



12Port Horizon

QUICK START GUIDE FOR LINUX

Table of Contents

- Introduction2
- Recommendations for Success2
- Terminology2
- Installation.....5
- Initial Setup.....5
- What to Expect7
- Assets7
- Linux Asset Status Check..... 11
- Policies 13
- Monitoring 16
- Enforcement 19
- Recap 24
- Appendix 25
 - Production Deployment Guide 25

Introduction

Welcome to the 12Port Horizon Quick Start Guide for Linux. This guide is tailored for users who are eager to quickly grasp the essentials, ensuring you can quickly get up and running with the software's core features. Whether you're a new or trial user looking to familiarize yourself with the basics or an experienced user seeking to refresh your knowledge, this guide will walk you through setup, product terminology, and key functionalities to configure your first network micro-segments. By the end of this guide, you'll have a solid foundation to leverage 12Port Horizon effectively in your continued micro-segmentation project. Let's get started!

Recommendations for Success

To begin, let's discuss specific recommendations to maximize your trial experience using this Quick Start Guide as a reference. These suggestions are crafted to guide you in effectively evaluating the software's capabilities, understanding its value proposition, and assessing its suitability for your needs.

For more advanced features and deeper walk-through guides, please visit our documentation portal: <https://docs.12port.com/horizon/>

- Two physical or virtual dedicated Linux machines that will be used to build our segmentation walkthrough example. These should be non-production servers.
 - Optional, but recommended, a third dedicated Linux machine where the software will be installed.
- SSH connectivity between all servers
- A user for each Linux machine with SSH access, sudo privileges, and NOPASSWD enabled

Terminology

In this section, we'll clarify key terms and concepts within 12Port Horizon, ensuring you have a solid understanding of its vocabulary. By familiarizing yourself with these definitions, you'll be better equipped to navigate the software's user interface and utilize its features effectively.

Micro-segmentation

Micro-segmentation is a network security practice that divides a network into isolated zones to prevent lateral movement (east-west traffic) across the network by bad actors or malicious operations. The micro-segmentation practice assumes that part of the network is compromised by a bad actor, either an external attacker or an insider, with the goal to prevent the compromise of the rest of the network.

Micro-segmentation is often included in Network Infrastructure Management, in particular, Zero Trust Network Architecture. The NSA provides a [good overview](#) of the value of micro-segmentation.

Assets

An asset is an electronic record describing a network device, an account, or any other physical or logical entity to hold data in a secure way. It could also be a container to logically group other assets for navigation and configuration purposes.

An asset must belong to one of the pre-configured asset types to categorize the asset, to define its metadata requirements and to define the asset behavior. Many other objects like tags, permissions, tasks, and policies are applied to assets to create network micro-segments.

Taxonomy, Terms, Tags

Taxonomy is a formal classification system that groups the words, terms and synonyms that describe something, and then arranges the terms into a hierarchy. In this structure a term might include other terms or servers as an endpoint of the term hierarchy. Each term might include synonyms to simplify tagging and search.

The software uses taxonomies to enable various operations such as

- tag assets to facilitate consistent use of metadata.
- assign labels to indicate various asset characteristics.
- group assets by multiple independent criteria to improve assets discoverability.
- simplify search for assets following taxonomy hierarchies.
- assign configuration, security policies and communication protocols to various groups of assets under management.

Versioning (Minor, Major) and Promotion

Within the software, asset version number includes Major and Minor versions. Every time an asset is created or updated a new minor version is assigned (i.e., 0.1, 1.2, 2.3, etc.). A minor version could be promoted to become the next major version (i.e., 1.0, 2.0, 3.0, etc.).

Micro-segmentation policies use major versions of assets for either selection or source selection criteria. It allows asset owners to review the changes in asset metadata and tagging (minor versions) before publishing the updates for micro-segmentation processing (major versions).

Services

A Service or Workload refers to specific applications, processes, or resources running within a network environment. These can include web servers, databases, communication protocols, or other types of services that require access control and segmentation. This software allows organizations to define and enforce segmentation policies that restrict communication between these different services or workloads, thereby enhancing security by minimizing the attack surface and reducing the risk of unauthorized access or data breaches.

Segmentation Policies

A segmentation policy refers to rules or configurations that regulate the communication and interaction between services or workloads within a network. These policies dictate how traffic flows between different segments, specifying permissions, imposing restrictions, and advancing security measures. By defining policies in 12Port Horizon, organizations can enforce granular control over network traffic, ensuring that only authorized interactions occur while mitigating the risks of potential threats.

Policy Monitoring

Segmentation policies can be set to a *Monitoring-only* state that observes and reports on network traffic and interactions between services or workloads. These policies do not enforce restrictions but rather monitor and analyze data flow for insights into network behavior. This monitoring only helps organizations gain visibility and understanding of their network environment, allowing them to make informed decisions about security measures and operational compatibility before enforcement is enabled.

Policy Enforcement

Segmentation policies can be enforced so they are actively controlling and managing network traffic between services or workloads. These policies enable security measures on the asset that dictate how data can flow within the network. Enforcing these policies ensures that only authorized traffic occurs, while unauthorized or potentially risky communications are restricted. This proactive enforcement enhances

network security by minimizing the attack surface and preventing unauthorized traffic between managed assets.

Installation

12Port Horizon can be installed on a Windows or Linux server OS and used with any Windows and/or Linux network. To keep things simple in this guide, we will walk-through using a Linux environment for segmentation.

For trial or test deployments, you may install 12Port Horizon on a laptop or workstation with minimal requirements, but for Production deployments please only install the software on a dedicated server. For specific requirements, please review our System Requirements (<https://docs.12port.com/horizon/guides/installation/#system-requirements>)

On your third Linux machine or a workstation you have chosen to be the 12Port Horizon host, download the Linux installation script `setup.sh` (<https://bin.12port.com/product/setup.sh>) to a directory such as `/opt/12port` and run it. *Do not install 12Port to a temp directory.* Follow the prompts to install the software and start the 12Port application service.

Initial Setup

After installing 12Port Horizon, you can access it via the server's web browser using the URL `https://localhost:6443/ztna`, where `<localhost>` is the hostname or address of the system where the 12Port application is installed.

When accessing the application for the first time, it will redirect to the **Administrator Registration** page. Enter a username and password to create the Administrator account. This first account will become the first Administrator of the base deployment. Be sure to safeguard your login credentials. Also, set up MFA on that account using TOTP in the configuration area or integrate with one of the other MFA providers or SSO with MFA.

As 12Port Horizon natively supports multitenancy, all network management operations are performed within the scope of an asset tenant which maintains independent data and configuration settings for each individual network. However, the application is initialized with only the base tenant upon installation. The base tenant is strictly used to manage other tenants, and network assets cannot be managed from it. After registering this administrator account, you will be prompted to create the first asset tenant so that you can

begin adding and managing assets and segmentation policies. Note the name of the tenant you created. You can use that to directly login to that tenant.

Depending on your planned usage, there are several choices to consider on this screen:

- **Tenant Update Type:** If you are using a single server deployment, choose "Create Standalone."
- **Name:** Enter a descriptive name for the asset tenant. This name will become part of the URL used to access the tenant.
- **Issuer:** Issuer, which is usually a tenant URL, identifies a tenant for external parties such as SSO identity providers, browsers, scripts, or applications integrating with this tenant using REST API calls. Tokens and exchange documents that the tenant signs are generated with this unique identifier.
When using the default *\${dynamic}* value, the tenant generates the issuer value based on the URL a client accesses the tenant during the request to use the issuer. The downside of using dynamic issuer generation is that all tokens and exchange documents generated with a different issuer will be invalid when the tenant is accessed using a different URL.
- **Language:** Select the tenant's default language.
- **Database:** If you would like to use the embedded database included with the application, choose "Embedded." The embedded database is suitable for trials and small-scale lab or production deployments. If you would instead like to use an existing standalone database, choose the DBMS variant, and enter your URL and credentials for access.

For this Quick Start Guide, we will select the following:

- **Tenant Update Type:** Create Standalone
- **Name:** QuickStart *or another alphanumeric name of your choosing*
- **Issuer:** *\${dynamic}*
- **Language:** English
- **Database:** Embedded

Once all fields are populated, click the **Save** button to add the asset tenant. This process may take a few seconds to complete. You will be automatically redirected to this new tenant once created.

Now your first new tenant has been created, we can start constructing the building blocks for your segmentation, beginning with your Assets.

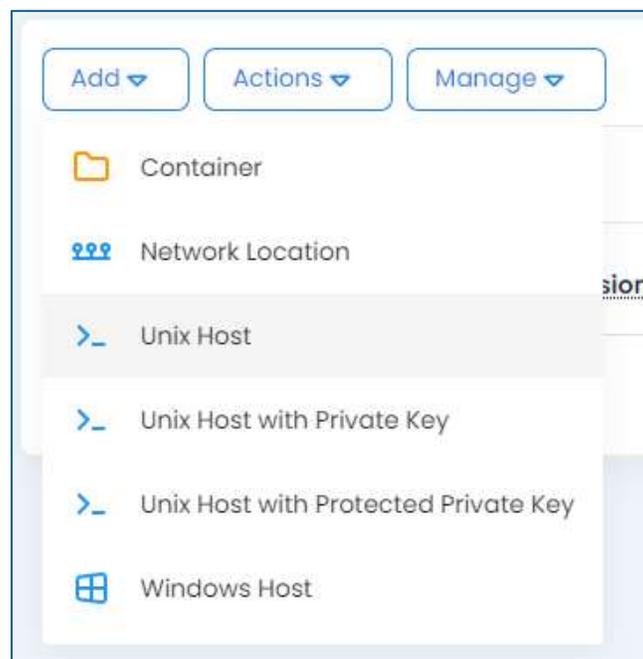
What to Expect

Between our two Linux servers, one will represent a Development server and the second a Production server. In the initial state, all connections between these machines are allowed, and an SSH connection can be established between the two. When we construct our micro-segment example using an SSH policy, 12Port Horizon will first monitor connectivity between the servers highlighting any violations that have occurred allowing us to observe this specific traffic. After we monitor and analyze the violations, we will then enforce the example SSH segmentation policy that will block this traffic, ensuring that the server in the Development environment can no longer connect to the server in the Production environment. Furthermore, we will create a DevOps asset, whose access to Production must remain open and uninterrupted after SSH policy enforcement.

Assets

Once you have been redirected to your new tenant, use the left side navigation to open the page Database > Assets. From this page, we will create a new Asset for each of your two (or three) Linux servers that will be used in this walkthrough guide.

1. From the Assets library, use the **Add > Unix Host** option to create your Linux server asset.



2. Use the following guidance for this first Asset, representing the Development server:
 - a. Name: Linux Development Server
 - b. Description: *optional*
 - c. Host: Enter the IP address or network accessible computer name of your first server
 - d. User: Enter the username of an account on this server with sudo privileges
 - e. Password: Enter the password of this account
 - f. Tags:
 - i. Type “Linux” and select **Component :: Server :: Linux** from the type ahead dropdown menu or use the **Select Term** button to select **Component > Server > Linux** as the tag.
 - ii. Type “Development” and select **Environment :: Development** from the type ahead dropdown menu or use the **Select Term** button to select **Environment > Development** as the tag.
3. Click the **Save** button to create the asset.

The screenshot shows a configuration form for an asset named "Linux Development Server". The form includes the following fields and values:

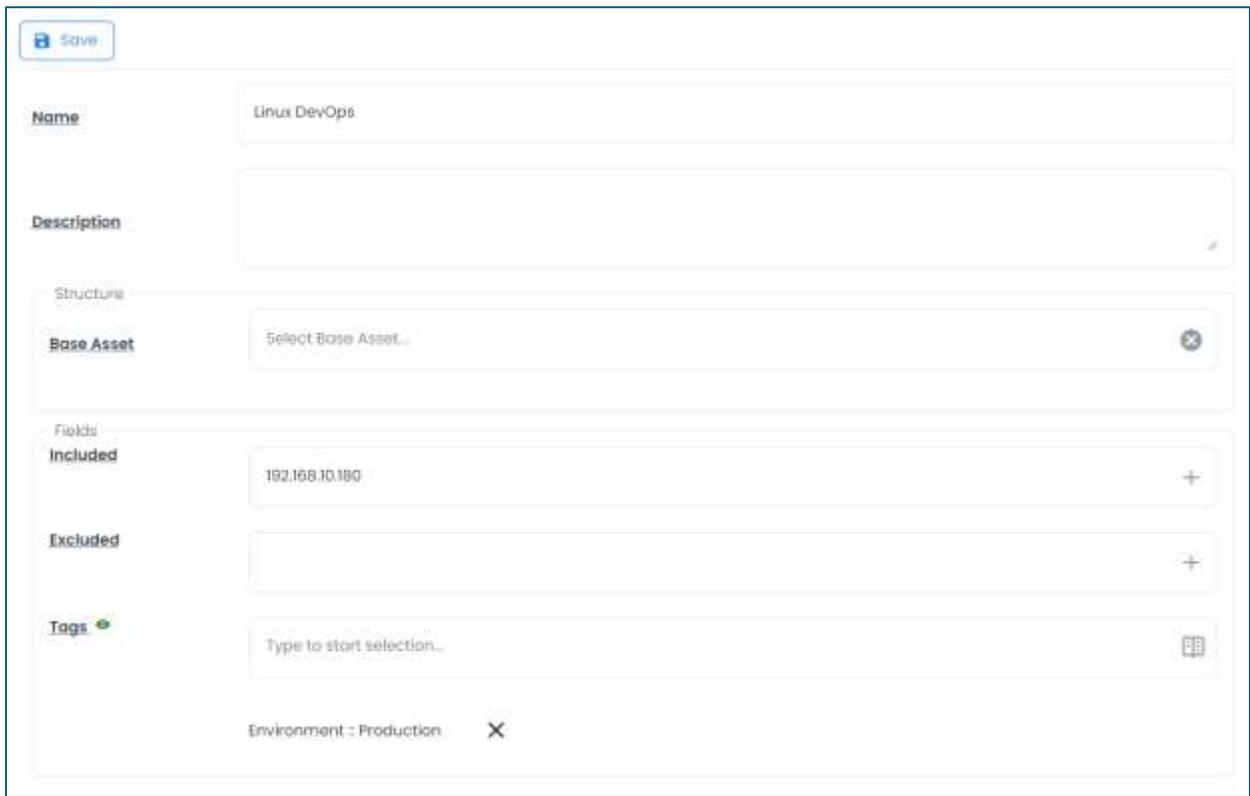
- Name:** Linux Development Server
- Type:** Unix Host (Unix or Linux Host)
- Version:** 0.1
- Created:** ztnaadmin (First Last) @ 08/22/2024 10:17:48
- Modified:** ztnaadmin (First Last) @ 08/22/2024 10:17:48
- Host:** 192.168.10.100
- User:** administrator
- Password:** *** SECURED ***
- Tags:** [Environment :: Development], [Component :: Server :: Linux]

4. From the *Assets* library, use the **Add > Unix Host** option to create your second Linux server asset.
5. Use the following guidance for your second Asset, representing your Production server:
 - a. Name: Linux Production Server
 - b. Description: *optional*
 - c. Host: Enter the IP address or network accessible Computer name of your second server
 - d. User: Enter the username of an account that has Administrator access to the server
 - e. Password: Enter the password of this account
 - f. Tags:
 - i. Type “Linux” and select **Component :: Server :: Linux** from the type ahead dropdown menu or use the **Select Term** button to select **Component > Server > Linux** as the tag.
 - ii. Type “Production” and select **Environment :: Production** from the type ahead dropdown menu or use the **Select Term** button to select **Environment > Production** as the tag.
6. Click the **Save** button to create the asset.
7. From the *Assets* library page, use the **Add > Network Location** option to create your DevOps asset. If you have a third server with native SSH access to the previous Linux Production Server asset, you may use the actual values to create this asset. Otherwise, we will simulate the asset using fictitious values.

If you do not have a third server, then later in the walkthrough you will only be able to use the Preview option to simulate policies. Instead, using an actual server asset will allow you to both monitor and enforce a policy that will confirm uninterrupted SSH access from this DevOps workstation to the Production server.

8. Use the following guidance for this DevOps asset:
 - a. Name: Linux DevOps
 - b. Description: *optional*
 - c. Included: Enter the IP address or network accessible Computer name of your second server or use the value 10.10.10.1 as a simulated IP address if you do not have a third server. Click the + icon to set the value.
 - d. Tags:
 - i. Type “Production” and select **Environment :: Production** from the type ahead dropdown menu or use the **Select Term** button to select **Environment > Production** as the tag.

9. Click the **Save** button to create the asset.

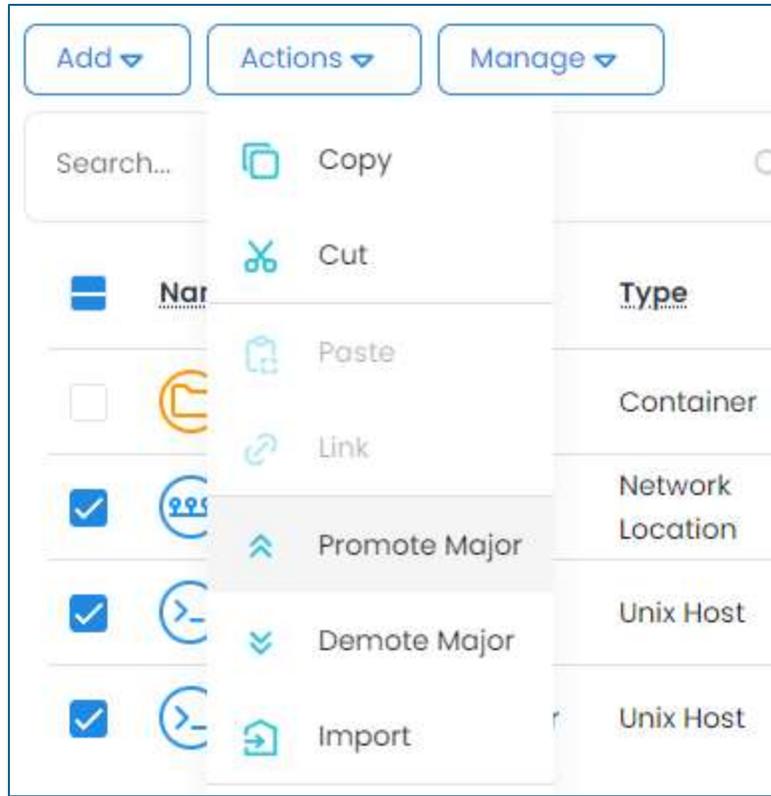


The screenshot shows a form for creating an asset. At the top left is a 'Save' button. The form has several sections:

- Name:** A text input field containing 'Linux DevOps'.
- Description:** A large empty text area.
- Structure:** A section containing a 'Base Asset' dropdown menu with the text 'Select Base Asset...' and a close icon.
- Fields:** A section containing two input fields:
 - Included:** A text input field containing '192.168.10.180' and a plus icon.
 - Excluded:** An empty text input field and a plus icon.
- Tags:** A text input field with the placeholder 'Type to start selection...' and a tag icon.

At the bottom of the form, there is a label 'Environment : Production' with a close icon.

10. Return to the Assets library, select all 3 assets by clicking the square to the left of each asset, and use the **Actions > Promote Major** option to promote these assets to a major version. A major version is a requirement for the policy selection built in the future Policies section.



Linux Asset Status Check

Before we continue, let's take a moment to ensure that 12Port Horizon has connectivity to each of your Linux server assets.

1. From the *Assets* library, open the *Linux Development Server* asset by clicking on this asset's Name.
2. When the *View Asset* page opens, open the **Execute** dropdown menu, and select the **Linux Status Check** option to execute this task.

Permissions

Linux Development Server
 Type: Unix Host (Unix or Linux Host)
 Version 0.1
 Created: ztnoadmin (First Last) @ 08/22/2024 10:17:48
 Modified: ztnoadmin (First Last) @ 08/22/2024 10:17:48

Host: 192.168.10.160

User: administrator

Password: *** SECURED ***

Tags: [Environment : Development] [Component : Server : Linux]

- IPTables Add Rule
- IPTables Delete Rule
- IPTables Initialization
- IPTables List
- Linux Connections
- Linux Firewall Management
- Linux Firewall Management Batch
- Linux Interfaces
- Linux Monitoring
- Linux Status
- Linux Workloads
- Unix Status Check

3. Click **Ok** on the confirmation dialog box to continue.
4. Repeat this procedure for the *Linux Production Server* asset.
5. After a brief period, use the left side menu to navigate to Reports > Statuses. From this report, you should see one entry for each of your Linux server assets. The presence of these report entries, and the information contained in the details, confirms the connectivity from 12Port Horizon was successful to each of these assets.

Export

Filter...

Page size: 25 | 1 - 2 of 2 | [Navigation icons]

| <input type="checkbox"/> | Created | Updated | Asset | Description | OS Version | Kernel | Host | HW | Actions |
|--------------------------|---------------------|---------------------|--|------------------|--------------------------|--------|--|--------|---------|
| <input type="checkbox"/> | 08/22/2024 10:27:36 | 08/22/2024 10:27:36 | Linux Production Server | Ubuntu 24.04 LTS | 24.04 LTS (Noble Numbat) | Linux | ubuntutest2-Standard-PC-i440FX-PiIX-1996 | x86_64 | ▼ |
| <input type="checkbox"/> | 08/22/2024 10:26:56 | 08/22/2024 10:26:56 | Linux Development Server | Ubuntu 24.04 LTS | 24.04 LTS (Noble Numbat) | Linux | ubuntutest-Standard-PC-i440FX-PiIX-1996 | x86_64 | ▼ |

If an expected entry did not appear in the Statuses report, navigate to Report > Jobs and search for any related Failed task executions. Review the Troubleshooting section at the end of this guide for further assistance with Job failures.

Policies

With the required assets created and properly tagged, and connectivity has been confirmed, we will move to creating our segmentation policy. This segmentation policy will be built so that traffic restrictions can be monitored or enforced preventing SSH connectivity between Development and Production, while not disrupting the SSH connectivity from DevOps to Production.

1. Using the left side menu, navigate to *Management > Policies* and click the **Add** button to create your segmentation policy.
2. Use the following guidance for this segmentation policy:
 - a. Description: SSH Segmentation for Linux Servers
 - b. Publishing: Monitored
 - c. Selector Taxonomy: LECA Segmentation
 - d. Selector Segment: Component :: Server :: Linux
 - e. Service: SSH (22/tcp)
 - f. Source Segment: Environment; Match: Same

Save
Preview

Description

SSH Segmentation for Linux Servers

Publishing

Monitoring
▼

Selector

Taxonomy

LECA Segmentation

Selector Segment

Type to start selection...

Component : Server - Linux
✕

Service

ssh (22/tcp)
📄 ✕

Source

Source Segment

Type to start selection...

📄
Some
▼

| Term | Match | Actions |
|-------------|-------|---------|
| Environment | Some | ✎ ✕ |

3. Click the **Save** button.
4. Still on this *Edit Segmentation Policy* page, click on the **Preview** button on the right side. This will open the *Policy Query Preview* screen for this policy that will allow you to review its current state and which Source(s) will be included when it is Monitored or Published.

| Policy Query Preview | | |
|--------------------------|------------------------------|--|
| Policy | | |
| Taxonomy: | LECA Segmentation | |
| Selector: | [Component : Server : Linux] | |
| Source: | [Environment : Same] | |
| Service: | ssh (22/top) | |
| Filter... | | |
| Name | Type | Terms |
| Linux Development Server | Unix Host | [Component : Server : Linux] [Environment : Development] |
| Linux Production Server | Unix Host | [Component : Server : Linux] [Environment : Production] |

Before we continue, let's take a moment to describe what is shown in the Policy Query Preview screen.

First, the Policy Query Preview screen is designed to allow users to review the effect of policy application and analyze the results of the queries performed by the policy selectors. It is an easy and quick way to check your policy configuration before you decide to monitor or publish the policy.

In the specific example above, what we can learn from this screen is quite important. We can see that two assets, Linux Development Server and Linux Production Server, are selected by way of the defined policy selector. Both assets are selected in this policy because of the selector query *Component :: Server :: Linux*, which both assets include in their Tags.

Next, we can expand the Linux Production Server asset and observe that its *Source Criteria* resolved from the policy definition "Environment : Same" to the actual criteria in this scenario which is "Environment : Production." This resolution occurred like this because the *Linux Production Server* asset was tagged with the term "Environment :: Production."

| Name | Type | Terms |
|--------------------------|-----------|--|
| Linux Development Server | Unix Host | [Component : Server : Linux] [Environment : Development] |
| Linux Production Server | Unix Host | [Component : Server : Linux] [Environment : Production] |

| Linux Production Server | | |
|-------------------------|---|--|
| Name | Linux Production Server | |
| Type | Unix Host | |
| Description | | |
| Terms | [Component : Server : Linux] [Environment : Production] | |
| Source Criteria | [Environment : Production] | |

| Source | | |
|--------------|------------------|----------------------------|
| Name | Type | Terms |
| Linux DevOps | Network Location | [Environment : Production] |

Furthermore, in the Source table you will also see that the *Linux DevOps* asset is present indicating that this asset will retain SSH connectivity to the Linux Production Server host. What you will not see in this Source table is *Linux Development Server* since that asset did not include the same “Environment :: Production” tag because its SSH connectivity to Linux *Production Server* is not allowed.

In summary, the Policy Query Preview screen demonstrates that when this policy is set to published and is enforced, Linux Development Server will no longer have SSH connectivity to the Linux Production Server, while Linux DevOps SSH connectivity will remain uninterrupted.

Monitoring

In the earlier *Policies* section, when we created our segmentation policy, we set the *Publishing* state to **Monitored**. This shows that the policy itself will not restrict our SSH connectivity, but rather it will monitor connections on our asset Hosts and report on detected violations. A violation is traffic that would be restricted if the policy itself is set for enforcement. We will cover enforcement in the next section of this guide.

Let’s explore Monitoring in our scenario using native SSH connections from the Linux Development Server host and the Linux Production Server host.

1. From the actual Linux Development Server host machine, establish an SSH connection to the actual Linux Production Server host machine.
2. Once this SSH connection is connected, return to the Assets library, and open the Linux Production Server asset by clicking on this name.
3. On the asset page, click on the **Execute > Linux Monitoring** option to execute this task manually that monitors connections to this Host machine (inbound).
4. Next, navigate to Reports > Connections and observe the results generated from this Monitored segmentation policy.

Linux Production Server
 Type: Unix Host (Unix or Linux Host)
 Host: 192.168.10.170
 Created: ztnoadmin (First Last) @ 06/22/2024 10:28:12
 Modified: ztnoadmin (First Last) @ 06/22/2024 10:28:36

Select Graph Type: None

Export Clear Filters

Conditions: Active: Active

Filter: Page Size: 25 1 - 4 of 4

| <input type="checkbox"/> | Local Address | Local Port | Foreign Address | Foreign Port | Active | Violation | Actions |
|--------------------------|----------------|------------|-----------------|--------------|--------|----------------------|---------|
| <input type="checkbox"/> | 192.168.10.170 | 22 | 192.168.10.200 | 39292 | ✓ | Disallowed by policy | ⌵ |
| <input type="checkbox"/> | 192.168.10.170 | 22 | 192.168.10.160 | 47842 | ✓ | Disallowed by policy | ⌵ |

Let's review this Connections report to understand what is included in the results.

In general terms, this report displays information about established network connections collected from this asset. This will include our previously established SSH connection along with other connections to this host.

Specific to this guide, bring your attention to two results in this connection list associated with asset Linux Production Server. Both connections are over SSH, port 22, and you should recognize the foreign addresses from each entry as your 12Port Horizon host and your Linux Development Server.

Starting with the SSH connection from the Linux Development Server, let's review the relevant metadata for this walkthrough.

| Timestamp | Asset | Protocol | Local Address | Local Port | Foreign Address | Foreign Port | Process | User | Space | Violation |
|---------------------|-------------------------|----------|----------------|------------|-----------------|--------------|---------|---------------|-------|----------------------|
| 08/22/2024 10:37:11 | Linux Production Server | IPv6 | 192.168.10.170 | 22 | 192.168.10.160 | 47842 | sshd | administrator | root | Disallowed by policy |

| | |
|-----------------|-------------------------|
| Created | 08/22/2024 10:37:11 |
| Updated | 08/22/2024 10:37:11 |
| Active | ✓ |
| Asset | Linux Production Server |
| Protocol | IPv6 |
| Local Address | 192.168.10.170 |
| Local Port | 22 |
| Foreign Address | 192.168.10.160 |
| Foreign Port | 47842 |
| Process | sshd |
| User | administrator |
| Space | root |
| Violation | Disallowed by policy |

- Active: the checkmark indicates this connection was active at the time of collection.
- Asset: displays the name of the asset where the script was run against, Linux Production Server in this guide.
- Local Address: this is the address of our Linux Production Server asset.
- Local Port: is the port number for this connection, 22.
- Foreign Address: displays the address of the host where the connection originates, the Linux Development server in this guide.
- Violation: this column is important for this section. *Disallowed by policy* is displayed for this connection indicating that if (or when) our policy is enforced, this connection would be blocked due to an enforcement segmentation policy.

We can use this Connections report, and more importantly, this Violation column to confirm which connections would be blocked if we moved a policy from state monitored to published with enforcement. A disallowed violation while monitored could be blocked when published.

The other result to look at is the SSH connection from the 12Port Horizon host to this Linux Production Server host. This connection must remain uninterrupted so the application can maintain connectivity as it is used for monitoring, enforcing, and applying policies. We will

resolve this violation in our Enforcement section, but for the purpose of Monitoring, you should remain aware that a violation like this from the 12Port Horizon server will result in failed connections that limit functionality after enforcement.

You may end your SSH session now.

Enforcement

When Monitoring is complete, Violations have been reviewed, and connections that should be detected are being detected, we can move to the next stage of segmentation policies which is to enforce the policy; resulting in these connections being blocked on the Linux Production Server host.

In our scenario, we will need to segment SSH access between our machines as well as maintain SSH access from the 12Port Horizon host. Since 12Port Horizon requires the SSH connection to publish enforcement, let's address this connection first:

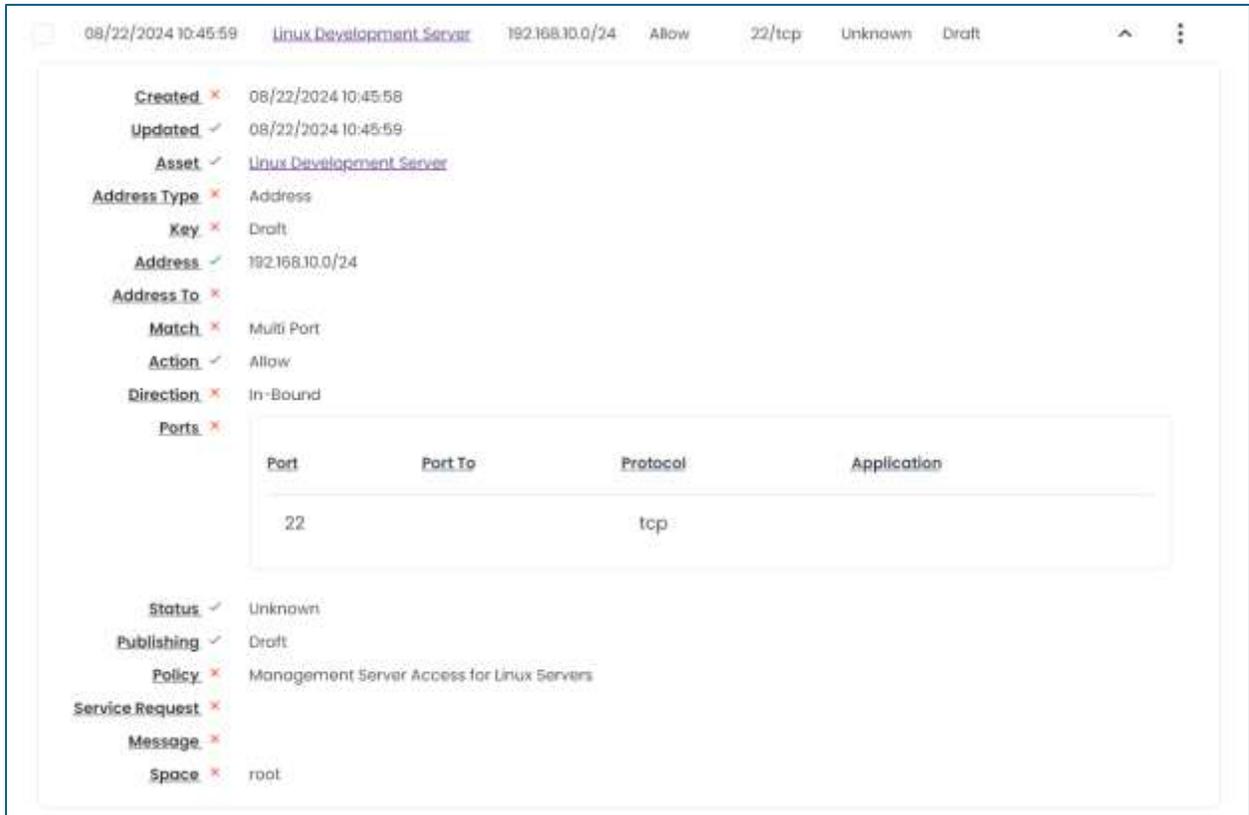
1. Return to the Management > Policies page and locate the policy with Selector **Component :: Server :: Linux**, Service **ssh (22/tcp)**, and Source **Component :: IP List :: Management Server > Exact**.
2. In the *Actions* column of this policy, use the vertical ellipsis menu to open and click the **Publish** button. This will publish the SSH policy to both the Linux Development Server and the Linux Production Server that allows inbound SSH traffic from the 12Port Horizon application server.
3. Locate the policy with **Component :: Server :: Linux**, Service **DNS (53/tcp)**, and Source **Component :: IP List :: Everything > Exact**. As we did with the first policy, click **Publish** in the actions menu.

Filter: Page Size: 20 1 - 4 of 4

| <input type="checkbox"/> | Taxonomy | Selector | Service | Source | Version | Publishing | Actions |
|--------------------------|-------------------|--------------------------------|------------------------|---|---------|------------|--|
| <input type="checkbox"/> | LECA Segmentation | [Component : Server : Linux] | DNS (53/udp) | [Component : IP List : Everything>Exact] | 1.0 | Published | ⌵ ⋮ |
| <input type="checkbox"/> | LECA Segmentation | [Component : Server : Linux] | ssh (22/tcp) | [Component : IP List : Management Server>Exact] | 1.0 | Disabled | ⌵ ⋮ |
| <input type="checkbox"/> | LECA Segmentation | [Component : Server : Windows] | WinRM (5985, 5986/tcp) | [Component : IP List : Management Server>Exact] | 1.0 | Disabl | ✎ Edit |
| <input type="checkbox"/> | LECA Segmentation | [Component : Server : Linux] | ssh (22/tcp) | [Environment>Same] | 0.1 | Monit | 🕒 History 📄 Demote Major 👁 Preview 📺 Monitor 📄 Publish 🗑 Delete |

Total: 4

- Now, navigate to the Reports > Firewall report to confirm policy publishing. For each of the Linux Development Server and the Linux Production Server report entries, take note of the following information:



- Publishing: **Draft** indicates that this rule was generated by a segmentation policy rather than detected in the firewall configuration at the remote endpoint.
- Policy: **Management Server Access for Linux Servers** means that the detected rule is associated with the policy, by its name, which caused it to be published.
- Address: **<IP subnet>** indicates that unrestricted access to the SSH ports for the subnet where the 12Port Horizon application server is located, is open.

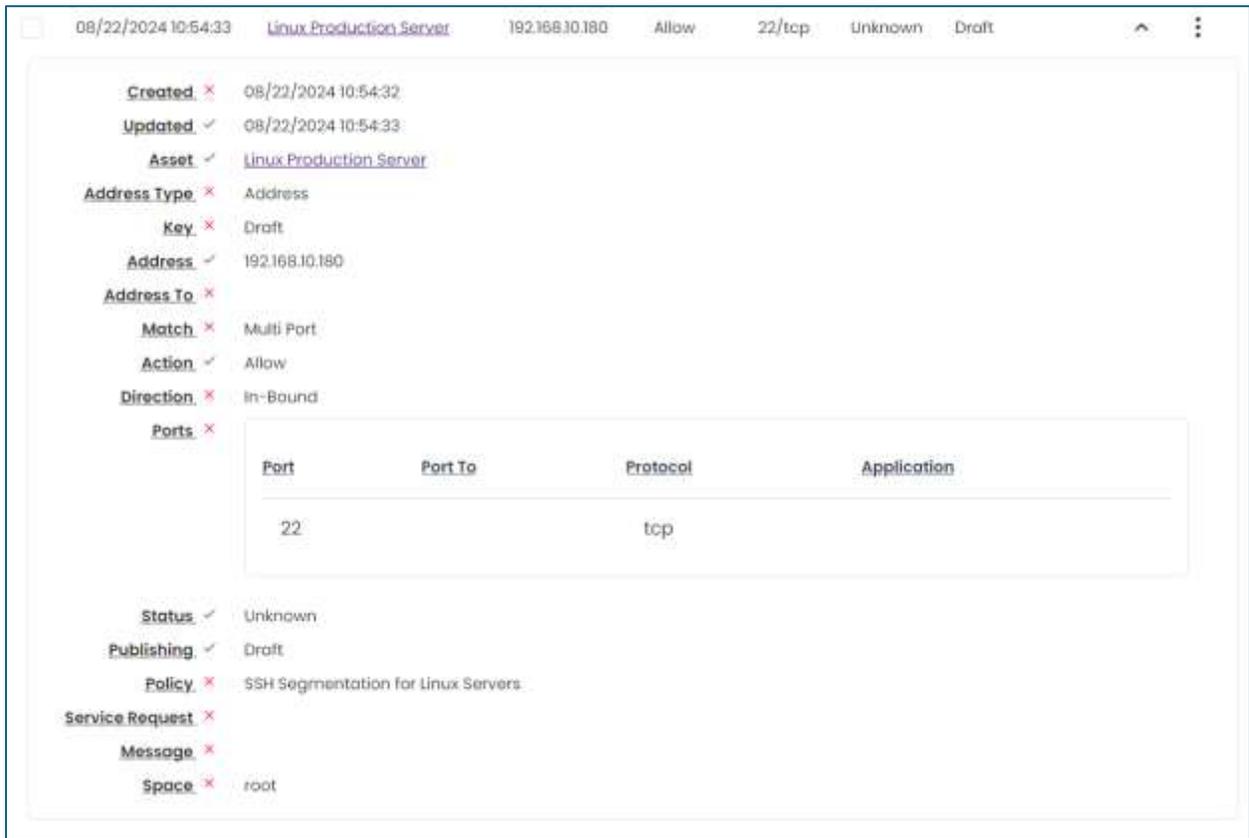
This subnet is derived from the default asset named *Management Server Access* located in Root Container > IP Lists. The asset is created during the tenant creation process using the IP subnet of the 12Port Horizon server. You may edit the asset if required to update this default Address value.

Now we want to repeat this process for our SSH segmentation policy.

- Return to the Management > Policies page and locate the policy we previously created. First, ensure that the policy is a major version and if not use the Actions > **Promote Major** option to make it one. Next, use this policy's Actions menu to click on the **Publish** button to publish this policy to your Linux Production Server. Again,

use the Reports > Firewall to confirm the rule was detected on the host server. Be sure the following is present in the report's entry for the *Linux Production Server* asset:

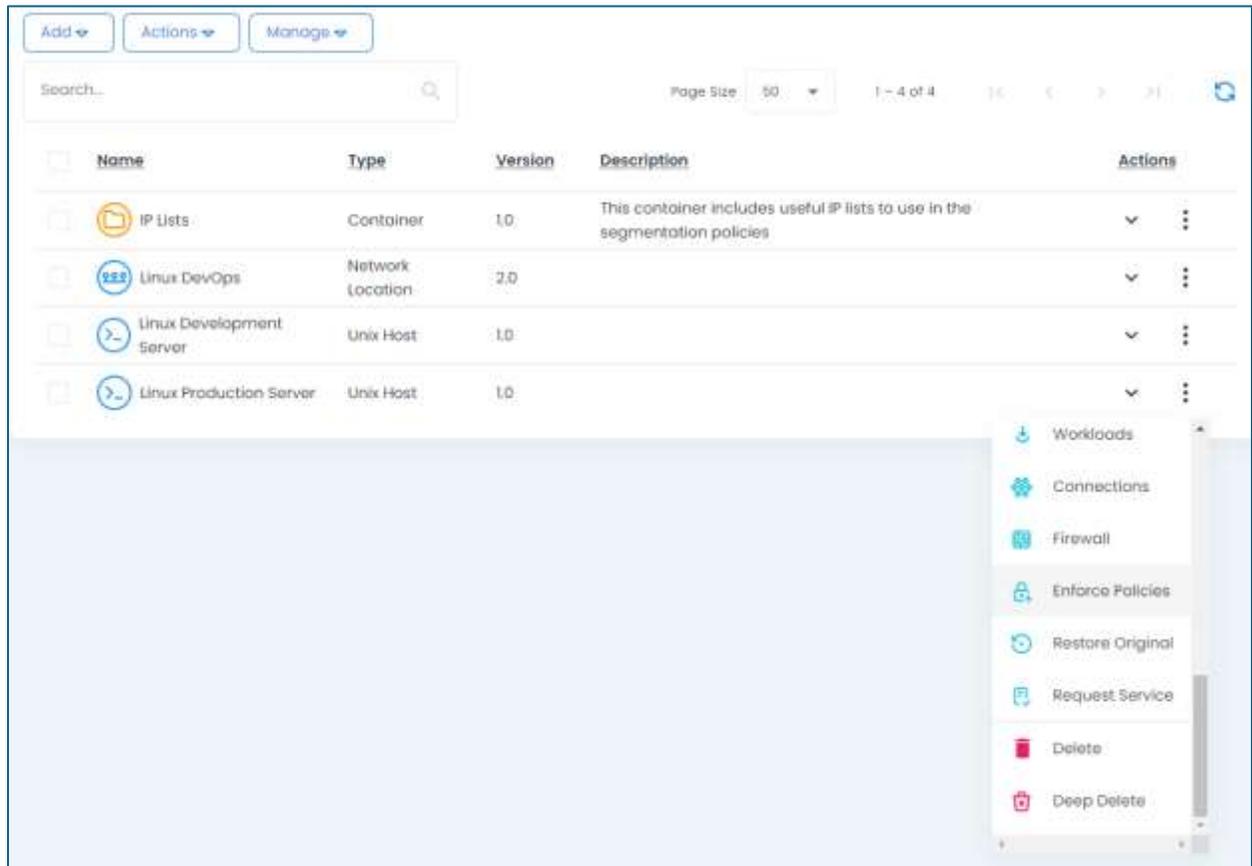
- a. Address: **<IP address>** of your Linux DevOps asset indicating that SSH access from this DevOps server is open using port 22.



For the last step of the Enforcement process, we must now disable the default Linux allow-all firewall rules so that only the connections permitted by our segmentation policies are allowed:

6. Return to the Assets library and locate your *Linux Development Server* asset. From this asset's Actions menu, open and select the option named **Enforce Policies**. Repeat the same Enforce Policies action with your *Linux Production Server* asset.

This action will connect to the host and enable the firewall, if it is not already, and configure a default drop-all behavior.



You can review the Reports > Jobs Report to verify the operation. In this report, filter on the name “Linux Firewall Management” and ensure it was completed for both asset hosts.

You can also review the iptables rules on this host and verify that the default rules are set to DROP, and the 12Port Horizon custom rules are enabled for this traffic.

And as a final test to confirm successful policy enforcement:

1. From the Linux Development Server, attempt to open an SSH connection to the Linux Production Server. Your session should fail to connect.

2. From your Linux DevOps machine, attempt to open an SSH connection to the Linux Production Server. Your SSH session should successfully connect like it had done prior to this guide.

Recap

You made it this far and all testing was performed successfully, and you created, and enforced, your first segmentation policy. If you are still left wondering what we just did, let's provide a quick review of the entire process.

1. We created assets for each of your hosts that are to be segmented. These assets were properly tagged with terms that create logical segments within your network.
2. We performed a 12Port Horizon connectivity test to ensure the software can communicate with each host. This connectivity is crucial to policy monitoring and enforcement.
3. We created a policy that segments traffic between your assets. Development should not connect to Production using SSH, but DevOps should retain SSH connectivity to Production.
4. We enabled monitoring to observe and analyze any policy violations. Ensuring that the traffic was managed properly and to our expectations.
5. We published our policies to the assets by which 12Port Horizon created rules on each to manage specific inbound traffic.
6. We enforced policies on the assets which both enabled the firewall and changed the default rules to drop all. Leaving in its place only the enabled 12Port Horizon rules that manage the SSH traffic to each host individually.
7. We confirmed throughout the process the results of each section and in the end, verified that SSH connectivity between the hosts was enforced as per our policy enforcement.

Congratulations on completing the quick start guide! You're now ready to dive further into the software with confidence. For further assistance, guides, how-to's, and documentation, refer to our online help portal here: <https://docs.12port.com/horizon/>

Appendix

Production Deployment Guide

This is a supplement to the Quick Start Guide, and we recommend that you become familiar with the basics of using the 12Port Horizon software described in the Quick Start Guide before continuing. This supplemental guide offers suggestions for advanced configuration to help you prepare your Horizon deployment for use in a production context. As the needs of each deployment are unique, this guide will focus primarily on best practices applicable to any system administrator getting started with the software. We hope that after following this guide, you will be equipped to tailor your 12Port Horizon configuration to the needs of your network and set up a production-ready deployment.

Intelligent Tagging

The Intelligent Tagging system allows you to create tagging rules which will automatically infer and apply tags to your assets as you add them to your asset database. While the optimal complete set of tagging rules depends largely on the details of your network, there are a few simple rules that are universally recommended. Recall from the Quick Start Guide that there are three out-of-the-box segmentation policies designed to keep your devices open for remote management by the 12Port Horizon host machine. These policies require that your assets are tagged with either *Component::Server::Windows* or *Component::Server::Linux*. Since each new asset will need one of these tags, it will be helpful at the outset to create some intelligent tagging rules to automate this process.

To add an intelligent tagging rule, navigate to Management > Tagging and click “Add.” Fill out the form to create the intelligent tagging rule. An example for Windows servers is shown below:

Add Tagging Rule
 Home > Tagging Rules > Add Tagging Rule

Name Windows Servers

Conditions Predicate And

Condition

Criteria Predicate And

| Type | Match | Query | Actions |
|------------|--------|--------------|------------------------------------|
| Asset Type | Equals | Windows Host | <input type="button" value="Add"/> |

Taxonomy LECA Segmentation

Tags Type to start selection...

Component :: Server :: Windows X

Enabled

This intelligent tagging rule will apply the *Component::Server::Windows* tag to any asset with the asset type “Windows Host.” A similar tagging rule could be created for Linux machines by changing the condition query to “Unix Host” and changing the applied tag to *Component::Server::Linux*. We recommend that you create one or both tagging rules depending on the asset types you expect to add to your asset database.

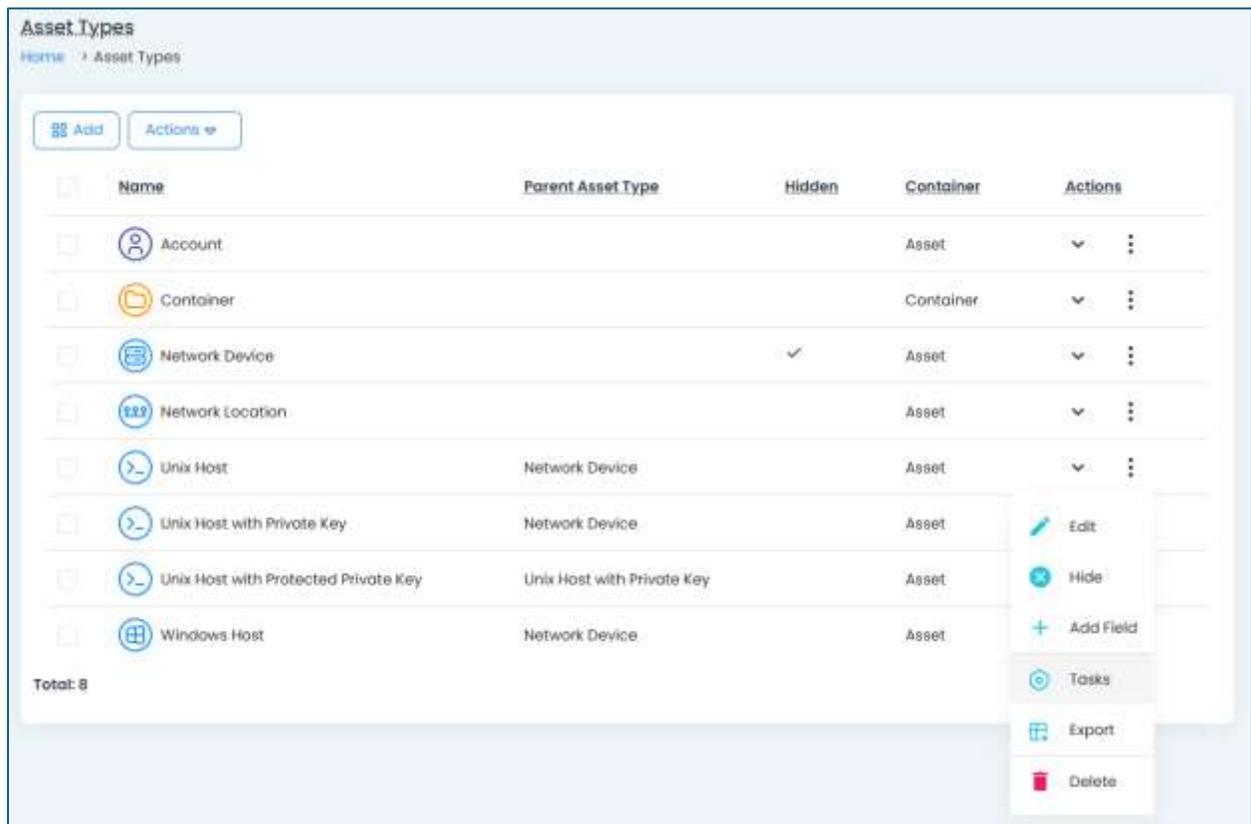
Task Configuration

In 12Port Horizon, all device management tasks are performed through remote execution of scripts. These scripts are assigned to assets by the Task construct which defines the trigger for execution of a script on a particular asset. By default, all tasks on each asset type have their execution trigger set to “Interactive.” This means that a script is only executed by manually running the task on a specific asset in the GUI. You may have already interacted

with scripts on this basis by running a *Status Check* task on a new asset to verify connectivity or by running a *Connections Monitoring* task to collect connection data from the endpoint. To configure your Horizon tenant to be more scalable and self-sustaining, many of these tasks should instead be automated. This section will outline the procedure to change your task configuration and offer recommendations as to the specific tasks you should modify.

Task Configuration Procedure

To view your global task configuration for an asset type, navigate to Management > Asset Types, open the Actions menu for the asset type, and select “Tasks.”



You will see a list of all tasks belonging to the asset type. To edit a task, open the actions menu for the task and choose “Edit.”

Edit Task
Home > Asset Types > Asset Type > Tasks > Edit Task

Asset Type : Unix Host

Script: Linux Monitoring

Trigger: Interactive

Delay (mins): 0

Run As: Main

The main setting you will likely need to modify during your initial configuration is “Trigger.” From the default of “Interactive,” it can be changed to execute the script upon creation, updating, or viewing of an asset. It can also be set to “Schedule” which will open a schedule builder below where you can specify how frequently you would like the script to run on your assets.

The screenshot shows the 'Schedule Builder' interface with the following configuration:

- Tab: Hours
- Option: Every Hour
- Option: Specific hours (choose one or many)
 - Hours 0-23: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
- Option: Range of hours
 - Every hour between hour [] and hour []
- Option: Hours Increment
 - Every [] hours started with hour []
- Option: Random Hour
- Horizontal scrollbar: []
- Field: Schedule Expression: 0 0 * * * ?
- Buttons: Close, Select

If you would like to create a unique task configuration for specific assets, this can be done by navigating to an asset, selecting “Tasks” from the Manage menu of the asset, and clicking “Make Unique.” You will then be able to edit the task configuration for that asset independently from the default configuration for the asset type. The task configuration for an asset can be returned to the default configuration by clicking “Make Inherit” on this same screen.

Task Configuration Recommendations

As your Horizon deployment scales, it will be helpful to have certain tasks run on a schedule or some other non-interactive trigger. Our suggestions for configuration of each task are listed below:

- IPTables Firewall Logs Enable / Windows Firewall Logs Enable

These scripts enable logging of connections on your assets so that the monitoring scripts can record connections that may not be active at the time the assets are polled. If you would like to enable this behavior globally without having to run this script manually on each new asset, you can set the trigger for this task to “After

Create.”

- Linux Monitoring / Windows Monitoring

This is a batch task which runs several minor monitoring scripts that will collect data about system status, network interfaces, running services, active connections, and firewall rules. To effectively monitor your network from Horizon, this data should be automated to update collected data frequently. We recommend setting the trigger to “Schedule” and choosing an appropriate recurrence like every hour.

- IPTables List / Windows List Firewall Rules

These scripts check the firewall rules of your assets and compare them to the rules maintained by the Horizon server. It will also regenerate any rules that it detects to be missing. To help maintain the integrity of your segmentation policies after enforcing them, this task should also be scheduled. We recommend a frequency between once per hour to once per day, depending on the level of security you wish to maintain on your assets.