criteria for an existing approved policy are excluded from the policy generation. This omission prevents redundant policies covering the same conversations from being generated. (This process differs from the exclusion filters (See Exclusion Filters), in which you define matching filters instead of policies. Exclusion filters prevent matching conversations from being visible to all parts of automatic policy discovery. )

Note that while redundant policies are not generated from these conversations, the conversations are still considered when automatic policy discovery analyzes and generates clusters.

To see which conversations are excluded from automatic policy discovery by existing approved policies:

In the conversations view (See Conversations), use the **excluded** flag to filter conversations. You can also explore which existing approved policies result in the exclusion of these conversations in the policy details view that opens on the right side of the page when you click the ports and protocols link in a policy, then click the exclusion icon next to the conversation. (Hover over the icons to find the right icon.)

## Iteratively Revise Policies

Defining and refining policies, for a single scope and for an entire network, will be an iterative process.

You can expect to revise both discovered and manually created policies.

### Re-running Automatic Policy Discovery

You can rerun automatic policy discovery at any time. The main reasons to rerun automatic policy discovery are to include additional information that was not included in the previous run, or to exclude information that is not helpful. For example, you can:

- Install additional agents or configure additional connectors, and allow some flow data to accumulate.

- Increase the timespan used for discovery, to include more data.
- Approve clusters (with or without editing them first), which can improve the clustering of other workloads upon rerun. See Approving Clusters, on page 80.
- Exclude flows that you know you don't want to influence policy so you don't have to edit them out. See Exclusion Filters, on page 29.
- Change advanced settings (for details, see Advanced Configurations for Automatic Policy Discovery, on page 35.)
- Capture changes after you have made changes to Address Policy Complexities, on page 82.

Automatically discovering policies again on an existing workspace may generate different clusters and policies in the workspace.

If a host is no longer in the scope of the workspace, upon a subsequent automatic policy discovery run, that host will not appear in any cluster; if it were in an approved cluster, it will no longer appear in that cluster. Even with the same set of member workloads but with a different timeframe or configuration, automatic policy discovery may result in different clusters.

**Note** For a list of the types of policies that are not modified during policy discovery, see Approved Policies, on page 47.

**Note** *Removal of Redundant Policies* On subsequent automatic policy discovery, approved policies in primary workspaces will remove matching conversations for policy generation, so redundant policies are not generated. Note that, as is the case for exclusion filters, this functionality may not work perfectly on non-primary workspaces if the policy uses a Cluster filter defined in the workspace. Cluster filters from a non-primary workspaces are not active, and will not match any flows, thus redundant policies may still be generated in non-primary workspaces during automatic policy discovery.

## Important: Before You Re-run Automatic Policy Discovery

**Important** Address each of the following before re-discovering policies in a workspace:

- By default, each time you discover policies in a particular workspace, the previous set of discovered policies and clusters are overwritten based on the data included in the new discovery period. If you want to keep some policies and clusters but not others, approve those policies and clusters.
- If you want to preserve any existing generated clusters, see Preventing Cluster Modification During Automatic Policy Discovery Reruns or Approving Clusters, on page 80.
- If you want to preserve any existing generated policies, see Approve Policies, on page 47.
- Any existing **Advanced Configuration** settings configured in the previous discovery run are used unless you change them.

  However, any configured *default* External Dependencies will be used over those of the previous run.
- If the currently displayed version of the discovered policies is not the latest version, and you want to keep previously discovered versions, click the version displayed at the top of the page and choose the latest v* version.

  If a previous version is displayed, any versions between that version and the new discovered version will be deleted.

  For details, see View, Compare, and Manage Discovered Policy Versions, on page 50.

To re-run policy discovery, see Automatically Discover Policies, on page 27. After you have addressed the points in this topic, the process is the same each time you discover policies.

## View, Compare, and Manage Discovered Policy Versions

Each time you discover policies in a workspace, the version number (v*) assigned to the set of policies increments.

For information, see About Policy Versions (v* and p*), on page 136.

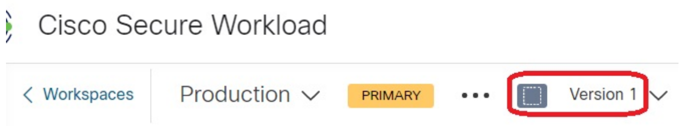
### Procedure

**Step 1** Click **Defend** > **Segmentation**.

**Step 2** Navigate to the workspace.

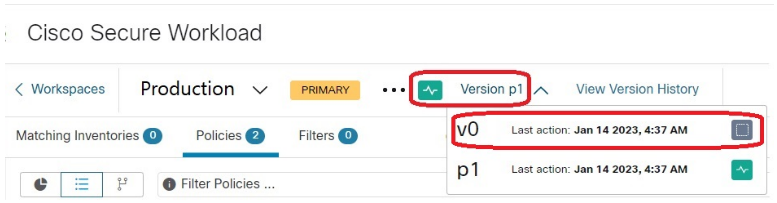
**Step 3** Click **Manage Policies**.

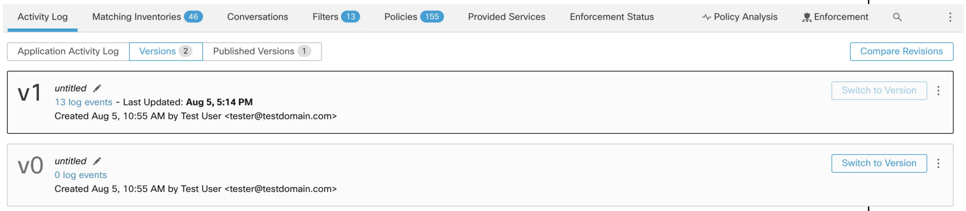
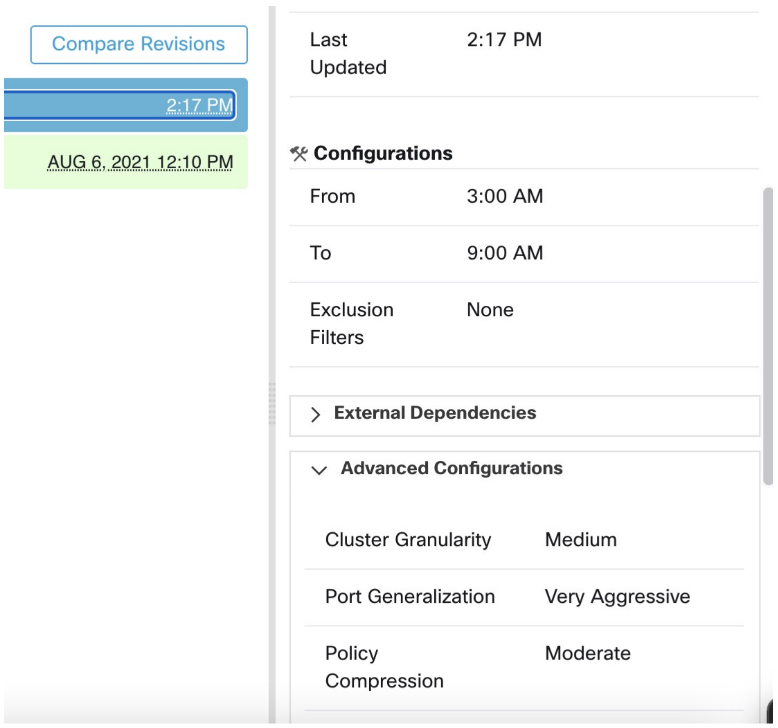
**Step 4** The currently displayed version of the policies generated by automatic policy discovery is shown at the top of the page:



If you have already analyzed or enforced policies, the displayed version may be a policy discovery version, an analyzed policy version, or an enforced version.

**Step 5** Do one of the following:

| | |
|---|---|
| Display a different version of the policies generated by automatic policy discovery: | Click the current version and choose a different v* version.<br>(If you see p* versions, those are analyzed and/or enforced versions, not versions of discovered policies.)<br> Cisco Secure Workload<br>Workspaces<br>Production<br>PRIMARY<br>Version p1<br>View Version History<br>Matching Inventories 0<br>Policies 2<br>Filters 0<br>Filter Policies ...<br>v0 Last action: Jan 14 2023, 4:37 AM<br>p1 Last action: Jan 14 2023, 4:37 AM<br>**Important!!** See the caveat in the What To Do Next section at the end of this procedure. |

| | |
|---|---|
| View details about a version | **a.** Click **View Version History** at the top of the page beside the current version.<br><br>**b.** Click the **Versions** tab to see the versions of discovered policies. (Not the Published Versions tab.)<br><br>The list of versions displays:<br><br>***Figure 22: List of generated policy versions with summary information***<br><br> Activity Log<br>Matching Inventories 46<br>Conversations<br>Filters 13<br>Policies 155<br>Provided Services<br>Enforcement Status<br>Policy Analysis<br>Enforcement<br>Application Activity Log<br>Versions 2<br>Published Versions 1<br>Compare Revisions<br>Switch to Version<br>Switch to Version<br>v1 *untitled*<br>13 log events - Last Updated: **Aug 5, 5:14 PM**<br>Created Aug 5, 10:55 AM by Test User <tester@testdomain.com><br>v0 *untitled*<br>0 log events<br>Created Aug 5, 10:55 AM by Test User <tester@testdomain.com><br><br>**c.** Click the **log events** link in the version.<br><br>**d.** Click a link in an event row.<br><br>Available details include statistics, exclusion filters, external dependencies, and configurations for the run.<br><br>***Figure 23: Configurations used for particular automatic policy discovery runs***<br><br> Compare Revisions<br>2:17 PM<br>AUG 6, 2021 12:10 PM<br>Configurations<br>External Dependencies<br>Advanced Configurations<br>Last Updated 2:17 PM<br>From 3:00 AM<br>To 9:00 AM<br>Exclusion Filters None<br>Cluster Granularity Medium<br>Port Generalization Very Aggressive<br>Policy Compression Moderate |
| Compare two versions to see what has changed: | **a.** Click **Compare Revisions**.<br><br>**b.** Choose the versions to compare.<br><br>**c.** For result details, see Comparison of Policy Versions: Policy Diff, on page 139. |

| | |
|---|---|
| Delete an unwanted version: | Click ⋮ for the version and choose **Delete**. You cannot delete the last remaining version generated by automatic policy discovery (v* version). |
| Export a version: | Click ⋮ for the version and choose **Export...**. |

### What to do next

**Important** If you want to preserve previous versions of the discovered policies, always display the current version of the discovered policies when you are done working with older versions.

If the most current version of the discovered policies is not displayed the next time you discover policies for this workspace, older versions may be deleted.

For example, if the most current version of discovered policies is v4, and v2 is displayed when you discover policies again, then the existing v3 and v4 will be deleted and the new discovered policy version will be v3.

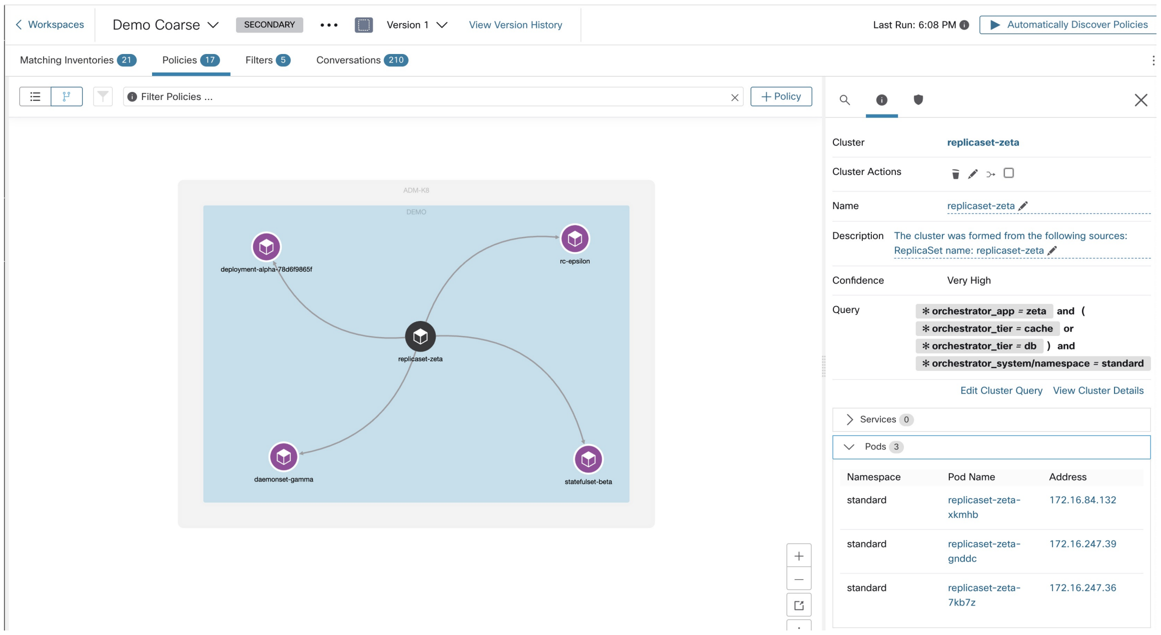
This behavior ensures a linear version history, which simplifies reverting to a previous version if desired.

In addition, you can manually create policies only if the latest v* version is displayed.

## Policy Discovery Kubernetes Support

Policy discovery uses the information on pods and services from Kubernetes configuration to create clusters for both pods and services and the respective policies are generated.

If the Cluster Granularity is set to COARSE or VERY COARSE, then the services and the pods backing them is clustered together.



| | |
|---|---|
| Cluster | replicaset-zeta |
| Cluster Actions | |
| Name | replicaset-zeta |
| Description | The cluster was formed from the following sources: ReplicaSet name: replicaset-zeta |
| Confidence | Very High |
| Query | ∗ orchestrator_app = zeta and ( ∗ orchestrator_tier = cache or ∗ orchestrator_tier = db ) and ∗ orchestrator_system/namespace = standard |

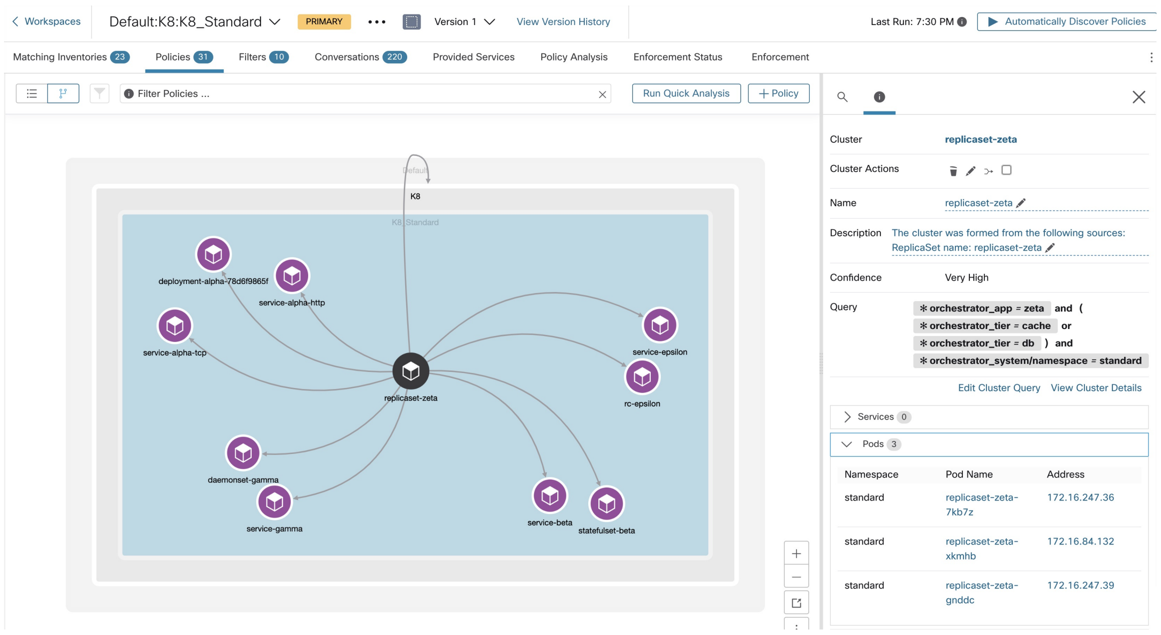
Edit Cluster Query View Cluster Details

Services 0

Pods 3

| Namespace | Pod Name | Address |
|---|---|---|
| standard | replicaset-zeta-xkmhb | 172.16.84.132 |
| standard | replicaset-zeta-gnddc | 172.16.247.39 |
| standard | replicaset-zeta-7kb7z | 172.16.247.36 |

If the Cluster Granularity is set to Medium or Fine or very fine, then the services, and the pods backing them is clustered separately.



| | |
|---|---|
| Cluster | replicaset-zeta |
| Cluster Actions | |
| Name | replicaset-zeta |
| Description | The cluster was formed from the following sources: ReplicaSet name: replicaset-zeta |
| Confidence | Very High |
| Query | ∗ orchestrator_app = zeta and ( ∗ orchestrator_tier = cache or ∗ orchestrator_tier = db ) and ∗ orchestrator_system/namespace = standard |

Edit Cluster Query View Cluster Details

Services 0

Pods 3

| Namespace | Pod Name | Address |
|---|---|---|
| standard | replicaset-zeta-7kb7z | 172.16.247.36 |
| standard | replicaset-zeta-xkmhb | 172.16.84.132 |
| standard | replicaset-zeta-gnddc | 172.16.247.39 |

For pod clusters, the source information is added as part of the cluster description and each cluster in the description contain the information of which entity has caused the cluster to be formed.

For example, **Description**: “*The cluster was formed from the following sources: ReplicaSet name: replicaset-zeta*”.

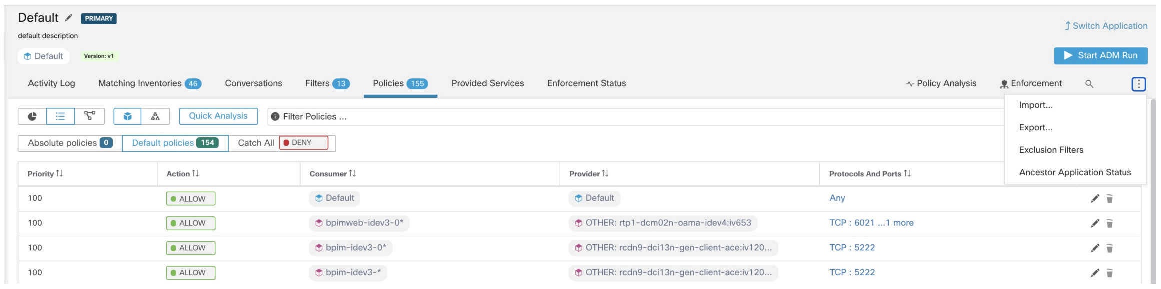
# Import/Export

## Export a Workspace

All the relevant contents of clusters and policies in each workspace can be downloaded as a single file in a number of popular structured document formats like JSON, XML and YAML. One can use such files for further in-house processing or ingestion by other policy enforcement or analysis tools.

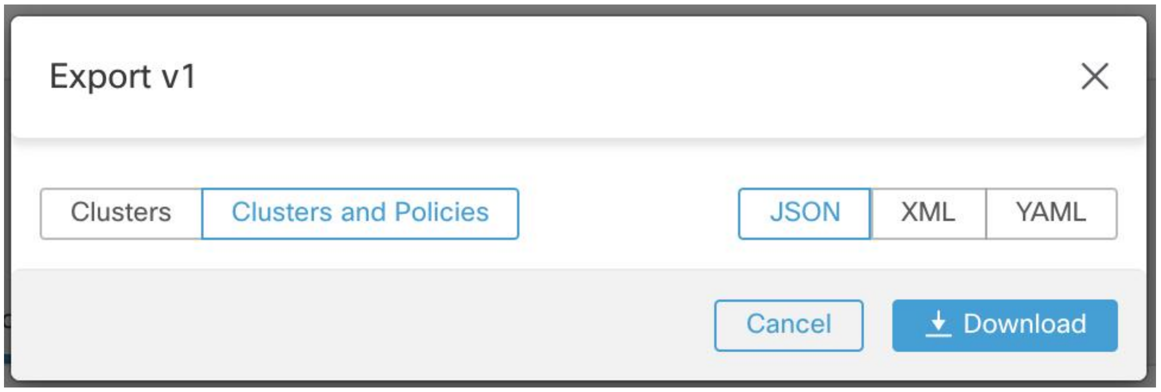
Navigate to the **. . .** menu item on the workspace header and click on the **export** item. This will show the export dialog. You can choose whether the exported file should include only the cluster contents or cluster contents as well as the security policies among the clusters generated by automatic policy discovery based on real network flows. Choose the desired format and click download to download the file into the local file system.

***Figure 24: Import/Export menu items***



| Priority | Action | Consumer | Provider | Protocols And Ports |
|---|---|---|---|---|
| 100 | ALLOW | Default | Default | Any |
| 100 | ALLOW | bpimweb-idev3-0* | OTHER: rtp1-dcm02n-oama-idev4:iv653 | TCP : 6021 ...1 more |
| 100 | ALLOW | bpim-idev3-0* | OTHER: rcdn9-dci13n-gen-client-ace:iv120... | TCP : 5222 |
| 100 | ALLOW | bpim-idev3-* | OTHER: rcdn9-dci13n-gen-client-ace:iv120... | TCP : 5222 |

***Figure 25: Exporting Policies of a workspace***



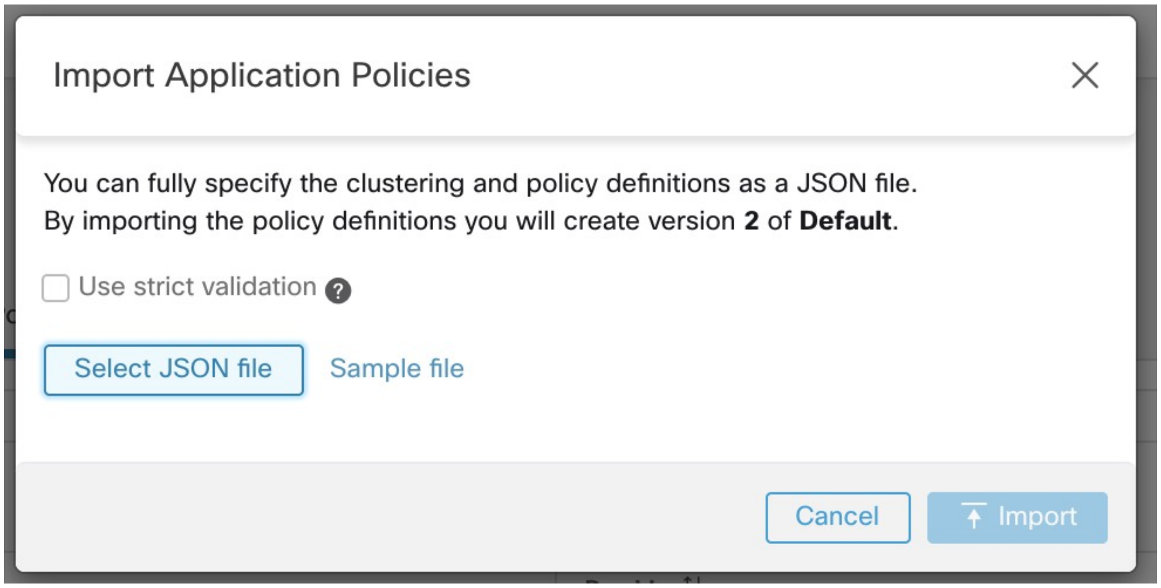
When you export a workspace, the "Auto accept outgoing policy connectors" setting in the automatic policy discovery configuration is included and will be active in the imported workspace.

## Import

You can import known cluster and policy definitions into a workspace by directly uploading a JSON file. Similar to automatic policy discovery, uploading policies into an existing workspace creates a new version and places the cluster and policy definitions under the new version. Missing filters and incorrect property values will return an error.

Click on the **Import** menu item from the **. . .** menu in the workspace header. In the import dialog, you can select a JSON file with a valid format. A small sample JSON file demonstrating the schema for policies and clusters can be found by clicking on the **Sample** button.

*Figure 26: Importing Clusters/Policies*


Import Application Policies

You can fully specify the clustering and policy definitions as a JSON file.
By importing the policy definitions you will create version **2** of **Default**.

**Strict Validation** if enabled, will return an error if the JSON contains unrecognized attributes. This is useful for locating typos or incorrectly identified optional fields.

**Note** All imported policies are marked as approved by default unless explicitly marked as `approved: false`. You have the option to maintain such approved policies during automatic policy discovery to generate a new set of policies. See Approved Policies, on page 47 for more info.

**Pro Tip**: The schema of the JSON file retrieved by exporting an application workspace is schema-compatible with the expected format for importing policies into a workspace. Therefore, you can clone policies from one application workspace to another using an export followed by an import. Note that many features may not work the same when exporting and then importing policies. For example, the conversations backing the policies are not included in the export and will not be present when importing the policies either.

## Platform-Specific Policies

For important details about how agents enforce policy on each platform, see Policy Enforcement with Agents. For Kubernetes/OpenShift, see Enforcement on Containers, on page 129.

### Windows

#### Recommended Windows OS-Based Policy Configuration

Always specify ports and protocols in policies when possible; we recommend not to allow ANY port, ANY protocol.

For example, a generated policy with port and protocol restrictions might look like this:

```
dst_ports {
      start_port: 22
      end_port: 22
      consumer_filters {
        application_name: "c:\\test\\putty.exe"
      }
     }}
     ip_protocol: TCP
```

In contrast, if you allow network connections that are initiated by iperf.exe with ANY protocol and ANY port, the generated policy looks like this:

```
  match_set {
    dst_ports {
      end_port: 65535
      consumer_filters {
        application_name: "c:\\test\\iperf.exe"
      }
    }
    address_family: IPv4
    inspection_point: EGRESS
    match_comment: "PolicyId=61008290755f027a92291b9d:61005f90497d4f47cedacb86:"
  }
```

For the above filter, Secure Workload creates a policy rule to allow the network traffic on the provider as follows:

```
  match_set {
      dst_ports {
      end_port: 65535
    }
    address_family: IPv4
    inspection_point: INGRESS
    match_comment: "PolicyId=61008290755f027a92291b9d:61005f90497d4f47cedacb86:"
  }
```

This network rule opens all the ports on the Provider. We strongly recommend not to create OS-based filters with *Any* protocol.

## Configure Policies for Windows Attributes

For more granularity when enforcing a policy on Windows-based workloads, you can filter network traffic by:

- Application Name
- Service Name
- User Names with or without User Groups

This option is supported in both WAF and WFP modes. Windows OS-based filters are categorized as *consumer filters* and *provider filters* in the generated network policy. The Consumer filters filter the network traffic that is initiated on the consumer workload and Provider filters filter the network traffic that is destined for the provider workload.

### Before you begin

This procedure assumes you are modifying an existing policy. If you have not yet created the policy to which you want to add a Windows OS-based filter, create that policy first.

**Important** See Caveats and Known limitations for policies involving Windows attributes.

### Procedure

**Step 1** In the navigation pane, click **Defend** > **Segmentation**.

**Step 2** Click the scope that contains the policy for which you want to configure Windows OS-based filters.

**Step 3** Click the workspace in which you want to edit the policy.

**Step 4** Click **Manage Policies**.

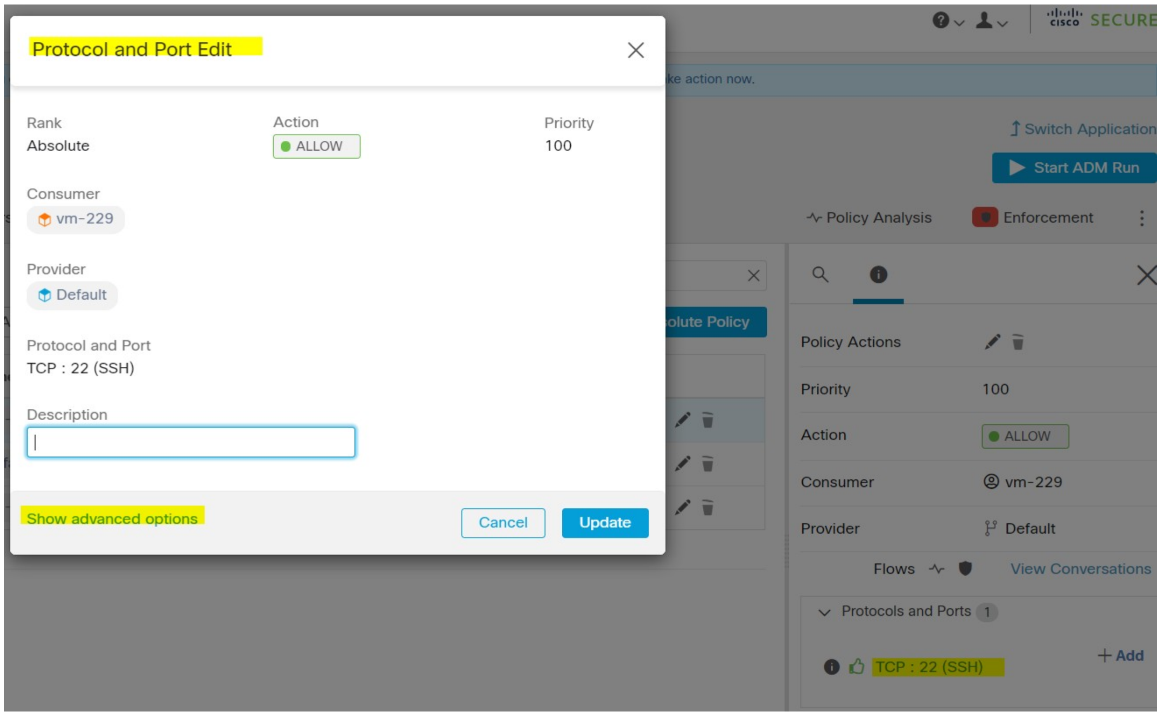
**Step 5** Choose the policy to edit.

**Important** Consumer and Provider must include only Windows workloads.

**Step 6** In the table row for the policy to edit, click the existing value in the **Protocols and Ports** column.

**Step 7** In the pane on the right, click the existing value under **Protocols and Ports**.

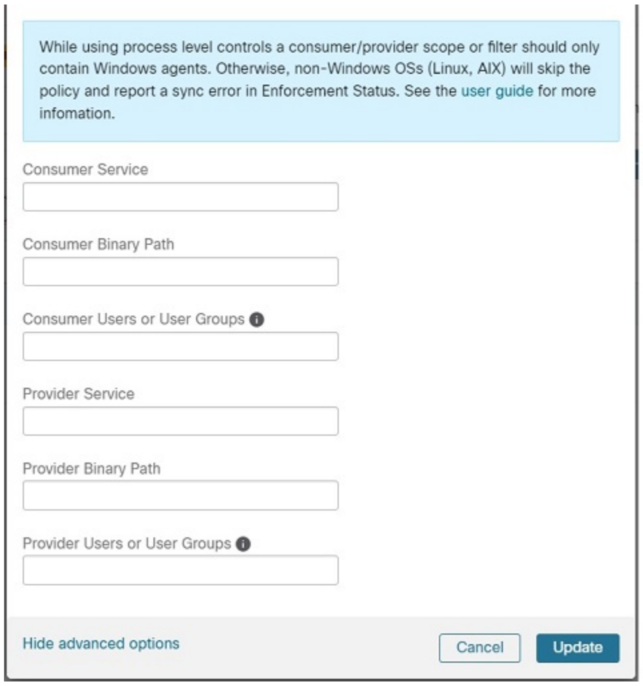
In the example, click `TCP : 22 (SSH)`.



Protocol and Port Edit

Rank: Absolute
Action: ALLOW
Priority: 100
Consumer: vm-229
Provider: Default
Protocol and Port: TCP : 22 (SSH)
Description:

**Step 8** Click **Show advanced options**.



While using process level controls a consumer/provider scope or filter should only contain Windows agents. Otherwise, non-Windows OSs (Linux, AIX) will skip the policy and report a sync error in Enforcement Status. See the user guide for more infomation.

Consumer Service

Consumer Binary Path

Consumer Users or User Groups

Provider Service

Provider Binary Path

Provider Users or User Groups

**Step 9** Configure consumer filters based on Application name, Service name, or User name.

- The application name must be a full pathname.
- Service name must be a short service name.
- User name can be a local user name (For example, tetter) or domain user name (For example, sensor-dev@sensor-dev.com or sensor-dev\sensor-dev)
- User group can be local user group (For example, Administrators) or domain user group (For example, domain users\\sensor-dev)
- Multiple user names and/ or user group names can be specified, separated by ",".(For example, sensor-dev\@sensor-dev.com,domain users\\sensor-dev)
- Service name and User name cannot be configured together.

**Step 10** Configure provider filters based on Application name, Service name, or User name.

Follow the same guidelines as given for consumer filters in the previous step.

**Step 11** Enter the paths to the binary, as applicable.

For example, enter **`c:\test\putty.exe`**

**Step 12** Click **Update**.

## *Known limitations*

- Windows 2008 R2 does not support Windows OS based filtering policies.

- Network policy can be configured with a single user name whereas MS Firewall UI supports multiple users.

#### *Caveats*

- While using the Windows OS-based policies, a consumer/ provider scope or filter should only contain Windows agents. Otherwise, non-Windows OSs (Linux, AIX) skip the policy and report a sync error in Enforcement Status.
- Avoid creating Windows OS filters with *loose* filtering criteria. Such criteria may open unwanted network ports.
- If OS filters are configured for consumer, then the policies are applicable only to consumer, similarly if it is configured for provider then it is applicable only to provider.
- Due to limited or no knowledge of the process context, user context or service context of the network flows, there will be discrepancy in the policy analysis if the policies have Windows OS-based filters.

#### *Verify and Troubleshoot Policies with Windows OS-Based Filtering Attributes*

If you use Windows OS-based filtering attributes, the following topics provide you with verification and troubleshooting information.

Cisco TAC can use this information as needed to troubleshoot such policies.

##### Policies Based on Application Name

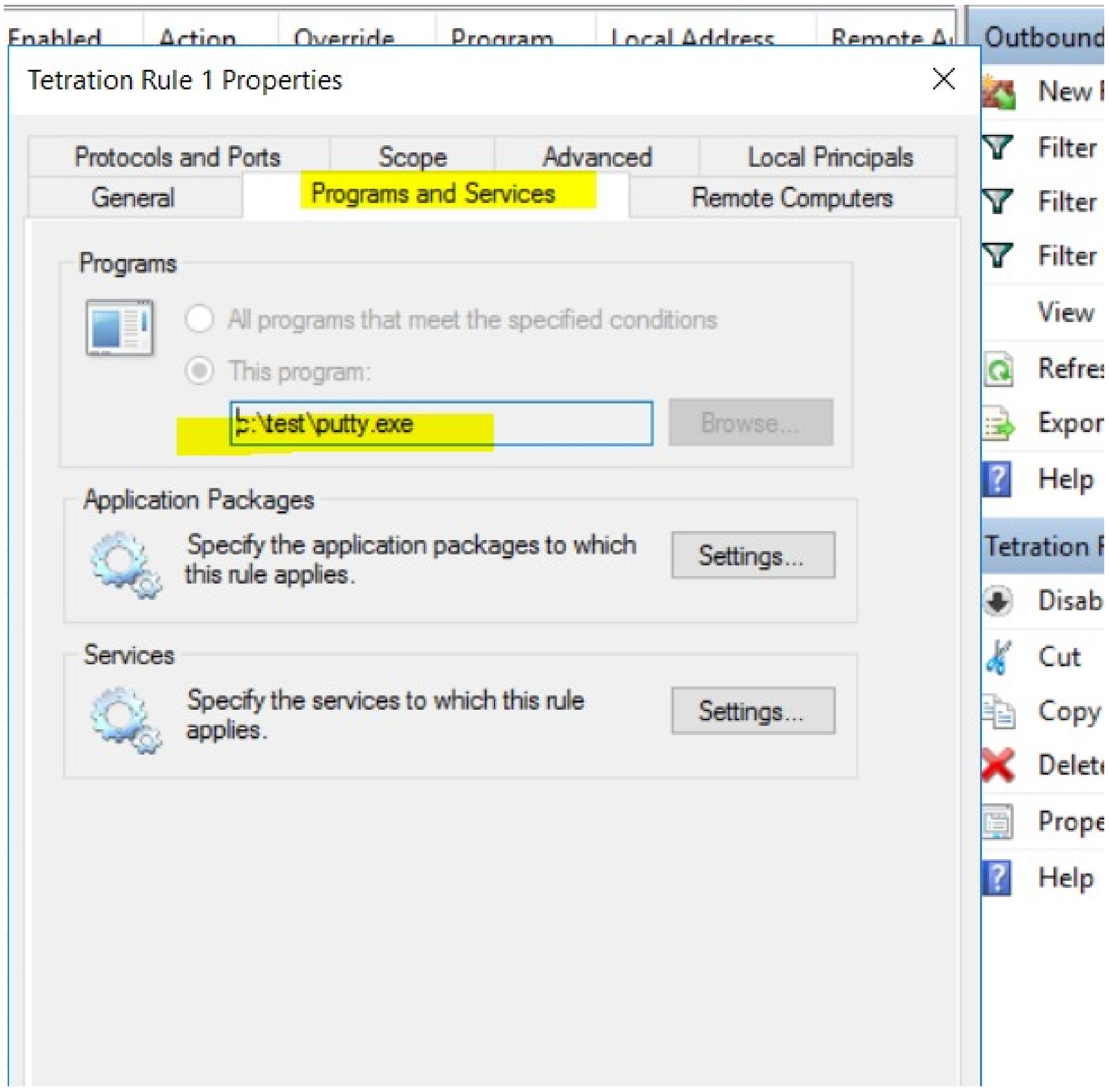
Use the following information to verify and troubleshoot policies based on application name on Windows OS workloads.

The following sections describe the way policies should appear on the workload for an application binary entered as **`c:\test\putty.exe`**.

**Sample Policy Based on Application Name**

```
dst_ports {
start_port: 22
end_port: 22
consumer_filters {
application_name: “c:\test\putty.exe”
}
}}
ip_protocol: TCP
address_family: IPv4
inspection_point: EGRESS
```

### Generated Firewall Rule



### Generated Filter Using netsh

To verify, using native Windows tools, that a filter has been added to an advanced policy:

- With administrative privileges, run `cmd.exe`.
- Run `netsh wfp show filters`.
- The output file, **filters.xml**, is generated in the current directory.
- Check FWPM_CONDITION_ALE_APP_ID for the application name in the output file: filters.xml.

```
<fieldKey>FWPM_CONDITION_ALE_APP_ID</fieldKey>
                     <matchType>FWP_MATCH_EQUAL</matchType>
                     <conditionValue>
                            <type>FWP_BYTE_BLOB_TYPE</type>
```

```
                                 <byteBlob>
                                        <data>
↪→5c006400650076006900630065005c006800610072006400640069007300 6b0076006f006
↪→</data>
                                       <asString>\device\harddiskvolume2\temp\putty.exe</
↪→asString>
                  </byteBlob>
          </conditionValue>
```

### Generated WFP Filter Using tetenf.exe -l -f

```
Filter Name:                   Secure Workload Rule 1
------------------------------------------------------
EffectiveWeight:               18446744073709551592
LayerKey:                      FWPM_LAYER_ALE_AUTH_CONNECT_V4
Action:                        Permit
RemoteIP:                      10.195.210.15-10.195.210.15
Remote Port:                   22
Protocol:                      6
AppID:                         \device\harddiskvolume2\test\putty.exe
```

### Invalid Application Name

- In WAF mode, Firewall rule is created for an invalid application name.
- In WFP mode, the WFP filter is not created for an invalid application name but the NPC is not rejected. The agent logs a warning message and configures the rest of the policy rules.

## Policies Based on Service Name

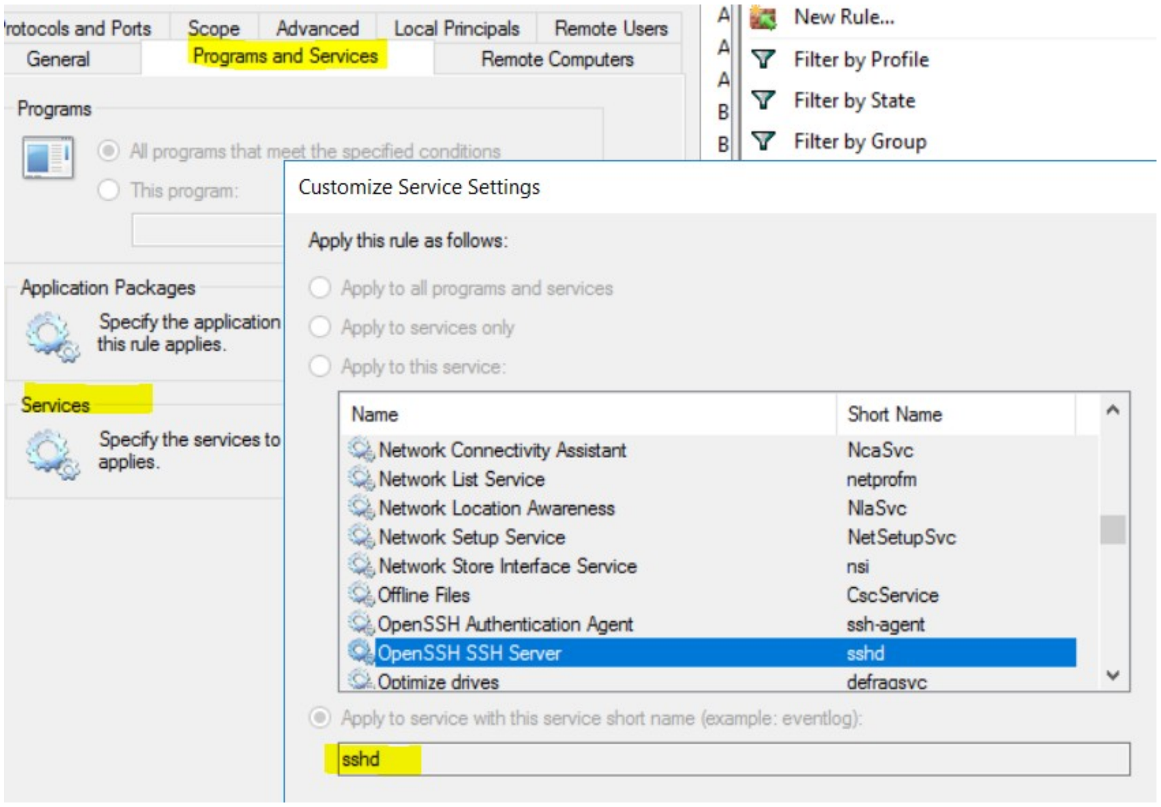
Use the following information to verify and troubleshoot policies based on Service name on Windows OS workloads.

The following sections describe the way that the policies should appear on the workload.

### Sample Policy Based on Service Name

```
    dst_ports {
             start_port: 22
             end_port: 22
             provider_filters {
                    service_name: “sshd”
             }
         }}
         ip_protocol: TCP
         address_family: IPv4
         inspection_point: INGRESS
```

### Generated Firewall Rule



Customize Service Settings

Apply this rule as follows:

- Apply to all programs and services
- Apply to services only
- Apply to this service:

| Name | Short Name |
|---|---|
| Network Connectivity Assistant | NcaSvc |
| Network List Service | netprofm |
| Network Location Awareness | NlaSvc |
| Network Setup Service | NetSetupSvc |
| Network Store Interface Service | nsi |
| Offline Files | CscService |
| OpenSSH Authentication Agent | ssh-agent |
| OpenSSH SSH Server | sshd |
| Optimize drives | defragsvc |

[✓ Apply to service with this service short name (example: eventlog):]

sshd

### Generated Filter Using netsh

To verify using native Windows tools, that a filter has been added for an advanced policy:

- With administrative privileges, run `cmd.exe`.
- Run `netsh wfp show filters`.
- The output file, **filters.xml**, is generated in the current directory.
- Check FWPM_CONDITION_ALE_USER_ID for user name in the output file: filters.xml.

```
<item>
                     <fieldKey>FWPM_CONDITION_ALE_USER_ID</fieldKey>
                     <matchType>FWP_MATCH_EQUAL</matchType>
                     <conditionValue>
                                <type>FWP_SECURITY_DESCRIPTOR_TYPE</type>

<sd>O:SYG:SYD:(A;;CCRC;;;S-1-5-80-3847866527-469524349-687026318-
→516638107)</sd>
                     </conditionValue>
</item>
```

### Generated WFP Filter Using tetenf.exe -l -f

```
Filter Name:       Secure Workload Rule 3
------------------------------------------------------
EffectiveWeight:            18446744073709551590
```

```
LayerKey:                    FWPM_LAYER_ALE_AUTH_RECV_ACCEPT_V4
Action:                      Permit
Local Port:                  22
Protocol:                    6
User or Service:             NT SERVICE\sshd
```

### Invalid Service Name

- In WAF mode, the Firewall rule is created for a nonexistent service name.
- In WFP mode, the WFP filter is not created for a nonexistent service name.
- Service SID type must be *Unrestricted* or *Restricted*. If the service type is *None*, the Firewall Rule and WFP filter can be added but they have no effect.

  To verify the SID type, run the following command:

  ```
  sc qsidtype <service name>
  ```

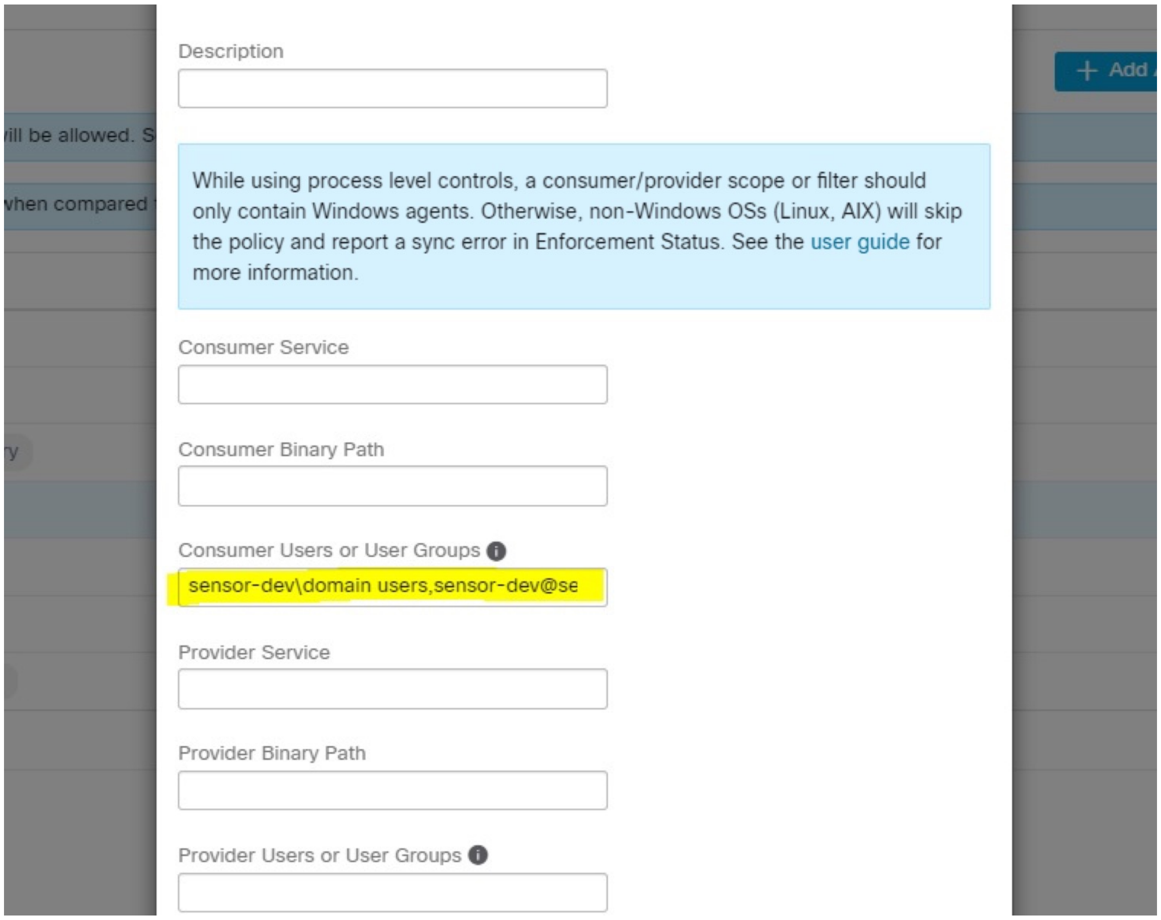
## Policies Based on User Group or User Name

Use the following information to verify and troubleshoot policies based on user name (with and without user group name) on Windows OS workloads.

Sections in this topic describe the way that the policies should appear on the workload.

Examples in this topic are based on policies that are configured with the following information:

*Figure 27: Policies Based on User Group or User Name*



Description

While using process level controls, a consumer/provider scope or filter should only contain Windows agents. Otherwise, non-Windows OSs (Linux, AIX) will skip the policy and report a sync error in Enforcement Status. See the user guide for more information.

Consumer Service

Consumer Binary Path

Consumer Users or User Groups: sensor-dev\domain users,sensor-dev@se

Provider Service

Provider Binary Path

Provider Users or User Groups

## Sample Policy Based on User Name

```
dst_ports {
          start_port: 30000
          end_port: 30000
          provider_filters {
               user_name: "sensor-dev\sensor-dev"
           }
}}
ip_protocol: TCP
address_family: IPv4
inspection_point: EGRESS
```

## Sample Policy Based on User Group and User Name

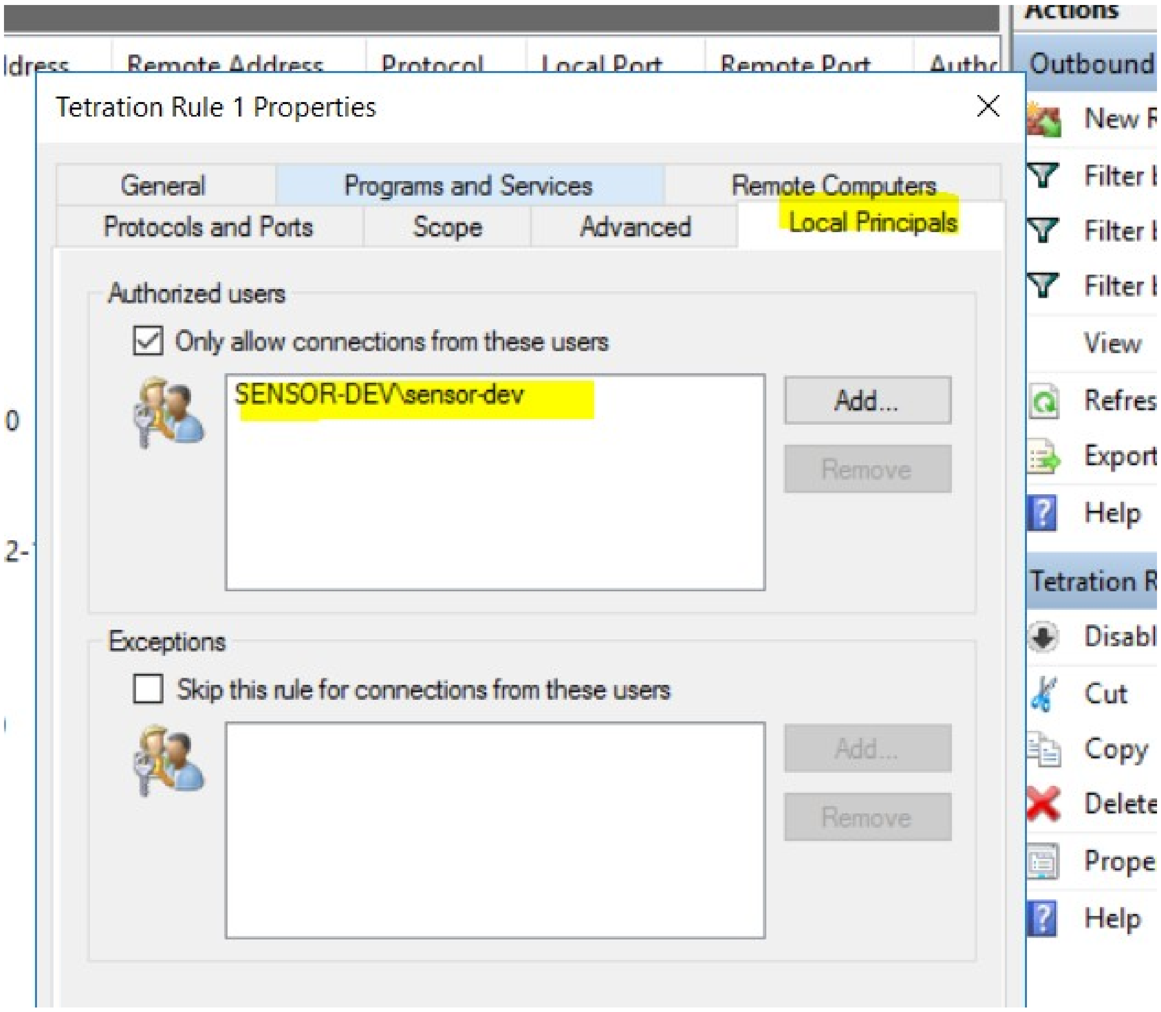
```
dst_ports {
start_port: 30000
end_port: 30000
provider_filters {
user_name: "sensor-dev\domain users,sensor-dev\sensor-dev"
}
}}
ip_protocol: TCP
```

```
address_family: IPv4
inspection_point: EGRESS
```

## Generated Firewall Rule

### Firewall Rule Based on User Name

Example: Firewall rule based on User Name, sensor-dev\\sensor-dev



Tetration Rule 1 Properties

Local Principals

Authorized users

☑ Only allow connections from these users

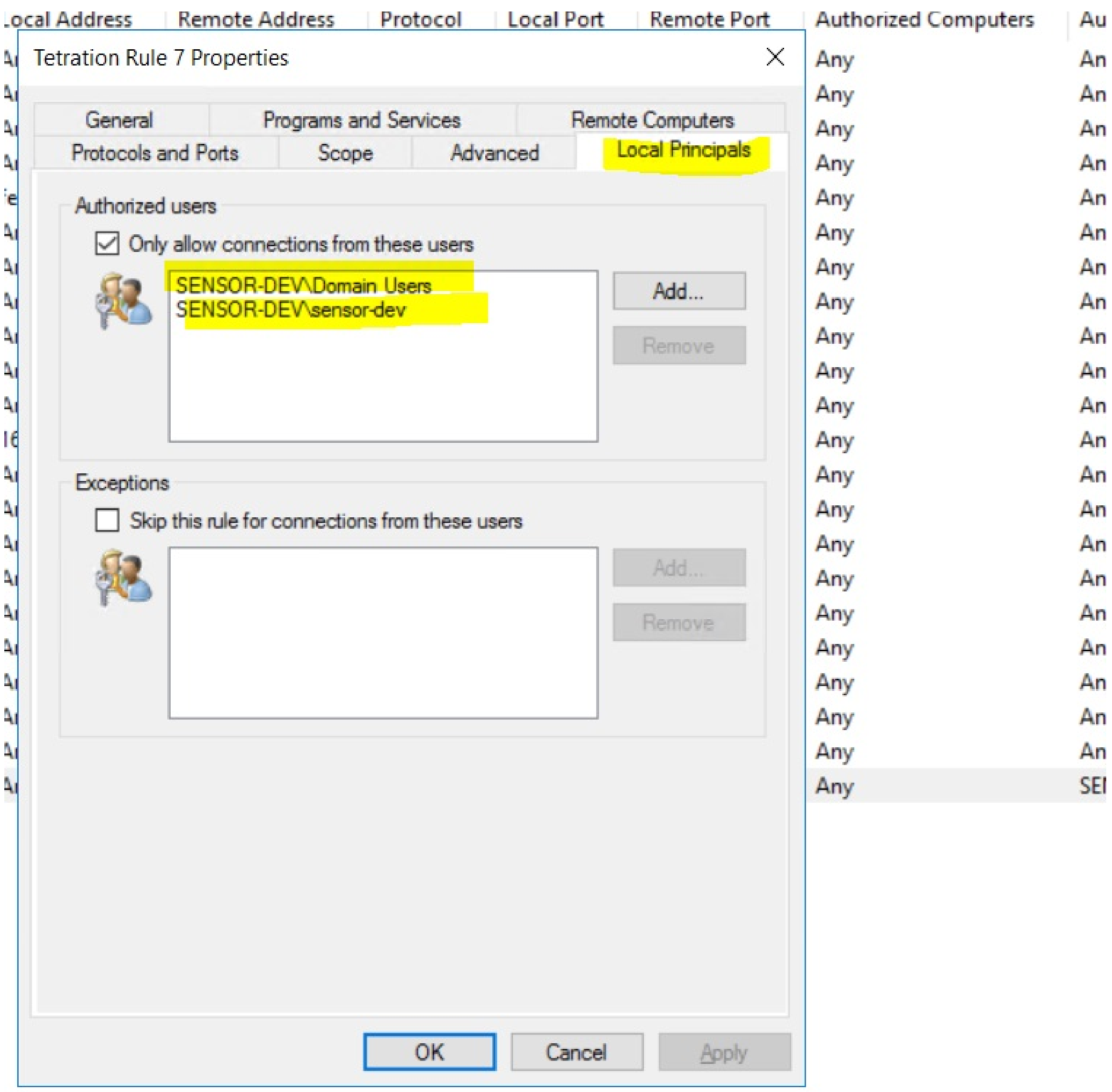
SENSOR-DEV\sensor-dev

Exceptions

☐ Skip this rule for connections from these users

### Firewall Rule Based on User Group and User Name

Example: Firewall rule based on User Name, sensor-dev\\sensor-dev and user group, domain users\\sensor-dev



Authorized users

☑ Only allow connections from these users

SENSOR-DEV\Domain Users
SENSOR-DEV\sensor-dev

Exceptions

☐ Skip this rule for connections from these users

## Generated Filter Using netsh

To verify using native Windows tools that a filter has been added for an advanced policy:

- With administrative privileges, run `cmd.exe`.
- Run `netsh wfp show filters`.
- The output file, **filters.xml**, is generated in the current directory.
- Check FWPM_CONDITION_ALE_USER_ID for user name in the output file: filters.xml.

```
<item>
            <fieldKey>FWPM_CONDITION_ALE_USER_ID</fieldKey>
            <matchType>FWP_MATCH_EQUAL</matchType>
            <conditionValue>
                   <type>FWP_SECURITY_DESCRIPTOR_TYPE</type>
```

```
                        <sd>O:LSD:(A;;CC;;;S-1-5-21-4172447896-825920244-2358685150)</sd>
                </conditionValue>
    </item>
```

### Generated WFP Filters Using tetenf.exe -l -f

**Filter based on User Name**

Example: WFP Rule based on User Name, SENSOR-DEV\sensor-dev

```
Filter Name:                   Secure Workload Rule 1
-------------------------------------------------------
EffectiveWeight:               18446744073709551590
LayerKey:                      FWPM_LAYER_ALE_AUTH_CONNECT_V4
Action:                        Permit
RemoteIP:                      10.195.210.15-10.195.210.15
Remote Port:                   30000
Protocol:                      6
User or Service:               SENSOR-DEV\sensor-dev
```

**Filter based on User Group and User Name**

Example: WFP Rule based on User Name, SENSOR-DEV\\sensor-dev and User Group name, SENSOR-DEV\\Domain Users

```
Filter Name:         Secure Workload Rule 1
-------------------------------------------------------
EffectiveWeight:             18446744073709551590
LayerKey:                    FWPM_LAYER_ALE_AUTH_CONNECT_V4
Action:                      Permit
RemoteIP:                    10.195.210.15-10.195.210.15
Remote Port:                 30000
Protocol:                    6
User or Service:             SENSOR-DEV\Domain Users, SENSOR-DEV\sensor-dev
```

*Service name and user name cannot be configured for a Network policy rule.*

**Note** The network policy is rejected by the Windows agent if the user name or the user group is invalid.

## Kubernetes and OpenShift

### (Optional) Additional Policies for Kubernetes Workloads

The following procedures are optional, depending on your Kubernetes environment.

#### *Policies for Kubernetes Nginx Ingress Controller Running in Host-network Mode*

Secure Workload enforces policies both at the nginx ingress controller and at the backend pods when the pods are exposed to the external clients using Kubernetes ingress object.

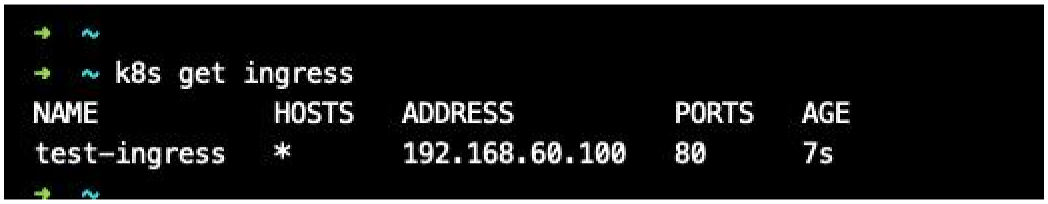
**Note** If the ingress controller is not running in host network mode refer IngressControllerAPI

**Note** IBM-ICP uses Kubernetes Nginx Ingress controller by default and runs on control plane nodes in host network mode.

Following are the steps to enforce the policy using the Kubernetes Nginx Ingress controller.

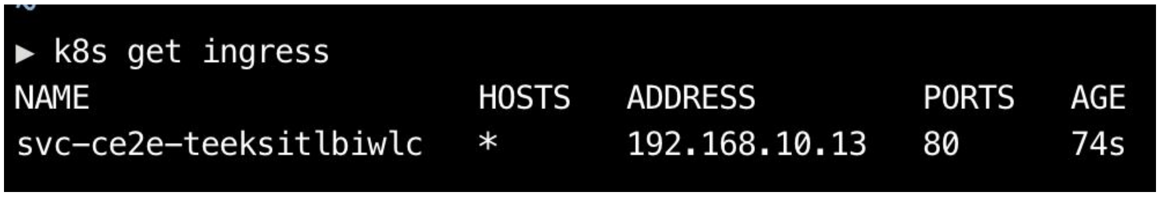
**Procedure**

**Step 1** Create an external orchestrator for Kubernetes/OpenShift as described here.

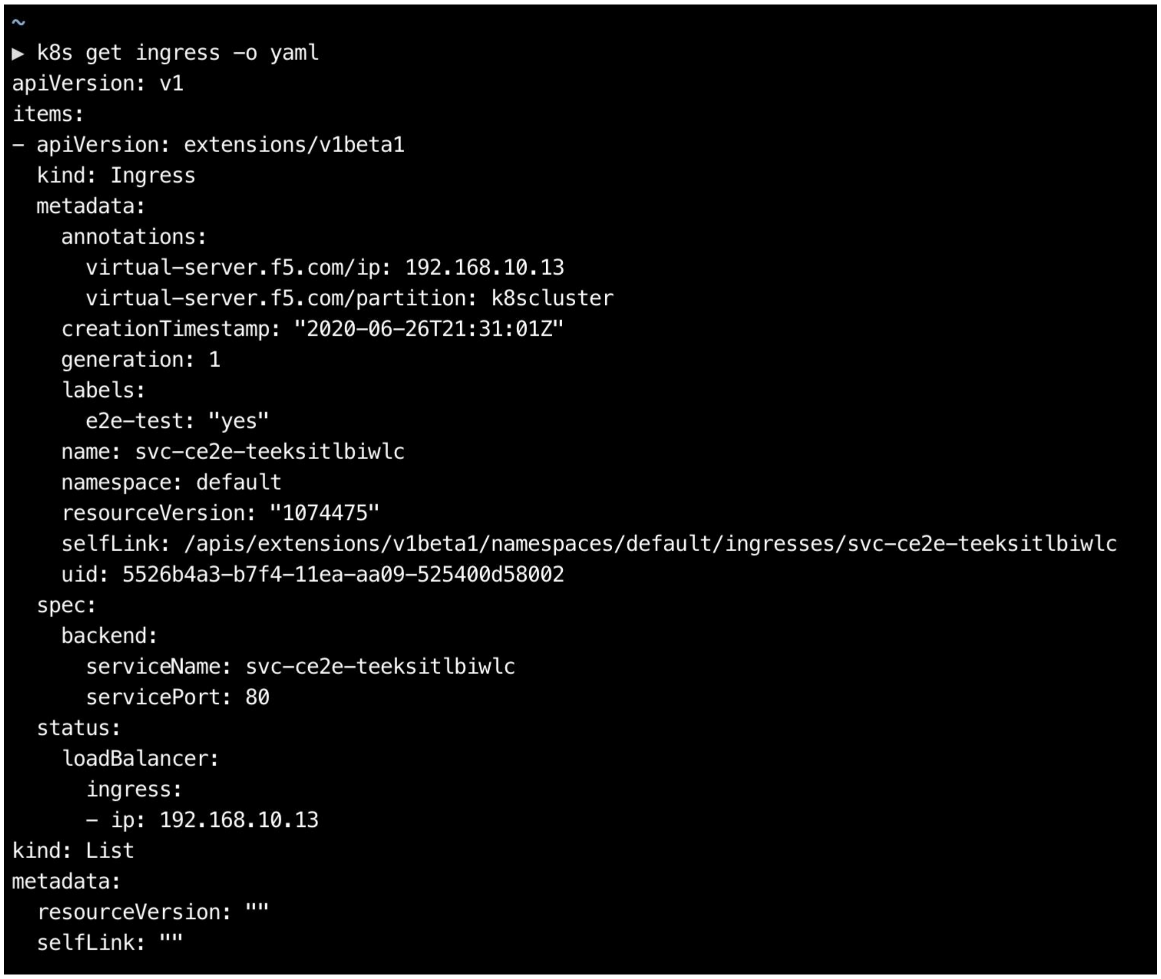


```
→  ~
→  ~ k8s get ingress
NAME           HOSTS   ADDRESS          PORTS   AGE
test-ingress   *       192.168.60.100   80      7s
→  ~
```

**Step 2** Create an ingress object in the Kubernetes cluster. A snapshot of the yaml file used to create the ingress object is provided in the following picture.

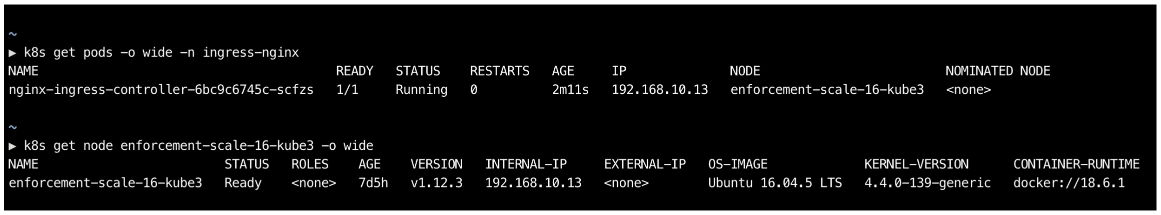


```
▶ k8s get ingress
NAME                     HOSTS   ADDRESS         PORTS   AGE
svc-ce2e-teeksitlbiwlc   *       192.168.10.13   80      74s
```



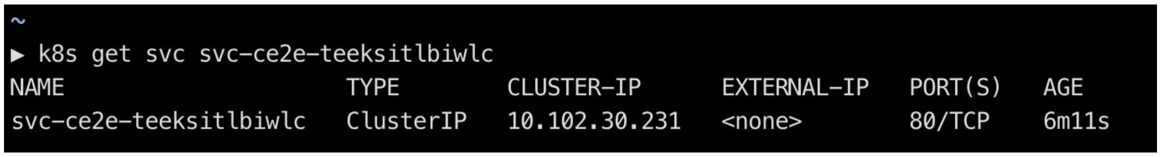
```
~
▶ k8s get ingress -o yaml
apiVersion: v1
items:
- apiVersion: extensions/v1beta1
  kind: Ingress
  metadata:
    annotations:
      virtual-server.f5.com/ip: 192.168.10.13
      virtual-server.f5.com/partition: k8scluster
    creationTimestamp: "2020-06-26T21:31:01Z"
    generation: 1
    labels:
      e2e-test: "yes"
    name: svc-ce2e-teeksitlbiwlc
    namespace: default
    resourceVersion: "1074475"
    selfLink: /apis/extensions/v1beta1/namespaces/default/ingresses/svc-ce2e-teeksitlbiwlc
    uid: 5526b4a3-b7f4-11ea-aa09-525400d58002
  spec:
    backend:
      serviceName: svc-ce2e-teeksitlbiwlc
      servicePort: 80
  status:
    loadBalancer:
      ingress:
      - ip: 192.168.10.13
kind: List
metadata:
  resourceVersion: ""
  selfLink: ""
```

**Step 3** Deploy Kubernetes Nginx Ingress controller in the Kubernetes cluster. IBM-ICP Ingress controller pods are running on control plane nodes by default.



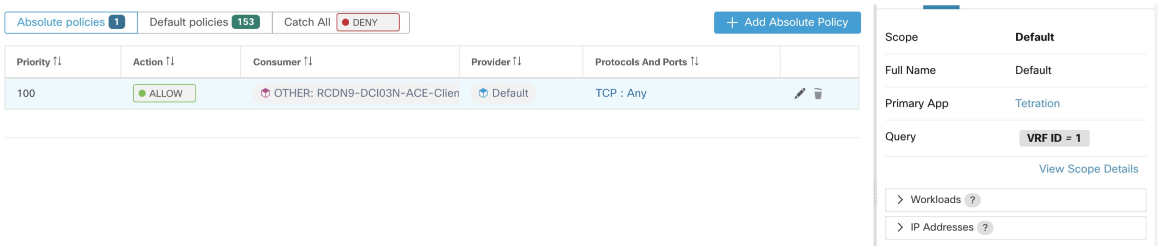
```
~
▶ k8s get pods -o wide -n ingress-nginx
NAME                                       READY   STATUS    RESTARTS   AGE     IP              NODE                          NOMINATED NODE
nginx-ingress-controller-6bc9c6745c-scfzs   1/1     Running   0          2m11s   192.168.10.13   enforcement-scale-16-kube3    <none>

~
▶ k8s get node enforcement-scale-16-kube3 -o wide
NAME                          STATUS   ROLES    AGE    VERSION   INTERNAL-IP     EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION      CONTAINER-RUNTIME
enforcement-scale-16-kube3    Ready    <none>   7d5h   v1.12.3   192.168.10.13   <none>        Ubuntu 16.04.5 LTS   4.4.0-139-generic   docker://18.6.1
```

**Step 4** Create a backend service which will be accessed by the consumers outside the cluster. In the example provided below we have created a simple *svc-ce2e-teeksitlbiwlc* (http-echo) service.



```
~
▶ k8s get svc svc-ce2e-teeksitlbiwlc
NAME                     TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
svc-ce2e-teeksitlbiwlc   ClusterIP   10.102.30.231   <none>        80/TCP    6m11s
```

**Step 5** Create a policy between external consumer and backend service.



| Priority | Action | Consumer | Provider | Protocols And Ports |
|---|---|---|---|---|
| 100 | ALLOW | OTHER: RCDN9-DCI03N-ACE-Clien | Default | TCP : Any |

| | |
|---|---|
| Scope | Default |
| Full Name | Default |
| Primary App | Tetration |
| Query | VRF ID = 1 |

View Scope Details

Workloads

IP Addresses

**Step 6** When you are ready, enforce the policy.

**Step 7** In case of Nginx ingress controller Secure Workload software applies the appropriate allow/drop rule where the source will be consumer specified in the above step and destination will be corresponding Ingress controller pod IP. In case of backend pods, Secure Workload software will apply the appropriate allow/drop rule where the source will be Ingress pod and destination will be the backend pod IP.

### *Policies for Kubernetes Nginx/Haproxy Ingress controller running as Deployment/Daemonset*

Secure Workload will enforce policies both at the ingress controller and at the backend pods when the pods are exposed to the external clients using Kubernetes ingress object.

Following are the steps to enforce policies on Ingress controller.

#### Procedure

**Step 1** Create/Update an external orchestrator for Kubernetes/OpenShift using OpenAPI. See Orchestrators for information on creating the external orchestrator using OpenAPI. Add information of Ingress Controllers for External Orchestrator config.

**Step 2** Create an ingress object in the Kubernetes cluster.

**Step 3** Deploy Ingress controller in the Kubernetes cluster.

**Step 4** Create a backend service which will be accessed by the consumers outside the cluster

**Step 5** Create a policy between external consumer and backend service.

**Step 6** When you are ready, enforce the policy.

**Step 7** In case of Ingress controllers Secure Workload software will apply the appropriate allow/drop rule where the source will be consumer specified in the above step and destination will be corresponding Ingress controller pod IP. In case of backend pods, Secure Workload software will apply the appropriate allow/drop rule where the source will be Ingress pod and destination will be the backend pod IP.

## Grouping Workloads: Clusters and Inventory Filters

Clusters and inventory filters serve similar purposes, but have some important differences:

*Table 4: Comparison of Clusters and Inventory Filters*

| Clusters | Inventory Filters |
|---|---|
| Are used to apply policy to a subset of the workloads in a scope. | Can be used to apply policy to a subset of the workloads in a scope.<br><br>Can also be used to apply policy to workloads regardless of scope (for example, to apply policy to all workloads running a particular operating system.) |
| Are defined by a query | Are defined by a query. |
| Can include only workloads in a single scope. | Can have membership restricted to a single scope or include workloads in any scope (for example, if the filter is based on operating system.) |
| Can only be used by policies in the same workspace and workspace version. | Can be used by policies in any scope and any workspace. |
| Can be automatically created during automatic policy discovery. | Must be manually created or converted from an existing cluster. |
| Can be overwritten during automatic policy discovery if not approved. Approving known good clusters can improve accuracy of other clusters in future discovery runs. | Are never modified by automatic policy discovery. |
| Benefit from important features of automatic policy discovery. They:<br>• Have a confidence rating that helps you evaluate whether the workloads in the group belong together.<br>• Can be compared with clusters generated during other policy discovery runs on the same workspace. | -- |
| Cannot be used when configuring External Dependencies, on page 31 and other features related to cross-scope policies and policy discovery. | Can be used to configure granular policies involving external dependencies and other features related to cross-scope policies, such as auto-pilot rules. |
| See Clusters, on page 72 and subtopics. | See Create an Inventory Filter and Convert a Cluster to an Inventory Filter, on page 77 |

## Clusters

A cluster is a set of workloads that are grouped together within a workspace. (A Secure Workload deployment can also be called a cluster, but the two usages are unrelated.)

For example, if your application scope includes several web servers among many other types of servers and hosts that comprise your application, you might want a cluster of web servers within this application scope, so you can assign specific policies only to these web servers.

Automatic policy discovery groups workloads into clusters based on the signals observed in the timeframe that is specified during the run configuration.

**Each cluster is defined by a query**

Cluster queries are dynamic unless you define them with specific IP addresses. With dynamic queries, cluster membership can change over time to reflect changes in your inventory: More, fewer, or different workloads can match the query.

For example, if a cluster query is based on hostname containing the substring ‘HR’, and more hosts with hostname containing HR are added to the workspace, the cluster automatically includes the additional hosts.

Automatic policy discovery examines the hostnames and labels that are associated with workloads. For each cluster, automatic policy discovery generates a short list of candidate queries based on the hostnames and these labels. From these queries, you can select one, possibly edit it, and associate it with the cluster. Note that, in certain cases, when automatic policy discovery cannot formulate simple enough queries based on the hostnames and labels, no (alternate) queries are suggested.

**Workloads in approved clusters are not affected by future policy discovery**

Only workloads that are not already members of an approved cluster in the relevant workspace are affected by policy discovery. An **approved cluster** is a cluster that you have manually approved. For details, see Approving Clusters, on page 80.

**Edit clusters to improve grouping**

In the following sections, we describe a few workflows to edit, enhance, and approve the clustering results. Note that one can change/approve clusters only in the latest version of a workspace (see Activity Logs and Version History).

See Making Changes to Clusters, on page 75.

**Clusters involving Kubernetes inventory**

**Note** If your workspace includes inventory from multiple Kubernetes namespaces, each cluster query must filter by namespace. Add the namespace filter to each query if it is not already present. If you change any query, then automatically discover the policies again.

**A cluster may consist of a single workload.**

You may want to create policies involving just a single workload.

**Clusters may be converted to inventory filters**

Like approved clusters, clusters promoted to inventory filters are not changed during subsequent policy discovery.

Unlike clusters, inventory filters are not tied to a workspace, but are globally available within your Secure Workload deployment.

For a comparison of clusters and inventory filters, see Grouping Workloads: Clusters and Inventory Filters, on page 71.

See Convert a Cluster to an Inventory Filter, on page 77.

## Cluster Confidence

Use the confidence or quality score of a cluster to identify clusters needing improvement.

The confidence for a cluster is the average of the confidences for member workloads. In general, the more similar a workload is to other members of the cluster it was assigned, and the more dissimilar it is to the workloads of the closest (most similar) alternative cluster, the higher the confidence for that workload.

When flows are used for clustering, two workloads are similar when they have a similar pattern of conversations (such as similar sets of neighbors in the conversation graph, i.e., similar sets of consumer and provider workloads and ports).

**Note**

- Cluster confidence is not computed (undefined) for:
  - clusters containing only one workload
  - approved clusters
  - workloads in the scope for which no communication was observed (or no process information is available, if process-based clustering was chosen)
- Clusters do not span partition boundaries (such as subnet boundaries, see route labels in the advanced automatic policy discovery configurations). However, in computing confidence and alternate cluster, such boundaries are ignored. This indicates the potential existence of workloads or clusters that behave very similarly even though they are in different subnets.
- After editing clusters, the confidence scores may become inaccurate as they are NOT recomputed until you discover policies again.

To view cluster confidence, see View Clusters, on page 74.

## View Clusters

The clusters view supports query-to-cluster association and query editing.

In the clusters view, you can click a table column heading to sort the clusters based on that column (such as name, the number of workloads, or confidence).

For each cluster, by clicking on its row, you can view further cluster information such as description, suggested or approved queries, and the member workloads in the right panel. Several of these fields are editable.

To view clusters and details about them:

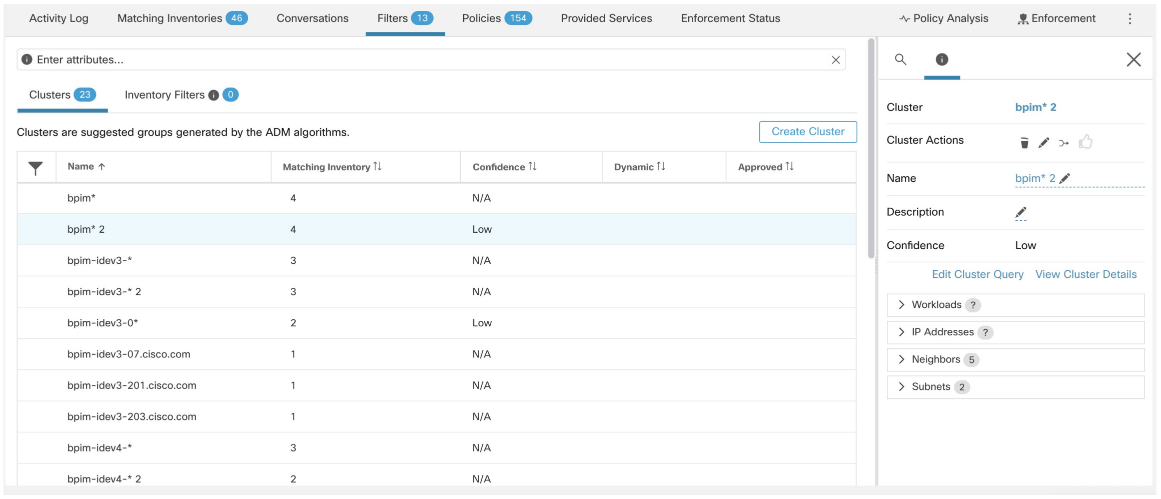
1. Navigate to the scope and workspace of interest.

   Clusters are specific to a workspace; each workspace in a scope can have different clusters. To make clusters available outside their current workspace, see Convert a Cluster to an Inventory Filter, on page 77.

2. Click **Manage Policies**.
3. Click **Filters**.
4. Click **Clusters**.
5. To view information about a cluster, click a cluster.
   a. Look in the panel that opens on the right.
   b. For more details, click **View cluster details**.

The Cluster Details page opens in a separate browser tab.

***Figure 28: Clusters View***



Activity Log | Matching Inventories 46 | Conversations | Filters 13 | Policies 154 | Provided Services | Enforcement Status | Policy Analysis | Enforcement

Enter attributes...

Clusters 23 | Inventory Filters 0

Clusters are suggested groups generated by the ADM algorithms. Create Cluster

| Name | Matching Inventory | Confidence | Dynamic | Approved |
|---|---|---|---|---|
| bpim* | 4 | N/A | | |
| bpim* 2 | 4 | Low | | |
| bpim-idev3-* | 3 | N/A | | |
| bpim-idev3-* 2 | 3 | N/A | | |
| bpim-idev3-0* | 2 | Low | | |
| bpim-idev3-07.cisco.com | 1 | N/A | | |
| bpim-idev3-201.cisco.com | 1 | N/A | | |
| bpim-idev3-203.cisco.com | 1 | N/A | | |
| bpim-idev4-* | 3 | N/A | | |
| bpim-idev4-* 2 | 2 | N/A | | |

| | |
|---|---|
| Cluster | bpim* 2 |
| Cluster Actions | |
| Name | bpim* 2 |
| Description | |
| Confidence | Low |

Edit Cluster Query View Cluster Details

Workloads ?

IP Addresses ?

Neighbors 5

Subnets 2

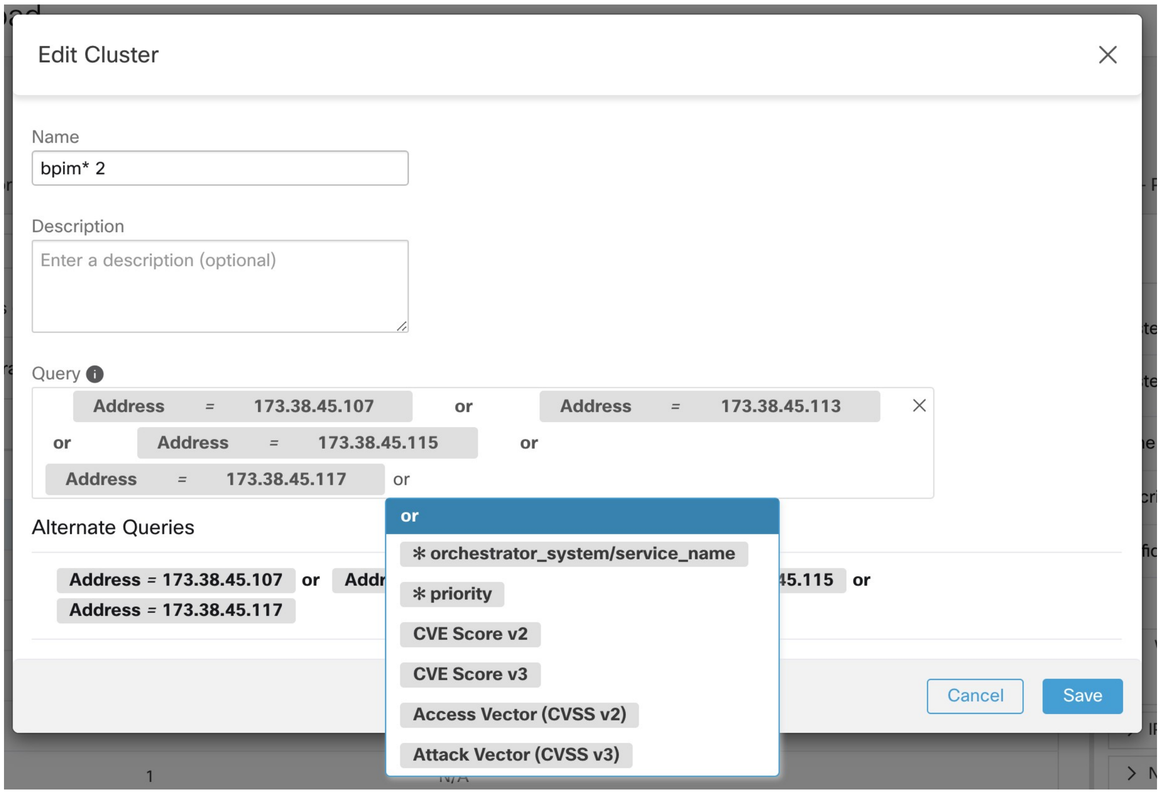
## Making Changes to Clusters

Automatic policy discovery creates one or more candidate queries for each cluster.

If clustering results do not completely match your expectations, you can improve the grouping by editing the query.

To browse and edit clusters: Click on the **clusters** box at the top of the page. To change a cluster (e.g. change the members of a cluster or select/change its query), select/edit the cluster’s query, as shown below.

*Figure 29: Edit Cluster*


Edit Cluster
Name
bpim* 2
Description
Enter a description (optional)
Query
Address = 173.38.45.107 or Address = 173.38.45.113
or Address = 173.38.45.115 or
Address = 173.38.45.117 or
or
✻ orchestrator_system/service_name
✻ priority
CVE Score v2
CVE Score v3
Access Vector (CVSS v2)
Attack Vector (CVSS v3)
Alternate Queries
Address = 173.38.45.107 or
Address = 173.38.45.117
Cancel
Save

You can add or remove explicit IP addresses, or pick another query from the list of alternatives provided and edit that query. A cluster's query can be any query filter expressed in terms of addresses, hostnames, and labels. If you define a query based on labels rather than explicit IP addresses, the cluster will be dynamic, and new, changed, or removed inventory that is properly labeled will automatically be included in or excluded from the cluster.

After query selection and possible editing is done, click save. Note that once the SAVE button is clicked, the cluster is automatically marked approved, the approved thumbs-up icon turns blue (whether or not a change was made). The approved icon can be toggled to change the approved status as desired. See details at Approving Clusters, on page 80.

**Important** When a cluster's membership is changed, it may be necessary to discover policies again to get an updated policy accurately reflecting the changes in flows among the changed clusters. This is because cluster memberships may have changed (such as new nodes added to a cluster). A similar situation can occur if the scope corresponding to the workspace is edited or in general when workspace membership changes. Similarly, cluster confidence scores may no longer be accurate with changes to cluster memberships. In all these cases, automatically discovering policies again is useful to get updated policies and cluster confidence scores (updated confidences on unapproved clusters).

If you edit cluster queries, it is possible that clusters associated with queries may overlap.

## Convert a Cluster to an Inventory Filter

Convert a cluster to an inventory filter if:

- You do not want the cluster to be modified by future automatic policy discovery runs, as a more versatile alternative to approving the cluster.
- You want the cluster to be independent of the workspace and workspace version.
- You are creating or discovering policies in which the consumer and provider belong to different scopes, and you want to create policies specific to a subset of workloads in a scope, not just policies involving the entire scope.

  You must use inventory filters instead of clusters for this purpose if you create cross-scope policies using the advanced method described in When Consumer and Provider Are in Different Scopes: Policy Options, on page 88 and you want policies to be more granular than scope-to-scope.

### Procedure

**Step 1** Navigate to the workspace that contains the cluster to promote.

**Step 2** Click **Manage Policies**.

**Step 3** Click **Filters**.

**Step 4** Click **Clusters**.

**Step 5** Click the cluster you want to use in the cross-scope policy.

**Step 6** In the panel on the right, in the **Cluster Actions** section, click ⤜ (Promote to Inventory Filter.)

**Step 7** Verify that the name, description, and query are as expected.

**Step 8** Select **Restrict Query to Ownership Scope**.

(Inventory filters can cross scope boundaries, but you do not want this behavior for this purpose; you want this filter to include only workloads in this scope.)

**Step 9** If you want the application defined by this inventory filter to be the provider in policies generated during automatic policy discovery, select **Provides a service external of its scope**.

If this application is a consumer rather than a provider, or if you will use this inventory filter only for manually created policies, you don't need to enable this option.

**Step 10** Click **Promote Cluster**.

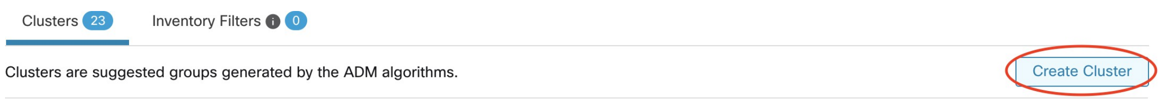
**Step 11** Verify that the cluster has moved to the **Inventory Filters** tab.

You may need to refresh the page to see this change.

## Creating or Deleting Clusters

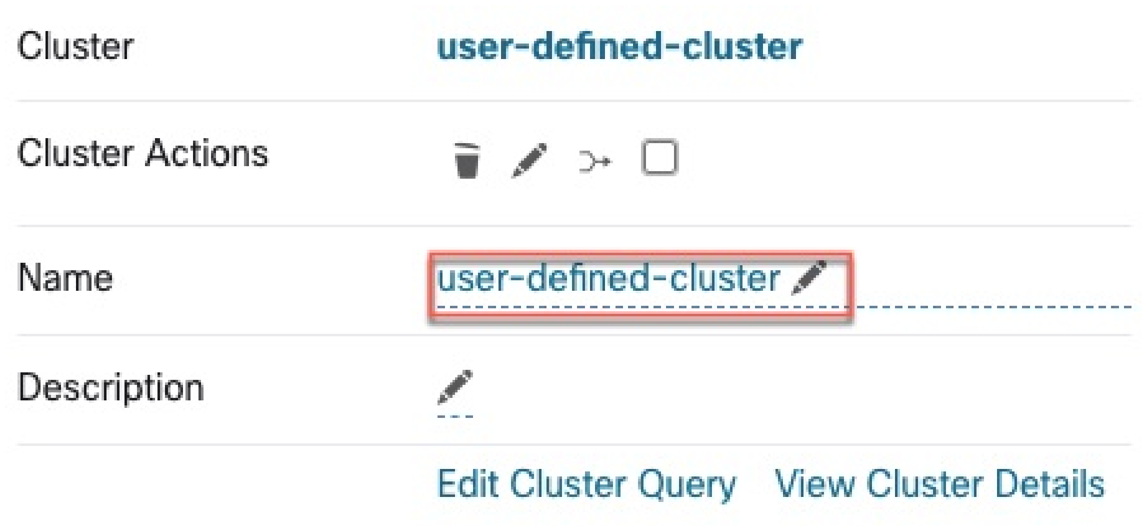
Click the **Create Cluster** button on the clusters page to create a new empty cluster. Alternatively, you can also create a cluster from the automatic policy discovery page by clicking on **Create Filter** button in Get Started sidebar and selecting Clusters in the modal.

*Figure 30: Creating a new Cluster*


Clusters 23
Inventory Filters 0
Clusters are suggested groups generated by the ADM algorithms.
Create Cluster

The new user defined cluster will show up on the side panel to be renamed, if necessary.

*Figure 31: Renaming a Cluster*



An empty cluster may be deleted by selecting the cluster in any of the views so that the details appear on the side panel and clicking the trash button on the header of cluster detail view. See figure above.

## Comparing Versions of Generated Clusters: Diff Views

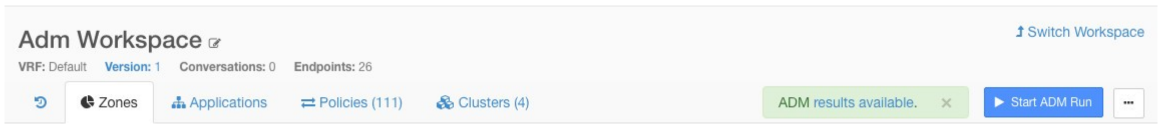
After you have automatically discovered policies at least twice for a workspace, you can compare the clusters generated in different discovery runs.

### Procedure

**Step 1** Navigate to the clusters diff view using one of the following paths:

- After successfully discovering policies, a message will appear indicating the success with a link that navigates to the diff view showing discovery results. Click the results link.

  *Figure 32: Successful automatic policy discovery run*

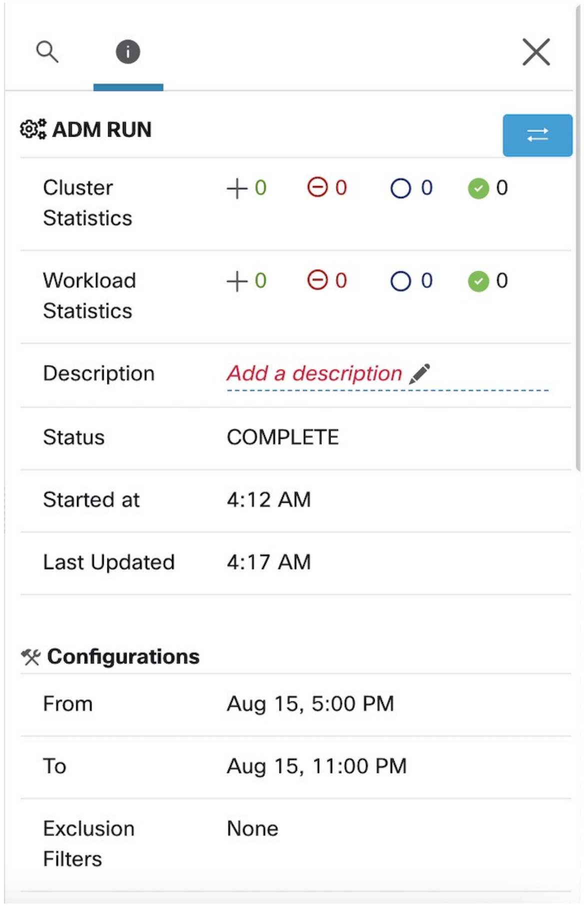
- Compare revisions from the versions view:

  **a.** Follow the steps in View, Compare, and Manage Discovered Policy Versions, on page 50.

  **b.** After you click **Compare Revisions**, click **Clusters**.

- From the version details side panel:
  a. Follow the steps to view version details in View, Compare, and Manage Discovered Policy Versions, on page 50.
  b. From the side panel, when it is showing context information for an automatic policy discovery run, click the double-arrow button on the top right corner of the side panel:

*Figure 33: Showing Context Information*



**ADM RUN**

| | |
|---|---|
| Cluster Statistics | + 0 ⊖ 0 ○ 0 ✓ 0 |
| Workload Statistics | + 0 ⊖ 0 ○ 0 ✓ 0 |
| Description | *Add a description* |
| Status | COMPLETE |
| Started at | 4:12 AM |
| Last Updated | 4:17 AM |

**Configurations**

| | |
|---|---|
| From | Aug 15, 5:00 PM |
| To | Aug 15, 11:00 PM |
| Exclusion Filters | None |

**Step 2** Choose the versions to compare.

**Step 3** Review the comparison results:

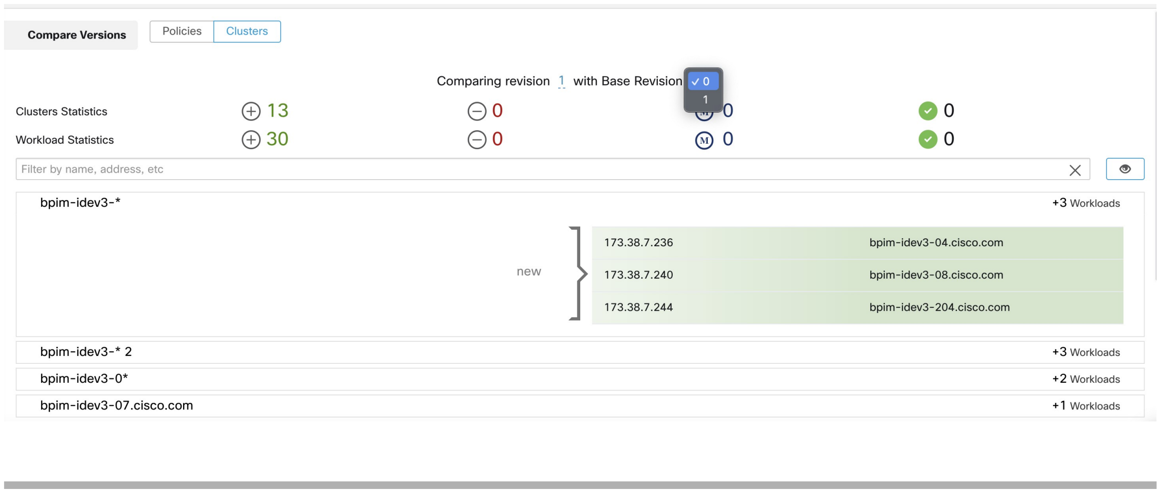
At the top level, the diff view for automatically discovered policies shows high level statistics about changes in clusters and workloads showing the number of added, deleted, modified, and unchanged clusters and workloads.

The rest of the view is organized as a list of clusters in the order of added, deleted, modified and unchanged, each color coded to reflect the status as well as the number of workloads added to or removed from the cluster.

You may search for a particular cluster or workload by name or IP address. To see how the contents of a cluster have changed, click any of the rows representing a cluster to expands that row.

**Note** By default, unchanged clusters are hidden. To display unchanged clusters, click the button with the eye icon.

*Figure 34: Cluster Diff View*



Comparing revision 1 with Base Revision 0

| | | | | |
|---|---|---|---|---|
| Clusters Statistics | ⊕ 13 | ⊖ 0 | Ⓜ 0 | ✓ 0 |
| Workload Statistics | ⊕ 30 | ⊖ 0 | Ⓜ 0 | ✓ 0 |

bpim-idev3-* +3 Workloads

new

| | |
|---|---|
| 173.38.7.236 | bpim-idev3-04.cisco.com |
| 173.38.7.240 | bpim-idev3-08.cisco.com |
| 173.38.7.244 | bpim-idev3-204.cisco.com |

bpim-idev3-* 2 +3 Workloads

bpim-idev3-0* +2 Workloads

bpim-idev3-07.cisco.com +1 Workloads

### What to do next

To view a similar comparison for policies, see Comparison of Policy Versions: Policy Diff.

## Preventing Cluster Modification During Automatic Policy Discovery Reruns

If you do not want automatic policy discovery (formerly known as ADM) to modify a cluster when you automatically discover policies for the workspace in future, approve the cluster.

For example, approve the cluster if you have edited the cluster query, and now you need to add new workloads to the scope and cluster them without affecting the existing policies. Approving the cluster freezes the cluster contents and attributes in the current state. Automatic policy discovery does not change approved clusters.

See Approving Clusters, on page 80.

Alternatively, you can promote the cluster to an inventory filter, which will never be modified by policy discovery. See Convert a Cluster to an Inventory Filter, on page 77.

## Approving Clusters

**Note** See also Convert a Cluster to an Inventory Filter, on page 77, which may be a more appropriate option for your needs.

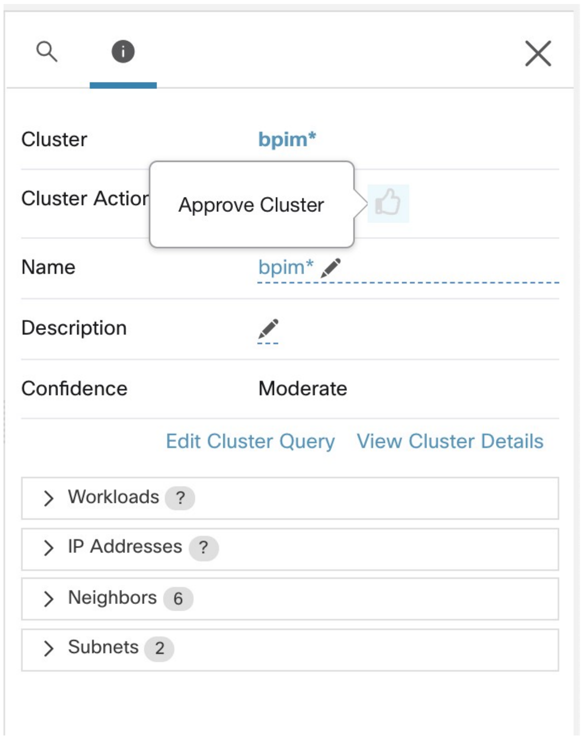
After you approve a cluster, subsequent automatic policy discovery does not change that cluster’s query. Memberships of approved clusters can change only if the members of the workspace change.

Workloads that are members of an approved cluster may be referred to as "approved workloads."

To approve a cluster:

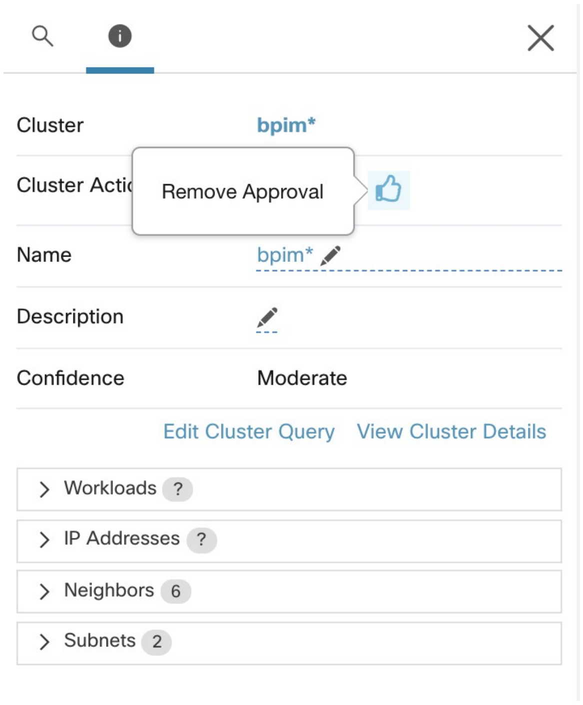
Make sure the cluster of interest is shown on the side panel. You can accomplish this via searching for the cluster, or clicking on the desired cluster on the chart in any of the views. Then select the check box on the top-right corner of the cluster info on the side panel as illustrated below. After a cluster is approved, it indicates that it will remain unchanged by future automatic policy discovery.

*Figure 35: Approving Clusters*



To remove approval of a cluster, click the approval icon.

*Figure 36: Removing Approval of a Cluster*



## Address Policy Complexities

Enforcement results are impacted by factors including:

- Rule type and rank:
  - Absolute vs Default policies
  - The catch-all setting for the workspace

  See Policy Rank: Absolute, Default, and Catch-All, on page 9.
- Order of policies within the workspace

  See Policy Priorities, on page 82.
- Policies inherited from parent or ancestor scopes, including the catch-all rule

  You will want to ensure that a higher-priority policy is not hitting traffic before the policy that you expect to hit that traffic.

  To see the impact of policies in ancestor scopes, run live policy analysis on all involved scopes. See Live Policy Analysis, on page 111.

  When you are ready to enforce the policies in a workspace, a wizard shows you which inherited policies impact workloads in the workspace. For information, see Policy Enforcement Wizard, on page 127.
- Cross-scope policy interactions

  (When consumer and provider are in different scopes, or one end of the conversation is in a different scope than the policy)

  See When Consumer and Provider Are in Different Scopes: Policy Options, on page 88.
- Situations in which the actual consumer or provider in a policy may differ from the default configured consumer and provider, for example in failover scenarios.

  See Effective Consumer or Effective Provider, on page 101.

## Policy Priorities

Traffic handling is affected by:

- The priority of policies within the scope, and
- Policy Global Ordering and Conflict Resolution, on page 83

### Policy priorities within a scope

Within a workspace, the order of the policies in the list reflects the relative priority of each policy, with the highest priority policy at the top of the list, and the lowest priority policy at the bottom of the list.

In each workspace, Absolute policies have priority over Default policies, and the Catch-All policy is the lowest priority policy in the workspace.

For details, see Policy Rank: Absolute, Default, and Catch-All, on page 9.

## Policy Global Ordering and Conflict Resolution

Conflicts can arise between different policies defined under different scopes. More specifically, conflicts arise for workloads (inventory items) that belong to multiple scopes, such as parent/child, when those scopes have contradictory policies).

It is not feasible to resolve such conflicts manually due to the dynamic nature of scope membership; workloads can enter and leave scopes as their properties change. Therefore, the system imposes a global order, as described below, for all policies according to the scope under which they are defined. For each workload, the list of relevant policies (according to consumer/provider/scope) is identified and sorted by the global order. The decision to permit or drop a flow is made based on the *first* matching policy in the sorted list.

By understanding the global ordering scheme of security policies, network admins can define the correct scopes and their priorities to apply the overall desired policies on workloads. Within each scope, application owners maintain their ability to enforce fine-grained policies on their respective workloads.

A global network policy has the following characteristics:

- A set of scopes ordered by priority (highest priority first).
- Each scope's primary workspace has absolute policies, default policies and a catch-all action.
- Each group of absolute or default policies within each workspace is sorted according to their local priorities (highest first).

The global order of policies is defined as follows:

- Groups of absolute policies from the primary workspaces of all scopes (arranged from highest to lowest priority).
- Groups of default policies from the primary workspaces all scopes (arranged from lowest to highest priority).
- Catch-all policies from all scopes (arranged from lowest to highest priority).

Note that the scope order applies to groups of policies in category 1 and 2, rather than individual policies. Within each group, individual policies with lower policy priority numbers taking precedence.

For a specific workload, first the subset of scopes it belongs to is determined, then the above order is applied. The catch-all policy from the lowest priority (enforced) workspace to which this workload belongs is the applicable catch- all (but an absolute or default policy may override). For a given flow on that workload, the action of the highest matching policy is applied.

**Note**

- If a workspace has neither Absolute nor Default policies defined, the workspace is ignored. The workspace's catch-all policy will not be included in the global order.
- The order of Default policies in the global order is the reverse of the scope priorities. This lets you define broad policies for all scopes to secure the perimeter of all workspaces including those that do not have policy enforcement enabled. At the same time application owners who have enabled enforcement on their scopes have the ability to override these default policies.
- Overlapping scopes are not recommended; see Scope Overlap for details. However, if a workload has two or more interfaces, in overlapping or disjoint scopes, the catch-all policy of the lowest priority workspace with enforcement enabled will apply (among all the applicable catch-all policies).

We expand our previous three-scope example to illustrate this ordering scheme. Assume that the three scopes are assigned the following priorities (See Use Workspaces to Manage Policies for instruction on how to change scope priorities):

1. Apps
2. Apps:HR
3. Apps:Commerce

The primary workspace of each of these scopes has absolute policies, default policies and a catch-all action. Each group of absolute or default policies within each workspace is sorted according their local priorities.

The global ordering of the policies are as follows:

1. Apps Absolute policies
2. Apps:HR Absolute policies
3. Apps:Commerce Absolute policies
4. Apps:Commerce Default policies
5. Apps:HR Default policies
6. Apps Default policies
7. Apps:Commerce Catch-all
8. Apps:HR Catch-all
9. Apps Catch-all

A workload that belongs to the *Apps* scope will receive only the following policies in the given order:

1. Apps Absolute policies that match the workload
2. Apps Default policies
3. Apps Catch-all

A workload that belongs to the *Apps* and *Apps:Commerce* scopes receive only the following policies in the given order:

1. Apps Absolute policies
2. Apps:Commerce Absolute policies
3. Apps:Commerce Default policies
4. Apps Default policies
5. Apps:Commerce Catch-all

A workload that belongs to the *Apps* and *Apps:HR* scopes will receive only the following policies in the given order:

1. Apps Absolute policies
2. Apps:HR Absolute policies
3. Apps:HR Default policies
4. Apps Default policies
5. Apps:HR Catch-all

### Policy Order and Overlapping Scopes

**Important** The following scenario involves overlapping scopes. You should avoid having overlapping sibling scopes – workloads should not be members of multiple branches of the scope tree. For more information, see Scope Overlap.

A workload that belongs to all three *Apps*, *Apps:HR* and *Apps:Commerce* scopes will receive the following policies in the given order:

1. Apps Absolute policies
2. Apps:HR Absolute policies
3. Apps:Commerce Absolute policies
4. Apps:Commerce Default policies
5. Apps:HR Default policies
6. Apps Default policies
7. Apps:Commerce Catch-all

Note that the relative ordering of the *Apps:HR* and *Apps:Commerce* scopes only matters if the two scopes overlap (that is, there are workloads that belong to both sibling scopes.) This is because policies are always defined under a scope. A workload belonging to one scope only will not be affected by policies from the other scope, thus the order does not matter.

## Validate the Order and Priority of Policies

To validate the order and priority of policies in parent/ancestor workspaces, click the **Analyzed Policies** or **Enforced Polices** tab at the top of the Defend > Segmentation page. These views provide a global view of the analyzed and enforced policies respectively.

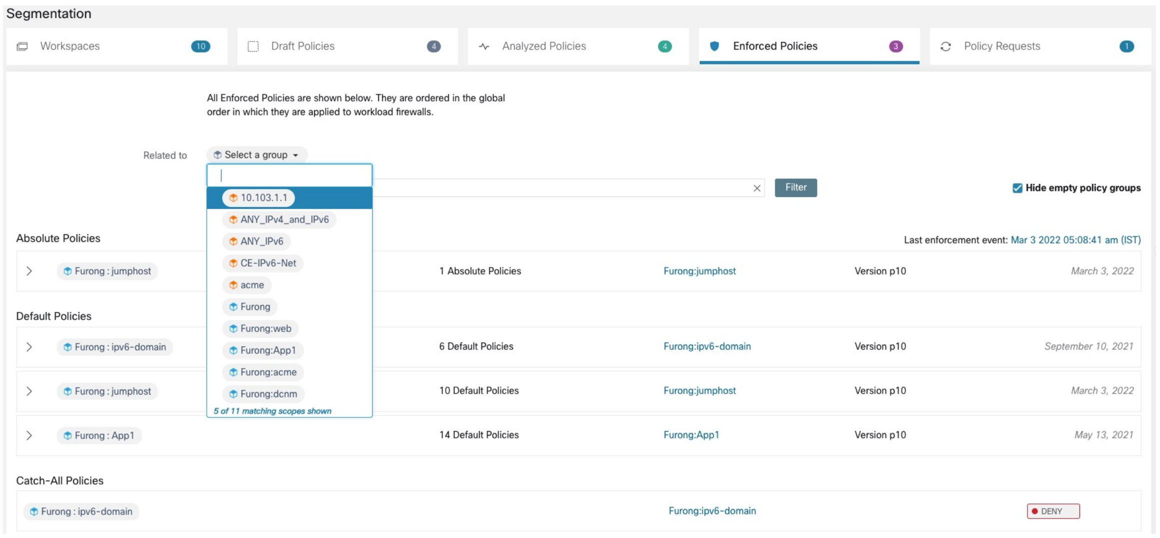


*Figure 37: Example: List of enforced policies in their policy priority order*

All Enforced Policies are shown below. They are ordered in the global order in which they are applied to workload firewalls.

Last enforcement event: Mar 3 2022 05:08:41 am (IST)

Absolute Policies

| | | | |
|---|---|---|---|
| Furong : jumphost | 1 Absolute Policies | Furong:jumphost | Version p10 | March 3, 2022 |

Default Policies

| | | | | |
|---|---|---|---|---|
| Furong : ipv6-domain | 6 Default Policies | Furong:ipv6-domain | Version p10 | September 10, 2021 |
| Furong : jumphost | 10 Default Policies | Furong:jumphost | Version p10 | March 3, 2022 |
| Furong : App1 | 14 Default Policies | Furong:App1 | Version p10 | May 13, 2021 |

Catch-All Policies

| | | |
|---|---|---|
| Furong : ipv6-domain | Furong:ipv6-domain | ● DENY |

- To limit the list of policies to only those which include a particular scope or filter as a consumer or provider, select a scope or enter a filter.
- Available filters:

| Filter Name | Definition |
|---|---|
| **Port** | Policy port to match, e.g. 80. |
| **Protocol** | Policy protocol to match, e.g. TCP. |
| **Approved** | Matches policies that have been marked as Approved Policies. |
| **External?** | Policies in which the consumer and provider are in different scopes. |
| **Action** | Policy action: Allow or Deny |

## (Advanced) Change Policy Priorities

**Caution** Scope policy priority order rarely needs to be changed. Since changing policy priorities can affect enforcement results on all workspaces, change with caution.

Access to this feature is limited to users with very high privilege roles such as site admin.

### Before you begin

Before changing scope priority order:

- Understand policy sorting logic and how policy priorities on scopes translate to ordering of individual policy intents. See Policy Priorities, on page 82.
- Make changes in a secondary workspace until you are confident that your new order will be as expected.
- Plan your changes, considering the following guidelines:

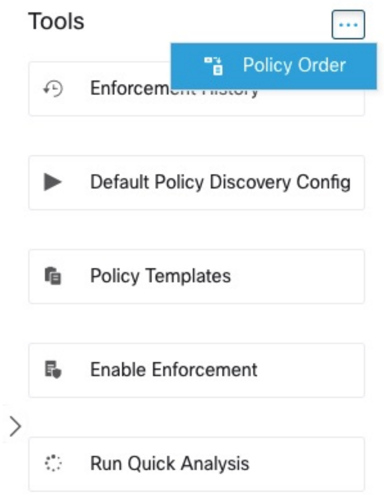
  When reordering, keep a parent-first ordering (parent scopes above child scopes) to take advantage of the hierarchical structure of your scope tree.

  (If you have overlapping sibling scopes, it may be necessary to reorder sibling scopes and their children. Overlapping sibling scopes are not recommended. Fix these by updating scope queries. See Scope Overlap.)

### Procedure

**Step 1** To reorder policy priority, click the menu icon next to **Tools** and select **Policy Order**:

*Figure 38: Navigating to Policy Priorities page*

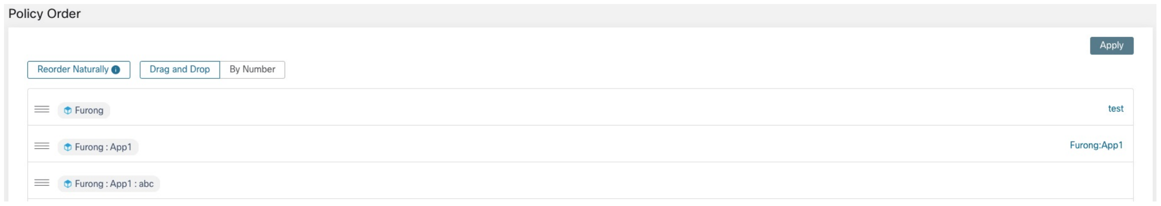


Once on the Policy Order page, you can see the list of all scopes and their corresponding primary workspaces according to the current policy priority.

**Step 2** There are several ways to reorder the scopes:

- To reorder the entire list to place parent scopes above child scopes ("pre-order"): Click **Reorder Naturally**. This is the recommended order and any deviation from this should be done with care.
- To reorder the list manually:
  - Drag the rows up and down.
  - Click **By Number** to set a number for each scope to be used for sorting. This can be easier for large lists.

*Figure 39: Setting Policy Priorities for Scopes*



### What to do next

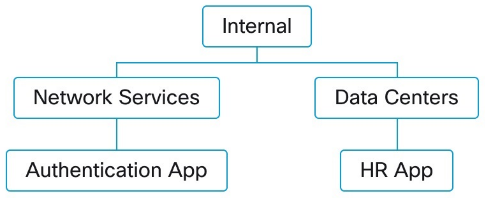
Run Quick Analysis to see the results of your changes.

## When Consumer and Provider Are in Different Scopes: Policy Options

### Example Scenario

The following situation is an example showing cross-scope traffic:

Your scope hierarchy includes a Network Services scope that includes an authentication application (the provider). An HR application, which is a member of a scope on a different branch of the scope hierarchy, is a consumer of the service provided by the authentication application.



### Policy Options

Secure Workload offers several ways to address this situation:

| Option | Instructions | Pros and Cons |
|---|---|---|
| Create these policies in a parent or ancestor scope that includes both consumer and provider as children or descendants | • Manually create one or more policies in the common-ancestor scope.<br>(Optional) For more precise policies, group workloads using inventory filters. For examples and instructions, see Create an Inventory Filter.<br>• Automatically discover policies in the common-ancestor scope, for the entire branch of the scope tree. | These methods are the simplest way to address cross-scope policies.<br>These methods require only one policy per consumer-provider pair.<br>If you are considering using automatic policy discovery, see important considerations in Discover Policies for One Scope or for a Branch of the Scope Tree, on page 23. |
| Use the advanced method for creating cross-scope policies | Automatically discover policies for each individual scope.<br>See (Advanced) Create Cross-Scope Policies, on page 89.<br>(This procedure applies to both manually created policies and discovered policies.) | This method requires two policies for each consumer-provider pair: A policy for the consumer and a policy for the provider.<br>This method allows policy creation when consumer and provider policies are owned by different people.<br>See other considerations in Discover Policies for One Scope or for a Branch of the Scope Tree, on page 23. |

## (Advanced) Create Cross-Scope Policies

This procedure describes the advanced method of creating cross-scope policies (policies in which consumer and provider are in different scopes.) It applies to both manually created policies and automatically discovered policies.

This method requires two policies for each consumer-provider pair, because both ends of the conversation must allow the conversation to happen:

- A policy in the consumer's scope must allow conversations with the provider,

  and

- A policy in the provider's scope must allow conversations with the consumer.

This procedure includes the steps that must be taken by the owner of each scope in order to create cross-scope policies. If your access privileges allow you to modify both scopes, you can perform all steps.

### Before you begin

- Consider simpler options for handling cross-scope traffic. See When Consumer and Provider Are in Different Scopes: Policy Options, on page 88.
- Policies using this method must be created in the primary workspace of both consumer and provider.

  If the provider scope to be specified in the policy does not yet have a primary workspace, create it before creating cross-scope policies using this method.

- The policies must have ALLOW action in order for policy requests to be created.
- For some additional details related to these requirements, see Policy Requests, on page 90.
- (Optional) Consider options for automatic handling of cross-scope policy requests. See Automate Handling of Cross-Scope Policy Requests, on page 95.
- (Optional) If you want cross-scope policies to apply only to the workloads in a cluster within the consumer or provider scope, and not to the entire scope, see Convert a Cluster to an Inventory Filter, on page 77. Clusters cannot be used in cross-scope policies created using this procedure.

  If you are discovering policies automatically, see also External Dependencies, on page 31 and Fine-Tune External Dependencies for a Workspace, on page 33.

### Procedure

**Step 1** In the consumer's primary workspace, create the desired policy, either manually or using automatic policy discovery.

For each cross-scope policy created, a policy request will automatically be created for the provider.

To view the policy requests, see Viewing, Accepting, and Rejecting Policy Requests, on page 91.

Note: If an existing policy in the provider application's workspace matches this traffic, a new policy is not needed and a request is not created. This situation is indicated as described in Resolved Policy Requests, on page 99.

**Step 2** You (or the owner of the provider application) must respond to each policy request:

See Viewing, Accepting, and Rejecting Policy Requests, on page 91.

Accepting a policy request automatically creates the required policy in the primary workspace of the provider, allowing traffic between the two applications.

If you do not want to allow traffic from the requesting application, reject the request.

**Step 3** (Optional) If you are automatically discovering policies, you may want to Fine-Tune External Dependencies for a Workspace, on page 33.

**Step 4** Review and analyze both primary workspaces.

### What to do next

When you are ready to enforce these policies, you must enforce both primary workspaces.

## Policy Requests

Policy requests are generated when you create cross-scope policies using the method described in (Advanced) Create Cross-Scope Policies, on page 89. Each time a policy is created in a consumer scope's primary workspace when the provider is a member of a different scope, if the policy does not yet exist in the primary workspace associated with the provider's scope, a policy request is generated.

This policy request alerts the owner of the provider application to allow dependent applications to access necessary services.

See options for viewing and responding to policy requests at Viewing, Accepting, and Rejecting Policy Requests, on page 91 and Automate Handling of Cross-Scope Policy Requests, on page 95.

#### Additional details about policy requests

- The provided services page (on which policy requests appear) is only available to primary workspaces. This is to ensure that isolated experiments on secondary workspaces do not create notifications on other primary workspaces.
- If an external scope (when the provider specified in the policy belongs to a different scope than the consumer) does not have a primary workspace, no requests are sent (for example, this could be the case for the root scope, or any scope defined for workloads outside the organization). If an external scope has not published any policy, policy analysis and enforcement are carried out on the consumer end only.
- Clusters are not supported when the provider is in a different scope than the consumer. If the policy's consumer is a cluster, the policy request will be made as if the policy request were from the consumer application's scope. Multiple policies consuming the same service from a provider could be grouped together.
- Policy requests are generated only for providers, not for consumers. If a consumer workspace is analyzing or enforcing policies, it has to explicitly include policies that allow all its legitimate consuming flows, either through automatic policy discovery or by explicitly manually crafting policies (no policy requests from external provider workspaces are generated to it).

### *Viewing, Accepting, and Rejecting Policy Requests*

When creating cross scope policies using the method described in (Advanced) Create Cross-Scope Policies, on page 89, a policy is required in the primary workspace of the provider's scope in addition to the policy in the consumer's scope. When a cross-scope policy is created in the primary workspace of the consumer's scope, a policy request is automatically created in the primary workspace of the provider's scope.

Use the information in this topic to accept the request (to create the required policy in the provider scope) or reject the request (in which case the cross-scope policy will not take effect.)

#### To view, accept, or reject policy requests:

| To | Do This |
|---|---|
| View all policy requests | **1.** Choose **Defend** > **Segmentation**.<br>**2.** Click **Policy Requests** at the top of the page.<br>**3.** Click a consumer scope to see policy requests from that scope. |

| To | Do This |
|---|---|
| View policy requests for a particular scope | To view pending policy requests for a provider scope:<br>1. Choose **Defend > Segmentation**.<br>2. Click the primary workspace of the applicable scope.<br>3. Click **Manage Policies**.<br>4. Click **Provided Services**.<br>If the tab does not display a number, there are no policy requests pending for this workspace.<br>5. Click **Policy Requests**.<br>6. Click a consumer scope to see policy requests from that scope.<br>Or<br>To view a policy request from the consumer scope:<br>In the Policies tab of the primary workspace of the consumer scope, click the value in the **Protocols and Ports** column, then look at the panel that opens on the right side of the page. In the **Protocols and Ports** section, click a yellow dot to see pending policy requests. |
| Manually accept a request and automatically create the required policy in the Provider scope | From either of the locations above, click **Accept** next to the policy request. |
| Manually reject a request | From either of the locations above, click **Reject** next to the policy request. |

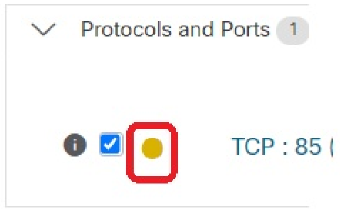
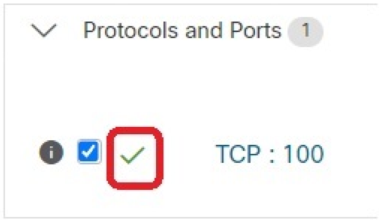
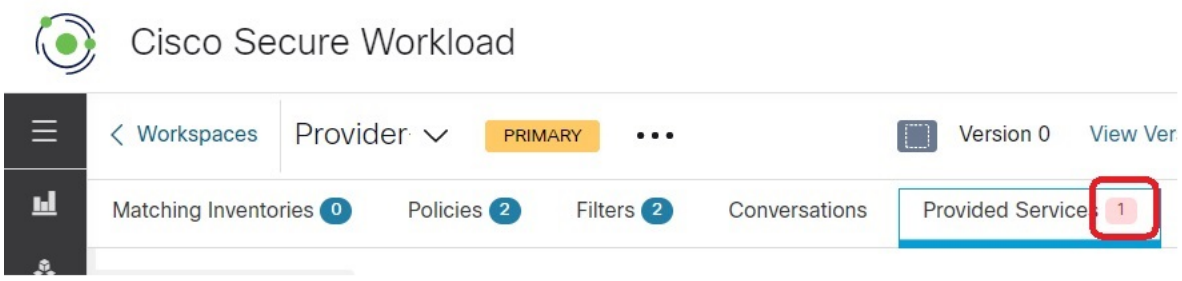
| To | Do This |
|---|---|
| View policy request status from the consumer workspace | On the Policies page of the primary consumer workspace, click the policy, then click the port/protocol value. Status is shown in the panel that opens on the right.<br>A pending request is shown with a yellow dot:<br><br>Protocols and Ports 1<br>TCP : 85 (<br>When the request is accepted, the dot changes to a green check <br>Protocols and Ports 1<br>TCP : 100<br>mark:<br>Click the indicator for details. |
| View policy request status from the provider's workspace | View request status in the **Provided Services** tab described above. |
| Allow policy discovery to create the required policy for the provider | Automatically discover policies in the provider scope's primary workspace, using a time range that ensures that the corresponding flows are seen, then publish the policy. |
| See also options for automating handling of policy requests | Automate Handling of Cross-Scope Policy Requests, on page 95 |

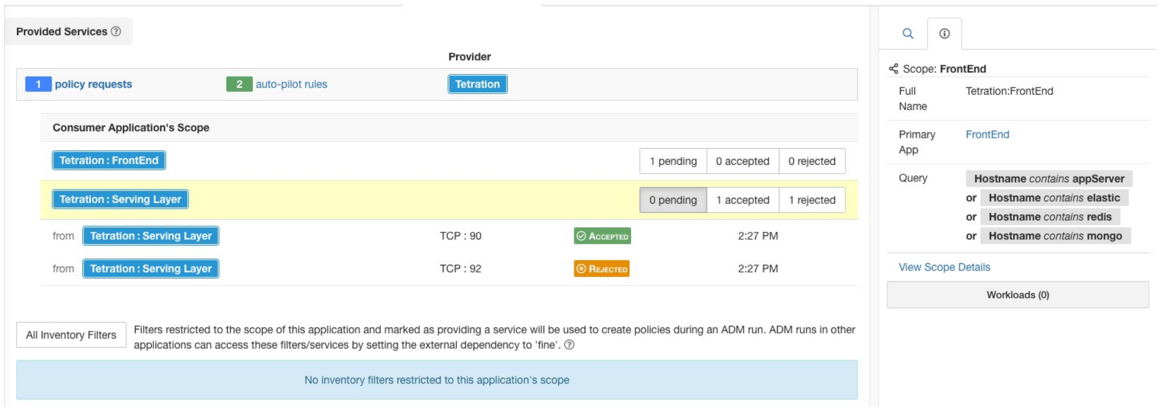
*Figure 40: Pending policy requests in the provider's workspace*



### Accepting Policy Requests: Details

Accepting a policy request on a service is equivalent to creating a policy from the requested filter as the consumer to the service as the provider. Additionally, upon accepting a policy request, the original policy from the consumer application's workspace (in the example, FrontEnd App and Serving Layer) will be marked as accepted (see figures below)

***Figure 41: Accepting/Rejecting policy requests***



Provided Services

| Provider | | |
|---|---|---|
| 1 policy requests | 2 auto-pilot rules | Tetration |

Consumer Application's Scope

| | | | | |
|---|---|---|---|---|
| Tetration : FrontEnd | | 1 pending | 0 accepted | 0 rejected |
| Tetration : Serving Layer | | 0 pending | 1 accepted | 1 rejected |
| from Tetration : Serving Layer | TCP : 90 | Accepted | 2:27 PM | |
| from Tetration : Serving Layer | TCP : 92 | Rejected | 2:27 PM | |

All Inventory Filters — Filters restricted to the scope of this application and marked as providing a service will be used to create policies during an ADM run. ADM runs in other applications can access these filters/services by setting the external dependency to 'fine'.

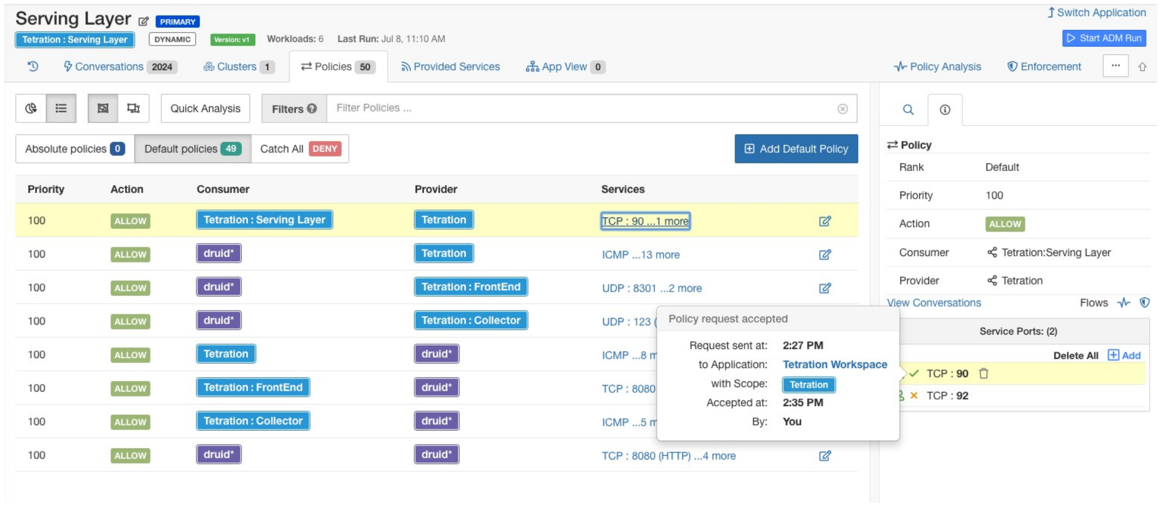
No inventory filters restricted to this application's scope

Scope: FrontEnd

| | |
|---|---|
| Full Name | Tetration:FrontEnd |
| Primary App | FrontEnd |
| Query | Hostname *contains* **appServer** or Hostname *contains* **elastic** or Hostname *contains* **redis** or Hostname *contains* **mongo** |

View Scope Details
Workloads (0)

***Figure 42: Policy status shown as Accepted***



Serving Layer PRIMARY
Tetration : Serving Layer DYNAMIC Version: v1 Workloads: 6 Last Run: Jul 8, 11:10 AM

Absolute policies 0 | Default policies 49 | Catch All DENY

| Priority | Action | Consumer | Provider | Services |
|---|---|---|---|---|
| 100 | ALLOW | Tetration : Serving Layer | Tetration | TCP : 90 ...1 more |
| 100 | ALLOW | druid* | Tetration | ICMP ...13 more |
| 100 | ALLOW | druid* | Tetration : FrontEnd | UDP : 8301 ...2 more |
| 100 | ALLOW | druid* | Tetration : Collector | UDP : 123 ( |
| 100 | ALLOW | Tetration | druid* | ICMP ...8 m |
| 100 | ALLOW | Tetration : FrontEnd | druid* | TCP : 8080 |
| 100 | ALLOW | Tetration : Collector | druid* | ICMP ...5 m |
| 100 | ALLOW | druid* | druid* | TCP : 8080 (HTTP) ...4 more |

Policy request accepted

| | |
|---|---|
| Request sent at: | 2:27 PM |
| to Application: | Tetration Workspace |
| with Scope: | Tetration |
| Accepted at: | 2:35 PM |
| By: | You |

Policy

| | |
|---|---|
| Rank | Default |
| Priority | 100 |
| Action | ALLOW |
| Consumer | Tetration:Serving Layer |
| Provider | Tetration |

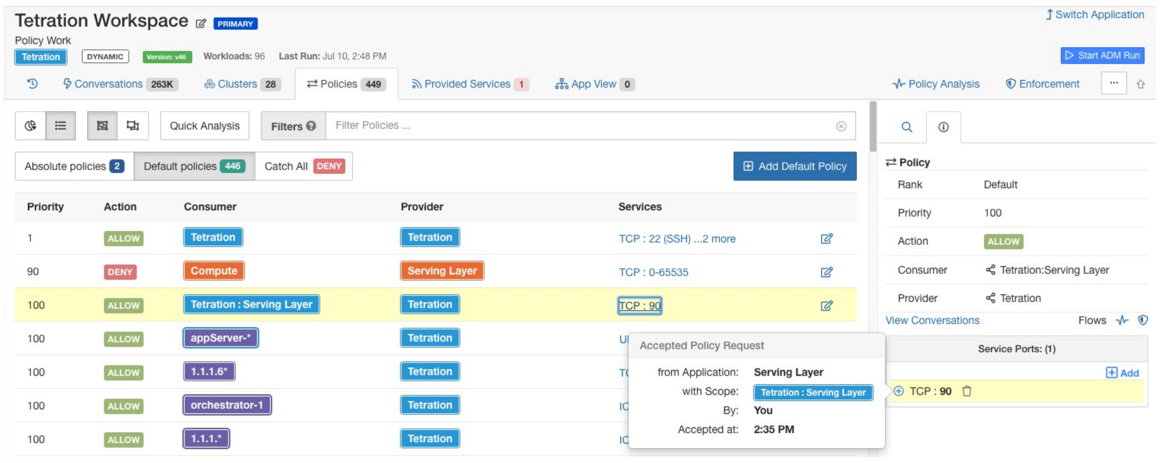
View Conversations

Service Ports: (2)
✓ TCP : 90
✗ TCP : 92

The new policy created on the provider application's workspace (in this example, the workspace is named Tetration) is marked with a **plus** icon indicating that this policy was created due to an external policy request.

**Note** If the original policy on the consumer side is deleted after the policy request is accepted, the policy on provider side will not be deleted. However, the tooltip next to the policy shows the original policy as deleted with the timestamp of the event:

**Figure 43: Provider side policy, created by accepting a policy request**


Tetration Workspace PRIMARY

Policy Work

Tetration DYNAMIC Version: v46 Workloads: 96 Last Run: Jul 10, 2:48 PM

Conversations 263K · Clusters 28 · Policies 449 · Provided Services 1 · App View 0 · Policy Analysis · Enforcement

Absolute policies 2 · Default policies 446 · Catch All DENY · Add Default Policy

| Priority | Action | Consumer | Provider | Services |
|---|---|---|---|---|
| 1 | ALLOW | Tetration | Tetration | TCP : 22 (SSH) ...2 more |
| 90 | DENY | Compute | Serving Layer | TCP : 0-65535 |
| 100 | ALLOW | Tetration : Serving Layer | Tetration | TCP : 90 |
| 100 | ALLOW | appServer-* | Tetration | U |
| 100 | ALLOW | 1.1.1.6* | Tetration | T |
| 100 | ALLOW | orchestrator-1 | Tetration | IC |
| 100 | ALLOW | 1.1.1.* | Tetration | IC |

Accepted Policy Request

| | |
|---|---|
| from Application: | Serving Layer |
| with Scope: | Tetration : Serving Layer |
| By: | You |
| Accepted at: | 2:35 PM |

Policy

| | |
|---|---|
| Rank | Default |
| Priority | 100 |
| Action | ALLOW |
| Consumer | Tetration:Serving Layer |
| Provider | Tetration |

View Conversations · Flows

Service Ports: (1)

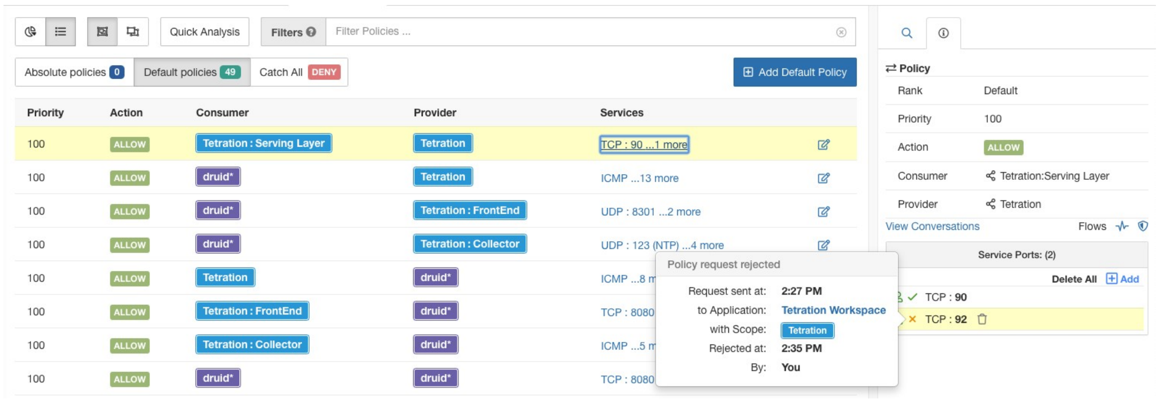
Add

TCP : 90

## Rejecting Policy Requests: Details

Rejecting a policy request does not create or update any policies. The original policy from the consumer application's workspace (in the example, Serving Layer App) will be marked as rejected, but the policy remains in effect, i.e., outbound traffic still will be allowed. The tooltip next to the reject policy has information about the provider application, the user that rejected the policy request as well as the time of the rejection.

**Figure 44: Policy status shown as Rejected**


Absolute policies 0 · Default policies 49 · Catch All DENY · Add Default Policy

| Priority | Action | Consumer | Provider | Services |
|---|---|---|---|---|
| 100 | ALLOW | Tetration : Serving Layer | Tetration | TCP : 90 ...1 more |
| 100 | ALLOW | druid* | Tetration | ICMP ...13 more |
| 100 | ALLOW | druid* | Tetration : FrontEnd | UDP : 8301 ...2 more |
| 100 | ALLOW | druid* | Tetration : Collector | UDP : 123 (NTP) ...4 more |
| 100 | ALLOW | Tetration | druid* | ICMP ...8 m |
| 100 | ALLOW | Tetration : FrontEnd | druid* | TCP : 8080 |
| 100 | ALLOW | Tetration : Collector | druid* | ICMP ...5 m |
| 100 | ALLOW | druid* | druid* | TCP : 8080 |

Policy request rejected

| | |
|---|---|
| Request sent at: | 2:27 PM |
| to Application: | Tetration Workspace |
| with Scope: | Tetration |
| Rejected at: | 2:35 PM |
| By: | You |

Policy

| | |
|---|---|
| Rank | Default |
| Priority | 100 |
| Action | ALLOW |
| Consumer | Tetration:Serving Layer |
| Provider | Tetration |

View Conversations · Flows

Service Ports: (2)

Delete All · Add

✓ TCP : 90

✗ TCP : 92

## *Automate Handling of Cross-Scope Policy Requests*

Policy requests are generated when you create cross-scope policies using the method described in (Advanced) Create Cross-Scope Policies, on page 89.

There are several options to reduce the number of policy requests that are generated when creating cross-scope policies:

*Table 5: Options for Automatically Handling Policy Requests*

| To | Do This |
|---|---|
| Specify handling of policy requests between specific consumer-provider pairs | See Auto-pilot Rules, on page 96.<br><br>You must have the required privileges. |
| Automatically create all required policies for providers for all cross-scope policies created during policy discovery in a particular workspace | When you start an automatic policy discovery run, enable the **Auto accept outgoing policy connectors** option in the Advanced Configurations section.<br><br>This option is available to root scope owners and site admins only.<br><br>For details, see:<br><br>Advanced Configurations for Automatic Policy Discovery, on page 35 and<br><br>Auto Accept Policy Connectors, on page 98 |
| Specify default handling for all policy requests from all workspaces | On the Default Policy Discovery Config page, enable the **Auto accept outgoing policy connectors** option in the Advanced Configurations section.<br><br>This option is available to root scope owners and site admins only.<br><br>For details, see:<br><br>Default Policy Discovery Config, on page 45 and<br><br>Advanced Configurations for Automatic Policy Discovery, on page 35 and<br><br>Auto Accept Policy Connectors, on page 98 |

## Auto-pilot Rules

This feature is applicable only if you create cross-scope policies using the method described in (Advanced) Create Cross-Scope Policies, on page 89.

Infrastructure applications that provide services to many other applications in a datacenter may receive a large number of policy requests from other applications.

You can reduce the volume of policy requests by creating auto-pilot rules to automatically accept or reject future matching policy requests.

**Note** Auto-pilot rules do not apply to existing policy requests. They affect only future policy requests.

### Automatically accept or reject policy requests using Auto-Pilot Rules

Configure auto-pilot rules to automatically accept or reject policy requests between a specified consumer-provider pair, on specified ports. Auto-pilot rules can be broad (scope-to-scope), or apply only to a subset of workloads within each scope (as configured by inventory filters. You can use an inventory filter for the consumer, for the provider, or for each.)

1. If you want your auto-pilot rule to apply to a subset of workloads within a scope rather than to the entire scope:

Create an inventory filter in the relevant scope(s) to group the workloads. Be sure the **Restrict Query to Ownership Scope** option is selected in each inventory filter, to ensure that the filter only includes workloads that are members of the scope.

2. Choose **Defend** > **Segmentation**.
3. Click the primary workspace of the consumer scope for which you want to automatically accept or reject policy requests related to a specific provider.
4. Click **Manage Policies**.
5. Click **Provided Services**.
6. If you are creating this rule for an inventory filter, perform the following steps for the desired inventory filter (inventory filters are identified by an orange icon.)

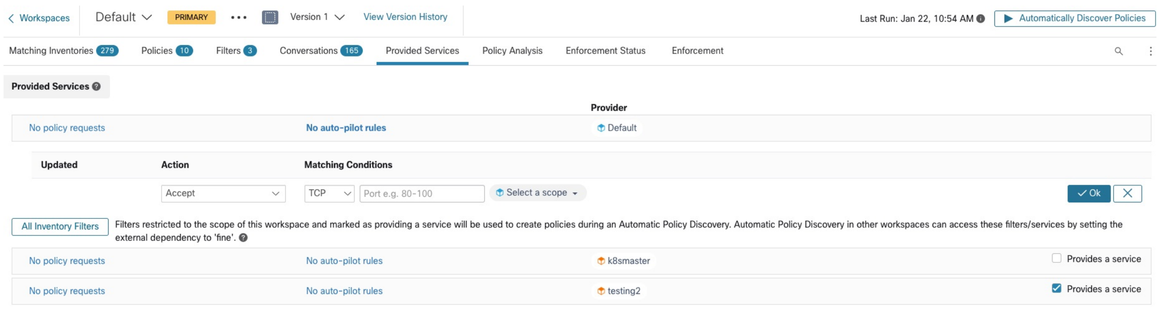
   Otherwise perform these steps for the scope (scopes are identified by a blue icon.)

   Make sure you are clicking in the correct place.
7. Click **No Auto-Pilot Rules** or **auto-pilot rules**, whichever is displayed.
8. Click **New Auto-Pilot Rule**.
9. Configure the auto-pilot rule. Select the scope or inventory filter that represents the provider.
10. Click **OK**.

### Example Auto-Pilot Rule

In the example below, we create a new auto-pilot rule to reject TCP policy requests in port range 1-200 from any consumer contained in Tetration:Adhoc to the provider service Tetration

*Figure 45: Creating/Updating Auto-pilot rules*



| | | Provider | |
|---|---|---|---|
| No policy requests | No auto-pilot rules | Default | |

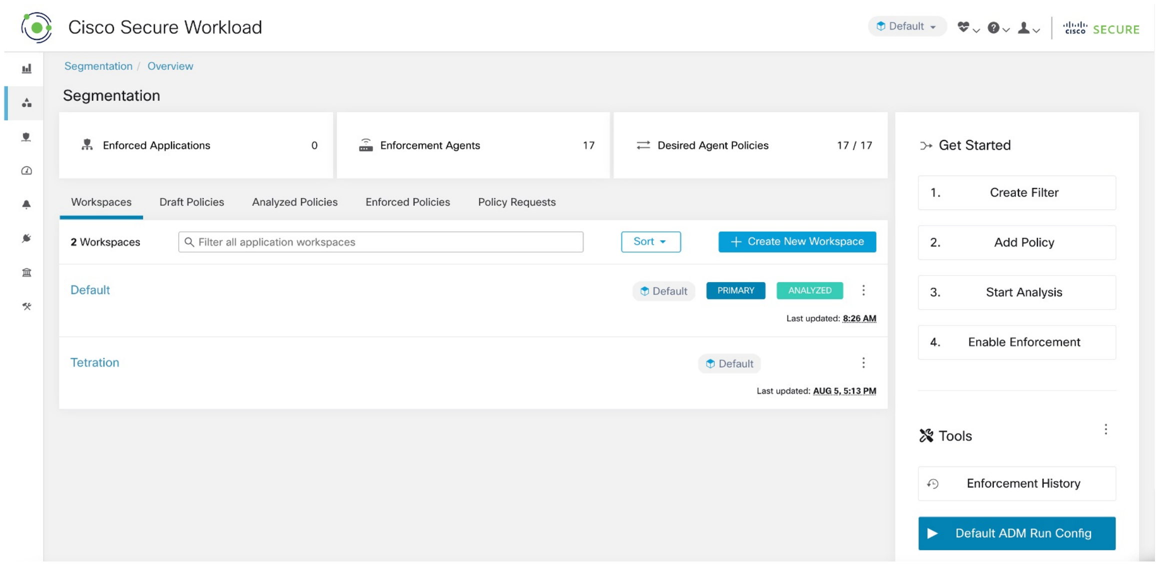
| Updated | Action | Matching Conditions | | |
|---|---|---|---|---|
| | Accept | TCP | Port e.g. 80-100 | Select a scope |

Filters restricted to the scope of this workspace and marked as providing a service will be used to create policies during an Automatic Policy Discovery. Automatic Policy Discovery in other workspaces can access these filters/services by setting the external dependency to 'fine'.

| | | | |
|---|---|---|---|
| No policy requests | No auto-pilot rules | k8smaster | ☐ Provides a service |
| No policy requests | No auto-pilot rules | testing2 | ✓ Provides a service |

Then we create a new policy in the workspace for the *FrontEnd App* on TCP port 23. Since the policy is a match for the auto-pilot rule, it will be automatically rejected. The status and reason for policy rejection is indicated on the tooltip next to the rejected policy.

*Figure 46: Policy automatically getting rejected by Auto-pilot rule*



### View a count of policies recently created by auto-pilot rules

To view the number of policies created in a workspace by auto-pilot rules since live policy analysis was last initiated (or re-initiated) for the workspace:

Navigate to the Provided Services page for the relevant primary workspace and look for the count of "Auto Created" policies.

## Auto Accept Policy Connectors

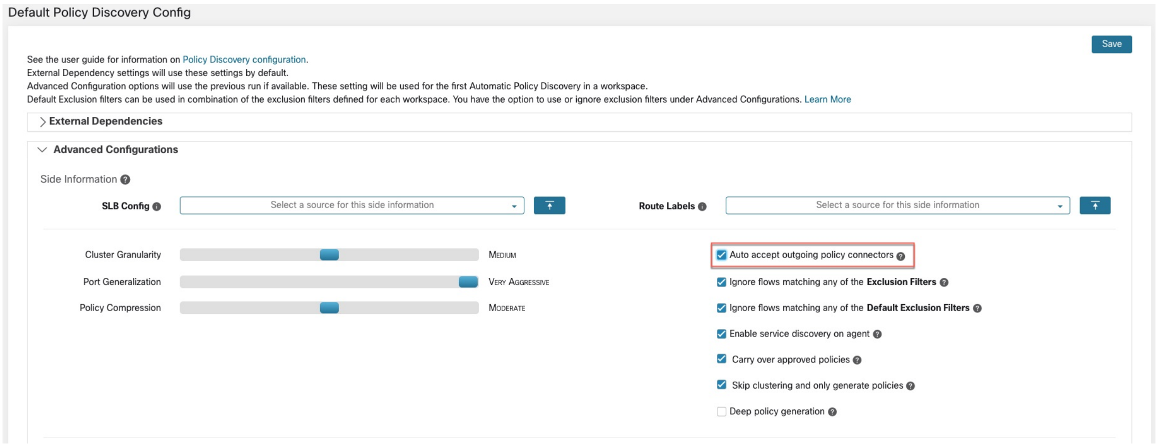
You can set this option as the default policy discovery configuration, or set it in the automatic policy discovery advanced options for each workspace.

The **Auto accept outgoing policy connectors** option on the automatic policy discovery configuration page allows you to automatically accept any policy requests created as part of automatic policy discovery.

If this option is enabled in Default automatic policy discovery config, policy requests created manually or by importing a workspace will be automatically accepted as well.

**Note** This option is only available for root scope owners or Site Admins.

*Figure 47: The Auto accept outgoing policy connectors option*


Default Policy Discovery Config

See the user guide for information on Policy Discovery configuration.
External Dependency settings will use these settings by default.
Advanced Configuration options will use the previous run if available. These setting will be used for the first Automatic Policy Discovery in a workspace.
Default Exclusion filters can be used in combination of the exclusion filters defined for each workspace. You have the option to use or ignore exclusion filters under Advanced Configurations. Learn More

External Dependencies

Advanced Configurations

Side Information

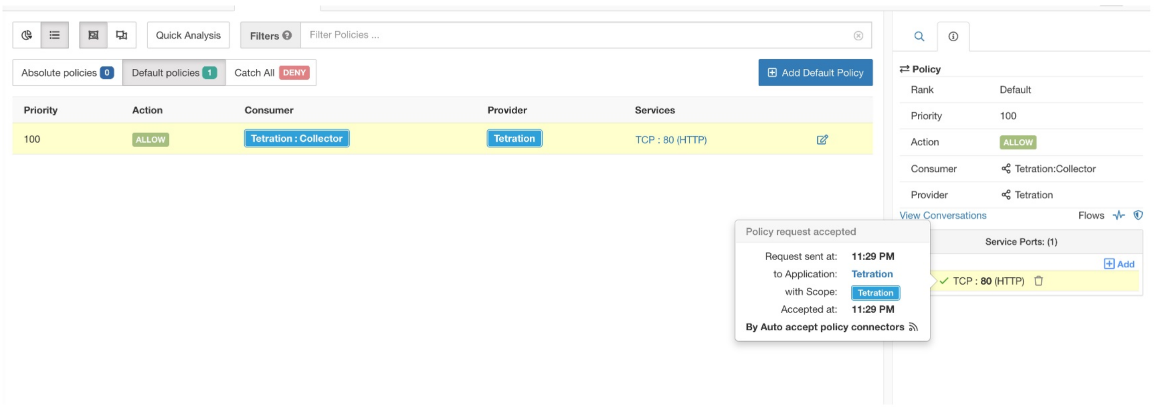
SLB Config: Select a source for this side information
Route Labels: Select a source for this side information

| | |
|---|---|
| Cluster Granularity | Medium |
| Port Generalization | Very Aggressive |
| Policy Compression | Moderate |

- [✓] Auto accept outgoing policy connectors
- [✓] Ignore flows matching any of the **Exclusion Filters**
- [✓] Ignore flows matching any of the **Default Exclusion Filters**
- [✓] Enable service discovery on agent
- [✓] Carry over approved policies
- [✓] Skip clustering and only generate policies
- [ ] Deep policy generation

After this option is set, any policy request created in any workspace the root scope or in the applicable workspace will be automatically accepted.

*Figure 48: Policy is automatically accepted by Auto accept policy connectors*



| Priority | Action | Consumer | Provider | Services |
|---|---|---|---|---|
| 100 | ALLOW | Tetration : Collector | Tetration | TCP : 80 (HTTP) |

Policy

| | |
|---|---|
| Rank | Default |
| Priority | 100 |
| Action | ALLOW |
| Consumer | Tetration:Collector |
| Provider | Tetration |

View Conversations — Flows

Service Ports: (1)
Add
✓ TCP : 80 (HTTP)

Policy request accepted

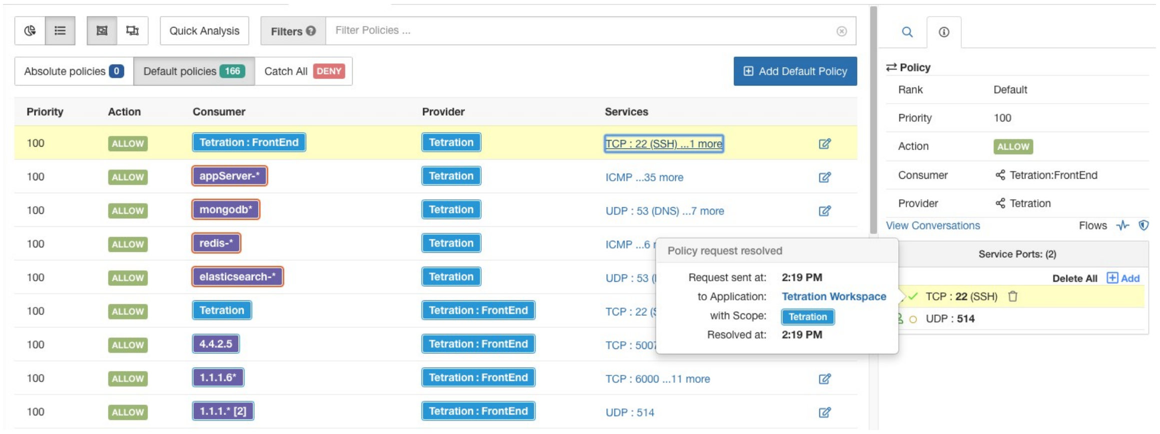
| | |
|---|---|
| Request sent at: | 11:29 PM |
| to Application: | Tetration |
| with Scope: | Tetration |
| Accepted at: | 11:29 PM |

By Auto accept policy connectors

## *Resolved Policy Requests*

If all conditions for creating a policy request are met, but there is already an existing matching policy on the provider application’s workspace, the policy created on the consumer application’s workspace will be marked as resolved indicating that the provider application’s workspace is already allowing the traffic through the requested port.

*Figure 49: Policy status shown as Resolved*



| Priority | Action | Consumer | Provider | Services |
|---|---|---|---|---|
| 100 | ALLOW | Tetration : FrontEnd | Tetration | TCP : 22 (SSH) ...1 more |
| 100 | ALLOW | appServer-* | Tetration | ICMP ...35 more |
| 100 | ALLOW | mongodb* | Tetration | UDP : 53 (DNS) ...7 more |
| 100 | ALLOW | redis-* | Tetration | ICMP ...6 |
| 100 | ALLOW | elasticsearch-* | Tetration | UDP : 53 |
| 100 | ALLOW | Tetration | Tetration : FrontEnd | TCP : 22 |
| 100 | ALLOW | 4.4.2.5 | Tetration : FrontEnd | TCP : 500 |
| 100 | ALLOW | 1.1.1.6* | Tetration : FrontEnd | TCP : 6000 ...11 more |
| 100 | ALLOW | 1.1.1.* [2] | Tetration : FrontEnd | UDP : 514 |

Policy request resolved

| | |
|---|---|
| Request sent at: | 2:19 PM |
| to Application: | Tetration Workspace |
| with Scope: | Tetration |
| Resolved at: | 2:19 PM |

Policy

| | |
|---|---|
| Rank | Default |
| Priority | 100 |
| Action | ALLOW |
| Consumer | Tetration:FrontEnd |
| Provider | Tetration |

TCP : 22 (SSH)
UDP : 514

## Provided Services

This page is used only for creating policies in which the consumer and provider are in different scopes, and only if you are using the method described in (Advanced) Create Cross-Scope Policies, on page 89.

For more information about options on this page, see:

- Policy Requests, on page 90
- Auto-pilot Rules, on page 96
- Create an Inventory Filter and External Dependencies, on page 31 (for information about the **Provides a service** option)

To access this page, navigate to a primary workspace, then click **Manage Policies**, then click **Provided Services**.

## Troubleshoot Cross-Scope Policies

If cross-scope policies were created using the method described in (Advanced) Create Cross-Scope Policies, on page 89, the primary workspaces for the consumer and provider workloads must each have a policy that allows the traffic. Ensure that the required policies exist in both workspaces.

No notification is given if one of the policies is deleted or modified.

If the policy pair was generated during policy discovery, see information about approving policies to protect them from subsequent discovery runs. See Approve Policies, on page 47.

Verify that other requirements are still being met, as listed in (Advanced) Create Cross-Scope Policies, on page 89.

Both consumer and provider workspaces that have the required policies must be enforced.

### Useful tools for cross-scope policies

- Use the **External?** filter option to find policies in which the provider is in a different scope from the scope in which you discovered policies.
- The policy visual view has an option to display external policies. See Policy Visual Representation, on page 107.

### If you are using Default Policy Discovery Config

Make sure you have clicked **Save** on the **Default Policy Discovery Config** page after making changes to make default external dependency configurations available to individual workspaces.

## Effective Consumer or Effective Provider

The consumer and provider specified in a policy determine:

- The set of workloads with Secure Workload agents that receive the policy.
- The set of IP addresses that are affected by the installed firewall rules.

By default, these are the same.

However, you may need to specify a group of IP addresses in the firewall rules that is different from the IP addresses of the workloads that receive the policy. (See an example below.)

To address this need, you can configure effective consumer and/or effective provider.

### Default behavior for consumer and provider

By default, when a Secure Workload agent receives a policy, the firewall rules are specific to that workload. This is best illustrated with the following example:

Consider an ALLOW policy with provider filter specifying 1.1.1.0/24 subnet. When this policy is programmed on a workload with IP address 1.1.1.2, the firewall rules look like the following:

- For incoming traffic firewall rules allow traffic destined to 1.1.1.2 specifically and not to the whole subnet 1.1.1.0/24.
- For outgoing traffic firewall rules allow traffic sourced from 1.1.1.2 specifically and not from the whole subnet 1.1.1.0/24 (to prevent spoofing).

As a corollary, any agent workloads belonging to the workspace that do not have IP address within 1.1.1.0/24 subnet will not receive the above firewall rules.

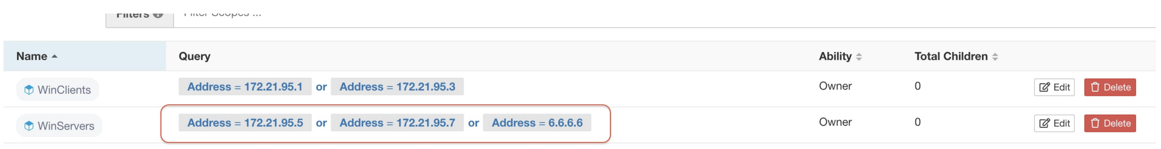
### Example: Effective Consumer or Effective Provider

In this example, suppose you are configuring policies for a fleet of workloads behind a virtual IP (VIP), similar to keepalive or windows failover clustering solutions. You will use effective consumer and /or effective provider to ensure that traffic is not disrupted during a failover event.

Consider a fleet of workloads with IP addresses (*172.21.95.5* and *172.21.95.7*) that provide a service sitting behind a VIP - *6.6.6.6*. This VIP is a floating VIP and only one workload owns the VIP at any point in time. The goal is to program firewall rules on all the workloads in the fleet to allow traffic to *6.6.6.6*.

In this setup, we have a scope and a corresponding workspace that contain a cluster of workloads that represents the fleet (*172.21.95.5* and *172.21.95.7*) as well as the VIP (*6.6.6.6*).

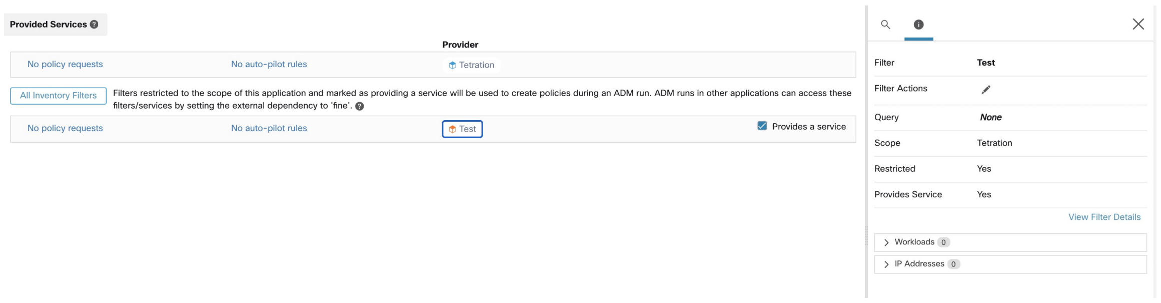
***Figure 50: Scopes including VIP and cluster of workloads***



| Name | Query | Ability | Total Children | |
|---|---|---|---|---|
| WinClients | Address = 172.21.95.1 or Address = 172.21.95.3 | Owner | 0 | Edit Delete |
| WinServers | Address = 172.21.95.5 or Address = 172.21.95.7 or Address = 6.6.6.6 | Owner | 0 | Edit Delete |

The VIP is exposed in this workspace as a provided service as shown below:

*Figure 51: VIP exposed as a provided service*



**Provided Services**

**Provider**

No policy requests | No auto-pilot rules | Tetration

All Inventory Filters — Filters restricted to the scope of this application and marked as providing a service will be used to create policies during an ADM run. ADM runs in other applications can access these filters/services by setting the external dependency to 'fine'.

No policy requests | No auto-pilot rules | Test | ✓ Provides a service

| | |
|---|---|
| Filter | **Test** |
| Filter Actions | |
| Query | *None* |
| Scope | Tetration |
| Restricted | Yes |
| Provides Service | Yes |

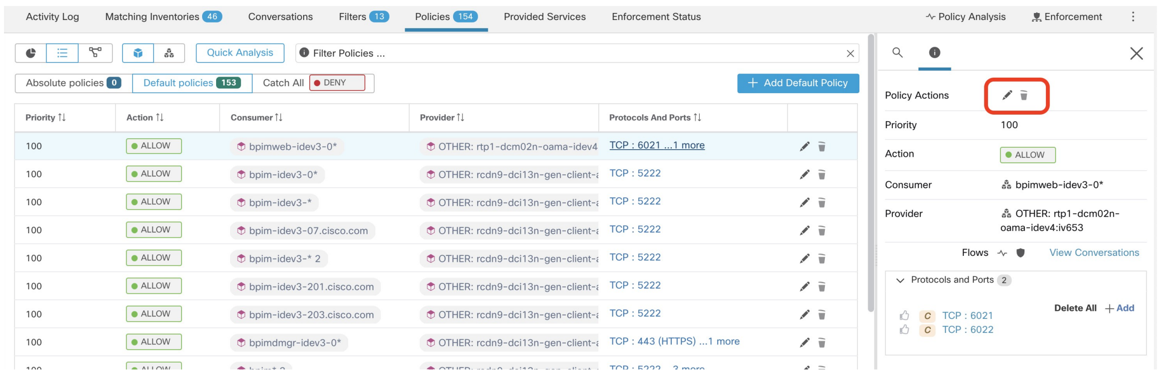
View Filter Details

Workloads 0

IP Addresses 0

If we were to add a policy from the clients of this service to the service VIP, then (by default) firewall rules allowing traffic to the VIP will only be programmed on the workload that owns the VIP. However, in case of a failover event, it may take some time for the new workload that subsequently owns the service VIP to get the right firewall rules and traffic may be disrupted for a brief while.

*Figure 52: Policy allowing traffic from clients to service VIP*



Activity Log | Matching Inventories 46 | Conversations | Filters 13 | Policies 154 | Provided Services | Enforcement Status | Policy Analysis | Enforcement

Quick Analysis | Filter Policies ...

Absolute policies 0 | Default policies 153 | Catch All ● DENY | + Add Default Policy

| Priority | Action | Consumer | Provider | Protocols And Ports |
|---|---|---|---|---|
| 100 | ● ALLOW | bpimweb-idev3-0* | OTHER: rtp1-dcm02n-oama-idev4 | TCP : 6021 ...1 more |
| 100 | ● ALLOW | bpim-idev3-0* | OTHER: rcdn9-dci13n-gen-client-a | TCP : 5222 |
| 100 | ● ALLOW | bpim-idev3-* | OTHER: rcdn9-dci13n-gen-client-a | TCP : 5222 |
| 100 | ● ALLOW | bpim-idev3-07.cisco.com | OTHER: rcdn9-dci13n-gen-client-a | TCP : 5222 |
| 100 | ● ALLOW | bpim-idev3-* 2 | OTHER: rcdn9-dci13n-gen-client-a | TCP : 5222 |
| 100 | ● ALLOW | bpim-idev3-201.cisco.com | OTHER: rcdn9-dci13n-gen-client-a | TCP : 5222 |
| 100 | ● ALLOW | bpim-idev3-203.cisco.com | OTHER: rcdn9-dci13n-gen-client-a | TCP : 5222 |
| 100 | ● ALLOW | bpimdmgr-idev3-0* | OTHER: rcdn9-dci13n-gen-client-a | TCP : 443 (HTTPS) ...1 more |
| 100 | ● ALLOW | bpim* 2 | OTHER: rcdn9-dci13n-gen-client-a | TCP : 5222 ...3 more |

| | |
|---|---|
| Policy Actions | |
| Priority | 100 |
| Action | ● ALLOW |
| Consumer | bpimweb-idev3-0* |
| Provider | OTHER: rtp1-dcm02n-oama-idev4:iv653 |

Flows | View Conversations

Protocols and Ports 2 | Delete All | + Add

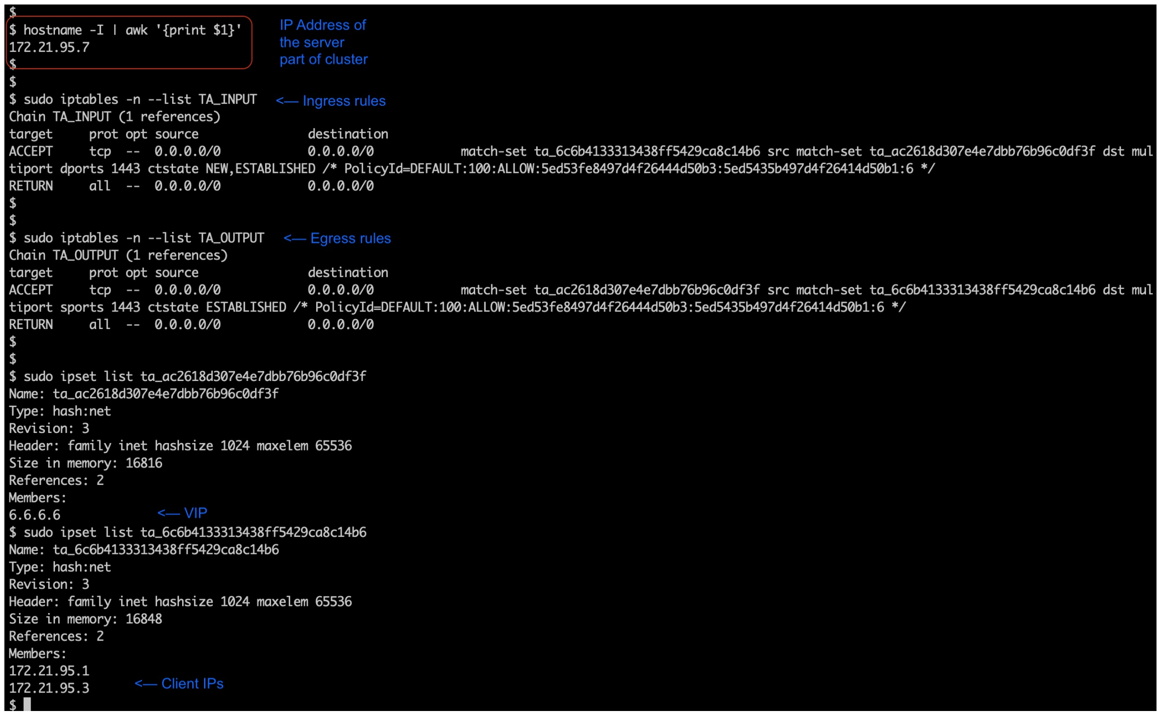
C TCP : 6021

C TCP : 6022

To address this issue, we configure the Effective Provider (using the procedure below.) Specifically, we set Effective Provider to include the group of workloads where firewall rules allowing traffic to the service VIP need to be programmed – it does not matter if any of these workloads own the VIP or not.

When Effective Provider is set, we can see on the workloads that firewall rules allowing traffic to 6.6.6.6 are programmed even when a workload does not own the VIP. When all workloads backing the service are programmed with these rules, traffic will not be disrupted during a failover event because the new primary workload (that owns the VIP) will have the necessary firewall rules programmed.

*Figure 53: Firewall rules on the host allowing traffic to service VIP*



```
$
$ hostname -I | awk '{print $1}'
172.21.95.7
$
$
$ sudo iptables -n --list TA_INPUT
Chain TA_INPUT (1 references)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0            match-set ta_6c6b4133313438ff5429ca8c14b6 src match-set ta_ac2618d307e4e7dbb76b96c0df3f dst mul
tiport dports 1443 ctstate NEW,ESTABLISHED /* PolicyId=DEFAULT:100:ALLOW:5ed53fe8497d4f26444d50b3:5ed5435b497d4f26414d50b1:6 */
RETURN     all  --  0.0.0.0/0            0.0.0.0/0
$
$
$ sudo iptables -n --list TA_OUTPUT
Chain TA_OUTPUT (1 references)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0            match-set ta_ac2618d307e4e7dbb76b96c0df3f src match-set ta_6c6b4133313438ff5429ca8c14b6 dst mul
tiport sports 1443 ctstate ESTABLISHED /* PolicyId=DEFAULT:100:ALLOW:5ed53fe8497d4f26444d50b3:5ed5435b497d4f26414d50b1:6 */
RETURN     all  --  0.0.0.0/0            0.0.0.0/0
$
$
$ sudo ipset list ta_ac2618d307e4e7dbb76b96c0df3f
Name: ta_ac2618d307e4e7dbb76b96c0df3f
Type: hash:net
Revision: 3
Header: family inet hashsize 1024 maxelem 65536
Size in memory: 16816
References: 2
Members:
6.6.6.6
$ sudo ipset list ta_6c6b4133313438ff5429ca8c14b6
Name: ta_6c6b4133313438ff5429ca8c14b6
Type: hash:net
Revision: 3
Header: family inet hashsize 1024 maxelem 65536
Size in memory: 16848
References: 2
Members:
172.21.95.1
172.21.95.3
$
```

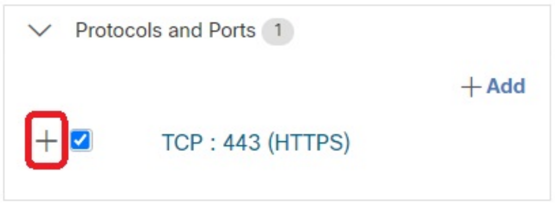
## How to Configure Effective Consumer or Effective Provider

1. Click the policy to edit.
2. Click the Edit button at the top right side of the policy to go to advanced policy options.
3. Click **Effective Consumer** or **Effective Provider**.
4. Specify the desired addresses.
5. You may need to specify addresses for both effective consumer and effective provider.

# About Deleting Policies

**Important** Before you delete a policy, check to be sure it is not one of a pair of policies required when consumer and provider are in different scopes.

To determine this: Click the link for the policy in the Protocols and Ports column. In the panel that opens on the right side of the page, look at the Protocols and Ports section. Policies created by accepting a cross-scope policy request are indicated by a plus sign beside the port and protocol:



Click the plus sign to see the creator of the cross-scope policy and a link to the corresponding consumer policy.

**Note** Policies suggested by automatic policy discovery that have not been approved may not be present after a subsequent policy discovery run, if the traffic flows that produced them are not seen during the subsequent run. To preserve suggested policies, see Approve Policies, on page 47.

# Review and Analyze Policies

It is essential that you ensure that your policies will have the intended effects (and not have any unintended effects) before you enforce them.

## Review Automatically Discovered Policies

Review policy discovery results on the Policies page of the workspace in which you discovered policies.

### Start your review here

We recommend that you start by checking to see if policies address each of the following areas, in this suggested order:

- Critical, common ports
- Internet-facing traffic
- Traffic between different applications (These flows may involve workloads in different scopes)
- Traffic within the same application (These flows are likely to involve workloads in the same scope)

### Helpful tools for reviewing policies

- To make this effort more manageable, filter and sort the policies so you can review related policies as a group.
    - Click table headings to sort the columns, for example by consumer, provider, or port/protocol.
    - Use the filter at the top of the policies list to view specific subsets.

      To see a list of properties that you can filter on, click the (i) button in the Filter Policies box.

- Look at the graphical representation of the generated policies:

  Click the Policy Visual View button ().

  For more information, see Policy Visual Representation, on page 107.

- To search or filter the rows based on ports, click the **Ungrouped** button.
- By default, the policies are grouped by consumer/provider/action. To return to this view, click the **Grouped** button.
- Use the **External?** filter option to find policies in which the provider is in a different scope from the scope in which you discovered policies.

  Create policies for this traffic using one of the methods described in When Consumer and Provider Are in Different Scopes: Policy Options, on page 88.

- Look at the confidence level of the generated policies. See Address Low-Confidence Policies, on page 106.
- Look at the Workload Profile for detailed information about a workload. Click the IP address, then click **View Workload Profile** in the pane on the right.
- To view the traffic flows that were used to produce a particular policy, click the value in the **Protocols and Ports** column for that policy, then click **View Conversations** in the side panel that opens.

  See Conversations, on page 142 for more information.

  If needed, you can drill down further by clicking **Flow Search** to view the flows for a conversation.

### Other things to do and check

- Identify unknown IP addresses (such as failover or other floating IPs) and tag them with labels so you know what they are.

  You may find helpful details on the Inventory Profile page. Click the IP address, then click **View Inventory Profile** in the pane on the right.

- Look for anything that is obviously not desirable or does not make sense.
- Group workloads using inventory filters so a single policy can address multiple workloads. See Create an Inventory Filter.
- Investigate and contact other network administrators as needed to understand the need for the policies you see.
- See the topics under Address Policy Complexities, on page 82, which can involve manual and approved policies as well as automatically discovered policies.

- In general, it is recommended that the maximum number of policies in a scope is not larger than about 500. If you have many more than this, see if you can consolidate similar policies or consider splitting the scope.
- As you review, approve any policies that you know are correct as-is to preserve them in future discovery runs.

## Address Low-Confidence Policies

After automatic policy discovery, confidence ratings indicate the accuracy and appropriateness of each discovered policy for each service (port and protocol) specified in the policy.

**To identify low-confidence discovered policies:**

1. Navigate to the applicable scope and workspace and click **Manage Policies**.
2. Click the **Policies** tab.
3. Click the **Ungrouped Policy List View** button.
4. Click the **Confidence** column heading to sort the list of policies by confidence level.
5. Click the value in the **Protocols and Ports** column to open a panel on the right side of the window.
6. In the **Protocols and Ports** section, the color of each **C** indicates the confidence for each service (port and protocol) specified in the policy.

   To interpret the confidence level, hover over the **C**.
7. Look for low-confidence indicators for any services in the list.
8. If applicable, delete or edit unwanted policies, or add additional policies.

**To view the confidence levels for a particular policy:**

1. In the Policies tab, click the value in the **Protocols and Ports** column for that policy.

   The Policy Side View panel opens at the right side of the window.
2. In the **Protocols and Ports** section, the color of each **C** indicates the confidence for each service (port and protocol) specified in the policy.

   To interpret the confidence level, hover over the **C**.

### Flow Direction and Policy Confidence

The accuracy of discovered policies depends on correct identification of the flow direction. If flow direction is incorrectly identified, the confidence rating of automatic policy discovery results may be reduced. For information about determination of flow direction for the conversation(s) analyzed for the creation of the policy, see Client Server Classification.

## Troubleshoot Automatic Policy Discovery Results

If automatic policy discovery results are not what you expect, check the following:

### Extend the selected time range to include more data

Extend the time window to include more data and to capture events that happen infrequently. For example, if an application generates a complex quarterly report using data drawn from several provider applications, be sure to include a time range that includes that traffic.

### Avoid data gathered before certain changes

If the scope definition has changed, or data gathered before a certain time has become invalid for some other reason, be sure your time range does NOT include data before that point.

### Exclude misleading traffic flows

Exclusion filters may need to be configured or modified.

There are multiple places exclusion filters can be configured, and multiple places they can be enabled or disabled. Check each location:

- Check the exclusion filters configured for the workspace.
- Check the default exclusion filters configured at the bottom of the Default Policy Discovery Config page.
- Check which exclusion filters are enabled in the Advanced Configurations section in the workspace's settings for automatic policy discovery.
- Check which exclusion filters are enabled in Advanced Configurations section on the Default Policy Discovery Config page.
- If you are using Default Exclusion Filters, make sure you have clicked **Save** on the **Default Policy Discovery Config** page to make those configurations available to individual workspaces.

For details, see Exclusion Filters, on page 29 and subtopics.

### Troubleshoot Policies in which Consumer and Provider Are in Different Scopes

See Troubleshoot Cross-Scope Policies, on page 100.

### Check Status of Approved Policies

See Troubleshoot Approved Policies, on page 48.

## Policy Visual Representation

Policy visual representation provides a graphical view of the policies.

To navigate to the policy visual representation page: On the Policies page, click the graph icon (⑂) to the right of the list icon.

### Policy View Elements

The visual elements on the policy view are:

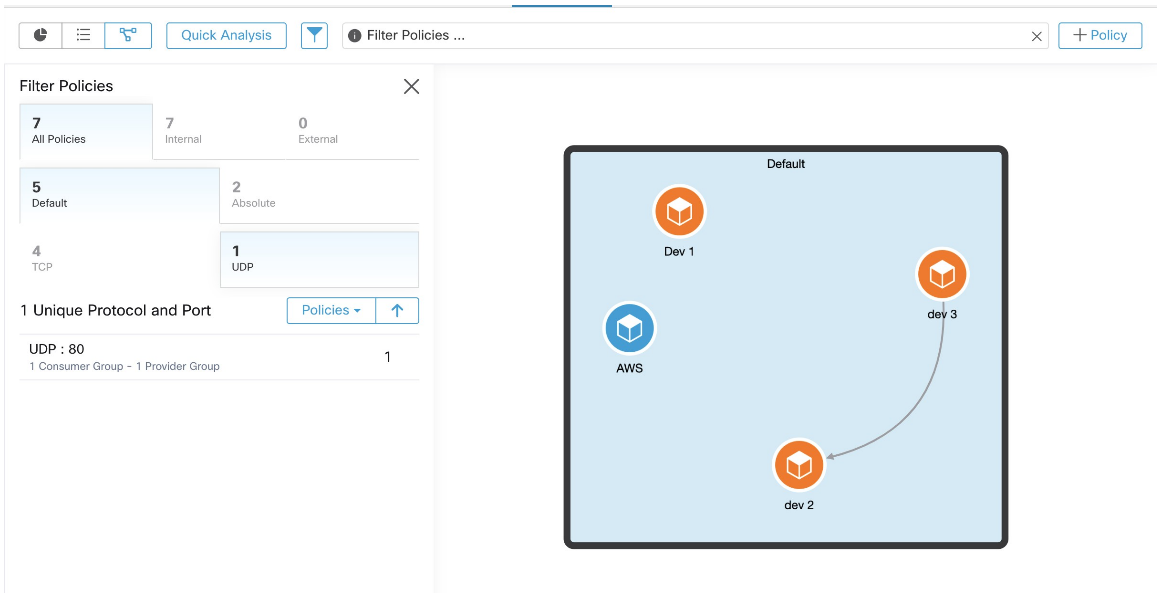
| This Element | Represents This |
|---|---|
| A blue, orange, or purple icon | A node (the consumer or provider of a policy) |

| This Element | Represents This |
|---|---|
| Blue icon | A scope |
| Orange icon | An inventory filter |
| Purple icon | A cluster |
| Line connecting two icons | One or more policies. |

## Policy View Options

| To | Do This |
|---|---|
| View the list of workloads included in a consumer or provider node | Double-click the node's icon. |
| View policy specifics such as services (ports), action (Allow/Deny) and protocol between a consumer and provider | Double-click the line connecting them. Details appear in the pane on the right. |
| View the policies entering and leaving a node | Click the icon. |
| View only the policies between workloads within the scope | Click the **Internal** button. |
| View only the policies in which the provider is in a different scope from the consumer | Click the **External** button. |
| Use advanced filtering options | Click the (i) button to the left of the filter text input box to see the options, then enter filter criteria. |

*Figure 54: Filtering policies in graphical view*



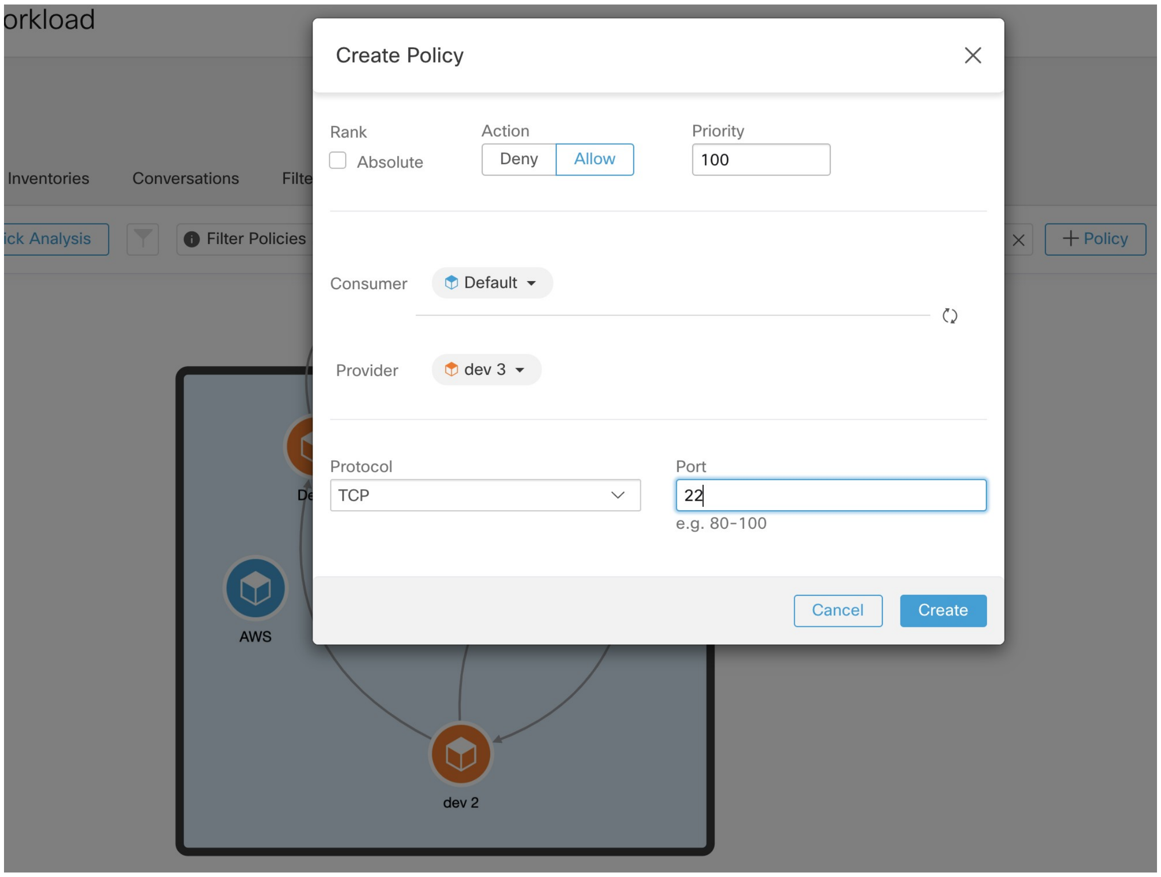
To download a high resolution image of the graphical view of the policies:

1. In the lower-right corner of the graph, click the ellipsis icon, and then click **Export Image**.
2. Select the required resolution and image type.
3. Click **Download**.

### Add a Policy (Policy View Page)

To create a policy, hover over the consumer until you see a "+" sign and then hold and drag the policy onto the provider. To create an Absolute policy, toggle the Absolute checkbox in the modal. Otherwise, the policy is created as a Default policy. Policies can also be managed by clicking a line and selecting a policy from the pop-up list. Policies will be displayed in the sidebar.

***Figure 55: Policy creation in graphical view***



Create Policy

Rank: Absolute [ ]

Action: Deny [✓ Allow]

Priority: 100

Consumer: Default

Provider: dev 3

Protocol: TCP

Port: 22

e.g. 80-100

## Quick Analysis

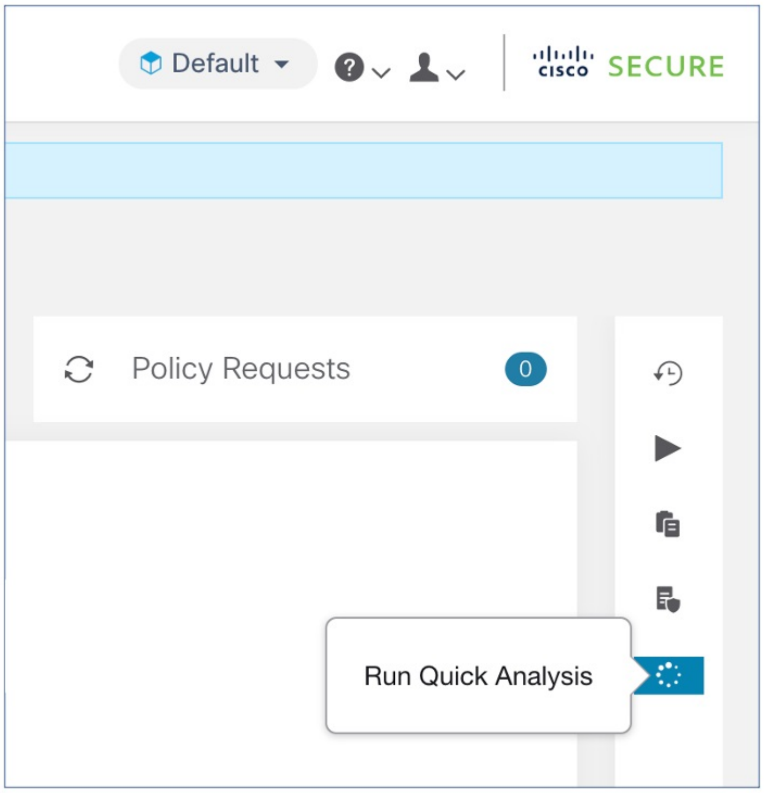
Quick analysis enables testing a hypothetical flow against all the policies in the current workspace and all other relevant policies from other workspaces. Quick analysis facilitates debugging and experimentation with different security policies, without the need to run live policy analysis for the workspace.

**Restriction**

- You can run Quick Analysis only on primary workspaces.
- Quick Analysis is not currently supported on flows from Kubernetes services.

Click the **Run Quick Analysis** tab on the right navigation pane to view the dialog.

*Figure 56: Quick Analysis Tab*



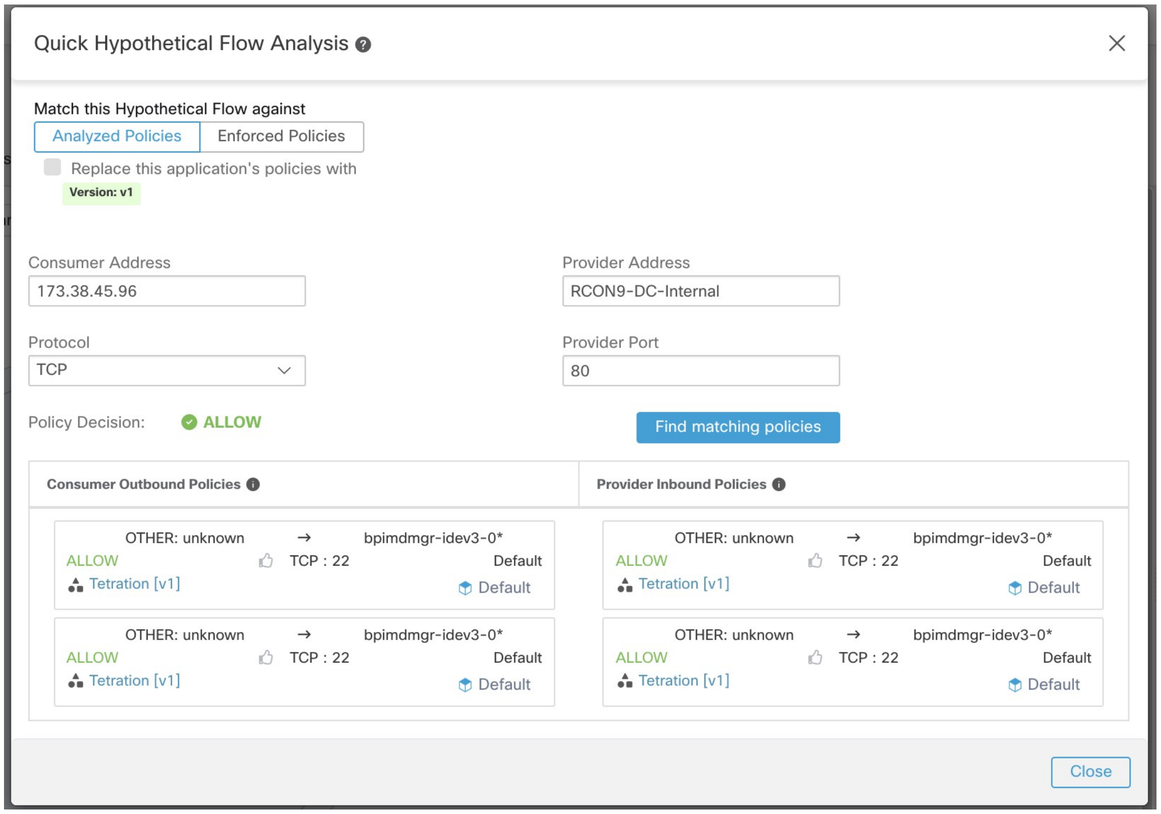
Enter the Consumer (client) IP, Provider (server) IP, port, and protocol for the hypothetical flow, then click **Find Matching Policies** button.

A policy decision will be shown indicating whether the hypothetical flow would be allowed or denied given the policy definitions in the latest version of the workspace and all other policies from relevant workspaces that are already pushed for live policy analysis.

At the bottom of the dialog, we show the matching outbound and inbound policies separately, and in their globally sorted order. It is only the first row on either side that has any effect. For a connection to successfully get established, we need both the top outbound rule on consumer and the top inbound rule on the provider side to be ALLOW rules.

Showing all other matching policies in their order, provides a valuable debugging tool to help sort out issues in policy definitions when a certain policy seems to not be taking any effect. You can add, update, or delete policies from the workspace, and repeat the analysis immediately without the need to run live policy analysis on the workspace.

*Figure 57: Quick Policy Analysis*



Quick Hypothetical Flow Analysis

Match this Hypothetical Flow against

Replace this application's policies with Version: v1

Consumer Address: 173.38.45.96

Provider Address: RCON9-DC-Internal

Protocol: TCP

Provider Port: 80

Policy Decision: ALLOW

| Consumer Outbound Policies | Provider Inbound Policies |
|---|---|
| OTHER: unknown → bpimdmgr-idev3-0* ALLOW TCP : 22 Default Tetration [v1] Default | OTHER: unknown → bpimdmgr-idev3-0* ALLOW TCP : 22 Default Tetration [v1] Default |
| OTHER: unknown → bpimdmgr-idev3-0* ALLOW TCP : 22 Default Tetration [v1] Default | OTHER: unknown → bpimdmgr-idev3-0* ALLOW TCP : 22 Default Tetration [v1] Default |

## Live Policy Analysis

After you have reviewed and approved the set of network security policies generated by automatic policy discovery, and before you enforce the policies, you should use live policy analysis to observe how the policies would affect actual traffic on your network.

Some questions that live policy analysis can help you answer:

- What would be the impact on this scope's application(s) if the policies in this workspace are enforced now?
- Could we have prevented a previously known security attack/risk by enforcing the new set of policies?

  See Run Policy Experiments to Test Current Policies Against Past Traffic, on page 118.
- Are our policies working the way we expect them to?

You should run policy analysis on any workspace that has policies. Because workloads in any particular scope can be affected by policies in other scopes, you should not run policy analysis only for a single scope before enforcing policy for that scope. Consider analyzing policies for all scopes that may affect traffic in a particular scope.

For example:

- Policies defined in scopes above this scope in the tree may apply to workloads in this scope.

- If workloads in this scope communicate with workloads in a different scope, policies in that scope may affect these communications. When policy analysis is started in that scope (or latest policies are analyzed after a policy change there), this can affect this scope's policy analysis results.

You should perform policy analysis any time you revise policies, to ensure that changes don't break applications.

Running live policy analysis on a workspace is sometimes referred to as "publishing" a workspace.

## Start Live Policy Analysis

Once you have reviewed the policies generated in a workspace by automatic policy discovery, and you believe that they are as you want them, you can start policy analysis.

### Before you begin

**Important** Live analysis includes the effects of policies in other workspaces that are also running live analysis. If you have enabled enforcement on any workspace, but analysis is not running on that workspace or the enforced version of the policies is not the same as the analyzed version of the policies, your live analysis results for this workspace may not be accurate.

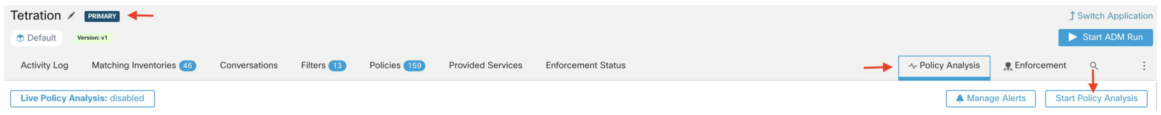
### Procedure

**Step 1** Toggle the workspace to **Primary** by clicking ••• to the right of "Secondary" next to the workspace name in the header.

**Step 2** Navigate to the **Policy Analysis** tab.

**Step 3** Click **Start Policy Analysis** on the right.

***Figure 58: Enable Policy Analysis***



### What to do next

- Because policies in other scopes can apply to workloads in this scope, consider simultaneously analyzing policies in other scopes that could affect analysis results in this scope. See Example: Impact of Policies Analyzed in Other Scopes, on page 114.
- If you want to be notified when escaped flows are detected, click **Manage Alerts**.
- Use the tools on the page to filter the data. To see the available filter criteria, click the (i) button in the filter box.
- If you add or change policies after initiating policy analysis, you must re-start analysis in order to include the changes in the analysis. See After Changing Policies, Analyze Latest Policies, on page 119.

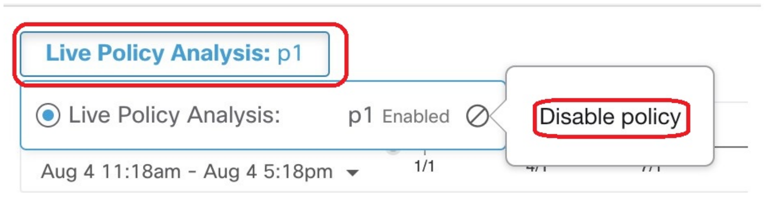
## Stop Live Policy Analysis

In general, you should let policy analysis continue to run, even after you enforce policies, since the policies in this workspace may affect policy analysis results in other workspaces you are analyzing.

To stop live policy analysis:

Click the **Live Policy Analysis: P<number>** button, then click **Disable policy**:

*Figure 59: Disable Live Policy Analysis*

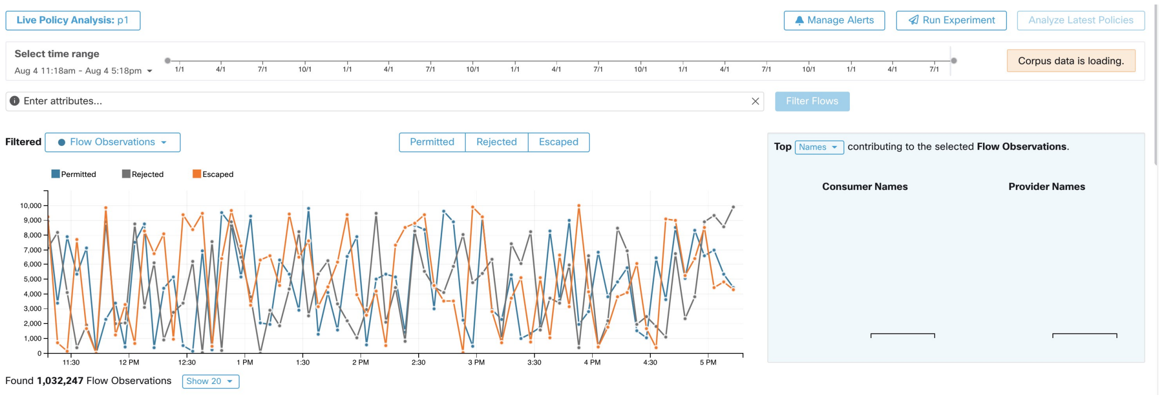

Live Policy Analysis: p1
Live Policy Analysis: p1 Enabled
Disable policy
Aug 4 11:18am - Aug 4 5:18pm
1/1

## Policy Analysis Results: Understand the Basics

During policy analysis, all flows traveling into, out of, and within the scope associated with the workspace are assigned one of the following results:

- **Permitted**: Flow was allowed by the network, and also by the analyzed policies.
- **Escaped**: Flow was allowed by the network, but should have been dropped according to the analyzed policies.
- **Rejected**: Flow was dropped by the network, and also by the analyzed policies.

*Figure 60: Policy Analysis Page*



Some things to look at to get oriented:

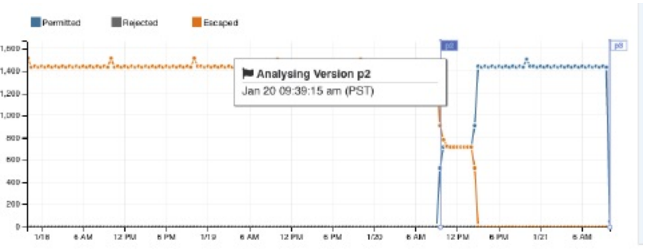
- You may filter the flow information presented in this page via a faceted filter bar. Clicking the **Filter Flows** button updates all the charts accordingly.
- Hovering on the chart shows the percentage of the aggregate observed flows at that timestamp.
- Clicking on a timestamp reveals a list of all filtered flows in a table below for further analysis.
- You can limit the interactions to one of the three result types by selecting or deselecting the types at the top of the time series charts.

- The Top N chart on the right shows the top Hostnames, Addresses, Ports, and so on contributing to the data shown in the time series chart on the left.

  You can limit the time series chart to escaped flows and select “Ports” in the Top N chart to see the top ports contributing to escaped flows.

## Example: Impact of Policies Analyzed in Other Scopes

In the following example, there are permitted flows up to about 12pm. At that time, policy analysis was started in a workspace associated with a different scope, affecting traffic with workloads in this scope and causing flows to be marked as escaped. (You know that this change did not result from policy changes newly analyzed in this workspace, because this would have created a label flag.)



### Analysis without Policies

The flows into, out of, and within the scope associated with the workspace may be affected by policies in other workspaces that are being analyzed. If live policy analysis is not enabled on this workspace, the flows will be marked with those of the other workspaces in the system that do have live policy analysis enabled.

**Note** If no workspaces are running live policy analysis, the timeseries chart will be empty.

## Policy Analysis Details

**Flow Disposition**

In policy live analysis, to decide on whether a flow is **Permitted**, **Escaped**, or **Rejected**, we have to first determine the **Disposition** of the flow from the network perspective. Each flow will receive an **ALLOWED**, **DROPPED** or **PENDING** disposition, based on the signals and observations given by Secure Workload agents. There are a number of scenarios based on the agent configurations along the path of the flow and the flow types.

First, regardless of flow types, if any agent along the path of a flow reports that the flow is DROPPED, the flow will receive a DROPPED disposition.

When there is no DROP reported by any agents along the path of the flow, we consider the case of bidirectional flows and unidirectional flows separately. When bidirectional flows are observed, we look at flows in pairs (forward and reverse) based on their source, destination ports and protocol, and timings. The same cannot be done for unidirectional flows.

For bidirectional flows, if there are agents installed and data plane enabled on both ends, a forward flow will receive an ALLOWED disposition if both the source and the destination agent report that the flow is observed. Otherwise, the forward flow will get a PENDING disposition. If an agent is installed on either the source or the destination workload, but not both, then the forward flow will received an ALLOWED disposition if and only if the agent observes subsequent reverse flow within a **60**second window. Otherwise a PENDING status will be assigned to the forward flow. The disposition of the reverse part of the bidirectional flow follows the same logic except that now the source and the destination are reversed. For example, if only one side has an agent, whether a reverse flow disposition is PENDING or ALLOWED depends on the observation and timing of its subsequent forward flow based on the same logic.

Note that we assume firewalls implement silent drop. If a reject message is sent on the *same* flow (for example, rejecting a TCP SYN with RST + ACK), a reverse flow will be detected, and the previous forward flow will be marked as ALLOWED. However if the reject message is sent on a *different* flow (for example, rejecting a TCP SYN with an ICMP message), the forward flow will remain as PENDING.

For a unidirectional flow, the flow will be considered DROPPED if it is reported as DROPPED by any agent as in the case of bidirectional flows. However, since there is no matching reverse flow, the flow will have PENDING disposition status if both agents observe the flow.

**Violation Types**

The flow dispositions are checked against the policies being analyzed to determine the final violation types.

A flow’s violation type will be

- **Permitted**, if its disposition is ALLOWED or PENDING, and its deciding policy action is ALLOW,
- **Escaped**, if its disposition is ALLOWED, and its deciding policy action is DENY,
- **Rejected**, if its disposition is DROPPED or PENDING, and its deciding policy action is DENY,

A DROPPED status is assigned only to flows whose relevant agents explicitly report their DROPPED status. When there is no explicit report of dropping for agents, the flow receives PENDING status.

When disposition is PENDING:

- and policy action is DENY, then violation type is set to Rejected.
- and policy action is ALLOW, then violation type is set to Permitted.

For a bidirectional flow, if the policy violation types of forward and reverse part of the flow agree, only a single type is shown in the policy analysis or enforcement analysis page. Otherwise, forward and reverse are shown separately, such as PERMITTED:REJECTED.

**Example scenarios:**

- Packets are dropped at the source-side enforcement.
    - In this case, the source side Secure Workload egress agent will report that the flow is DROPPED.
- Packets leave the source.
    - If there is only an agent on the source side, the flow will be reported as ALLOWED by the egress agent if a reverse packet is also observed by the agent within 60 seconds.
    - If there is a visibility-only agent on both the source and the destination side, the flow will be given a DROPPED disposition status, if and only if the ingress agent reports that the flow is DROPPED. Otherwise, the flow will be reported as ALLOWED.
    - Flow packets are received at the destination, but no reverse traffic.

      If there is no destination side agent, the flow will receive a PENDING status. Otherwise, it will be assigned ALLOWED status.

## Suggested Steps for Investigating Flows

When drilling into specific flows when examining policy results, the following suggestions and filters may be helpful:

1. Focus on *ESCAPED FLOWS* initially:

   **Escaped** flows require special attention as their actual flow dispositions differ from the intended actions based on the currently analyzed policies. Investigate to ensure that enforcing these policies does not block needed flows and adversely impact your applications.

   Click the violation type, such as **Escaped**.

   (Later, you can look at rejected and permitted flows as needed.)

   Escaped flows can occur for many reasons, including but not limited to:

   • Another policy higher in the priority order is taking effect

   • The traffic is taking a different path than the route that your policies address, or

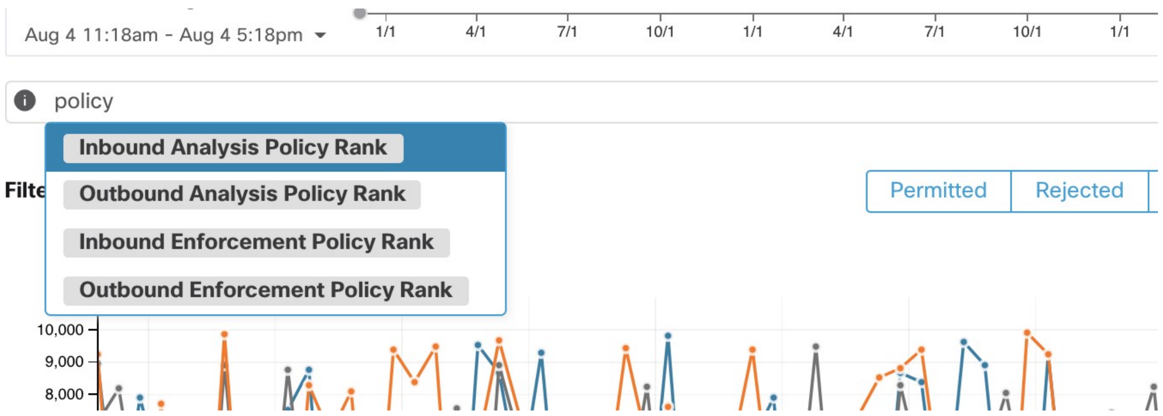
   • The policy that you expect the traffic to hit is in a workspace that is not being analyzed (if you are looking at escaped flows on the Policy Analysis page) or enforced (if you are looking at escaped flows on the Enforcement page), for example in an ancestor scope or even in a secondary workspace in the same scope.

2. Identify flows that matched the catch-all policy (inbound and outbound) :

   It is important to understand what flows are matched to catch-all policies, especially in an allow-list policy model. If these flows are legitimate but do not have explicit allow policies configured for them, you may want to add appropriate explicit policies in the corresponding inbound or outbound scopes. If they are suspicious flows, you want to quickly identify them and further investigate their details.

   To focus on these flows, apply filters based on the *catch-all* value of **inbound_policy_rank** or **outbound_policy_rank**, depending whether you are looking at the inbound, outbound or both sides, shown below.

   ***Figure 61: Policy Analysis Filtering Options for Rank***


Aug 4 11:18am - Aug 4 5:18pm
1/1
4/1
7/1
10/1
1/1
4/1
7/1
10/1
1/1
policy
Inbound Analysis Policy Rank
Outbound Analysis Policy Rank
Inbound Enforcement Policy Rank
Outbound Enforcement Policy Rank
Filte
Permitted
Rejected
10,000
9,000
8,000

3. Filter out TCP flows with RST: *Fwd flags do not contain RST, Rev flags do not contain RST*

   Some escaped TCP flows have RST flags set. These flows are reset by either their consumers or providers. They are unestablished connections without data exchange, but may be reported as ALLOWED because the agents see their handshaking packets. Since they do not have established connections to begin with, they will not be affected when currently analyzed policies are enforced. Filtering out TCP flows that have the RST flag on either side allows you to focus on more meaningful and important escaped flows whose established connection will be blocked by the currently analyzed policies.

4. If most traffic is using IPv4, focus only on IPv4 flows:

   Filter using *address type = IPv4, address type != IPv6*. It is also helpful to filter out *link-local* address.

5. Prioritize which flows to focus on in the next diagnostic step by identifying the most frequent hostnames, ports, addresses, scopes, and so on involved in the escaped traffic:

   Select *Hostname, Ports, or Addresses* from the TopN feature pane. You can usually combine these with other filters to drill down to a particular type of traffic when diagnosing policies.

6. Search flow data for the hostnames, ports, protocols, etc. identified in the previous step

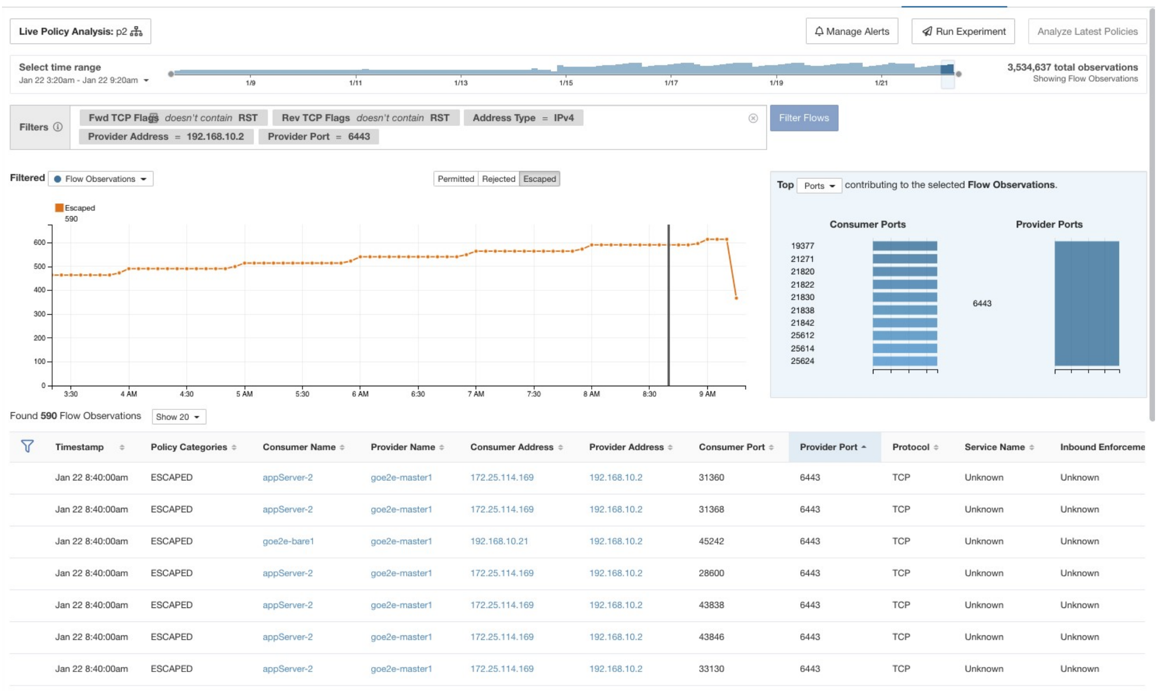
   Once you have an idea about the top candidates based on the targeted flows' hostnames, port and and so on, you can choose to drill down into the flows by either applying drill-down filters directly from values given in the top N query window, or by manually entering relevant filters into the flow search filters bar. For example, *Consumer Hostname contains {something}, Provider Hostname contains {something}, Provider Port = {some port number}, Protocol = TCP Protocol != ICMP*

7. Check individual flows and quick analysis:

   Finally, you can focus on a specific flow to examine its policy result by clicking the table row corresponding to the flow. Pay attention to the policies matched to the flow and the scopes of both the consumer and the provider addresses. If the policy action does not match your intended action, you need to create appropriate policies in workspaces associated with the consumer’s and/or the provider’s scopes to change the policy action.

   The figure below shows an example workflow of narrowing down escaped flows using some of the filtering described above. The search input also supports “,” and “-” for Port, Consumer Address and Provider Address, by translating “-” into range queries.

*Figure 62: Policy analysis diagnosis example*



| Timestamp | Policy Categories | Consumer Name | Provider Name | Consumer Address | Provider Address | Consumer Port | Provider Port | Protocol | Service Name | Inbound Enforceme |
|---|---|---|---|---|---|---|---|---|---|---|
| Jan 22 8:40:00am | ESCAPED | appServer-2 | goe2e-master1 | 172.25.114.169 | 192.168.10.2 | 31360 | 6443 | TCP | Unknown | Unknown |
| Jan 22 8:40:00am | ESCAPED | appServer-2 | goe2e-master1 | 172.25.114.169 | 192.168.10.2 | 31368 | 6443 | TCP | Unknown | Unknown |
| Jan 22 8:40:00am | ESCAPED | goe2e-bare1 | goe2e-master1 | 192.168.10.21 | 192.168.10.2 | 45242 | 6443 | TCP | Unknown | Unknown |
| Jan 22 8:40:00am | ESCAPED | appServer-2 | goe2e-master1 | 172.25.114.169 | 192.168.10.2 | 28600 | 6443 | TCP | Unknown | Unknown |
| Jan 22 8:40:00am | ESCAPED | appServer-2 | goe2e-master1 | 172.25.114.169 | 192.168.10.2 | 43838 | 6443 | TCP | Unknown | Unknown |
| Jan 22 8:40:00am | ESCAPED | appServer-2 | goe2e-master1 | 172.25.114.169 | 192.168.10.2 | 43846 | 6443 | TCP | Unknown | Unknown |
| Jan 22 8:40:00am | ESCAPED | appServer-2 | goe2e-master1 | 172.25.114.169 | 192.168.10.2 | 33130 | 6443 | TCP | Unknown | Unknown |

## Run Policy Experiments to Test Current Policies Against Past Traffic

If a known attack or other significant short-term traffic pattern occurred in the past, and you want to see how your current policies (or another versioned policy set) would have handled that traffic, you can use the Run Experiments feature.

### Before you begin

**Tip** As an alternative to this procedure, you can run automatic policy discovery again, including the relevant time range, and see what different policies are suggested.

### Procedure

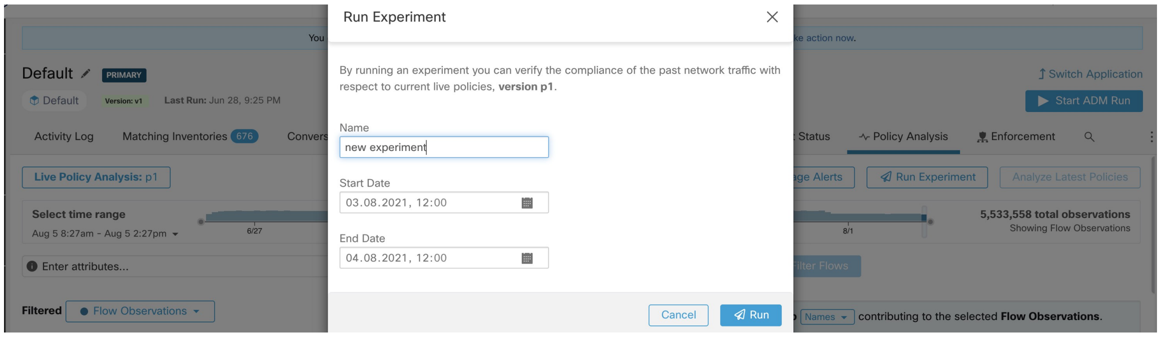
**Step 1** Navigate to the Policy Analysis page of your selected workspace.

**Step 2** From the top of the page, select the policy version to test.

**Step 3** Click **Run Experiment**.

**Step 4** Enter a name and a duration for the policy experiment.

*Figure 63: Run Experiment Form*


Run Experiment

By running an experiment you can verify the compliance of the past network traffic with respect to current live policies, **version p1**.

Name: new experiment

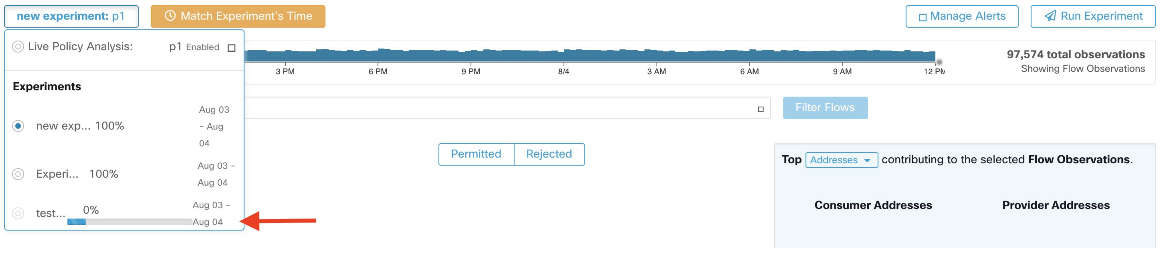
Start Date: 03.08.2021, 12:00

End Date: 04.08.2021, 12:00

This will start a new policy analysis job which goes back in time and re-analyzes all the flows in the selected duration against the selected versioned policy.

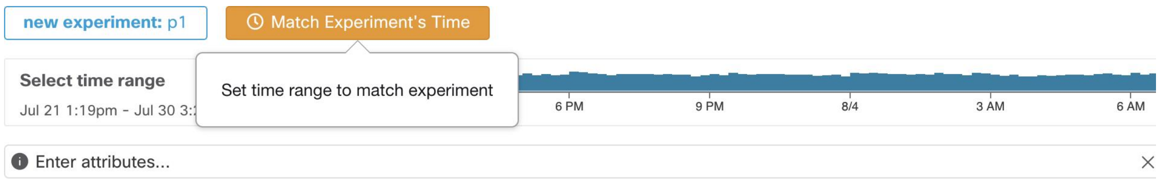
This job may take a few minutes, depending on the selected duration. The progress is shown in the policy selector menu. When the results are ready to be presented, you should be able to select the policy experiment like any other versioned policy and the time series charts showing different flow categories will be updated accordingly.

*Figure 64: View Experiment Status*


new experiment: p1
Match Experiment's Time
Manage Alerts
Run Experiment
Live Policy Analysis: p1 Enabled
Experiments
new exp... 100% Aug 03 - Aug 04
Experi... 100% Aug 03 - Aug 04
test... 0% Aug 03 - Aug 04
97,574 total observations
Showing Flow Observations
Filter Flows
Permitted
Rejected
Top Addresses contributing to the selected Flow Observations.
Consumer Addresses
Provider Addresses

**Note** If you cannot see any flows when selecting a policy experiment, it might be due to time range mismatch, for example, the current time range of the charts is the past 1 hour, but the experiment duration is 6 hours in the past. To reset the time range to the duration of the experiment, click the clock icon next to the policy selector.

*Figure 65: Match Time Range*



## After Changing Policies, Analyze Latest Policies

Policy analysis does not automatically reflect policy changes in the workspace. When you are ready to analyze the current set of policies after making changes, click **Analyze Latest Policies** so policy analysis reflects the changes.

If the policies in the workspace have not changed since policy analysis was last initiated, or if policy analysis is not currently enabled, the Analyze Latest Policies button is not available. If the button is clickable, there are policy changes that have not yet been included in analysis.

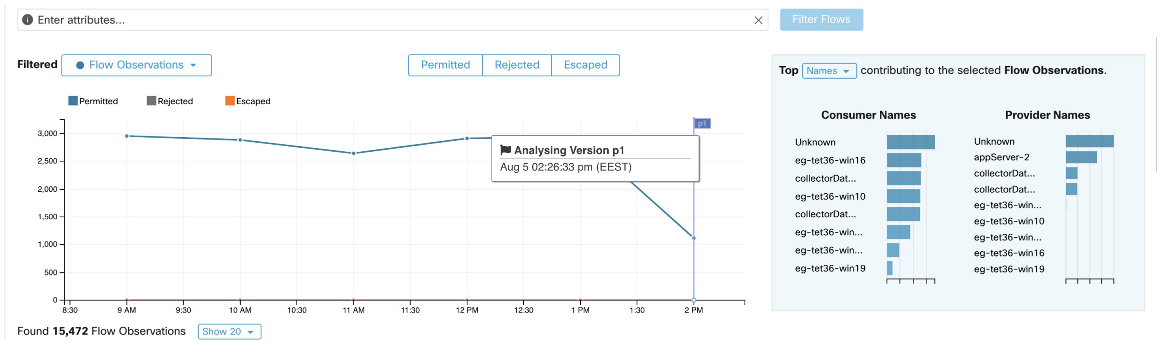
See also View, Compare, and Manage Analyzed Policy Versions, on page 120.

## Policy Label Flags

On the policy analysis timeseries chart, policy label flags mark the point at which analysis was initiated and at each point analysis was re-initiated to reflect the latest policy and cluster changes.

Click a flag to view the version of the policies associated with that flag:

***Figure 66: Policy label flag in timeseries chart***



Clicking a policy label flag opens the corresponding version of the Policies page and displays the policies analyzed by that policy analysis version.

## View, Compare, and Manage Analyzed Policy Versions

Each time you analyze or re-analyze policies in a workspace after making changes, a new analysis version (p*) is created.

For details about versioning, see About Policy Versions (v* and p*), on page 136.

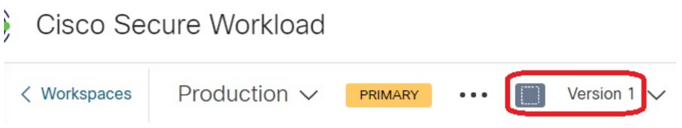
### Procedure

**Step 1** Click **Defend** > **Segmentation**.

**Step 2** Navigate to the relevant scope and primary workspace.

**Step 3** Click **Manage Policies**.

**Step 4** The currently displayed version of the policies is shown at the top of the page:



The displayed version may be a policy discovery version, an analyzed policy version, or an enforced version.

**Step 5** You can:

| | |
|---|---|
| To display a different version of the policies: | Click the current version and choose a different version.<br>For descriptions of the versions, see About Policy Versions (v* and p*), on page 136.<br>**Important!** If you choose a v* version, see View, Compare, and Manage Discovered Policy Versions, on page 50 instead of this topic, including the important caveat at the end of the topic. |
| To view details about the analyzed versions: | **a.** Click **View Version History** at the top of the page beside the current version.<br>**b.** Click the **Published Versions** tab to see the versions of analyzed and enforced policies.<br>**c.** To view log entries for a version, click the link in the version.<br>Pale green rows represent analysis activity.<br>Bright green rows represent enforcement activity. |
| To compare two versions to see what has changed: | **a.** Click **Compare Revisions**.<br>**b.** Choose the versions to compare.<br>You can compare the latest draft version, analyzed and enforced versions.<br>**c.** For result details, see Comparison of Policy Versions: Policy Diff, on page 139. |
| To delete an unwanted version: | Click ⋮ for the version and choose **Delete**.<br>Published policy versions (p* versions) can be deleted as long as the version is not being actively analyzed or enforced. |
| To export a version: | Click ⋮ for the version and choose **Export...**.<br>See also Export a Workspace, on page 55. |

### What to do next

When you are done working with versions, change the version at the top of the workspace page to the latest discovered policy version (v*).

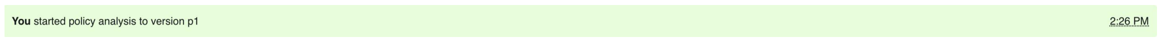
This avoids unintentional deletion of discovered policy versions and allows you to manually create policies in the workspace.

## Activity Logs of Policy Analysis

All workspace users may view activity logs associated with changes done on the policy analysis page in the workspace history (see Activity Logs and Version History ).

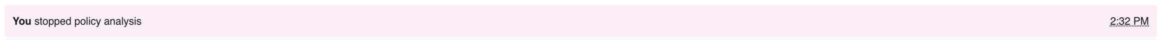
- Enable policy analysis

*Figure 67: Enable policy analysis*


**You** started policy analysis to version p1 2:26 PM

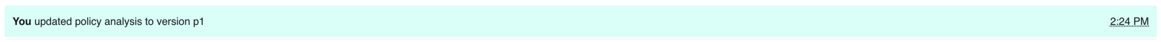
- Disable policy analysis

*Figure 68: Disable policy analysis*


**You** stopped policy analysis 2:32 PM

- Update policy analysis

*Figure 69: Update policy analysis*


**You** updated policy analysis to version p1 2:24 PM

## Enforce Policies

Secure Workload can enforce policies using:

- Deploy Software Agents on Workloads installed on individual workloads:
    - Linux
    - Windows
    - Kubernetes/OpenShift

  For technical details about how agents work on each platform, see Policy Enforcement with Agents and Enforcement on Containers, on page 129.
- Cloud connectors:
    - AWS through AWS Connector
    - Azure through Azure Connector
- Integrate load balancers through an external orchestrator:
    - F5 BIG-IP
    - Citrix Netscaler
- Integration with Cisco Secure Firewall Management Center
- Streaming to third-party orchestrators for enforcement in third-party infrastructure

**Caution** When you enforce policies, the system inserts new firewall rules on affected hosts and deletes any existing rules on the relevant hosts.

## Check Agent Health and Readiness to Enforce

Some of these checks can be done before or after enforcing policy.

Permissions may be required to modify agent or connector capability; see the requirements and prerequisites in the relevant chapters.

You do not need to perform these checks for any workloads on which you do not intend to enforce policies.

| **Verify That:** | **More Information** |
|---|---|
| Agents are installed on all workloads in the scope that are associated with the enforced workspace | Click **Defend** > **Segmentation** and navigate to the relevant scope and workspace. Click **Matching Inventories**, then click **IP Addresses**.<br>IP addresses on this tab generally do not have agents that are installed, and agents generally must be installed to enforce policy.<br>Exceptions: Enforcement occurs for the following types of inventory that appears on the IP Addresses tab:<br>• Cloud-based inventory on which policy is enforced using a cloud connector. (Installing agents on individual workloads is optional.)<br>• Kubernetes addresses appear in the IP Addresses list if agents are installed on individual workload pods; Kubernetes inventory with installed agents appears on the Pods tab. |
| The installed agent version is current and supported | For an overview of installed agent versions, click **Manage** > **Agents**, then click **Distribution** and look at the **Agent Software Version Distribution** chart.<br>For details, click **Manage** > **Agents**, then click **Agents List**. |
| Installed agents have enforcement capability | Click **Manage** > **Agents**, then click **Convert to Enforcement Agent**.<br>In the **Filter** box, enter **`Agent Type = Deep Visibility`**<br>Convert any agents that must enforce policy. |
| Enforcement is enabled for all agents | (This requirement is distinct from ensuring that agents have enforcement capability and from enabling enforcement in the workspace.)<br>**Important!** Depending on your deployment, this may need to be done before or after you enforce the workspace.<br>See that the Verify Enforcement is Enabled for Agents section. |
| Enforcement is enabled for nonagent enforcement mechanisms | **Important!!!** Do not enable enforcement on cloud connectors without agents until AFTER you enforce policy on the workspace.<br>External orchestrators that support enforcement must also be enabled before they can enforce. |

| Verify That: | More Information |
|---|---|
| The **Preserve Rules** setting in the Agent Config Profile is appropriate for the workload platform | • For Kubernetes/OpenShift, see Enforcement on Containers section.<br>• For other platforms, see information for each platform in Software Agents section.<br><br>Tip: Search this document for "Preserve Rules" to find useful information. |
| (After the workspace is enforced) All agents have received the applicable policies for the workload | See the Verify Enforced Policies are being pushed to Agents section. |
| Agents are healthy | In addition to the sources above, the following locations have information about agent health:<br>• Click **Manage > Agents**, then click **Monitor**.<br>Look at the information under **Enforcement Agents**. For details, see theAgent Monitoring section.<br>• Click **Manage > Agents**, then click **Distribution**.<br>Choose the agent type from the top of the page.<br>For information about this page, see the Agent Status and Statistics.<br>• Click **Organize > Scopes and Inventory**, filter to find a specific workload of interest, and click the IP address.<br>The **Workload Profile** page opens in a separate browser window including an Agent Health panel.<br>For details, see Workload Profile section. |

## Enable Policy Enforcement

**Caution** Enforcing policies delete existing firewall rules and write new firewall rules on every workload in the scope that is affected by this workspace.

If you have not properly verified that your policies are working correctly, enforcing policies can change the way that your applications work and disrupt business operations.

### Before you begin

- Initially, when you enforce policies, consider setting the catch-all to Allow. Then, monitor traffic to see what matches the catch-all rule. When no necessary traffic is matching the catch-all rule, you can set the catch-all to Deny.
- If you enforce workspaces in multiple scopes at once, you can enforce only analyzed workspaces. If you enforce a single workspace using the second method that is described in the procedure below, analyzing the policies in the workspace before you enforce them is recommended but not required.

See Live Policy Analysis and subtopics.

- The wizard for enforcing a single scope is more detailed than the wizard that offers the option to enforce multiple scopes simultaneously. If you require the features in Policy Enforcement Wizard, on page 127, use the second method that is described in the procedure below.
- **IMPORTANT!** Verify that the policies are correct.

  Policy results in any workspace may be affected by enforced policies in other scopes. Before policy enforcement is enabled on a workspace, the Policy Enforcement page shows how flows are affected by enforced policies in the workspaces associated with other scopes. For example, a broad "Production hosts should not talk with Non-Production hosts" policy in the enforced workspace of a parent scope may impact traffic on workloads belonging to an application in a child scope.

  If no new information is being shown in the Enforcement charts, make sure that the correct time range is selected.

  For information about the information you see on the Enforcement page, see Live Policy Analysis and subtopics. (The same information for Live Analysis also applies to the Policy Enforcement page.)

  If live analysis results differ from results on the Enforcement page, make sure the scopes, policy versions, and time range being analyzed are the same as the scopes, policy versions, and time range being used to generate results on the Enforcement page.
- Understand how agents enforce policies on each platform. See:
  - For Windows and Linux workloads, see Policy Enforcement with Agents and subtopics.
  - For Kubernetes and OpenShift, see

    Enforcement on Containers, on page 129.
  - For load balancers, see

    Policy enforcement for Citrix Netscaler and

    Policy enforcement for F5 BIG-IP.
  - For cloud-based workloads configured using cloud connectors, see:
    - Best Practices When Enforcing Segmentation Policy for AWS Inventory and linked topics.
    - Best Practices When Enforcing Segmentation Policy for Azure Inventory and linked topics.

- You must have the required permissions to enforce policies:

  You must have the Enforce ability or higher for the scope. Users with other abilities on the scope can still view this page but will not be able to enforce (or disable) new policies.
- Verify that all relevant installed agents and other enforcement endpoints such as cloud connectors are ready to enforce policy. For a list of agent health and readiness checks, see Check Agent Health and Readiness to Enforce, on page 123.

**Note** Some of these checks must wait until after enforcement; for example, you should enable enforcement on cloud connectors only after you have enabled enforcement in the workspace. For installed agents, you will typically enable enforcement in the agent configuration before you enforce the workspace.

### Procedure

**Step 1** From the navigation pane, choose **Defend** > **Segmentation**.

**Step 2** You can enforce policies for one scope or for multiple scopes at the same time:

To enforce policy for multiple scopes at the same time:

(Only workspaces that have been analyzed can be enforced using this process.)

a) Click the caret on the right side of the page to display the Tools pane:
b) Click **Enable Enforcement**.
c) Click **Next** to start the wizard.
d) Select one workspace to enforce.

   (The option to enforce workspaces for additional scopes is on the last page of the wizard.)

e) Click **Next**.
f) Choose the version of that workspace to enforce, then click **Next**.
g) To simultaneously enforce policies for another scope, click **+ Add Another Workspace** and complete the steps.

   Repeat as needed for additional scopes.

h) Click **Accept and Enforce**.

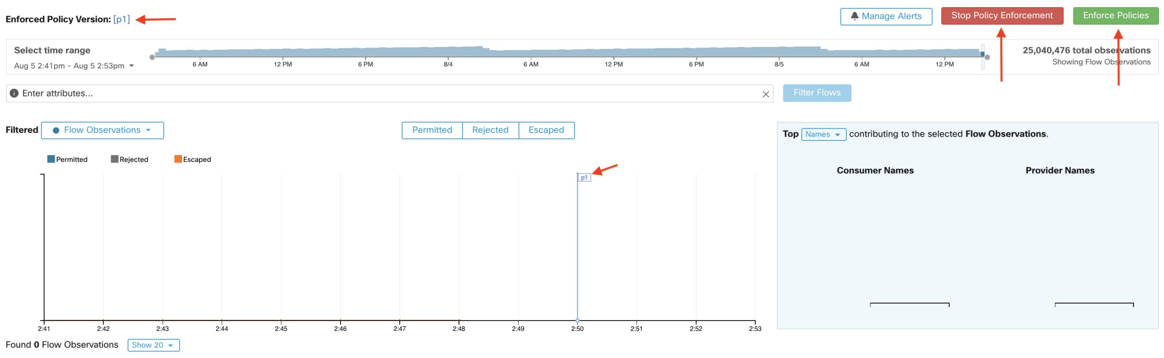
To enforce policies for a single scope:

a) Navigate to the primary workspace for the scope for which you want to enforce policy.
b) Click **Manage Policies**.
c) Click **Enforcement**.
d) Click **Enforce Policies**.
e) Step through the wizard.

   For wizard details, see Policy Enforcement Wizard, on page 127.

**Step 3** Click **Accept and Enforce** on the final page of the wizard to push new firewall rules to the assets that are affected by policies in this workspace. A label flag is created at the time of enforcement:

***Figure 70: Policy Enforcement Page with Enforcement Enabled***



You may need to refresh the page to see the flag.

### What to do next

- If you enforced policy for a single workspace, consider whether workspaces for other scopes must also be enforced to achieve the expected enforcement outcomes.

  For example, workspaces for ancestor scopes or scopes that include workloads that are involved in cross-scope policies may also need to be enforced.

- Enforcement will not occur until enforcement is enabled for the agents, cloud connectors, and/or external orchestrators that enforce the policies:
  - For workloads with installed agents, enforce policy in the agent config for the relevant scopes and inventory filters. See Software Agent Config and subtopics.
  - For cloud-based workloads configured using cloud connectors, see:
    - Best Practices When Enforcing Segmentation Policy for AWS Inventory and linked topics.
    - Best Practices When Enforcing Segmentation Policy for Azure Inventory and linked topics.
  - For Kubernetes and OpenShift, see:
    - Enforcement on Containers, on page 129
    - Software Agent Config
  - For load balancers, see:
    - Policy enforcement for F5 BIG-IP

      Policy Enforcement for F5 Ingress Controller
    - Policy enforcement for Citrix Netscaler

- Check to be sure enforcement is working as expected. See Verify That Enforcement Is Working as Expected, on page 130.

- Configure alerts so you are notified of any issues, for example if flows are rejected after enforcement is enabled.

## Policy Enforcement Wizard

When you enforce policies for a single workspace from the Enforcement page of the workspace, the policy enforcement wizard lets you:

- Review policies before they are implemented on the workloads.

  This includes policies that are inherited from ancestor scopes.

- Download policy changes for review.

- Compare policy versions.

- Choose which analyzed version of the workspace to enforce.
- Roll back policies to a previous version.

Steps in the policy enforcement wizard:

1. Select Policy Updates

   You can select which version of policies to be enforced on the workloads.

   The difference between the currently enforced policies and policies in the selected version is displayed.

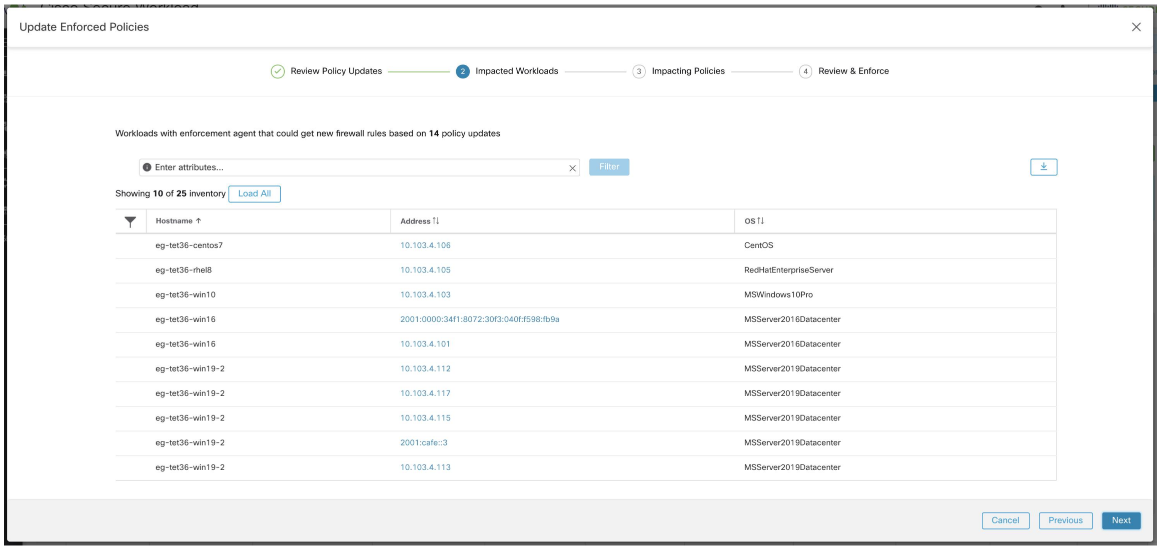
   Similarly to the Comparison of Policy Versions: Policy Diff, you can filter and review the policy changes and download them as CSV.

2. Impacted Workloads

   This step shows the workloads that will be affected by the new firewall rules generated from the selected policy changes. The result comes from searching all the workloads that have enforcement agents within the union of the consumers/providers of the selected policy changes.

   The number of potentially impacted workloads cannot exceed the total number of workloads in the scope. However, the actual impacted workloads might be smaller due to other factors such as agent config intents.

   ***Figure 71: List of Impacted Workloads***

   Workloads with enforcement agent that could get new firewall rules based on **14** policy updates

   Showing **10** of **25** inventory

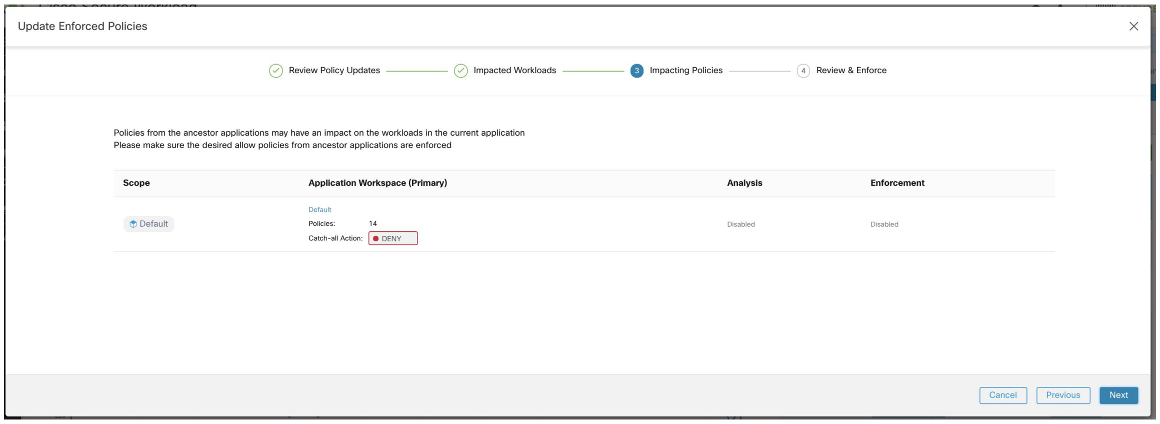
   | Hostname ↑ | Address ↑↓ | OS ↑↓ |
   |---|---|---|
   | eg-tet36-centos7 | 10.103.4.106 | CentOS |
   | eg-tet36-rhel8 | 10.103.4.105 | RedHatEnterpriseServer |
   | eg-tet36-win10 | 10.103.4.103 | MSWindows10Pro |
   | eg-tet36-win16 | 2001:0000:34f1:8072:30f3:040f:f598:fb9a | MSServer2016Datacenter |
   | eg-tet36-win16 | 10.103.4.101 | MSServer2016Datacenter |
   | eg-tet36-win19-2 | 10.103.4.112 | MSServer2019Datacenter |
   | eg-tet36-win19-2 | 10.103.4.117 | MSServer2019Datacenter |
   | eg-tet36-win19-2 | 10.103.4.115 | MSServer2019Datacenter |
   | eg-tet36-win19-2 | 2001:cafe::3 | MSServer2019Datacenter |
   | eg-tet36-win19-2 | 10.103.4.113 | MSServer2019Datacenter |

   For more details on viewing, filtering, and downloading inventory items, see Manage Inventory for Secure Workload.

3. Impacting Policies

   Policies from the ancestor workspaces may impact workloads in the current workspace. Therefore, you should make sure the desired allow policies from ancestor workspaces are enforced.

*Figure 72: List of ancestor workspaces and enforced versions*



Policies from the ancestor applications may have an impact on the workloads in the current application
Please make sure the desired allow policies from ancestor applications are enforced

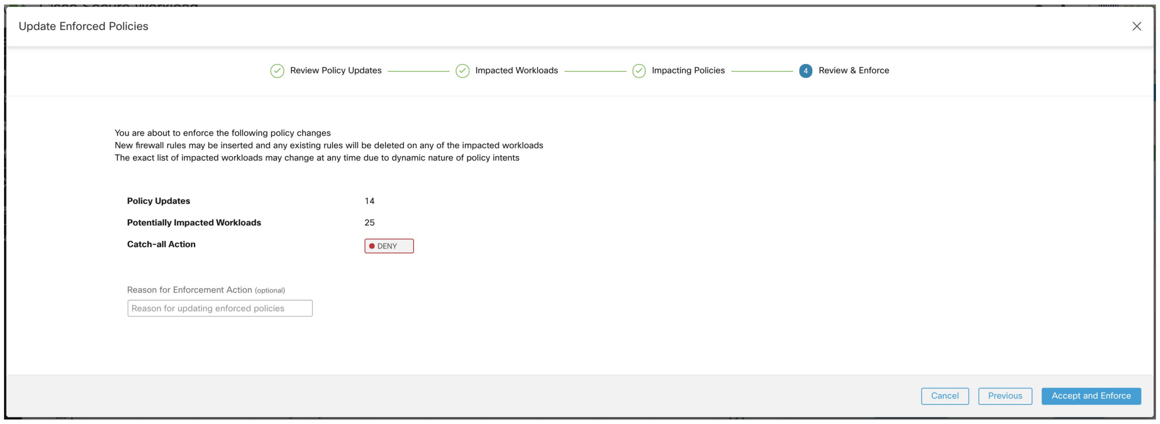
| Scope | Application Workspace (Primary) | Analysis | Enforcement |
|---|---|---|---|
| Default | Default<br>Policies: 14<br>Catch-all Action: DENY | Disabled | Disabled |

4. Review & Accept

   This final step summarizes the policy changes to be enforced, the number of potentially impacted workloads, and the catch-all action that will be enforced. When you click **Accept and Enforce**, the policies in the workspace will be used to calculate the new firewall rules that will be configured on the relevant workloads.

   You have the option to provide a name, description, and reason for action for the newly enforced policies for future reference. In case of rollback, you can provide only the reason, as name and description for a past version cannot be changed.

*Figure 73: Review the Summary and Enforce Policy Changes*



You are about to enforce the following policy changes
New firewall rules may be inserted and any existing rules will be deleted on any of the impacted workloads
The exact list of impacted workloads may change at any time due to dynamic nature of policy intents

| | |
|---|---|
| Policy Updates | 14 |
| Potentially Impacted Workloads | 25 |
| Catch-all Action | DENY |

Reason for Enforcement Action (optional)
Reason for updating enforced policies

## Enforcement on Containers

For an overview of the steps that are required to set up segmentation on container-based workloads that are managed by Kubernetes and OpenShift, see Set Up Microsegmentation for Kubernetes-Based Workloads.

**Attention** **Agents running on Kubernetes/OpenShift hosts must be configured to preserve existing rules.**

In order to prevent enforcement from interfering with iptables rules added by Kubernetes, the agent must be configured with a profile that has the **Preserve Rules** option enabled. See Creating an Agent Config Profile

When enforcing policies on containers, Secure Workload allows Kubernetes/OpenShift service abstractions to be used as providers. Internally, the policies for service abstractions are transformed into rules for the provider pods and the nodes they are running on. This transformation depends on the type of the Kubernetes/OpenShift service, and it is dynamically updated whenever changes are received from the API server.

The following example illustrates the flexibility made possible by this feature. Consider the following policy which allows traffic from all hosts and pods with the label *environment = production* to a Kubernetes service of type *NodePort* with the name *db* which exposes TCP port 27017 on a set of pods.

| Consumer | Provider | Protocol/Port | Action |
|---|---|---|---|
| environment = production<br>OR<br>orchestrator_environment = production | orchestrator_system/service_name = db | TCP 27017 | Allow |

This policy would result in the following firewall rules:

- On hosts and pods that are labeled with *environment = production*, allow outgoing connections to all Kubernetes nodes of the cluster to which the service belongs. This rule uses the node port that is assigned to this service by Kubernetes.
- On pods with the label *environment = production*, allow outgoing connections to the ClusterIP assigned to this service by Kubernetes. This rule uses the port that is exposed by the service (TCP 27017).
- On Kubernetes nodes of the cluster to which the service belongs, allow outgoing connections to the provider pods. This rule uses the target port that is exposed by the service (TCP 27017).
- On pods providing the service db, all incoming connections from all kubernetes nodes and consumer hosts and pods. This rule uses the target port that is exposed by the service (TCP 27017).

Changes to the type of the service, ports, and set of provider pods will immediately be picked up by Secure Workload rule generator and used to update the generated firewall rules.

**Caution** **Policies including Kubernetes/OpenShift inventory must be designed carefully to avoid conflicting with the internal operation of the kubernetes cluster.**

Kubernetes/OpenShift items imported by Secure Workload include the pods and services constituting the kubernetes cluster (for example, pods in the kube-system namespace). This allows precise policies to be defined to secure the kubernetes cluster itself, but it also means that badly designed policies can affect the operation of the cluster.

## Verify That Enforcement Is Working as Expected

### Check Agents

See Check Agent Health and Readiness to Enforce, on page 123.

### Check for Escaped and Rejected Flows

In the menu on the left side of the screen, click **Overview**.

On the **Security Dashboard** page, look at the **Segmentation Compliance Score**.

If this is less than 100, you may have escaped or rejected flows, either of which indicates a policy configuration issue.

For details, see Segmentation Compliance Score.

For more information about investigating these situations, see Policy Analysis Results: Understand the Basics, on page 113 and subtopics. (The information in these topics applies to enforced policies shown on the Enforcement tab and to analyzed policies shown on the Policy Analysis tab.)

Add any missing policies, or modify existing policies, for example, by adding additional protocols/ports, to allow required legitimate traffic.

Then reanalyze before reenforcing.

## View Enforced Policies for a Specific Workload (Concrete Policies)

Use this procedure to view all enforced policies for a specific workload (that is, the *concrete policies* for that workload). This view is useful because all policies in a workspace may not apply to every workload in the workspace, and because policies in multiple workspaces may apply to a particular workload (for example, inherited policies in parent or ancestor scopes).

Concrete policies are listed in priority order. For more information about the effects of priority, see the Policy Priorities section.

### Before you begin

**Note** Concrete policies include only policies in enforced workspaces. If a workspace is not enforced, any policies that would apply to the workload if the workspace were enforced do not appear in the list.

### Procedure

**Step 1** You can navigate to the Concrete Policies page for a workload from the Inventory page or from the workspace:

To navigate from the Scopes and Inventory page:

a) Choose **Organize** > **Scopes and Inventory**.
b) Search for the IP address of the workload of interest and click it.

   The Workload Profile opens in a separate tab.

   In general, except for cloud-based workloads that are managed without agents, Kubernetes, and OpenShift workloads, if the IP address appears in the **IP Addresses** tab and not in the **Workloads** tab, this means that an agent is not installed on the workload, so policies cannot be enforced, and there is no concrete policies list.

To navigate from the Segmentation page:

a) Choose **Defend** > **Segmentation**.

b) Click the scope.
c) Click the Primary workspace.
d) Click **Manage Policies**.
e) Click the **Matching Inventories** tab.
f) Search for the IP address of the workload of interest and click it.
g) In the panel that opens on the right, click **View Workload Profile**.

   The Workload Profile opens in a separate tab.

**Step 2** From the menu on the left side of the Workload Profile page, click **CONCRETE POLICIES**.

**Step 3** Click a row to view details.

For more information, see the Concrete Policies tab.

**Step 4** To see the amount of traffic that has hit each policy:

a) Click **Fetch All Stats**.
b) Click each policy of interest.

**Step 5** To view information about Kubernetes or OpenShift workloads, click **CONTAINER POLICIES**.

---

**What to do next**

Choose **Monitor** > **Enforcement Status** for status of concrete policies, for example to see if any policies have been skipped. For details, see the Enforcement Status section.

## Verify That Enforcement Is Enabled for Agents

**Procedure**

---

**Step 1** Click **Defend** > **Enforcement Status**.

**Step 2** To view only the enforcement status for a specific scope, toggle the **Filter by Scope** control and select a scope.

**Step 3** Look at the **Agent Enforcement Enabled** chart.

If the chart shows that any agents are **Not Enforced**, continue with this procedure.

Otherwise, skip the rest of this procedure, as all agents are enabled for enforcement.

**Step 4** Click the orange **Not Enforced** section of the chart to display the affected workloads in the table below the chart.

**Step 5** Enable enforcement on these workloads by modifying the Agent Config profile.

See Create an Agent Configuration Profile.

---

## Verify That Enforced Policies Are Being Pushed to Agents

For enforcement to occur, policies specific to each workload must be successfully pushed to the agent installed on that workload. Status is also shown for policy enforcement that is managed by cloud connectors even if agents are not installed.

### Before you begin

Enforce policies for at least one scope.

### Procedure

**Step 1** Click **Defend** > **Enforcement Status**.

**Step 2** To view only the enforcement status for a specific scope, toggle the **Filter by Scope** control and select a scope.

**Step 3** Look at the **Agent Concrete Policies** chart.

If the chart shows that any are **Skipped**, continue with this procedure.

Otherwise, skip the rest of this procedure.

**Step 4** To display the list of workloads affected by this issue, click the red **Skipped** slice of the chart.

The affected workloads are listed in the table below the charts.

**Step 5** To see the reasons for this issue:

For each workload in the search results, click the **(i)** button beside **Skipped** in the **Concrete Policies** column.

| Error Message | More Information |
|---|---|
| Agent doesn't have Windows OS | At least one policy that is applicable only to Windows workloads includes consumers and/or providers that are not running Windows OS.<br><br>Remove those workloads from those policies. |
| Maximum number of policies has been reached | See If There Are Too Many Policies for the Agent, on page 133. |

### What to do next

(Optional) Configure an alert so you are notified if this situation occurs in future. See Configure Alerts.

## If There Are Too Many Policies for the Agent

If the complete set of applicable concrete policies cannot be pushed to a particular agent, the latest version of the policies is not pushed.

Background: There is a limit to the number of policies supported on each agent. Limits also apply to policies enforced using cloud connectors. You may find the information in Configuration Limits in Secure Workload helpful.

### Before you begin

Use this procedure to resolve this problem if Verify That Enforced Policies Are Being Pushed to Agents, on page 132 indicates that the agent cannot accommodate the full set of enforced policies.

**Procedure**

**Step 1** Navigate to the primary workspace for an affected scope.

**Step 2** Modify the policies in the primary workspace:

Try to reduce the number of policies and reduce any long lists of IP addresses in consumer or provider.

For example, consolidate existing policies, and/or base policies on subnets rather than on huge lists of IP addresses.

For policies enforced using a cloud connector, you may also be able to increase any limits that are imposed by the platform. See the documentation for your cloud platform.

**Step 3** After you have made changes, enforce the latest version of the workspace and check again for skipped policies.

**Step 4** Repeat this procedure for any other scopes experiencing this issue.

# Modify Enforced Policies

## Enforce New and Revised Policies

If you must revise policies after enforcement, generally you will make the changes in the same primary workspace. Then, review your changes carefully, and analyze the workspace again to be sure it has the effect you expect. When you are confident that the changes will have the desired effect, click the **Enforce Latest Policies** button in the upper right of the page.

## View, Compare, and Manage Enforced Policy Versions

Each time that you enforce or reenforce policies in a workspace after making changes, a new version (p*) is created.

For detailed information about versioning, see About Policy Versions (v* and p*), on page 136.

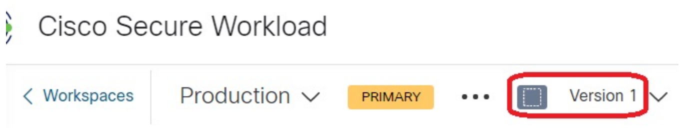
**Procedure**

**Step 1** Click **Defend** > **Segmentation**.

**Step 2** Navigate to the relevant scope and primary workspace.

**Step 3** Click **Manage Policies**.

**Step 4** The currently displayed version of the policies is shown at the top of the page:



The displayed version may be a policy discovery version, an analyzed policy version, or an enforced version.

**Step 5** Do one of the following:

| | |
|---|---|
| To display a different version of the policies: | Click the current version and choose a different version.<br><br>For descriptions of the versions, see About Policy Versions (v* and p*), on page 136.<br><br>**Important!** If you choose a v* version, see View, Compare, and Manage Discovered Policy Versions, on page 50 instead of this topic, including the important caveat at the end of the topic. |
| To view details about the analyzed versions: | **a.** Click **View Version History** at the top of the page beside the current version.<br><br>**b.** Click the **Published Versions** tab to see the versions of analyzed and enforced policies.<br><br>**c.** To view log entries for a version, click the link in the version.<br><br>Pale green rows represent analysis activity.<br><br>Bright green rows represent enforcement activity. |
| To compare two versions to see what has changed: | **a.** Click **Compare Revisions**.<br><br>**b.** Choose the versions to compare.<br><br>You can compare the latest draft version, analyzed and enforced versions.<br><br>**c.** For result details, see Comparison of Policy Versions: Policy Diff, on page 139. |
| To delete an unwanted version: | Click ⋮ for the version and choose **Delete**.<br><br>Published policy versions (p* versions) can be deleted as long as the version is not being actively analyzed or enforced. |
| To export a version: | Click ⋮ for the version and choose **Export...**.<br><br>See also Export a Workspace, on page 55. |

### What to do next

When you are done working with versions, change the version at the top of the workspace page to the latest discovered policy version (v*).

This avoids unintentional deletion of discovered policy versions and allows you to manually create policies in the workspace.

## Revert Enforced Policies to an Earlier Version

To roll back enforced policies to a previous version, follow one of the processes that are described in Enable Policy Enforcement, on page 124 and choose an earlier version to enforce.

## Disable Policy Enforcement

- **To disable policy enforcement for multiple scopes simultaneously**:

  Follow the procedure for enforcing policy in multiple scopes simultaneously, as described in Enable Policy Enforcement, on page 124. On the Select Version page of the wizard, click **Select a version** and choose **Disable enforcement**.

- **To disable policy enforcement for a single scope**:

  Navigate to the Policy Enforcement page for the scope's primary workspace and click the red **Stop Policy Enforcement** button. This writes new firewall rules to assets in the scope based on enforced policies in ancestor workspaces. A Label Flag with an ‘x’ will be created on the time series chart.

## Enforcement History

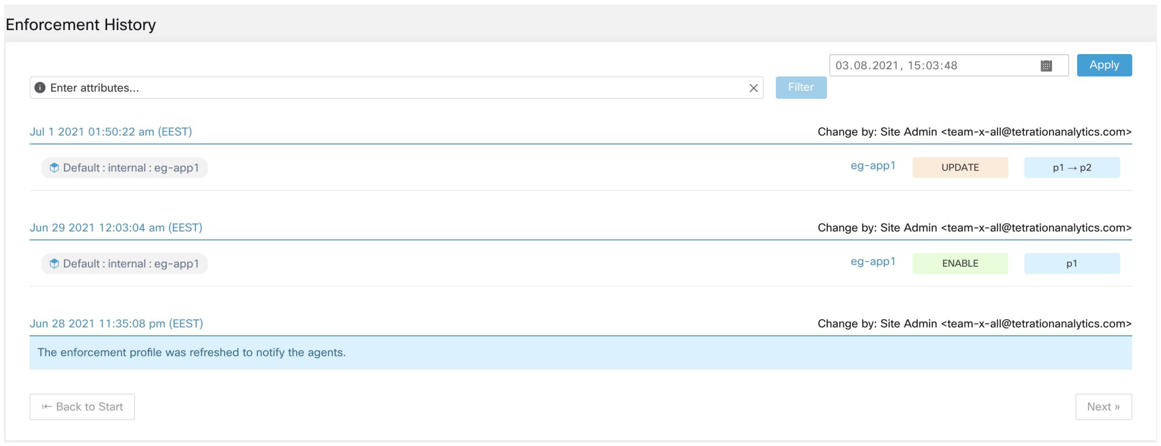
Enforcement History provides a list of changes to the list of enforced workspaces and their version.

To view the enforcement history:

1. Click the caret at the right side of the Segmentation page to expand the Tools menu.
2. Click **Enforcement History**.

   Each section describes an event and displays a summary of what has changed.
3. Click an event for details about all the policies that were enforced at that time.

***Figure 74: Enforcement history view***


Enforcement History

03.08.2021, 15:03:48

Enter attributes...

Jul 1 2021 01:50:22 am (EEST) Change by: Site Admin <team-x-all@tetrationanalytics.com>

Default : internal : eg-app1 eg-app1 UPDATE p1 → p2

Jun 29 2021 12:03:04 am (EEST) Change by: Site Admin <team-x-all@tetrationanalytics.com>

Default : internal : eg-app1 eg-app1 ENABLE p1

Jun 28 2021 11:35:08 pm (EEST) Change by: Site Admin <team-x-all@tetrationanalytics.com>

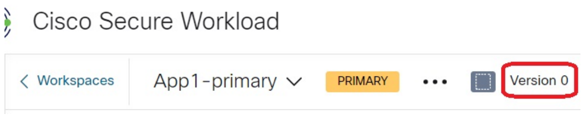
The enforcement profile was refreshed to notify the agents.

# About Policy Versions (v* and p*)

Policy versions are sometimes called workspace versions.

### Displayed Version

The version of the policies (and clusters) that you are currently working with is shown at the top of the workspace page:



- V* versions are generated by automatic policy discovery

  For details, see below

- P* versions are analyzed and/or enforced versions

  For details, see below

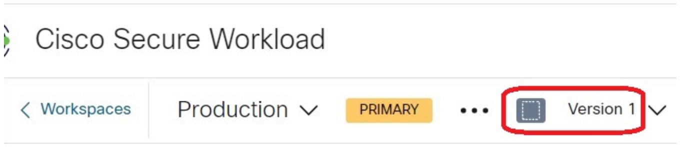
The following icons may appear beside the version number:

***Table 6: Version Icons***

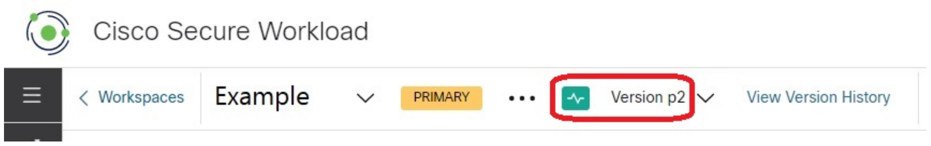
| | |
|---|---|
| 🟩 | Indicates the version of the policies that is currently being analyzed |
| 🟪 | Indicates the version of the policies that is currently being enforced |
| ▨ | Indicates the latest version of automatically discovered policies |
| (no icon) | Indicates that the version is not the latest version of its type |

Examples:

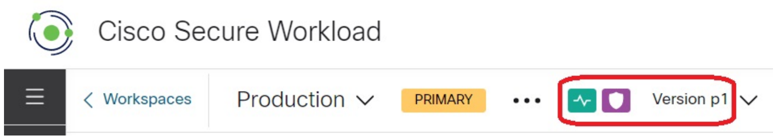
- Displayed version is the latest discovered version of the policies:



- Displayed version is the version of the policies that is currently being analyzed:



- Displayed version is the version of the policies currently being analyzed and enforced:



## Policy Discovery Version (v*)

Each time you automatically discover policies for a workspace, the version (v*) increments.

The first time you automatically discover policies, version 1 is generated, and all modifications after that run, such as editing or approving clusters (but not a rerun), are also grouped under version 1. When you subsequently automatically discover policies, a new version is generated (unless discovery failed).

The v* version also increments if you import policies.

To work with v* versions, see View, Compare, and Manage Discovered Policy Versions, on page 50.

### Published Policy Version (p*)

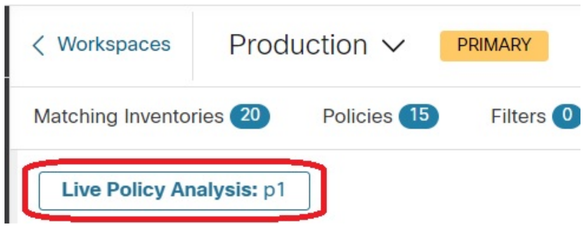
The term "published" policy version (p*) for a workspace can refer to either:

- The version of the policies that was analyzed, or
- The version of the policies that was enforced

These are two separate but parallel versions that depend on the context:

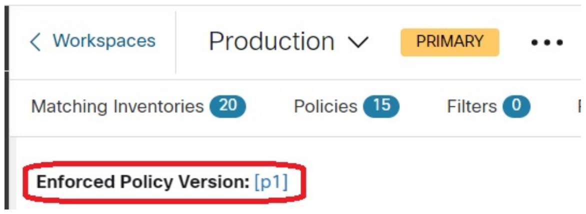
- Policy version for analysis:

  Each time you analyze policies in a workspace, or click **Analyze Latest Policies** after making a change, the system takes a snapshot of all the clusters and policies that are defined in that workspace, and the "published" policy version (p*) number for analysis increments. The latest **Live Policy Analysis** version is shown at the top-left side of the page on the Policy Analysis tab of the primary workspace.

- Policy version for enforcement:

  Each time you enable enforcement of the policies in a workspace, or enable enforcement again after making changes, the "published" policy version (p*) for enforcement becomes the number of the analyzed version that you choose in the enforcement wizard. So, if you enforce analyzed version 5, the enforced version is also version 5, even if it is, for example, the first time policy has been enforced for the workspace. The current **Enforced Policy Version** is shown at the top-left side of the page on the Enforcement tab of the primary workspace.

### Managing Published (p*) Versions

Published policy versions cannot be edited, only deleted entirely.

**Note** Published policy versions (p*) are limited to 100 total. Once this limit is reached, you must delete old versions.

To manage and delete p* versions, see View, Compare, and Manage Analyzed Policy Versions, on page 120, or View, Compare, and Manage Enforced Policy Versions, on page 134.

You can also use the API to delete published versions.

## Comparison of Policy Versions: Policy Diff

To compare policies, see any of the following topics: View, Compare, and Manage Discovered Policy Versions, on page 50, View, Compare, and Manage Analyzed Policy Versions, on page 120, or View, Compare, and Manage Enforced Policy Versions, on page 134

Policy changes will be displayed in three categories: Absolute, Default and Catch All. In the comparison table:

- Different services that belongs to the same policy are grouped together
- Filter policy changes by facet or by diff type
- Policy changes and services are paginated
- Download filtered policy changes as CSV

*Table 7: Facet filter properties*

| Property | Description |
|---|---|
| **Priority** | e.g. 100 |
| **Action** | e.g. ALLOW, DENY |
| **Consumer** | e.g. Consumer Cluster |
| **Provider** | e.g. Provider Cluster |
| **Port** | e.g. 80 |
| **Protocol** | e.g. TCP |

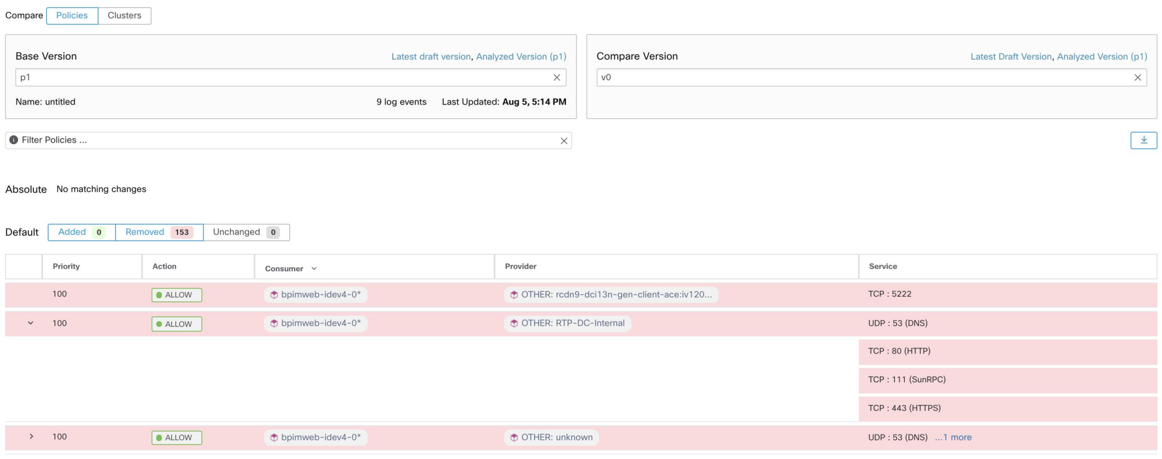
*Table 8: CSV output columns*

| Column | Description |
|---|---|
| **Rank** | The category of the policy. e.g. ABSOLUTE, DEFAULT, CATCH_ALL |
| **Diff** | The diff type of the change. e.g. ADDED, REMOVED, UNCHANGED |
| **Priority** | e.g. 100 |
| **Action** | e.g. ALLOW, DENY |

| Column | Description |
|---|---|
| **Consumer Name** | The name of the consumer cluster. |
| **Consumer ID** | The ID of the consumer cluster. |
| **Provider Name** | The name of the provider cluster. |
| **Provider ID** | The ID of the provider cluster. |
| **Protocol** | e.g. TCP |
| **Port** | e.g. 80 |

In the figure below, policy versions p1 and v1 are compared.

***Figure 75: Policy Diff View***



Compare Policies Clusters

Base Version — Latest draft version, Analyzed Version (p1)
p1
Name: untitled — 9 log events — Last Updated: **Aug 5, 5:14 PM**

Compare Version — Latest Draft Version, Analyzed Version (p1)
v0

Filter Policies ...

Absolute No matching changes

Default Added 0 Removed 153 Unchanged 0

| | Priority | Action | Consumer | Provider | Service |
|---|---|---|---|---|---|
| | 100 | ALLOW | bpimweb-idev4-0* | OTHER: rcdn9-dci13n-gen-client-ace:iv120... | TCP : 5222 |
| ⌄ | 100 | ALLOW | bpimweb-idev4-0* | OTHER: RTP-DC-Internal | UDP : 53 (DNS) |
| | | | | | TCP : 80 (HTTP) |
| | | | | | TCP : 111 (SunRPC) |
| | | | | | TCP : 443 (HTTPS) |
| › | 100 | ALLOW | bpimweb-idev4-0* | OTHER: unknown | UDP : 53 (DNS) ...1 more |

***Figure 76: Policy Diff View Download Button***

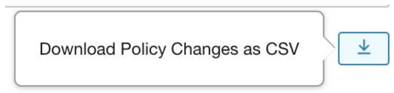
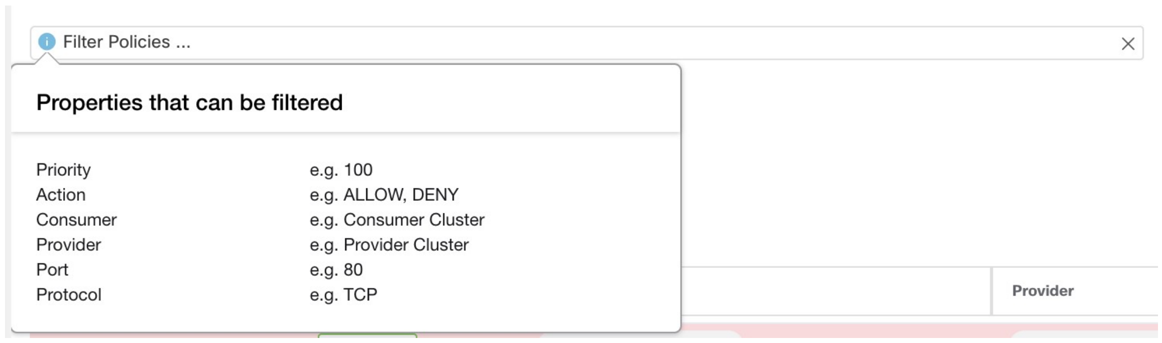


***Figure 77: Filtering Policy Diff View***



Filter Policies ...

Properties that can be filtered

| | |
|---|---|
| Priority | e.g. 100 |
| Action | e.g. ALLOW, DENY |
| Consumer | e.g. Consumer Cluster |
| Provider | e.g. Provider Cluster |
| Port | e.g. 80 |
| Protocol | e.g. TCP |

*Figure 78: Policy Diff View Diff Type Filter*

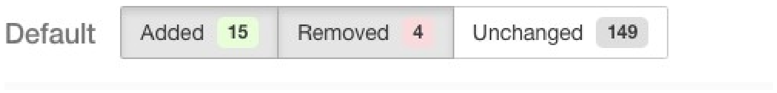


*Figure 79: Policy Diff View Grouping*

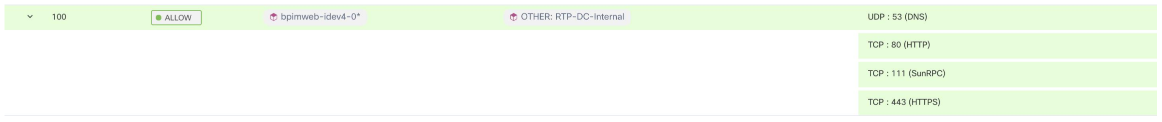
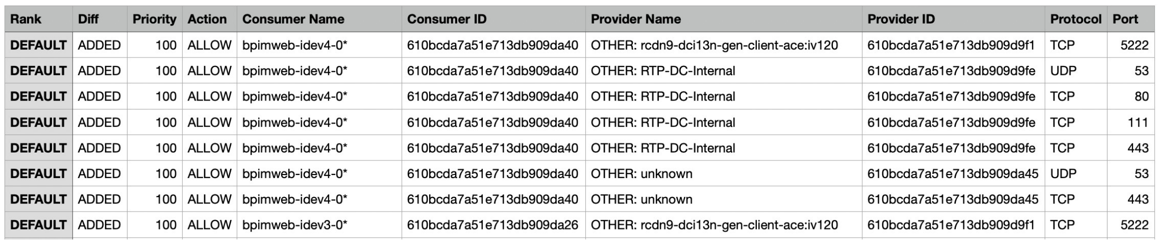


*Figure 80: Policy Diff View CSV Output*



| Rank | Diff | Priority | Action | Consumer Name | Consumer ID | Provider Name | Provider ID | Protocol | Port |
|---|---|---|---|---|---|---|---|---|---|
| **DEFAULT** | ADDED | 100 | ALLOW | bpimweb-idev4-0* | 610bcda7a51e713db909da40 | OTHER: rcdn9-dci13n-gen-client-ace:iv120 | 610bcda7a51e713db909d9f1 | TCP | 5222 |
| **DEFAULT** | ADDED | 100 | ALLOW | bpimweb-idev4-0* | 610bcda7a51e713db909da40 | OTHER: RTP-DC-Internal | 610bcda7a51e713db909d9fe | UDP | 53 |
| **DEFAULT** | ADDED | 100 | ALLOW | bpimweb-idev4-0* | 610bcda7a51e713db909da40 | OTHER: RTP-DC-Internal | 610bcda7a51e713db909d9fe | TCP | 80 |
| **DEFAULT** | ADDED | 100 | ALLOW | bpimweb-idev4-0* | 610bcda7a51e713db909da40 | OTHER: RTP-DC-Internal | 610bcda7a51e713db909d9fe | TCP | 111 |
| **DEFAULT** | ADDED | 100 | ALLOW | bpimweb-idev4-0* | 610bcda7a51e713db909da40 | OTHER: RTP-DC-Internal | 610bcda7a51e713db909d9fe | TCP | 443 |
| **DEFAULT** | ADDED | 100 | ALLOW | bpimweb-idev4-0* | 610bcda7a51e713db909da40 | OTHER: unknown | 610bcda7a51e713db909da45 | UDP | 53 |
| **DEFAULT** | ADDED | 100 | ALLOW | bpimweb-idev4-0* | 610bcda7a51e713db909da40 | OTHER: unknown | 610bcda7a51e713db909da45 | TCP | 443 |
| **DEFAULT** | ADDED | 100 | ALLOW | bpimweb-idev3-0* | 610bcda7a51e713db909da26 | OTHER: rcdn9-dci13n-gen-client-ace:iv120 | 610bcda7a51e713db909d9f1 | TCP | 5222 |

**Tip** See also Comparing Versions of Generated Clusters: Diff Views, on page 78.

## Activity Logs and Version History

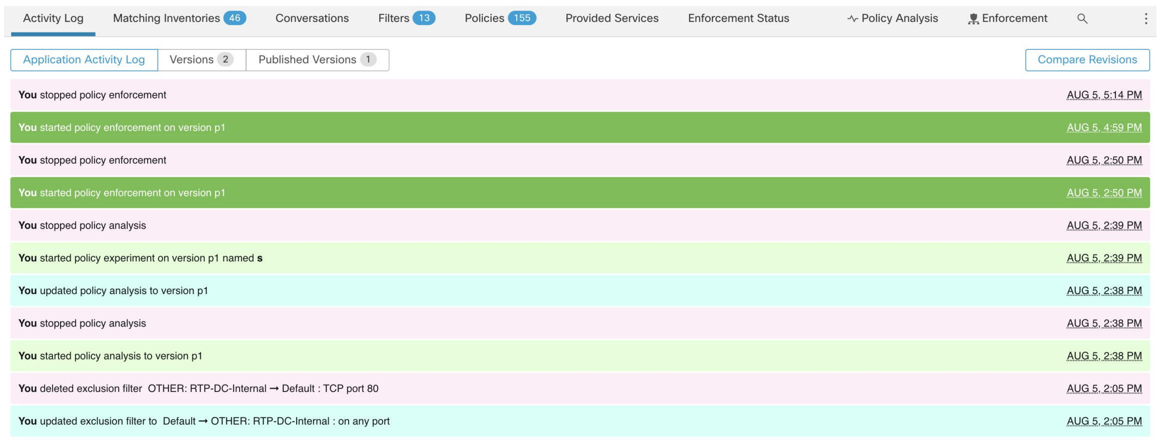
Activity logs record the history of modifications that are applied to a workspace by you. The events that are shown include adding, removing, and renaming workloads and clusters, moving workloads between clusters, uploading side information, submitting and aborting automatic policy discovery, and so on. The view shows which user has made each modification.

To view the modification history for a workspace, click any **Activity Log** link in the workspace.

For example:

1. Click **Defend** > **Segmentation**
2. Click the relevant scope and workspace.
3. Click the **View Activity Logs** link.
4. Click the **Workspace Activity Log** tab.

*Figure 81: Log of Events Applicable to Version v1 of this Workspace*



**You** stopped policy enforcement AUG 5, 5:14 PM
**You** started policy enforcement on version p1 AUG 5, 4:59 PM
**You** stopped policy enforcement AUG 5, 2:50 PM
**You** started policy enforcement on version p1 AUG 5, 2:50 PM
**You** stopped policy analysis AUG 5, 2:39 PM
**You** started policy experiment on version p1 named **s** AUG 5, 2:39 PM
**You** updated policy analysis to version p1 AUG 5, 2:38 PM
**You** stopped policy analysis AUG 5, 2:38 PM
**You** started policy analysis to version p1 AUG 5, 2:38 PM
**You** deleted exclusion filter OTHER: RTP-DC-Internal → Default : TCP port 80 AUG 5, 2:05 PM
**You** updated exclusion filter to Default → OTHER: RTP-DC-Internal : on any port AUG 5, 2:05 PM

For information about the version-related tabs and options on the page, see:

- About Policy Versions (v* and p*), on page 136
- View, Compare, and Manage Discovered Policy Versions, on page 50
- View, Compare, and Manage Analyzed Policy Versions, on page 120
- View, Compare, and Manage Enforced Policy Versions, on page 134

## Automatic Deletion of Old Policy Versions

Every week, the following are automatically deleted: Workspace versions that have not been accessed for six months and policy experiments that have not been accessed in the last 30 days.

# Conversations

A conversation is defined as a service provided by one host on a particular port and consumed by another host. Such a conversation is materialized from many flows over different times. Automatic policy discovery takes all such flows, ignores the ephemeral/client ports, and deduplicates them to generate the conversation graph. For any given conversation between host A and host B on server (provider) port N, there has been at least one flow observation from A to B on port N in the timeframe for which automatic policy discovery has been performed.

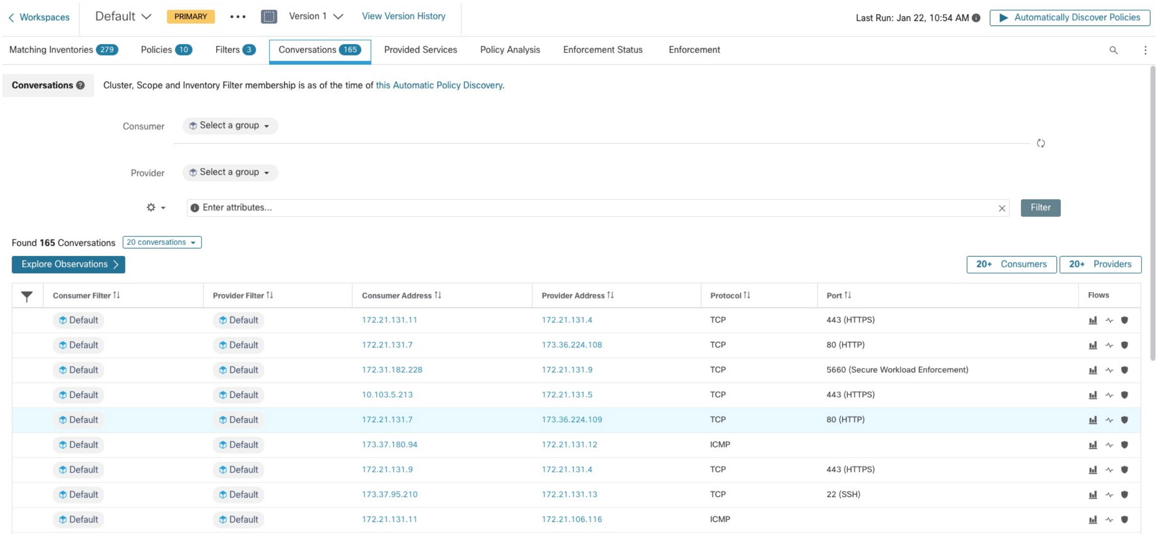
Use flow data to better understand what flows are associated with what process while evaluating clusters generated during automatic policy discovery.

In addition, information that is collected by agents provides visibility of unused L4 ports. Unused ports are the ones for which no communication was seen for the interval selected for automatic policy discovery. This information can be used to open up policies for communication on those ports OR to close those applications binding to the unused ports, thereby reducing the attack surface of the workload.

Note that client-server classification affects the automatic policy discovery conversation view – it dictates which port is dropped (is deemed ephemeral) in the aggregation: See Client Server Classification.

## Conversations Table View

The Conversations Table view provides a simple way to view aggregated flows from the duration of automatic policy discovery where the consumer port is removed and there is only one record for all time. While policies go from filter to filter, conversations are from IP address to IP address.



Cluster, Scope and Inventory Filter membership is as of the time of this Automatic Policy Discovery.

Found 165 Conversations

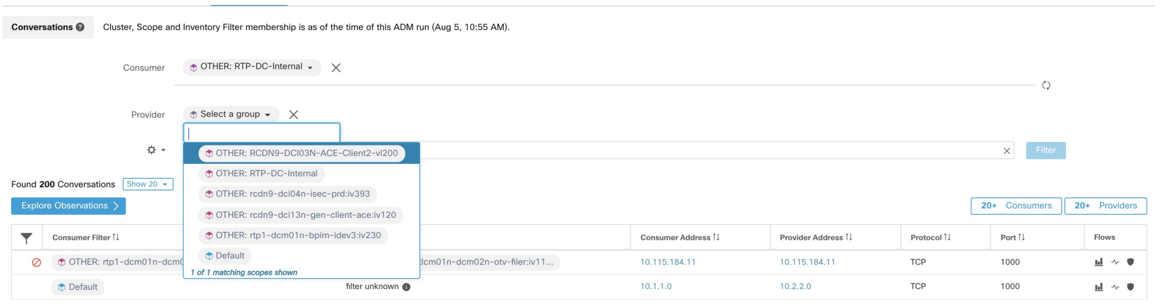
| Consumer Filter | Provider Filter | Consumer Address | Provider Address | Protocol | Port | Flows |
|---|---|---|---|---|---|---|
| Default | Default | 172.21.131.11 | 172.21.131.4 | TCP | 443 (HTTPS) | |
| Default | Default | 172.21.131.7 | 173.36.224.108 | TCP | 80 (HTTP) | |
| Default | Default | 172.31.182.228 | 172.21.131.9 | TCP | 5660 (Secure Workload Enforcement) | |
| Default | Default | 10.103.5.213 | 172.21.131.5 | TCP | 443 (HTTPS) | |
| Default | Default | 172.21.131.7 | 173.36.224.109 | TCP | 80 (HTTP) | |
| Default | Default | 173.37.180.94 | 172.21.131.12 | ICMP | | |
| Default | Default | 172.21.131.9 | 172.21.131.4 | TCP | 443 (HTTPS) | |
| Default | Default | 173.37.95.210 | 172.21.131.13 | TCP | 22 (SSH) | |
| Default | Default | 172.21.131.11 | 172.21.106.116 | ICMP | | |

*Figure 82: Conversations Table View*

## Choosing Consumer or Provider

Consumers and Providers can be selected by a typeahead dropdown selector which allows one to choose Inventory Filters, Scopes, and Clusters as shown in the example below. All conversations between the chosen Consumer and Provider are displayed. Note: to delete an existing filter, click on the 'x' icon (erasing the filter may not work).

By default, the Consumer and Provider match against all of the inventory filters an IP address is a member of during automatic policy discovery. For example, searching for the "root scope" will match all the conversations even though some IPs may be better matched by more specific scopes. To perform a more specific match, select "Restrict scope filtering to an IP's best match" from the settings dropdown to the left of the faceted filter input.


Conversations
Consumer
OTHER: RTP-DC-Internal
Provider
Select a group
OTHER: RCDN9-DCI03N-ACE-Client2-vl200
OTHER: RTP-DC-Internal
OTHER: rcdn9-dci04n-isec-prd:iv393
OTHER: rcdn9-dci13n-gen-client-ace:iv120
OTHER: rtp1-dcm01n-bpim-idev3:iv230
Default
1 of 1 matching scopes shown
Filter
Show 20
Explore Observations
20+ Consumers
20+ Providers

Cluster, Scope and Inventory Filter membership is as of the time of this ADM run (Aug 5, 10:55 AM).

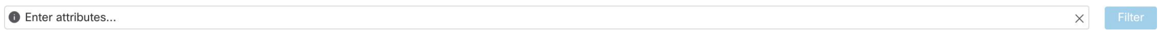
Found 200 Conversations

| Consumer Filter | | Consumer Address | Provider Address | Protocol | Port | Flows |
|---|---|---|---|---|---|---|
| OTHER: rtp1-dcm01n-dcm | dcm01n-dcm02n-otv-filer:iv11... | 10.115.184.11 | 10.115.184.11 | TCP | 1000 | |
| Default | filter unknown | 10.1.1.0 | 10.2.2.0 | TCP | 1000 | |

*Figure 83: Choosing Consumer or Provider*

## Conversation Filters

*Figure 84: Conversation Filters*



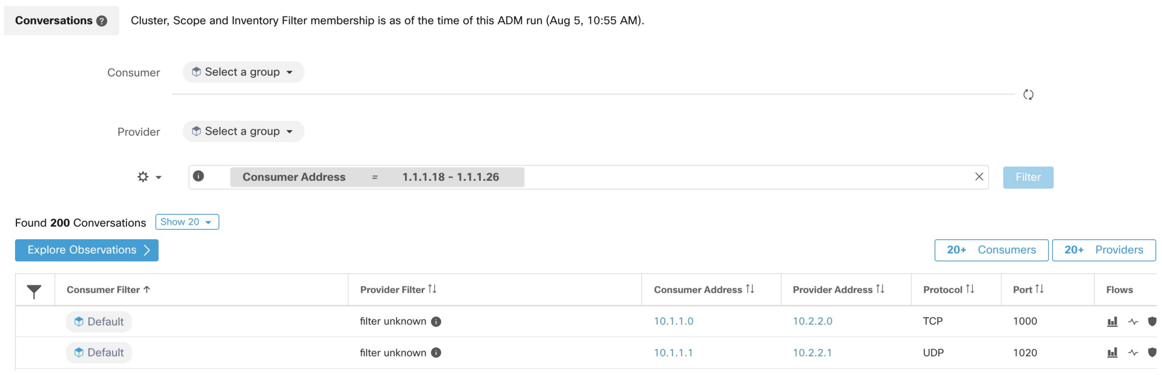
This is where you define filters to narrow-down the search results. All the possible dimensions can be found by clicking on the (?) icon next to the word Filters. For any User Labels data, those columns will also be available for the appropriate intervals. This input also supports and, or, not, and parenthesis keywords, use these to express more complex filters. For example, a direction-agnostic filter between IP 1.1.1.1 and 2.2.2.2 can be written:

Consumer Address = 1.1.1.1 and Provider Address = 2.2.2.2 or Consumer Address = 2.2.2.2 and Provider Address = 1.1.1.1 And to additionally filter on Protocol = TCP:

(Consumer Address = 1.1.1.1 and Provider Address = 2.2.2.2 or Consumer Address = 2.2.2.2 and Provider Address = 1.1.1.1) and Protocol = TCP

The filter input also supports "," and "-" for Port, Consumer Address and Provider Address, by translating "-" into range queries. The following are examples of a valid filter:

*Figure 85: Filter input Supports Range Query for Consumer Address*



Cluster, Scope and Inventory Filter membership is as of the time of this ADM run (Aug 5, 10:55 AM).

Consumer Address = 1.1.1.18 - 1.1.1.26

Found 200 Conversations

| Consumer Filter | Provider Filter | Consumer Address | Provider Address | Protocol | Port | Flows |
|---|---|---|---|---|---|---|
| Default | filter unknown | 10.1.1.0 | 10.2.2.0 | TCP | 1000 | |
| Default | filter unknown | 10.1.1.1 | 10.2.2.1 | UDP | 1020 | |

Available filters:

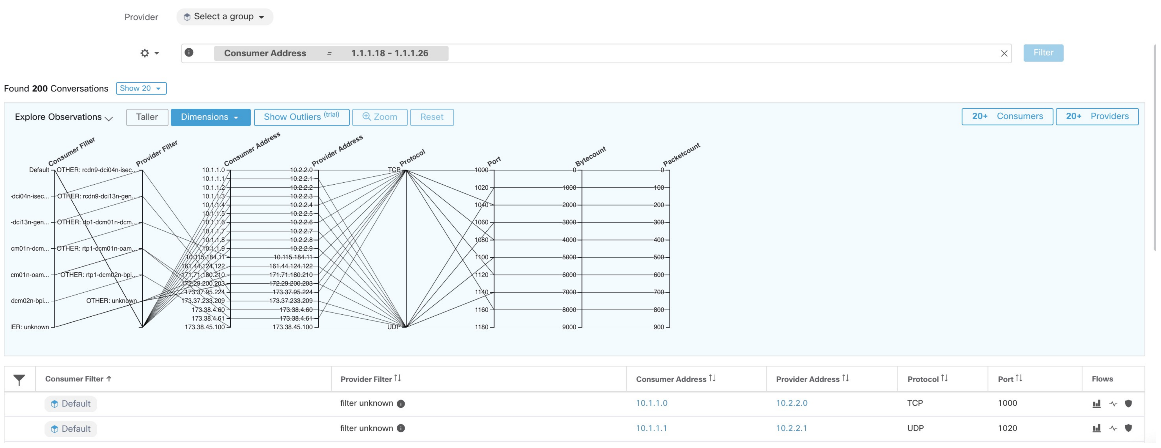
| Filters | Description |
|---|---|
| **Consumer Address** | Enter a subnet or IP Address using CIDR notation (for example, 10.11.12.0/24). Matches conversation flow observations whose consumer address overlaps with the provided IP Address or subnet. |
| **Provider Address** | Enter a subnet or IP Address using CIDR notation (for example, 10.11.12.0/24) Matches conversation flow observations whose provider address overlaps with provided IP address or subnet. |
| **Port** | Matches conversation flow observations whose port overlaps with provided port. |
| **Protocol** | Filter conversation flow observations by Protocol type (TCP, UDP, ICMP). |

| Filters | Description |
|---|---|
| **Address Type** | Filter conversation flow observations by Address type (IPv4, IPv6, DHCPv4). |
| **Confidence** | Indicated the confidence in the direction of flow. Possible values: High, Very High, Moderate. |
| **Excluded?** | Match conversations that are excluded by an exclusion filter or approved policy. |
| **Excluded By** | Match conversations excluded by a specific filter. Possible values: Exclusion Filter, Policy. |

## Explore Observations

Clicking on the Explore Observations button enables a chart view that allows quick exploration of the high-dimensional data via a “Parallel Coordinates” chart. A bit overwhelming at first, this chart can be useful when enabling only the dimensions you’re interested in (by unchecking items in the Dimensions dropdown), and when rearranging the order of the dimensions. A single line in this chart represents a single observation, and where that line intersects with the various axes indicates the value of that observation for that dimension. This can become clearer when hovering over the list of observations below the chart to see the highlighted line representing that observation in the chart:

***Figure 86: Explore Observations***



Consumer Address = 1.1.1.18 - 1.1.1.26

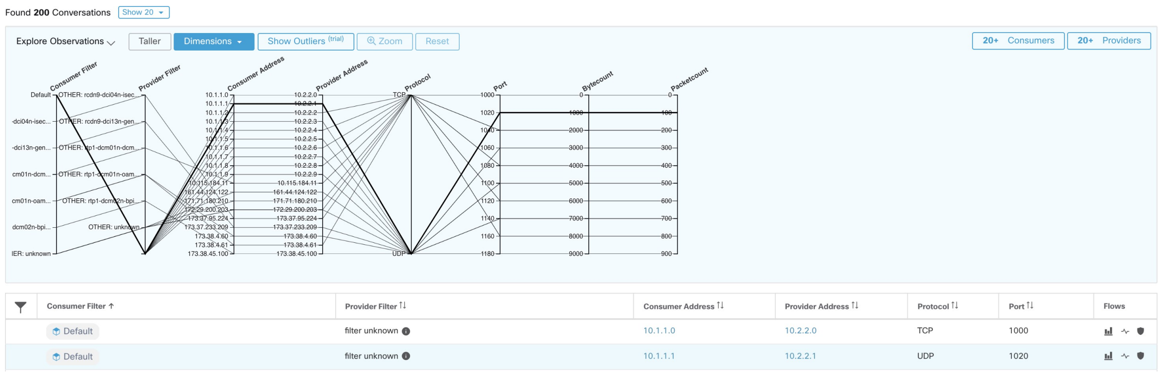
Found **200** Conversations

| Consumer Filter ↑ | Provider Filter ↑↓ | Consumer Address ↑↓ | Provider Address ↑↓ | Protocol ↑↓ | Port ↑↓ | Flows |
|---|---|---|---|---|---|---|
| Default | filter unknown | 10.1.1.0 | 10.2.2.0 | TCP | 1000 | |
| Default | filter unknown | 10.1.1.1 | 10.2.2.1 | UDP | 1020 | |

## Conversation Observation Hovered

Due to the high-dimensional nature of the conversations data, this chart is wide by default, and requires scrolling right to see the entire chart. For this reason, it’s useful to disable all but the dimensions you are interested in. Hover state in Explore Conversations is provided to map (hover) each conversation with the table list view.

*Figure 87: Conversation Observation Hovered*

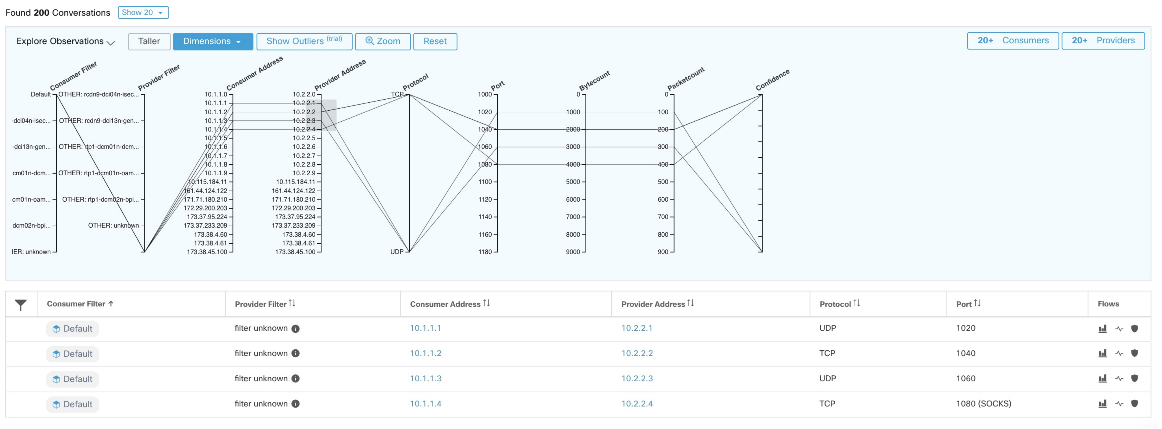


| Consumer Filter ↑ | Provider Filter ↑↓ | Consumer Address ↑↓ | Provider Address ↑↓ | Protocol ↑↓ | Port ↑↓ | Flows |
|---|---|---|---|---|---|---|
| Default | filter unknown | 10.1.1.0 | 10.2.2.0 | TCP | 1000 | |
| Default | filter unknown | 10.1.1.1 | 10.2.2.1 | UDP | 1020 | |

## Filtering

Dragging the cursor along any of the axes creates a selection that will show only observations that match that selection. Click again on the axis to remove the selection at any time. Selections can be made on any number of axes at a time. The list of observations will update to show only the selected conversations.

*Figure 88: Filtering*



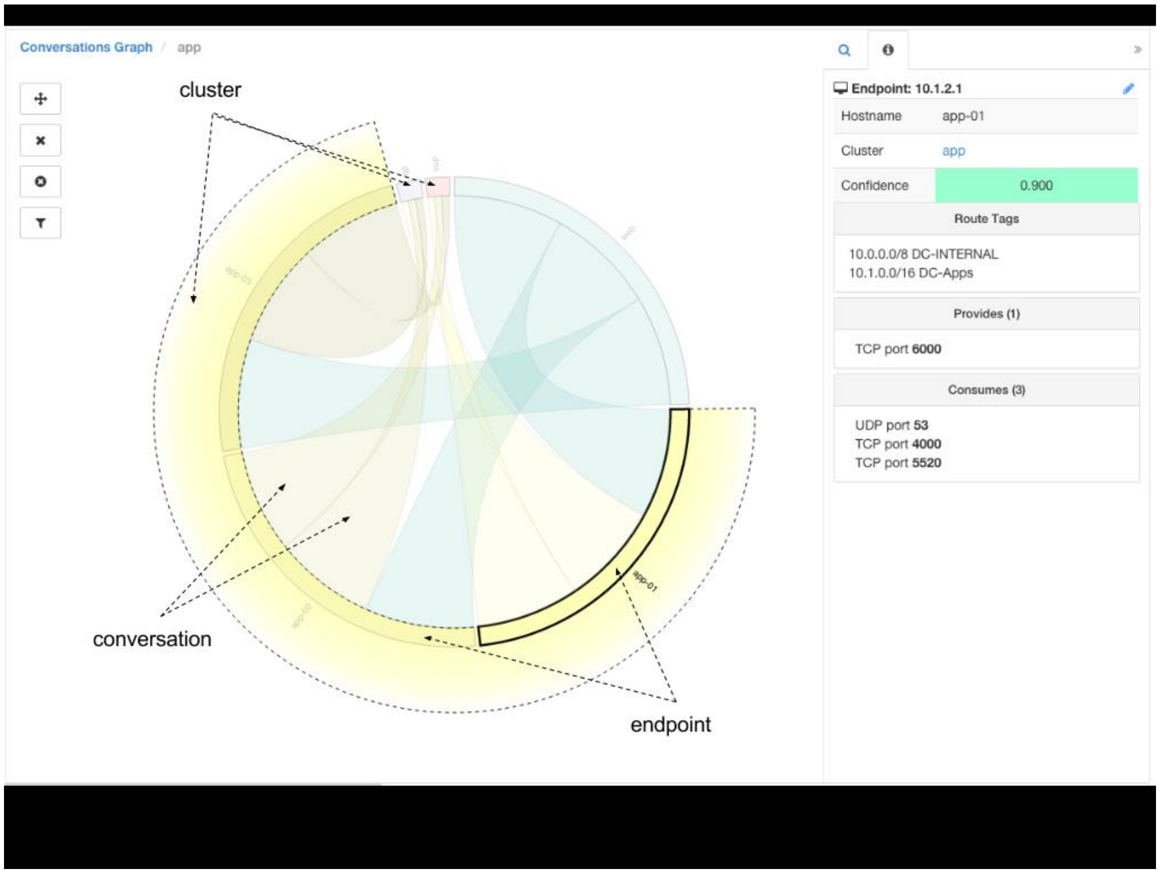
| Consumer Filter ↑ | Provider Filter ↑↓ | Consumer Address ↑↓ | Provider Address ↑↓ | Protocol ↑↓ | Port ↑↓ | Flows |
|---|---|---|---|---|---|---|
| Default | filter unknown | 10.1.1.1 | 10.2.2.1 | UDP | 1020 | |
| Default | filter unknown | 10.1.1.2 | 10.2.2.2 | TCP | 1040 | |
| Default | filter unknown | 10.1.1.3 | 10.2.2.3 | UDP | 1060 | |
| Default | filter unknown | 10.1.1.4 | 10.2.2.4 | TCP | 1080 (SOCKS) | |

# Conversations Chart View

Conversation chart view has a similar look and feel to the policy view page, except that instead of partitions/clusters/policies, it focuses on clusters/workloads/conversations. As illustrated in the figure below, the outer arcs at the high level represent clusters and can be expanded to show the member hosts/workloads as inner arcs. The chords represent the conversations or connections.

The controls and side panel on conversation view behave similarly to the policy view, except for the fact that the side panel information also shows detailed information about selected workloads such as consumed/provided services as well as link to parent cluster and process information, if available.

*Figure 89: Conversations Chart View*



Endpoint: 10.1.2.1

| | |
|---|---|
| Hostname | app-01 |
| Cluster | app |
| Confidence | 0.900 |

Route Tags

10.0.0.0/8 DC-INTERNAL
10.1.0.0/16 DC-Apps

Provides (1)

TCP port **6000**

Consumes (3)

UDP port **53**
TCP port **4000**
TCP port **5520**

## Top Consumers/Providers of Conversations

The number of top Consumers or Providers based on total conversations reflecting chosen filters can be seen from two buttons on top of the Conversations table. Click on each one to see a dialog containing a table with the Conversation Count column along with each Consumer/Provider’s Address, Hostname, and other User Annotated columns.

*Figure 90: Above the Conversations Table*

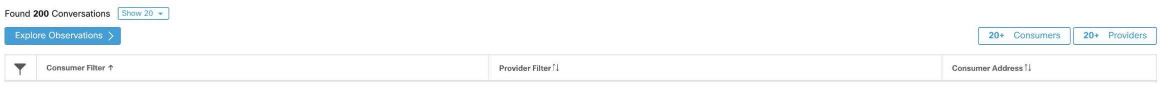
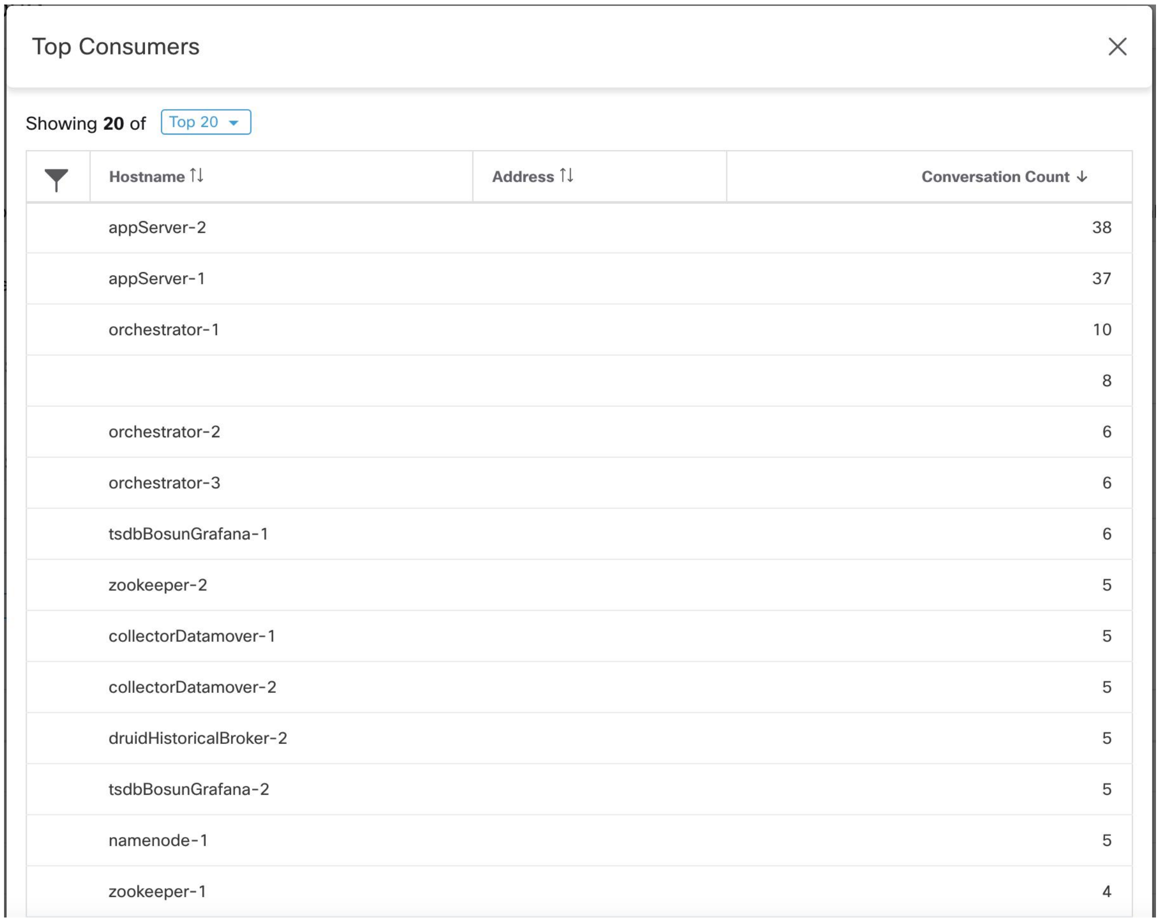


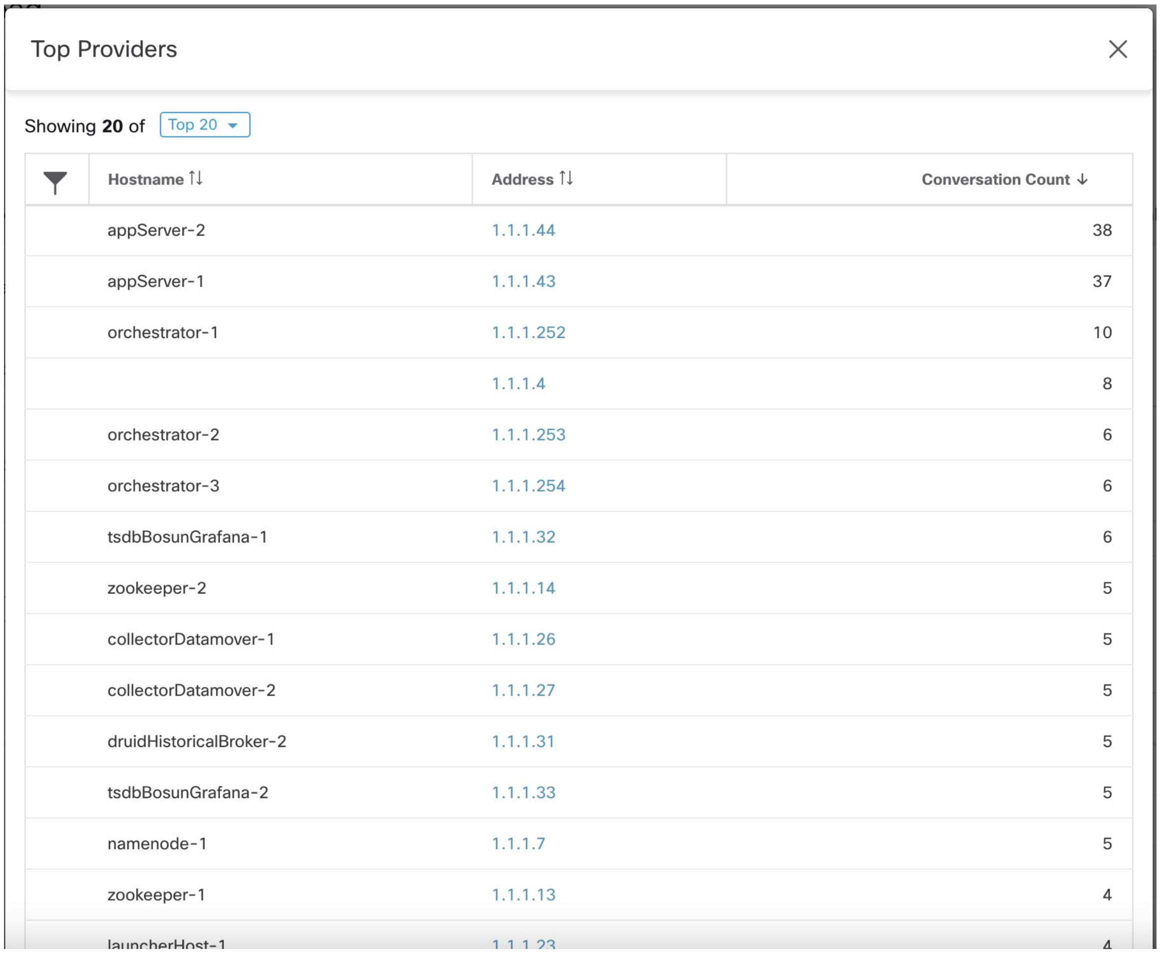
*Figure 91: Top Consumers Modal*


Top Consumers

Showing **20** of Top 20

| Hostname | Address | Conversation Count |
|---|---|---|
| appServer-2 | | 38 |
| appServer-1 | | 37 |
| orchestrator-1 | | 10 |
| | | 8 |
| orchestrator-2 | | 6 |
| orchestrator-3 | | 6 |
| tsdbBosunGrafana-1 | | 6 |
| zookeeper-2 | | 5 |
| collectorDatamover-1 | | 5 |
| collectorDatamover-2 | | 5 |
| druidHistoricalBroker-2 | | 5 |
| tsdbBosunGrafana-2 | | 5 |
| namenode-1 | | 5 |
| zookeeper-1 | | 4 |

*Figure 92: Top Providers Modal*


Top Providers

Showing **20** of Top 20

| Hostname | Address | Conversation Count |
|---|---|---|
| appServer-2 | 1.1.1.44 | 38 |
| appServer-1 | 1.1.1.43 | 37 |
| orchestrator-1 | 1.1.1.252 | 10 |
| | 1.1.1.4 | 8 |
| orchestrator-2 | 1.1.1.253 | 6 |
| orchestrator-3 | 1.1.1.254 | 6 |
| tsdbBosunGrafana-1 | 1.1.1.32 | 6 |
| zookeeper-2 | 1.1.1.14 | 5 |
| collectorDatamover-1 | 1.1.1.26 | 5 |
| collectorDatamover-2 | 1.1.1.27 | 5 |
| druidHistoricalBroker-2 | 1.1.1.31 | 5 |
| tsdbBosunGrafana-2 | 1.1.1.33 | 5 |
| namenode-1 | 1.1.1.7 | 5 |
| zookeeper-1 | 1.1.1.13 | 4 |
| launcherHost-1 | 1.1.1.23 | 4 |

## Automated Load Balancer Config for Automatic Policy Discovery (F5 Only)

**Important** **This is an experimental feature.**

This feature and its APIs are in **ALPHA** and are subject to changes and enhancements in future releases.

Automatic policy discovery generates policies from configuration for load balancers connected to an external orchestrator. Generating policies from configuration minimizes reliance on flow data and improves the accuracy of discovered clusters and policies.

It relies on clients to report flows to the load balancer for generating policies to permit this traffic.

## Terminology

**VIP** Virtual IP: IP to which the client sends traffic that is destined for a service.

**SNIP** SNAT IP: IP used by the load balancer for sending traffic to backend hosts.

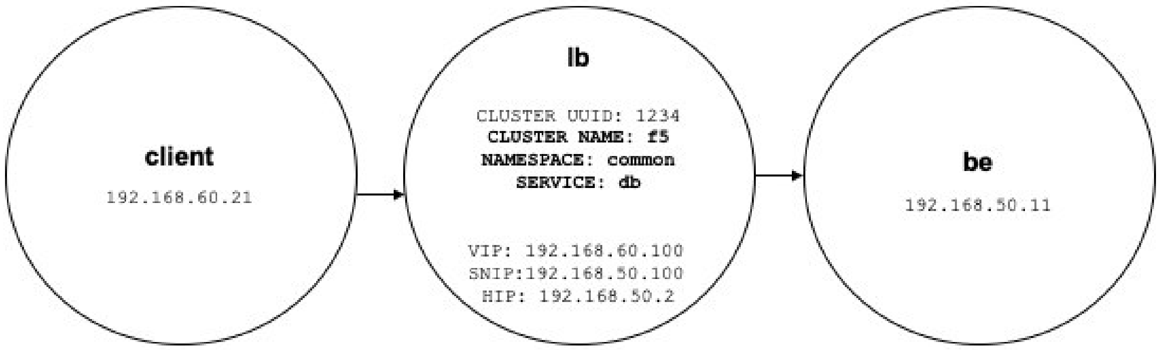
**BE** Backend Endpoint: IP of the backend host.

**HIP** Health-check IP: Source IP used by the load balancer for sending health-check traffic to backend hosts.

**Note** HIPs are the same as SNIPs in automap mode. However, HIPs and SNIPs can differ when a SNAT pool is configured.

## Deployment

*Figure 93: Deployment*



Consider the following deployment where load balancer VIPs, SNIPs, and HIPs are part of the *lb* scope, and BEs are part of the *be* scope. Scopes are created as follows.

- Client

  The client scope includes clients communicating with the load balancer. For the example above, the *client* scope query is as follows:

  ```
  address eq 192.168.60.21 or address eq 192.168.60.22
  ```

- lb

  The F5 external orchestrator labels VIPs, SNIPs, HIPs, and BEs used by the load balancer. These labels can be used to construct scope queries, where *orchestrator_system/service_name* is used for selecting VIPs, *orchestrator_system/service_startpoint* SNIPs, and *orchestrator_system/service_healthcheck_startpoint* HIPs for the service. For the example above, a scope query that includes VIPs, SNIPs, and HIPs for service *db* is as follows:

  ```
  user_orchestrator_system/cluster_id eq 1234 and
  (user_orchestrator_system/service_name eq db or
  user_orchestrator_system/service_startpoint eq db or
  user_orchestrator_system/service_healthcheck_startpoint eq db)
  ```

**Note** It is required that SNIPs and VIPs be part of the same scope.

- Be

  *user_orchestrator_system/service_endpoint* selects BEs for a service. For the example above, a scope query that includes BEs for service *db* is as follows:

  ```
  user_orchestrator_system/cluster_id eq 1234 and
  user_orchestrator_system/service_endpoint eq db
  ```

## Clusters

Each service generates up to four discovered clusters, of which only the service cluster is visible to the user. SNIP, HIP, and BE clusters appear as related clusters for the service cluster. HIP and BE clusters are generated only when HIPs and BEs are present in the *lb* scope.

For the example above, automatic policy discovery generates a SNIP cluster and HIP cluster in the *lb* scope that include SNIPs and HIPs for service. Since BEs lie outside the *lb* scope, automatic policy discovery does not generate a backend cluster but instead adds the *be* scope to the list of related clusters for *db*.

Clusters are generated as follows:

- Service

  The service cluster includes VIPs for service. The query for the service cluster as follows:

  ```
  user_orchestrator_system/cluster_id eq 1234 and
  user_orchestrator_system/namespace eq common and
  user_orchestrator_system/service_name eq db
  ```

- SNIP

  SNIPs for a service are included in the SNIP cluster. The query for the SNIP cluster is as follows:

  ```
  user_orchestrator_system/cluster_id eq 1234 and
  user_orchestrator_system/service_startpoint eq db
  ```

- HIP

  HIPs for a service are included in the HIP cluster. The query for the HIP cluster is as follows:

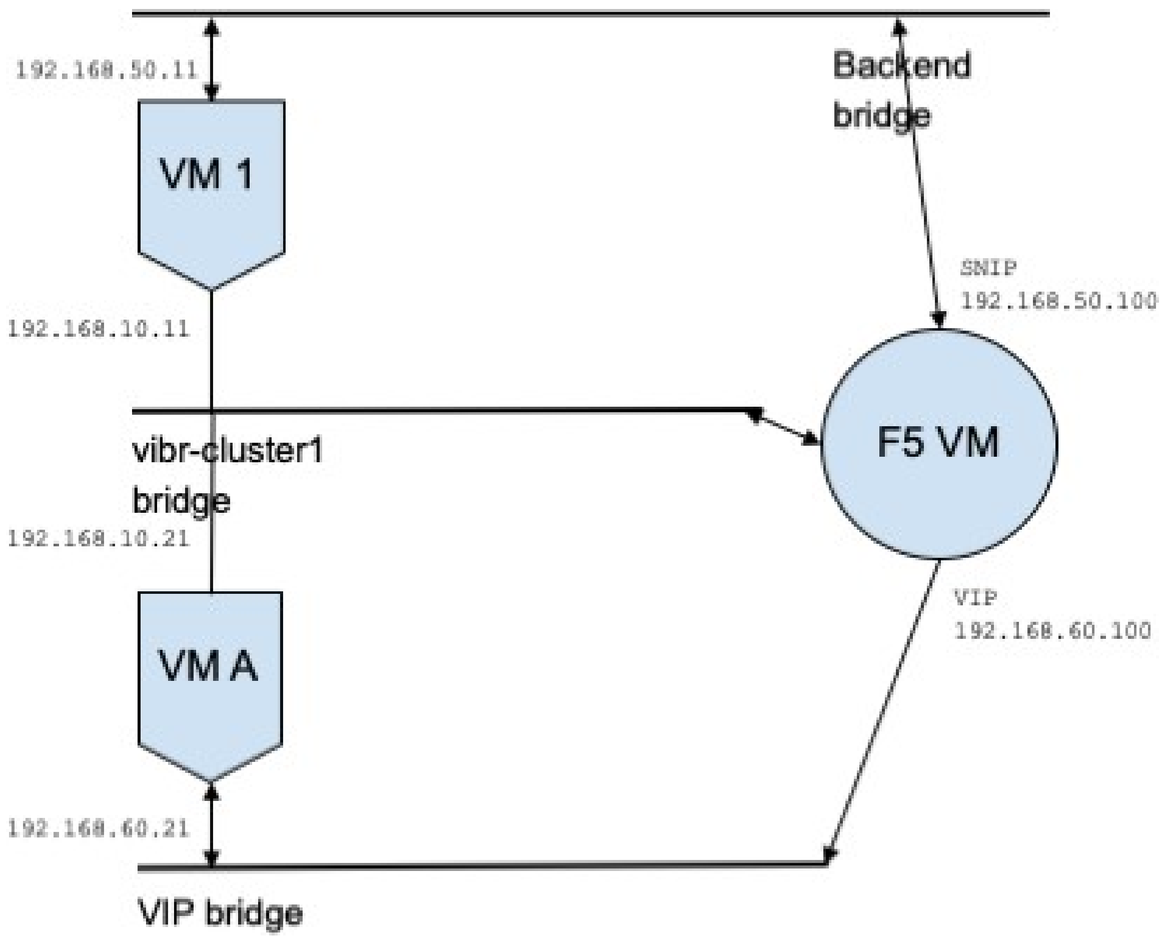
  ```
  user_orchestrator_system/cluster_id eq 1234 and
  user_orchestrator_system/service_healthcheck_startpoint eq db
  ```

- Backend

  A backend cluster for the service is generated when one or more BEs are part of the *lb* scope. This doesn’t apply to the example above, resulting in a backend cluster not being generated in the *lb* scope.

## Policies

*Figure 94: Policy Generation*



Assume we have a service *db* with VIP *192.168.60.100*, SNIP *192.168.50.100*, and a backend VM with IP *192.168.50.11* listening on port 10000. Traffic from client VM *192.168.60.21* to *db* results in the following policies:

- Policy from client to VIP

  The following policy permits from the client VM to service *db*.

```
{
"src": "<uuid of client scope>",
"dst": "<uuid of service cluster>",
"l4_params": [
{
"port": [
10000,
10000
],
"proto": 6,
}
]
}
```

- Policy from SNIP to BE.

  A policy permitting traffic from the SNIP to the BE is autogenerated from configuration, and shows up as a related policy for *db*.

  ```
  {
  "src": "<uuid of SNIP cluster>",
  "dst": "<uuid of be scope>",
  "l4_params": [
  {
  "port": [
  10000,
  10000
  ],
  "proto": 6,
  }
  ]
  ```

  A policy connector from the *lb* scope to the *be* scope pushes the following policy to it.

  | Consumer | Provider | Port | Protocol | Action |
  |---|---|---|---|---|
  | SNIP | be | 10000 | TCP | Allow |

  This generates firewall rules on BE host 192.168.50.11 allowing incoming traffic from LB SNIP 192.168.50.100 on port 10000.

- Policy from HIP to BE.

  A policy permitting traffic from the HIP to the BE is autogenerated from configuration, and shows up as a related policy for *db*.

  ```
  {
  "src": "<uuid of HIP cluster>",
  "dst": "<uuid of be scope>",
  "l4_params": [
  {
  "port": [
  0,
  0
  ],
  "proto": ICMP,
  }
  ]
  }
  ```

  A policy connector from the *lb* scope to the *be* scope pushes the following policy to it.

  | Consumer | Provider | Port | Protocol | Action |
  |---|---|---|---|---|
  | HIP | be | 0 | ICMP | Allow |

  This generates firewall rules on BE host *192.168.50.11* allowing incoming ICMP traffic from LB HIP *192.168.50.2*.

## Caveats

- When multiple services from the same load balancer instance have the same name, backend rules generated for any of these services will include backend pools for all of them, i.e. rules will be more permissive than needed.

# Policies Publisher

*Policies Publisher* is an advanced Cisco Secure Workload feature allowing third -party vendor to implement their own enforcement algorithms that are optimized for network appliances such as load balancers or firewalls. This feature is realized by publishing defined policies to a Kafka instance residing within Secure Workload cluster and by providing customers with Kafka client certificates, which allows third-party vendor code to retrieve policies from Kafka and to translate them into their network appliances configuration appropriately.

This section aims to describe the procedure third-party vendors, in short users in the following, must perform to exploit the *Policies Publisher* feature with Java on Linux.

## Prerequisites

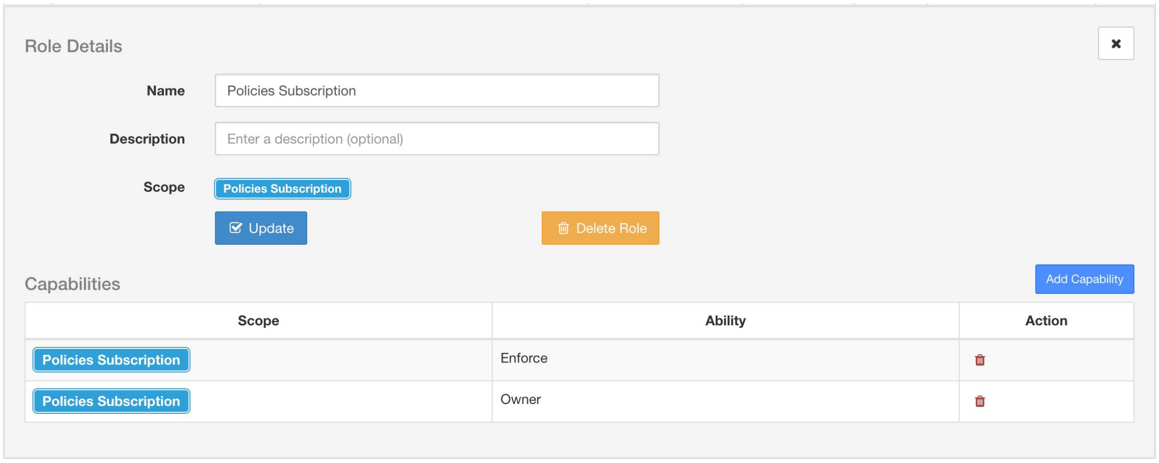
The following software packages are installed on a Linux system, such as Ubuntu 16.04.

- Java 8 JDK
- Apache Kafka Clients: kafka-clients-1.0.0.jar
- Protocol Buffers Core: protobuf-java-3.4.1.jar
- Apache Log4j: log4j-1.2.17.jar
- Simple Logging Facade for Java: slf4j-api-1.7.25.jar, slf4j-log4j12-1.7.25.jar
- Snappy compressor/decompressor for Java: snappy-java-1.1.4.jar

## Getting Kafka Client Certificates

- Create a user role with capability *“Owner”* and assign it to a user account of choice:

*Figure 95: User Role Configuration to Receive Policies from Kafka*



Role Details

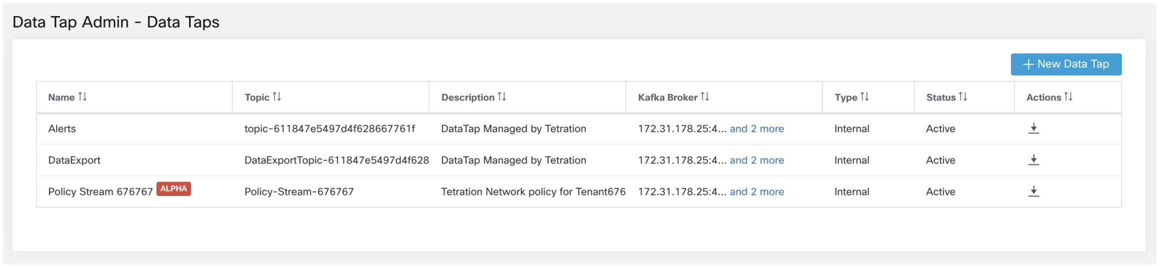
| | |
|---|---|
| Name | Policies Subscription |
| Description | Enter a description (optional) |
| Scope | Policies Subscription |

Capabilities

| Scope | Ability | Action |
|---|---|---|
| Policies Subscription | Enforce | 🗑 |
| Policies Subscription | Owner | 🗑 |

- Perform policies enforcement as described in Enforce Policies. This first step is necessary as it creates a Kafka topic that is associated with active scope.
- Navigate to **Manage** > **Data Tap Admin**
- Select the tab *"Data Taps"* and download Kafka client certificates by clicking on the download button under column *"Actions"*. Make sure to select the *Java Keystore* format in the download dialog.

*Figure 96: Data Taps View*



Data Tap Admin - Data Taps

| Name | Topic | Description | Kafka Broker | Type | Status | Actions |
|---|---|---|---|---|---|---|
| Alerts | topic-611847e5497d4f628667761f | DataTap Managed by Tetration | 172.31.178.25:4... and 2 more | Internal | Active | ⤓ |
| DataExport | DataExportTopic-611847e5497d4f628 | DataTap Managed by Tetration | 172.31.178.25:4... and 2 more | Internal | Active | ⤓ |
| Policy Stream 676767 ALPHA | Policy-Stream-676767 | Tetration Network policy for Tenant676 | 172.31.178.25:4... and 2 more | Internal | Active | ⤓ |

- The downloaded clients certificates file usually has a name like *Policy-Stream-10-Policies-Subscription.jks.tar.gz*. Create a directory and unpack it underneath the created directory as below:

```
    mkdir Policy-Stream-10-Policies-Subscription
    tar -C Policy-Stream-10-Policies-Subscription -zxf
Policy-Stream-10-Policies-Subscription.jks.tar.gz
```

## Protobuf Definition File

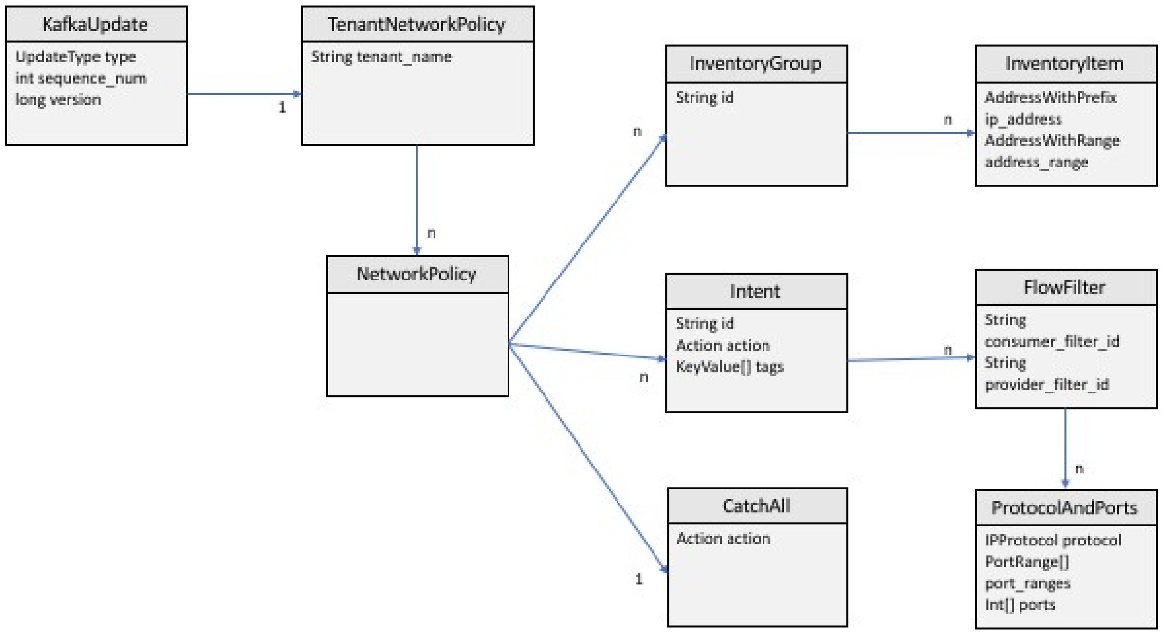
The network policies exposed by Secure Workload backend to Kafka are encoded in Google Protocol Buffers format. Refer to this guide for instructions how to download and install it on your Linux system.

The proto file of Secure Workload network policy can be downloaded from here.

## Data Model of Secure Workload Network Policy

Picture below shows a simplified UML diagram of Secure Workload entities exposed to Kafka:

***Figure 97: Data Model of Secure Workload Network Policy***



A *Secure Workload Network Policy* as modeled in protobuf consists of a list of *InventoryGroups*, a list of *Intents* and a *CatchAll* policy. Each policy contains all the items belonging to one root scope. An *InventoryGroup* contains a list of *InventoryItems*, which represent Secure Workload entities such as servers or appliances by specifying their network address, be it a singular network address, subnet or address range. An *Intent* describes action (allow or deny) to be taken when a network flow matches with the given consumer's *InventoryGroup*, provider's *InventoryGroup* and network protocols and ports. The *CatchAll* represents the catch-all action that is defined for the root scope inside Secure Workload. If no workspace with enforcement enabled exists for the root scope, a default policy of *ALLOW* is written to the produced policy.

When an enforcement is triggered by the users or by a change of inventory groups, Secure Workload backend sends a full snapshot of defined network policies to Kafka as a sequence of messages that are represented as *KafkaUpdates*. Refer to *KafkaUpdate*'s comments in *tetration_network_policy.proto* file for details how to reconstruct those messages to a full snapshot and how to handle error conditions.

In case *KafkaUpdate* message size is greater than 10MB, Secure Workload backend splits this message into multiple fragments, each of size 10MB. If there is multiple fragments, only the first fragment has the *ScopeInfo* field of *TenantNetworkPolicy*. The *ScopeInfo* will be set to nil in the remaining fragments of *KafkaUpdate* message.

## Reference Implementation of Secure Workload Network Policies Client

For implementation and instructions on how to compile and run a demo client, see tnp-enforcement-client in Java.

This implementation provides common code to read network policies from Secure Workload policy stream via Kafka only. Vendor-specific code to program the actual policies to a network device can be plugged in by implementing the required interface PolicyEnforcementClient.