EVERTRUST

Horizon

Version 2.12, 2026-11-02

Table of Contents

1. Installation . . . 1
   1.1. Installing on Linux . . . 1
   1.2. Installing on Kubernetes . . . 52
   1.3. Installing on Openshift . . . 98
   1.4. Running with Docker/Compose . . . 102
   1.5. FIPS mode . . . 108
   1.6. Analytics . . . 110
   1.7. Tinkey . . . 112
   1.8. Monitoring . . . 125
   1.9. Troubleshooting . . . 127
   1.10. Logging . . . 131
   1.11. Multi-Tenancy . . . 137
   1.12. Endpoint configuration . . . 140
   1.13. Advanced configuration . . . 150
   1.14. Upgrade . . . 185
   1.15. Uninstallation . . . 198
2. Admin guide . . . 201
   2.1. User Information . . . 201
   2.2. Certification Authorities . . . 202
   2.3. PKIs . . . 206
   2.4. DCV . . . 248
   2.5. Security . . . 266
   2.6. Notifications . . . 285
   2.7. Discovery . . . 291
   2.8. Automation . . . 293
   2.9. Monitored profiles . . . 296
   2.10. Protocols . . . 303
   2.11. Datasources . . . 424
   2.12. Third parties . . . 427
   2.13. MDM . . . 467
   2.14. System configuration . . . 510
   2.15. Common configuration elements . . . 520
   2.16. Reports . . . 551
   2.17. Archives . . . 553
   2.18. Page Moved . . . 555
   2.19. Logging . . . 555
   2.20. Event Codes . . . 555
3. User guide . . . 569
   3.1. Managing requests on the WebRA . . . 569

3.2. Requesting a SCEP challenge . . . 576
3.3. Requesting an EST challenge . . . 577
3.4. Requesting a WebRA challenge . . . 579
3.5. Managing requests (operator) . . . 580
3.6. Searching requests and certificates . . . 581
3.7. Sharing custom dashboards . . . 587
4. Knowledge base . . . 589
4.1. Configure tunnels . . . 589

Chapter 1. Installation

Description

Horizon is EverTrust Certificate lifecycle management solution. This document is an installation procedure detailing how to install and bootstrap Horizon server on your infrastructure. It does not describe how to configure and operate a Horizon instance. Please refer to the administration guide for administration related tasks.

Prerequisites

Choose an installation method

We offer three installation modes:

- Enterprise Linux 8.x/9.x/10.x x64
- Debian based Linux x64
- A cloud-native installation using Kubernetes

Depending on your needs, you'll have to choose the solution that fits your use cases the best. Reach out to our support team to get suggestions on how to deploy on your infrastructure.

Gathering your credentials

All methods require that you download the binaries of the Horizon software from our software repository. The access to this repository is protected by username and password, which you should have got from our tech team. If you don't, you won't be able to continue with the installation. Email us to get your credentials, and come back to this step.

1.1. Installing on Linux

1.1.1. Pre-requisites

This section describes the system and software pre-requisites to install Horizon. Please select your OS type in the available tabs.

System pre-requisites

The following elements are considered as system pre-requisites:

RHEL

- A server running EL [8.x-10.x] x64 (CentOS / RHEL) with the network configured and **SELinux** as well as **FIPS mode** disabled;
- Base and EPEL CentOS / RHEL [8.x-10.x] x64 repositories activated;

- An access with administrative privileges (root) to the server mentioned above;
- The IP address / DNS Name of an SMTP relay;
- The email address of the Horizon server administrator.

Debian

- A server running Debian 11/12 or Ubuntu 22/24 x64 with the network configured and **AppArmor** as well as **FIPS mode** disabled;
- Standard Debian/Ubuntu repositories activated;
- An access with administrative privileges (root) to the server mentioned above;
- The IP address / DNS Name of an SMTP relay;
- The email address of the Horizon server administrator.

Software pre-requisites

The following elements are considered as software pre-requisites:

RHEL

- The Horizon installation package: `horizon-2.12.X-1.noarch.rpm`;
- Java 17 or 21;
- The MongoDB Community Edition package available from the MongoDB web site;
- EPEL repository activated.



As a reminder, EPEL can be activated on CentOS / RHEL by doing the following:

```
$ yum install epel-release
```

Debian

- The Horizon installation package: `horizon_2.12.X_all.deb`;
- Java 17 or 21;
- The MongoDB Community Edition package available from the MongoDB web site;

1.1.2. Installation

Install MongoDB



Mongo DB version 7.0 to 8.0 are supported by Horizon

RHEL

Download the latest version of the following Mongo DB 8.x RPMs from the MongoDB web site:

- `mongodb-org`
- `mongodb-org-mongos`
- `mongodb-org-server`
- `mongodb-org-shell`
- `mongodb-org-tools`

Download the last version of the `mongosh` RPM from the MongoDB GitHub.

- `mongodb-mongosh`

Upload the downloaded RPMs through SCP on the server under `/root`.

Using an account with privileges, install the RPMs using 'yum'. For example, to install MongoDB, run the following command from the folder containing the RPMs:

```
$ yum install mongodb-org*
$ yum install mongodb-mongosh
```

Debian

Download and install the latest version of the following Mongo DB 8.x DEB using the steps of the Official Mongo DB documentation:

- `mongodb-org`

Download the last version of the `mongosh` DEB from the MongoDB GitHub.

- `mongodb-mongosh`

Upload the downloaded DEB through SCP on the server under `/root`.

Using an account with privileges, install the DEBs using 'apt'. For example, to install MongoDB, run the following command from the folder containing the DEBs:

```
$ apt install mongodb-mongosh
```

Enable the service at startup with the following command:

```
$ systemctl enable mongod
```

Start the `mongod` service with the following command:

```
$ systemctl start mongod
```

Verify that you can connect to the Mongo instance by running the mongo shell:

```
$ mongosh
```



You can disconnect from the shell with ^D

Install NGINX

1. Access the server through SSH with an account with administrative privileges;
2. Install the NGINX web server using the following command:

RHEL

```
$ yum install nginx
```

Debian

```
$ apt install nginx
```

3. Enable NGINX to start at boot using the following command:

```
$ systemctl enable nginx
```

4. Stop the NGINX service with the following command:

```
$ systemctl stop nginx
```

Install Horizon

RHEL



Installing the Horizon package will install the following dependencies:

- `dialog`

Please note that these packages may have their own dependencies.

Debian



Installing the Horizon package will install the following dependencies:

- `dialog`
- `zip`
- `unzip`

Please note that these packages may have their own dependencies.

Installation from the EverTrust repository

RHEL

Create a `/etc/yum.repos.d/horizon.repo` file containing the EverTrust repository info:

```
[horizon]
enabled=1
name=Horizon Repository
baseurl=https://repo.evertrust.io/repository/horizon-rpm/
gpgcheck=1
gpgkey=https://evertrust.io/.well-known/rpm/gpg.pub
username=<username>
password=<password>
```

Replace `<username>` and `<password>` with the credentials you were provided.

Make sure the Evertrust GPG key is trusted:

```
# rpm --import https://evertrust.io/.well-known/rpm/gpg.pub
```

You can then run the following to install the latest Horizon version:

```
# yum install horizon
```

To prevent unattended upgrades when running yum update, you should pin the Horizon version by adding

```
exclude=horizon
```

at the end of the `/etc/yum.repos.d/horizon.repo` file after installing Horizon.

Debian

If you haven't already, to add the EVERTRUST repository to your APT repositories, run the following commands:

1. Install the required tools (`gpg`)

```
# sudo apt install gnupg
```

2. Download and install the EVERTRUST GPG key

```
# curl https://evertrust.io/.well-known/apt/gpg.pub | sudo gpg -o
/usr/share/keyrings/evertrust.gpg --dearmor
```

3. Add the repository

```
# echo "deb [ arch=all,amd64 signed-by=/usr/share/keyrings/evertrust.gpg ]
https://repo.evertrust.io/repository/apt all main" | sudo tee
/etc/apt/sources.list.d/evertrust.list
```

Once the repository has been added, authentication to it must be provided. To do so, edit the `/etc/apt/auth.conf` file and add the following lines:

```
machine repo.evertrust.io
login <your EVERTRUST login>
password <your EVERTRUST password>
```

Once the repository has been added, run the following command to update the APT repository list.

```
# sudo apt update
```

You can then run the following command to install the latest Horizon version:

```
# sudo apt install horizon
```

To prevent unattended upgrades when running `apt upgrade`, you should pin the Horizon version by creating a `/etc/apt/preferences.d/horizon` file:

```
Package: horizon
Pin: version <installed-version>
Pin-Priority: 1001
```

After installing, services must be reloaded to take the change into account

```
$ systemctl daemon-reload
```

Installing from the package file

RHEL

Download the latest RPM for Horizon on the Official EVERTRUST repository.

Upload the file `horizon-<latest>.noarch.rpm` to the server;

Access the server with an account with administrative privileges;

Install the Horizon package with the following command:

```
# yum localinstall /root/horizon-<latest>.noarch.rpm
```

If you wish to verify the signature of the RPM package, the EVERTRUST key can be added to your trusted keys using the following command:

```
# rpm --import https://evertrust.io/.well-known/rpm/gpg.pub
```

The signature can then be verified using the following command:

```
# rpm -K /root/horizon-<latest>.noarch.rpm
```

Debian

Download the latest DEB for Horizon on the Official EVERTRUST repository.

Upload the file `horizon_<latest>_all.deb` to the server;

Access the server with an account with administrative privileges;

Install the Horizon package with the following command:

```
# apt install /root/horizon_<latest>_all.deb
```

Enabling the service

After installing, services must be reloaded to take the change into account

```
$ systemctl daemon-reload
```

Enable Horizon to start at boot using the following command

```
$ systemctl enable horizon
```

Installing Tinkey

RHEL

Tinkey requires java 17 or 21 installed

Debian

Tinkey requires java 17 or 21 installed

If your environment is subject to FIPS constraints, use the FIPS flavor of Tinkey:

- RPM / DEB package: `tinkey-fips`

Installation from the EverTrust repository

RHEL

Create a `/etc/yum.repos.d/tinkey.repo` file containing the EverTrust repository info:

```
[tinkey]
enabled=1
name=Tinkey Repository
baseurl=https://repo.evertrust.io/repository/tinkey-rpm/
gpgcheck=1
gpgkey=https://evertrust.io/.well-known/rpm/gpg.pub
username=<username>
password=<password>
```

Replace `<username>` and `<password>` with the credentials you were provided.

Make sure the Evertrust GPG key is trusted:

```
# rpm --import https://evertrust.io/.well-known/rpm/gpg.pub
```

You can then run the following to install the latest Tinkey version:

```
# yum install tinkey
```

To prevent unattended upgrades when running yum update, you should pin the Tinkey version by adding

```
exclude=tinkey
```

at the end of the `/etc/yum.repos.d/tinkey.repo` file after installing Tinkey.

Debian

If you haven't already, to add the EVERTRUST repository to your APT repositories, run the following commands:

1. Install the required tools (`gpg`)

```
# sudo apt install gnupg
```

2. Download and install the EVERTRUST GPG key

```
# curl https://evertrust.io/.well-known/apt/gpg.pub | sudo gpg -o
/usr/share/keyrings/evertrust.gpg --dearmor
```

3. Add the repository

```
# echo "deb [ arch=all,amd64 signed-by=/usr/share/keyrings/evertrust.gpg ]
https://repo.evertrust.io/repository/apt all main" | sudo tee
/etc/apt/sources.list.d/evertrust.list
```

Once the repository has been added, authentication to it must be provided. To do so, edit the `/etc/apt/auth.conf` file and add the following lines:

```
machine repo.evertrust.io
login <your EVERTRUST login>
password <your EVERTRUST password>
```

Once the repository has been added, run the following command to update the APT repository list.

```
# sudo apt update
```

You can then run the following command to install the latest Tinkey version:

```
# sudo apt install tinkey
```

To prevent unattended upgrades when running `apt upgrade`, you should pin the Tinkey version by creating a `/etc/apt/preferences.d/tinkey` file:

```
Package: tinkey
Pin: version <installed-version>
Pin-Priority: 1001
```

Installing from the package file

RHEL

Download the latest RPM for Tinkey on the Official EVERTRUST repository.

Upload the file `tinkey-<latest>.noarch.rpm` to the server;

Access the server with an account with administrative privileges;

Install the Tinkey package with the following command:

```
# yum localinstall /root/tinkey-<latest>.noarch.rpm
```

If you wish to verify the signature of the RPM package, the EVERTRUST key can be added to your trusted keys using the following command:

```
# rpm --import https://evertrust.io/.well-known/rpm/gpg.pub
```

The signature can then be verified using the following command:

```
# rpm -K /root/tinkey-<latest>.noarch.rpm
```

Debian

Download the latest DEB for Tinkey on the Official EVERTRUST repository.

Upload the file `tinkey_<latest>_all.deb` to the server;

Access the server with an account with administrative privileges;

Install the Tinkey package with the following command:

```
# apt install /root/tinkey_<latest>_all.deb
```

Configure the Firewall

RHEL

Access the server through SSH with an account with administrative privileges;

Open port TCP/443 on the local firewall with the following command:

```
$ firewall-cmd --permanent --add-service=https
```

Reload the firewall configuration with:

```
$ systemctl restart firewalld
```

Debian

Make sure to allow port 443 if you have any firewall set up.

1.1.3. Configuration

Initial Configuration

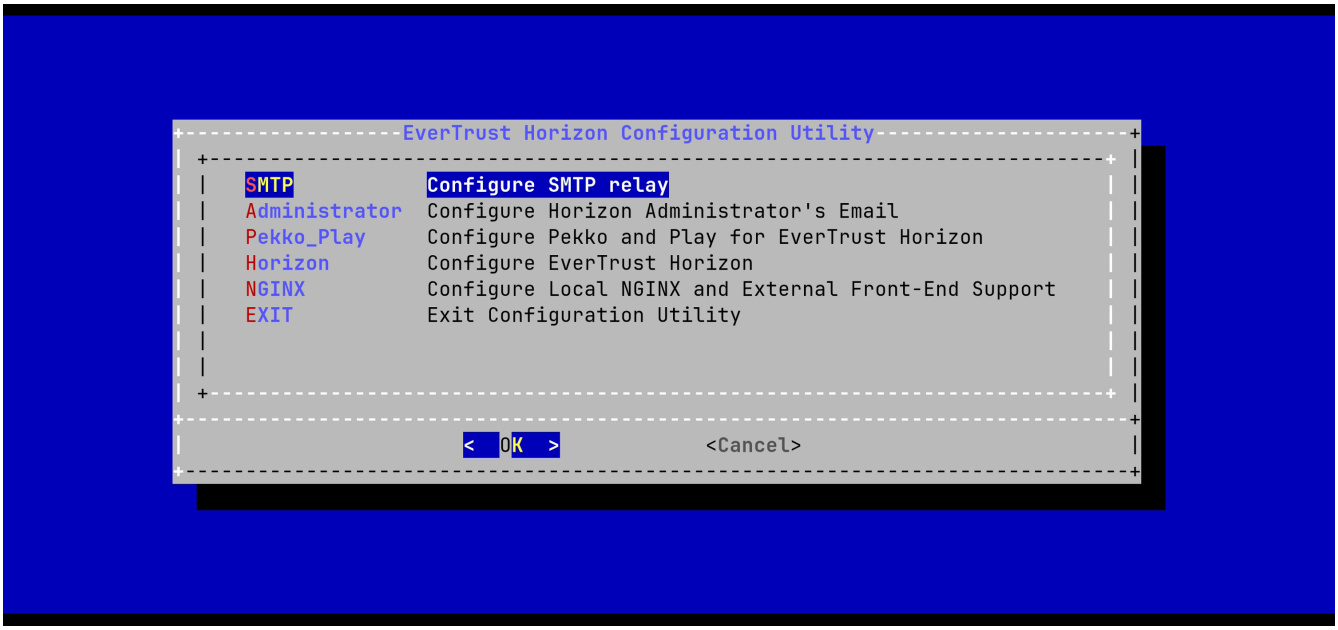
Configuring the SMTP Relay

Connect to the server with an account with administrative privileges;

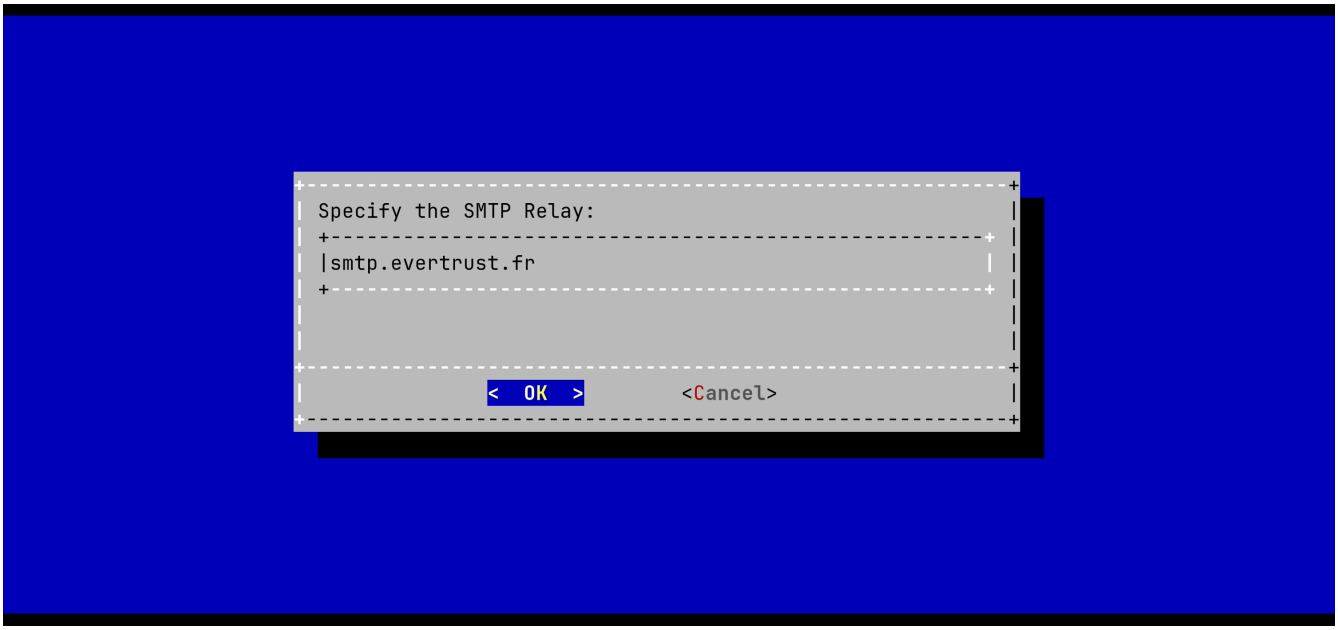
Start the Horizon configuration utility by running:

```
# /opt/horizon/sbin/horizon-config
```

In the main menu, select '**SMTP**':



Specify IP address or the DNS name of the SMTP relay and validate:


Specify the SMTP Relay:
smtp.evertrust.fr
< OK >
<Cancel>

The Postfix configuration is updated:



Exit the configuration utility and restart the Postfix service with the following command:

```
$ systemctl restart postfix
```

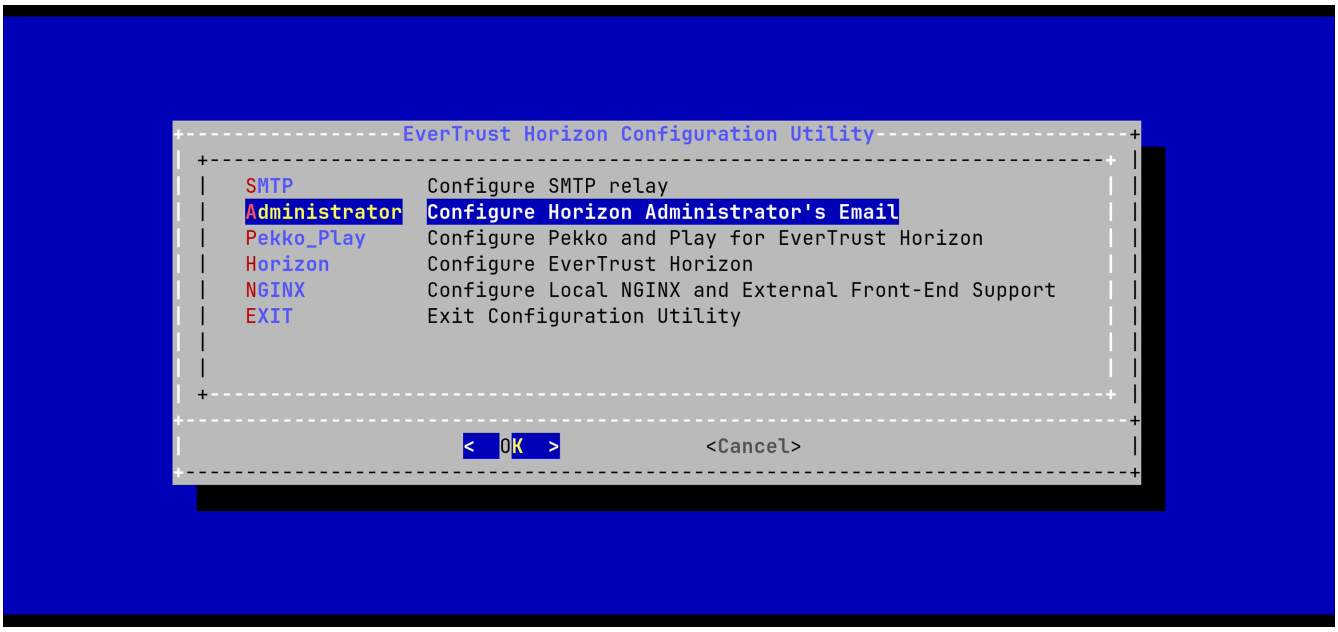
Configuring the Horizon Administrator's Email Address

Connect to the server with an account with administrative privileges;

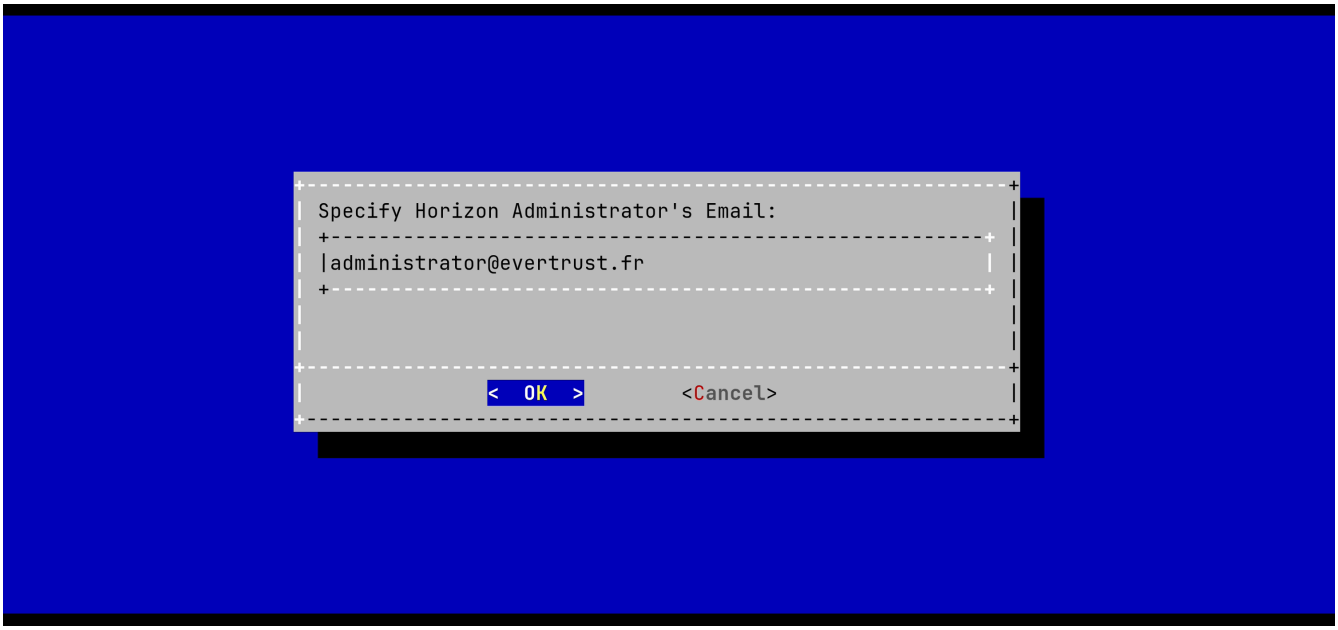
Start the Horizon configuration utility by running:

```
# /opt/horizon/sbin/horizon-config
```

In the main menu, select **Administrator**:



Specify the email address of the Horizon Administrator and validate:



Exit the Configuration Utility;

Validate the SMTP relay and Administrator Email Address with the following commands:

```
$ yum install mailx
$ mail -s "Hello Horizon root"
  > Hello From Horizon
  .
```

Ensure that the email receives the test email.

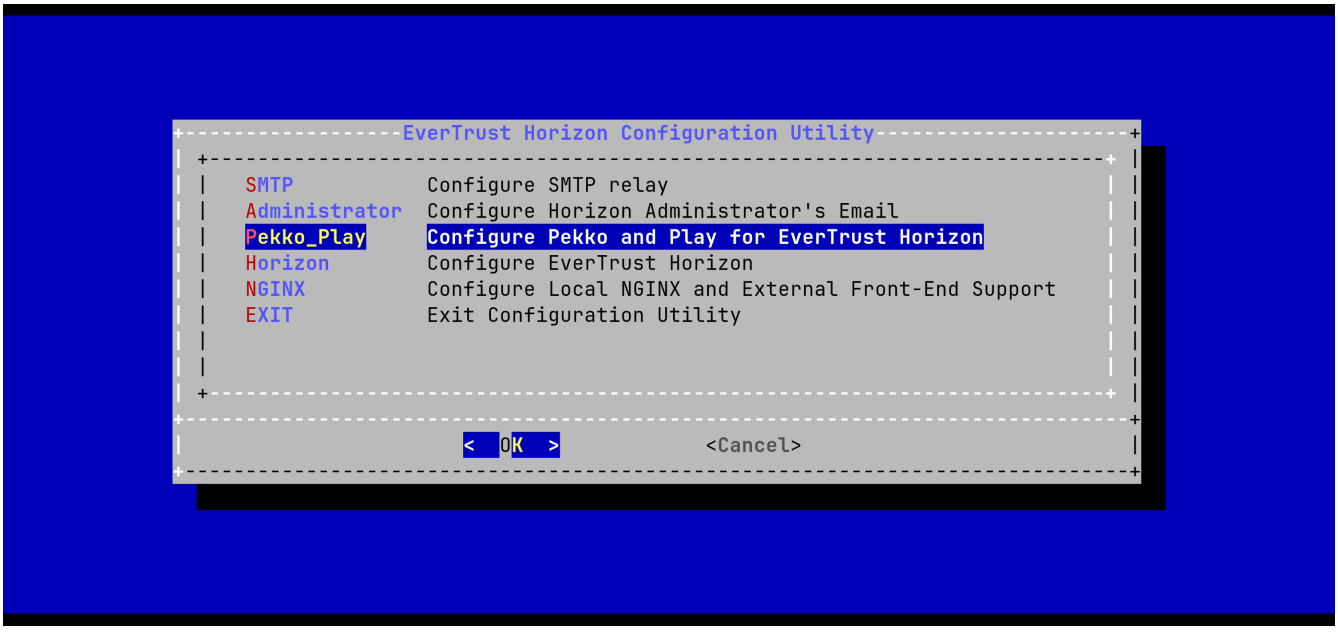
Generating a new Horizon Application Secret

Connect to the server with an account with administrative privileges;

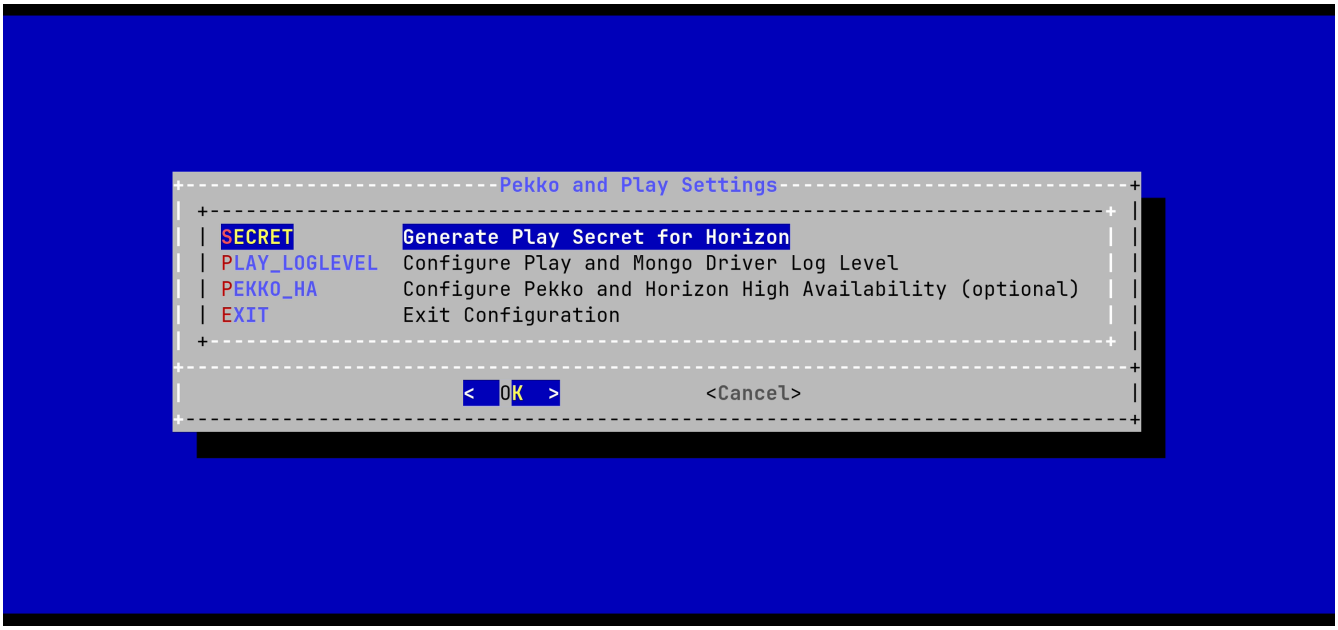
Start the Horizon configuration utility by running:

```
# /opt/horizon/sbin/horizon-config
```

In the main menu, select '**Pekko_Play**':



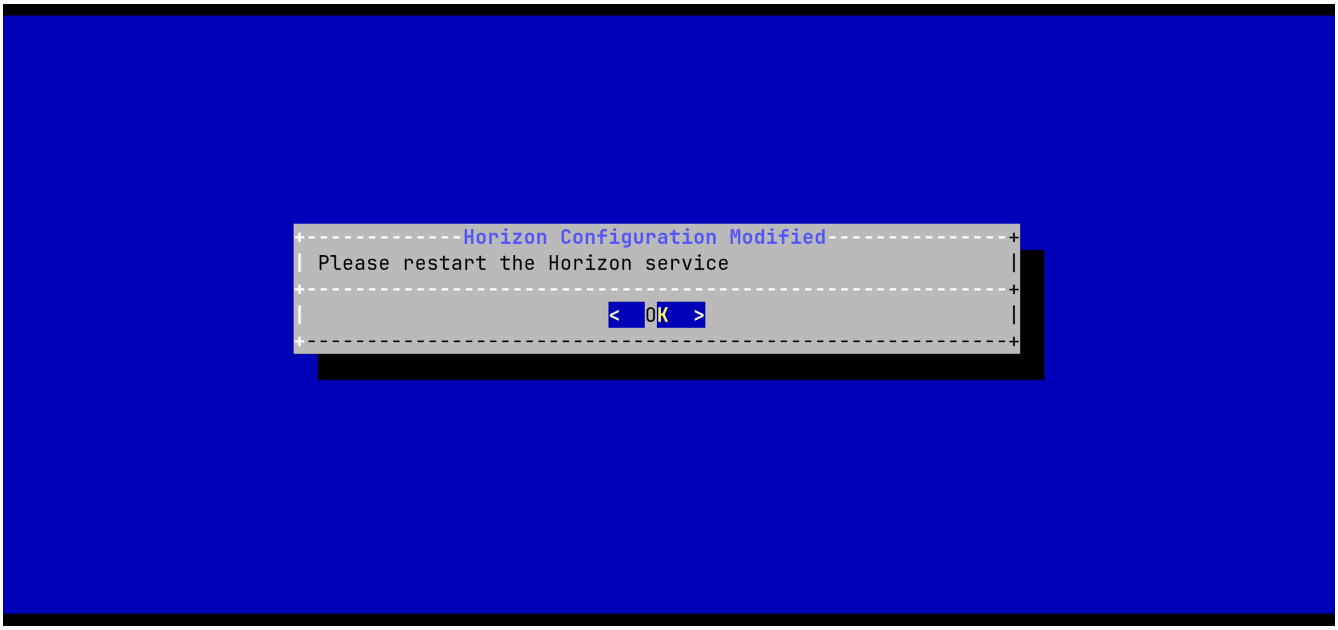
In the Pekko_Play menu, select '**SECRET**':



Validate the new Horizon Application Secret:



The Horizon configuration is updated:



For the changes to take effect, you must restart the Horizon service by running:

```
# systemctl restart horizon
```

JVM Configuration

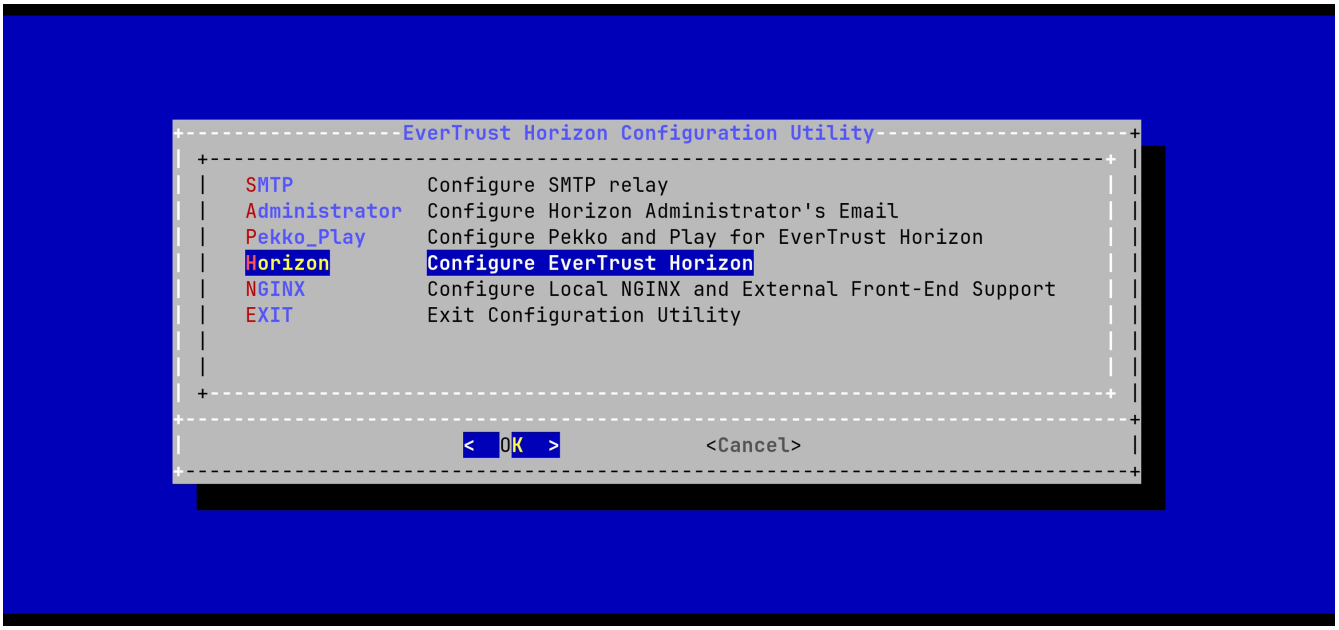
Horizon allows you to configure the ***xms*** (minimum memory allocation pool) and ***xmx*** (maximum memory allocation pool) parameters of the JVM running Horizon using the configuration tool.

Connect to the server with an account with administrative privileges;

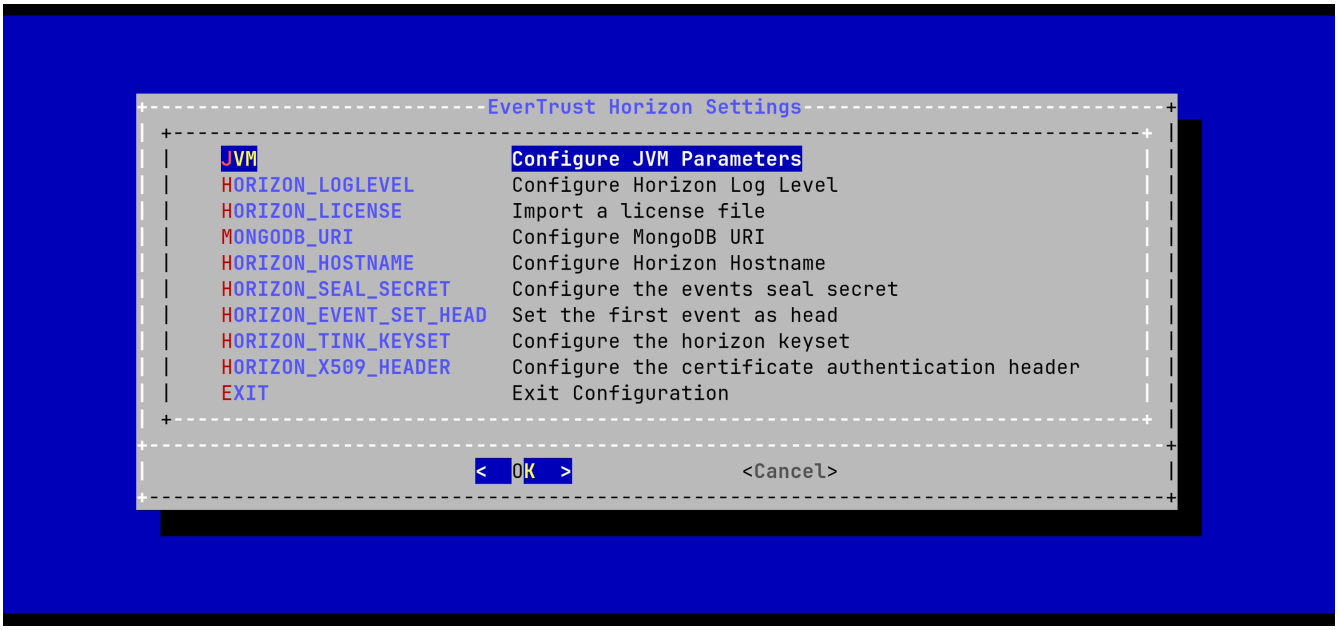
Start the Horizon configuration utility by running:

```
# /opt/horizon/sbin/horizon-config
```

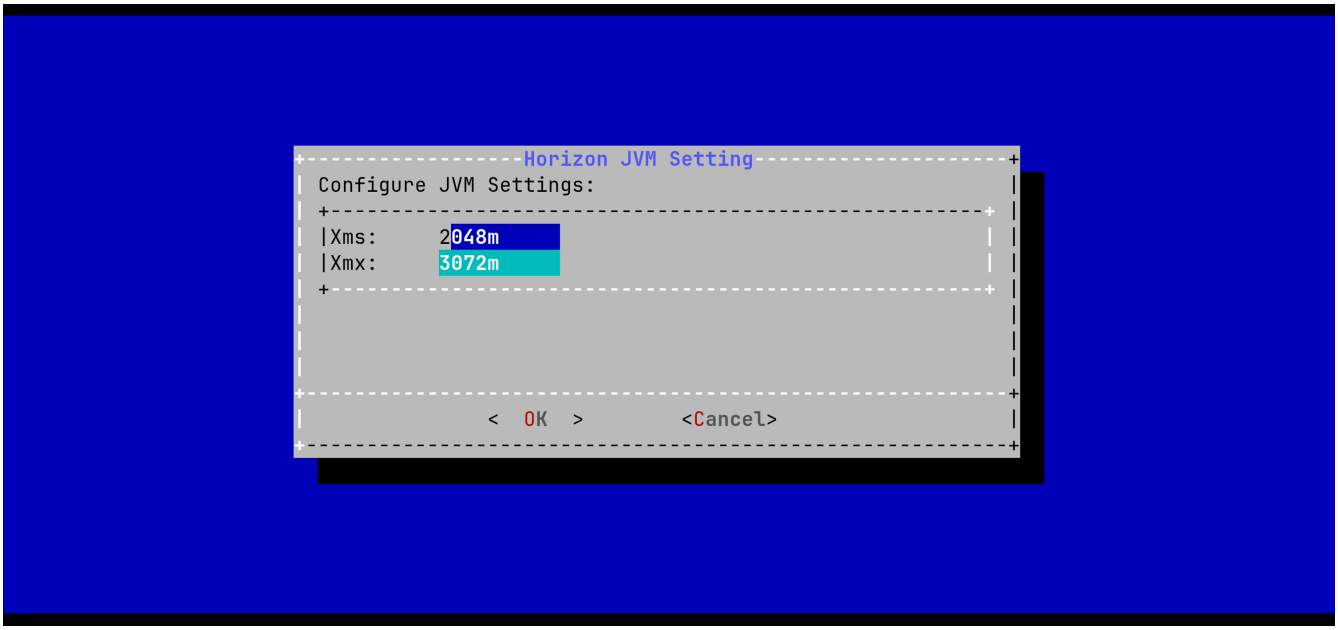
In the configuration menu, select '**Horizon**':



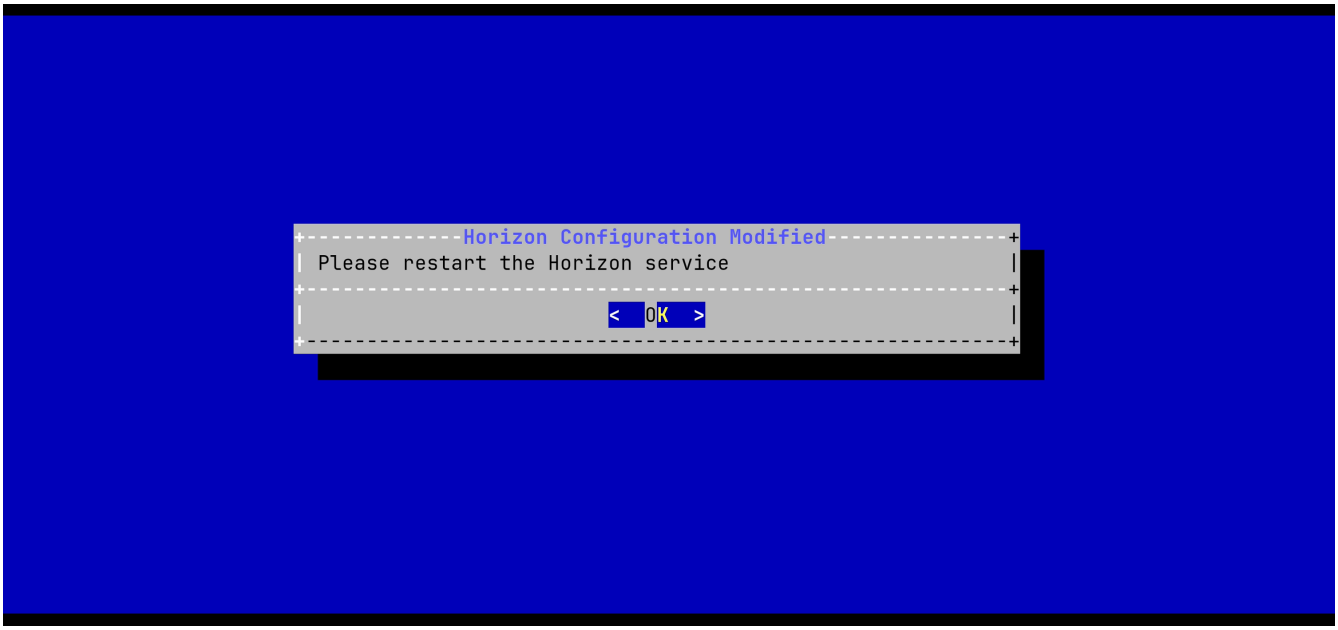
In the Horizon configuration menu, Select '**JVM**':



Specify the 2048 for ***xms*** and 3072 for ***xmx*** parameters and select '**OK**':



The new JVM parameters are configured:



For the changes to take effect, you must restart the Horizon service by running:

```
# systemctl restart horizon
```

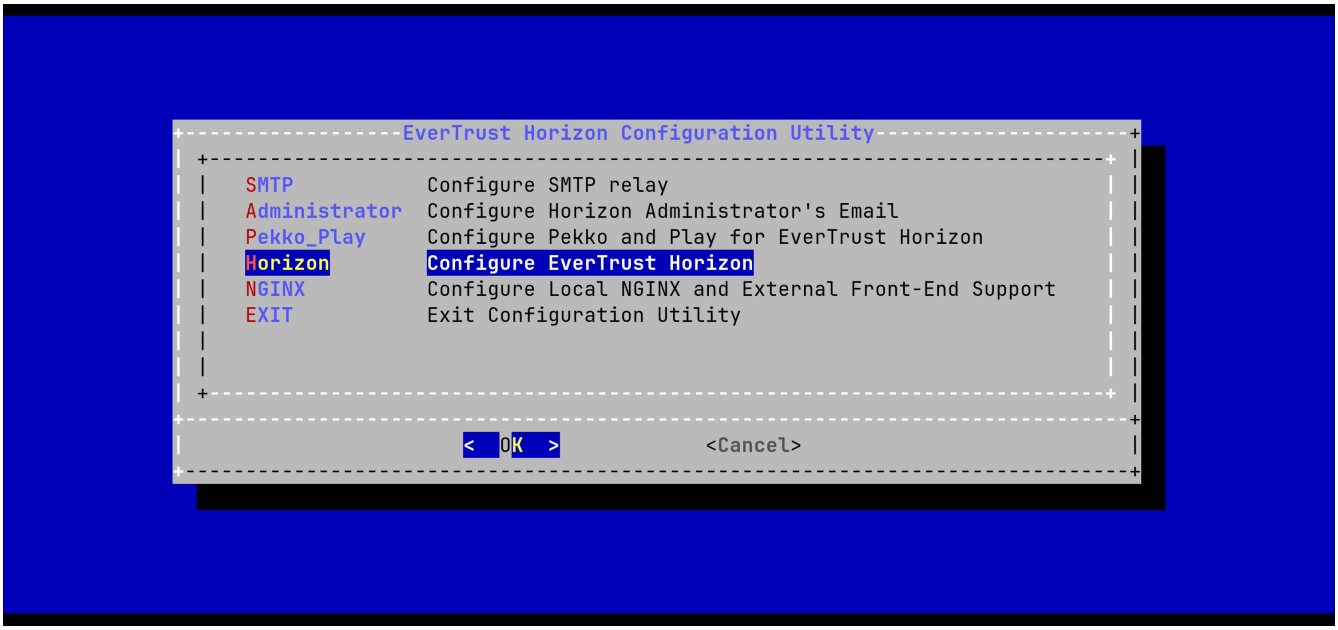
MongoDB URI Configuration

Connect to the server with an account with administrative privileges;

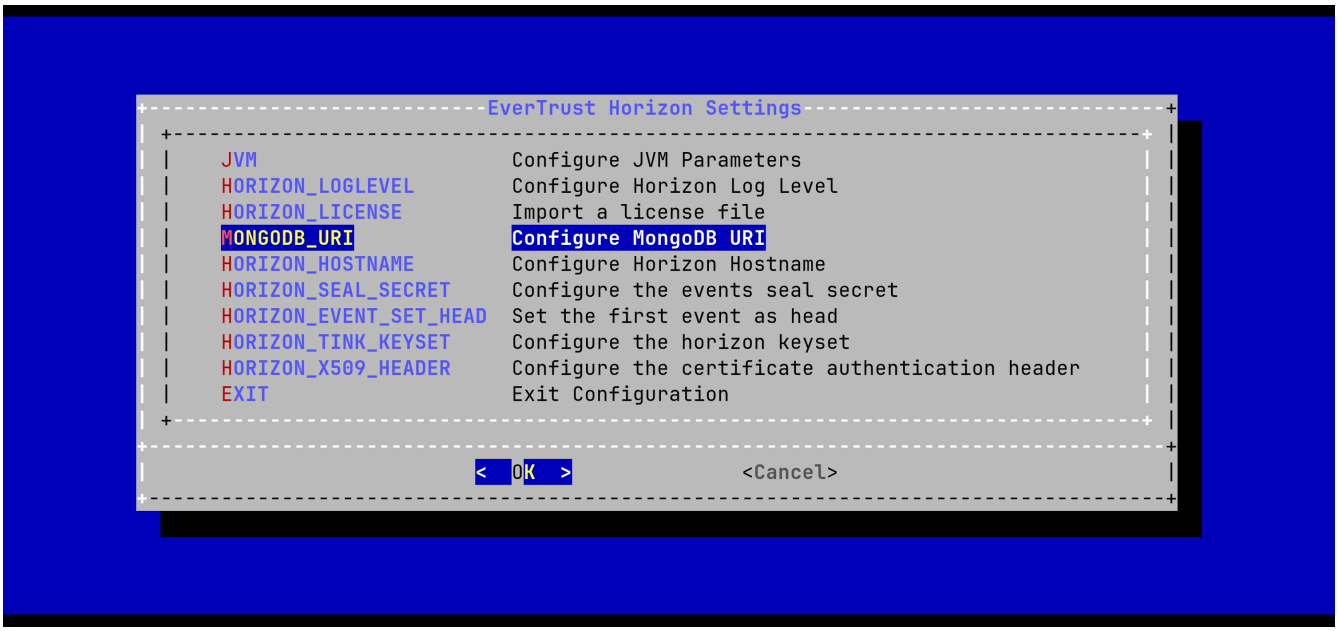
Start the Horizon configuration utility by running:

```
# /opt/horizon/sbin/horizon-config
```

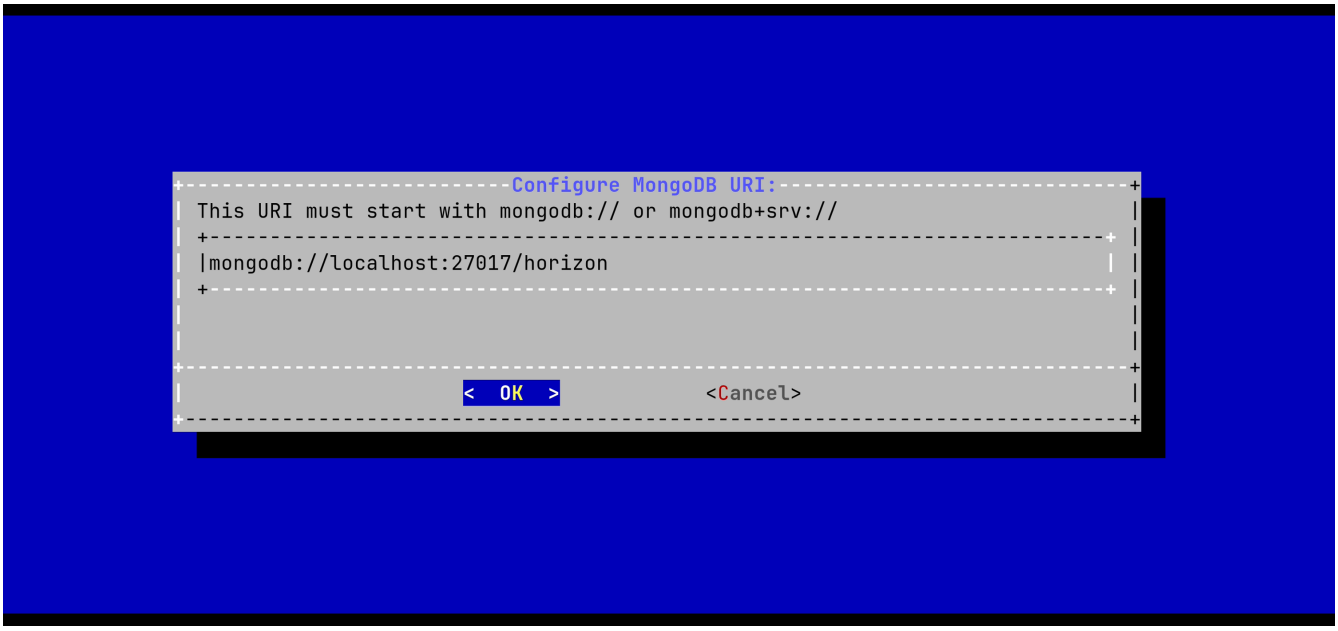
In the main menu, select **Horizon**:



In the Horizon configuration menu, Select **MONGODB_URI**:



Specify the MongoDB URI to target your MongoDB instance:



Horizon is installed to target a local MongoDB instance by default.

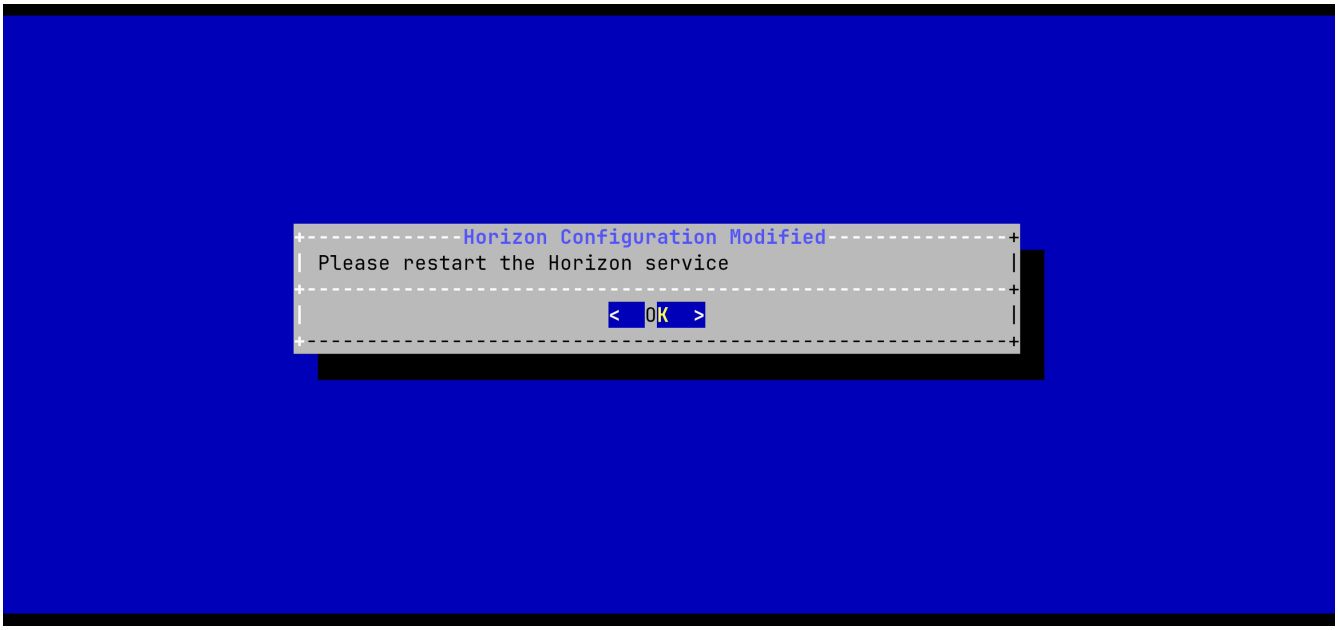
If you use an external MongoDB (such as MongoDB Atlas Database or dedicated On-premises database) instance:

- Create a user with "read/write" permissions on your MongoDB instance;
- Create a replicaSet if using a MongoDB cluster;
- Specify a MongoDB URI that does match your context.

External MongoDB database URI syntax: `mongodb+srv://<user>:<password>@<Mongo-DB-hostname>:<Mongo-DB-Port>/horizon`

External MongoDB cluster of databases URI syntax: `mongodb+srv://<user>:<password>@<Mongo-DB-hostname-1>,<Mongo-DB-hostname-2>:<Mongo-DB-Port>/horizon?replicaSet=<Horizon-ReplicaSet-Name>&authSource=admin`

The MongoURI is configured:



For the changes to take effect, you must restart the Horizon service by running:

```
# systemctl restart horizon
```

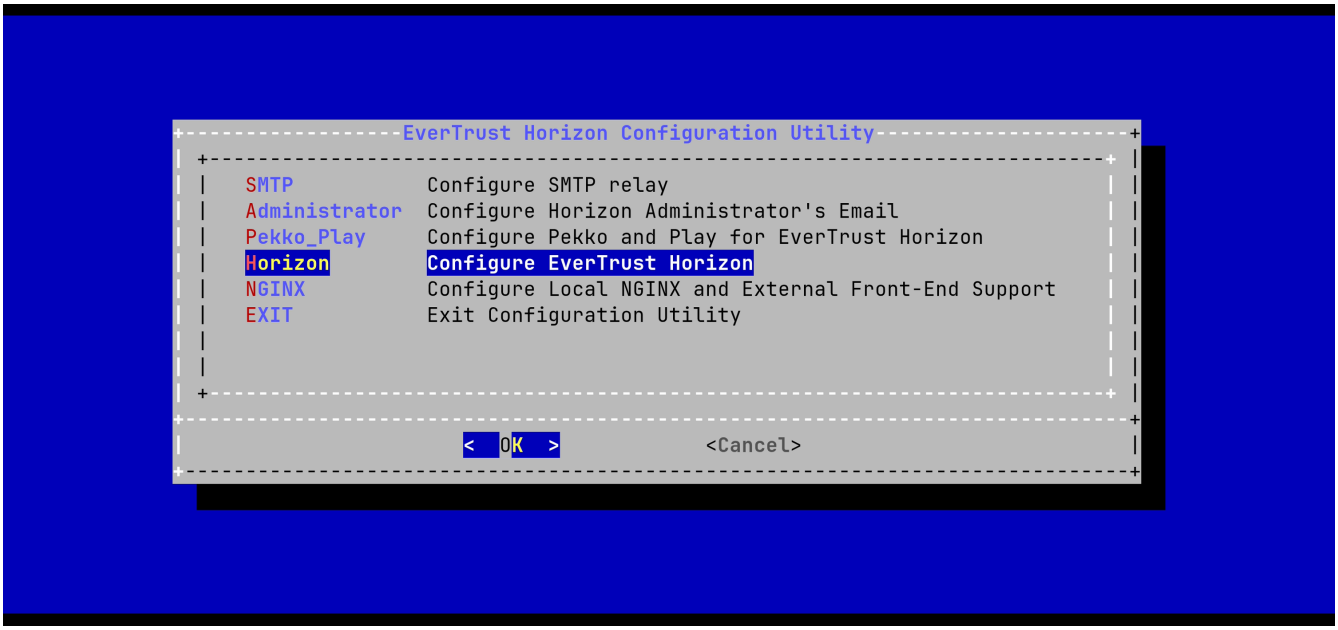
Horizon Hostname Configuration

Connect to the server with an account with administrative privileges;

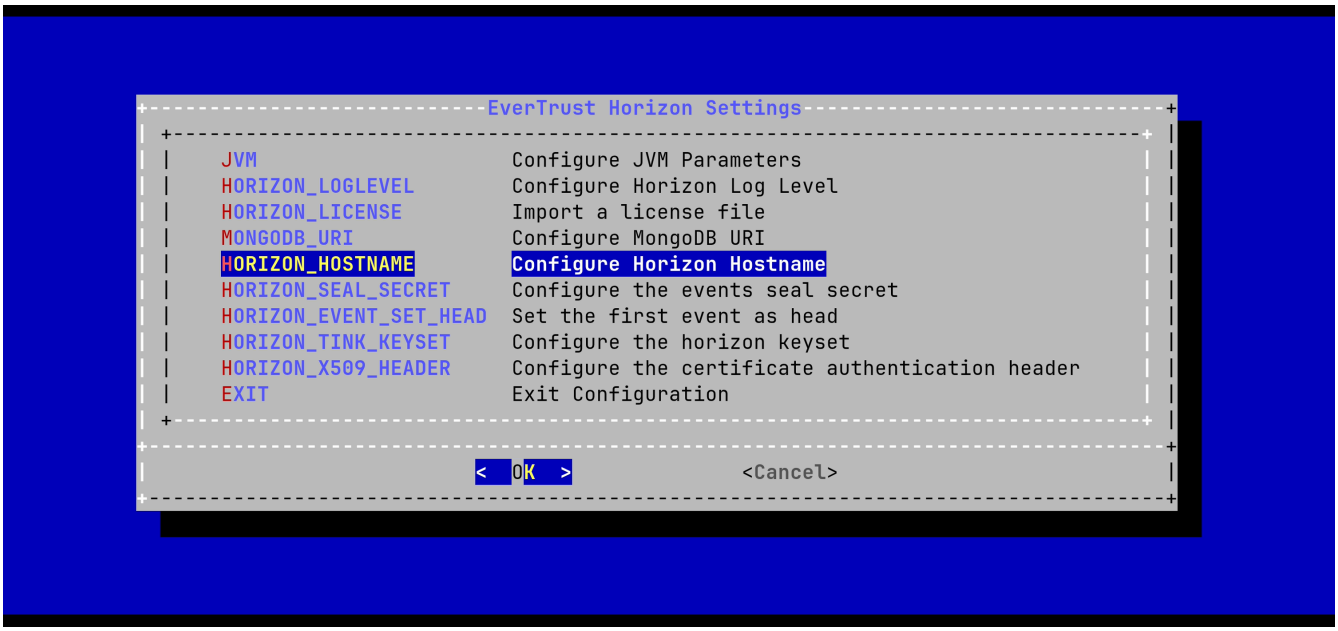
Start the Horizon configuration utility by running:

```
# /opt/horizon/sbin/horizon-config
```

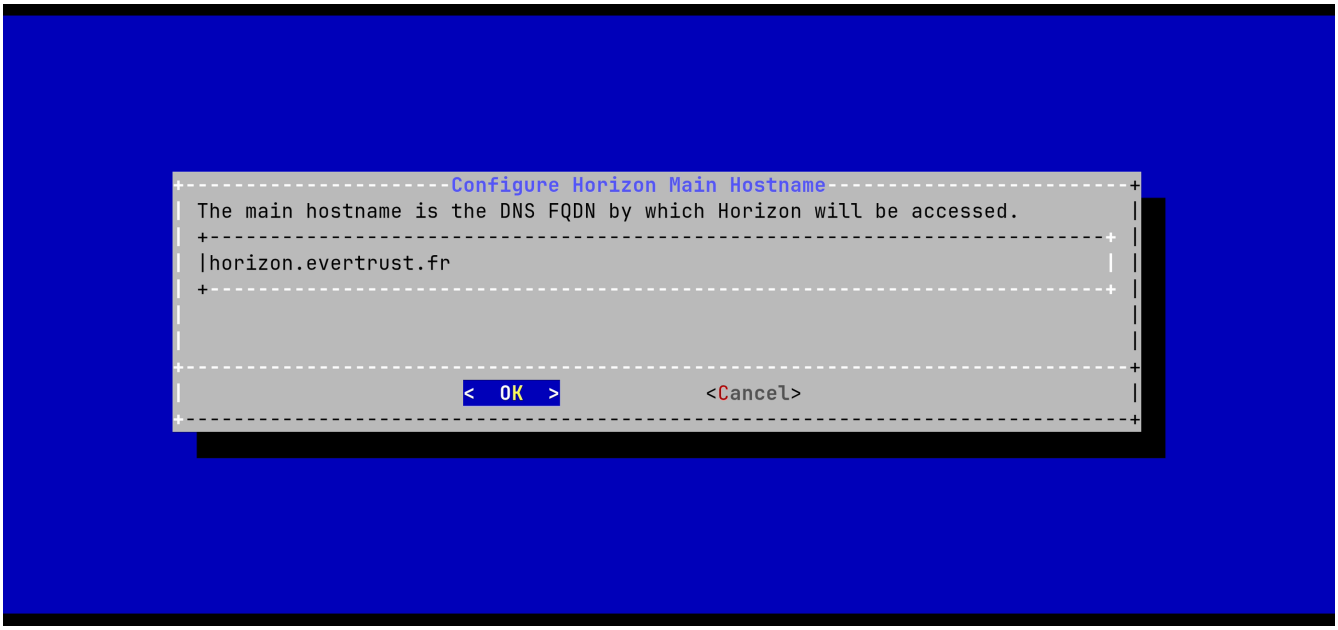
In the main menu, select '**Horizon**':



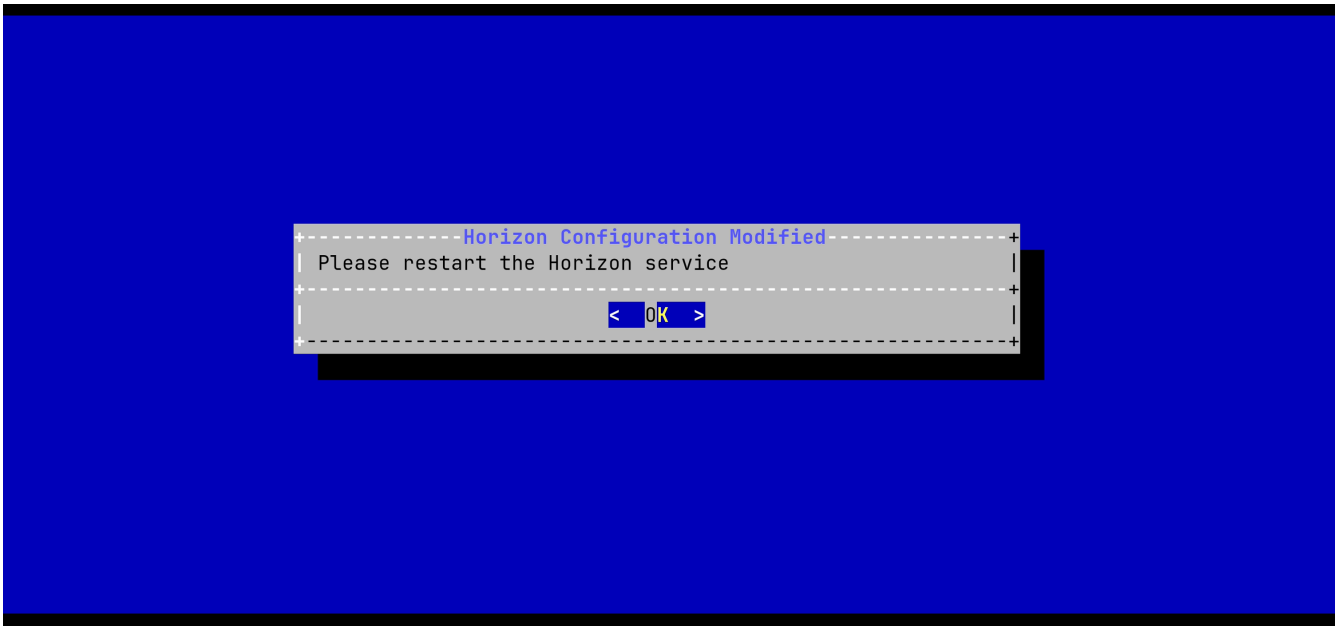
In the Horizon configuration menu, select **HORIZON_HOSTNAME**:



Specify the DNS FQDN by which Horizon will be accessed:



The Horizon Hostname is configured:



For the changes to take effect, you must restart the Horizon service by running:

```
# systemctl restart horizon
```

Generating an event seal secret

Horizon will generate functional events when using the software.

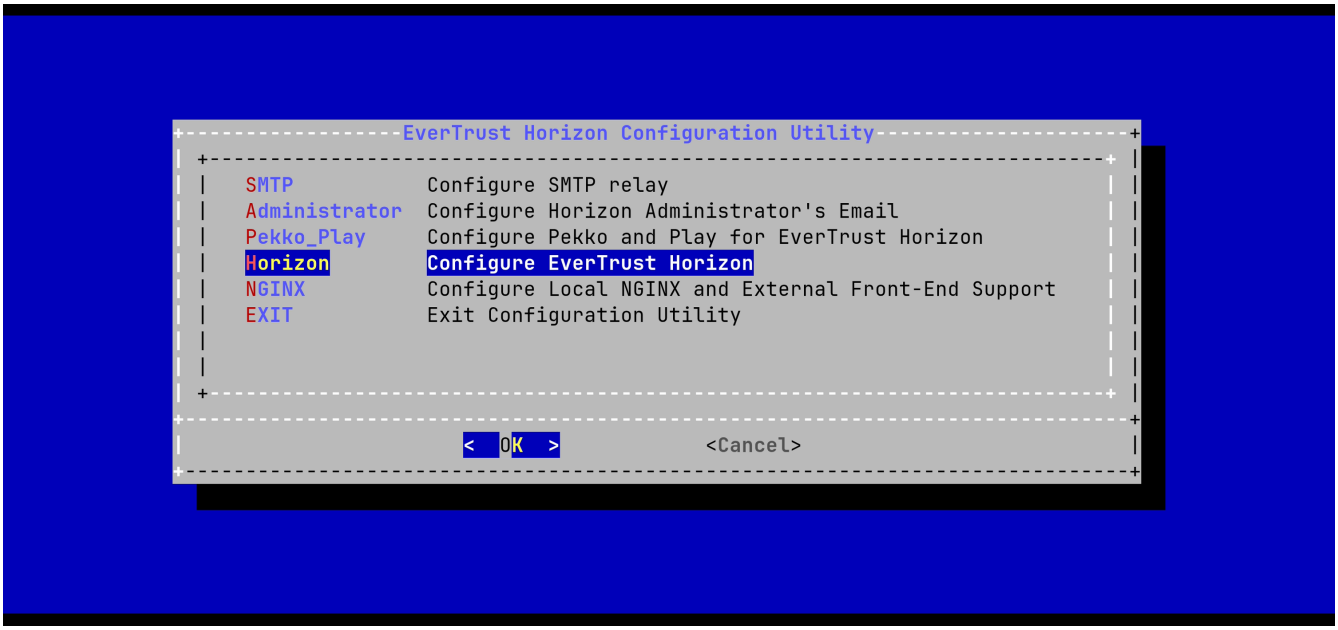
These events are typically signed and chained to ensure their integrity. Therefore, you must specify a sealing secret for this feature to work correctly.

Connect to the server with an account with administrative privileges;

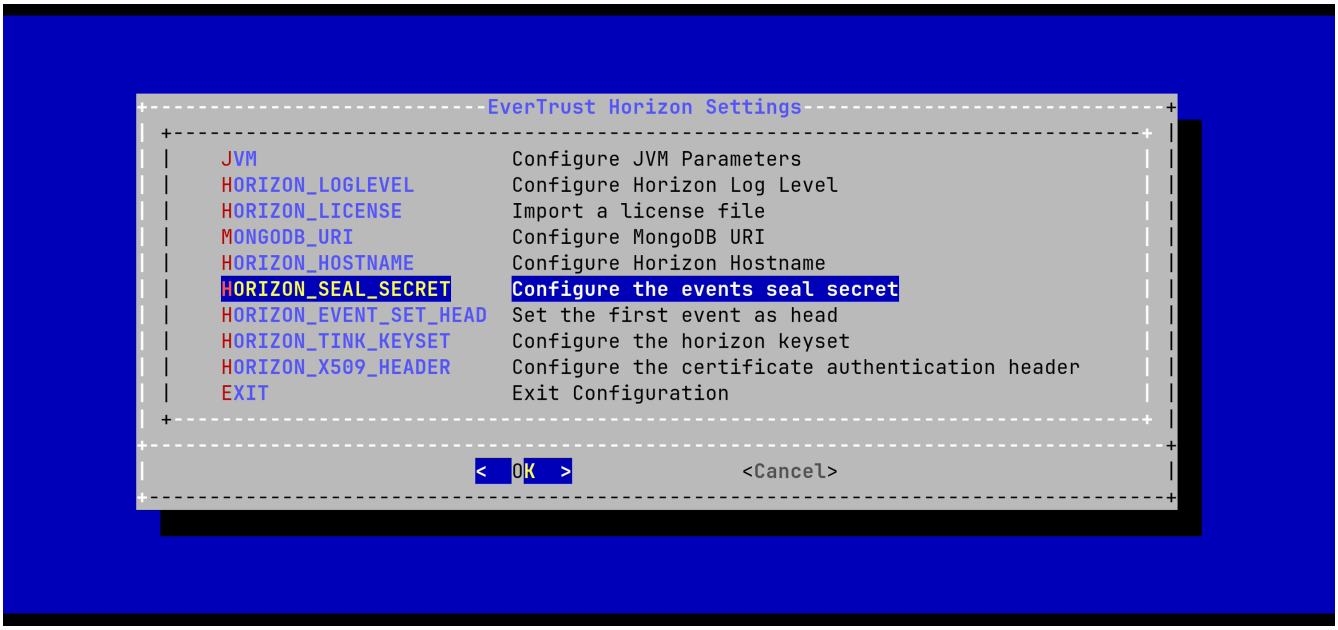
Start the Horizon configuration utility by running:

```
# /opt/horizon/sbin/horizon-config
```

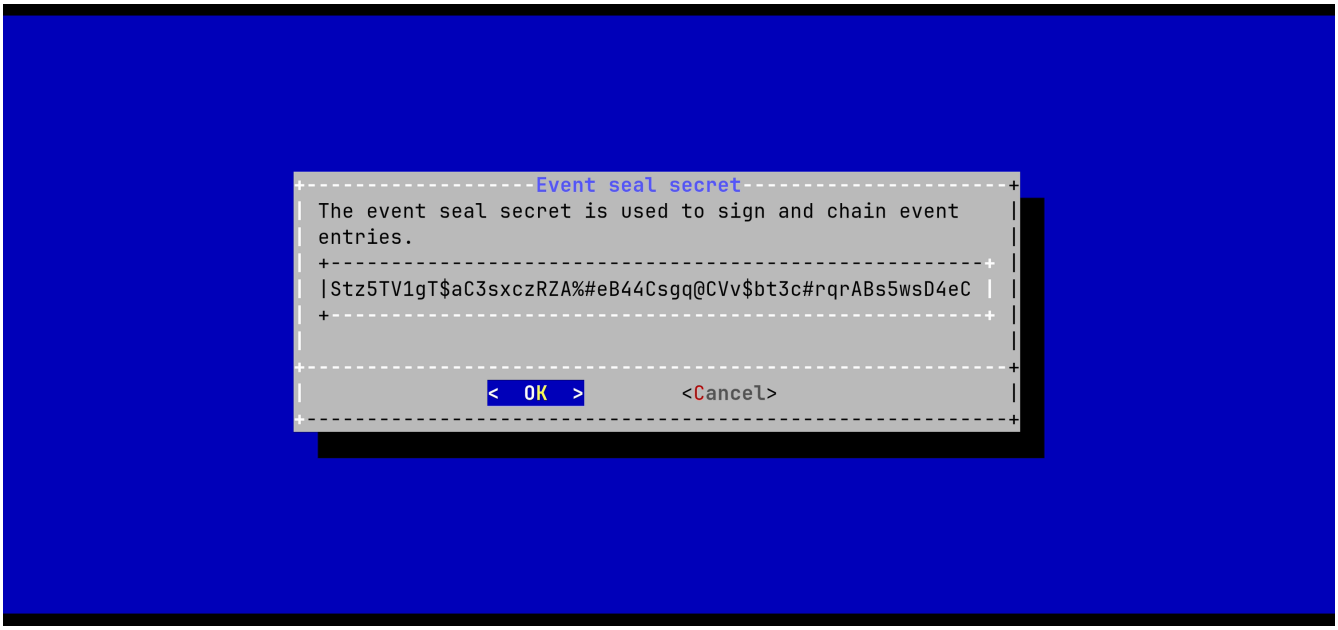
In the main menu, select '**Horizon**':



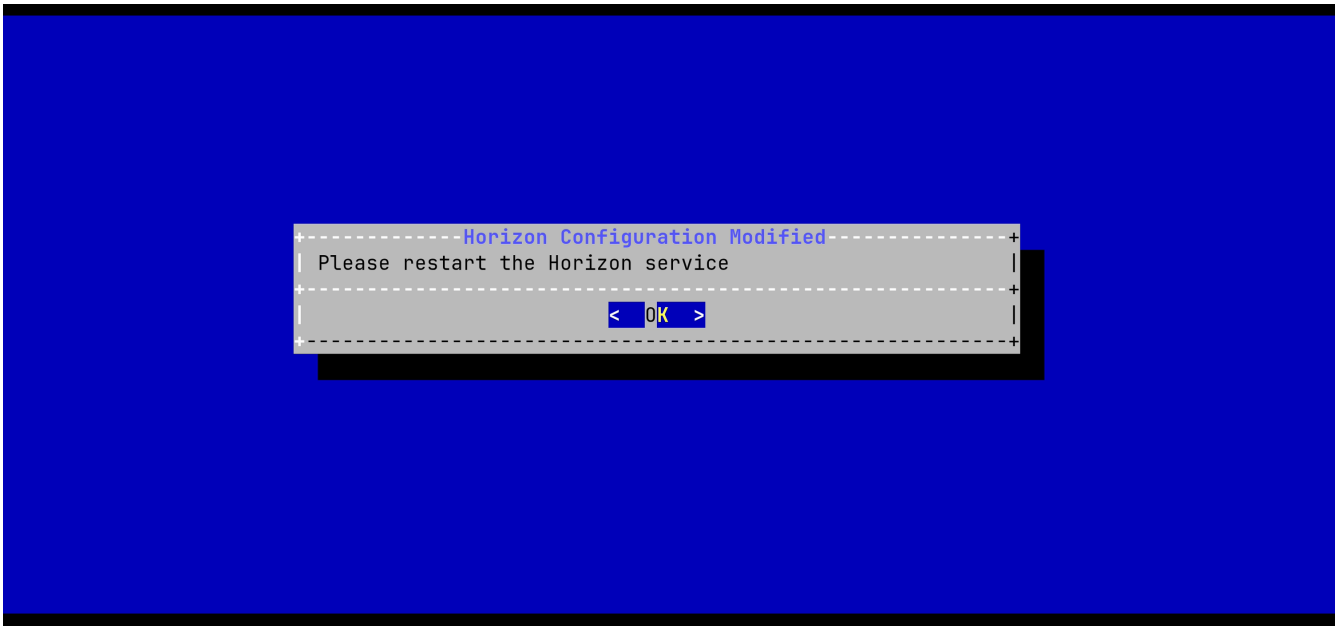
In the Horizon menu, select '**HORIZON_SEAL_SECRET**':



Validate the new event seal secret:



The event seal secret is now configured:



For the changes to take effect, you must restart the Horizon service by running:

```
# systemctl restart horizon
```

Installing the Horizon license



You should have been provided with a '***horizon.lic***' file. This file is a license file and indicates:

- The horizon entitled module(s)
- The limitation in terms of holder per module if any
- A end of support date

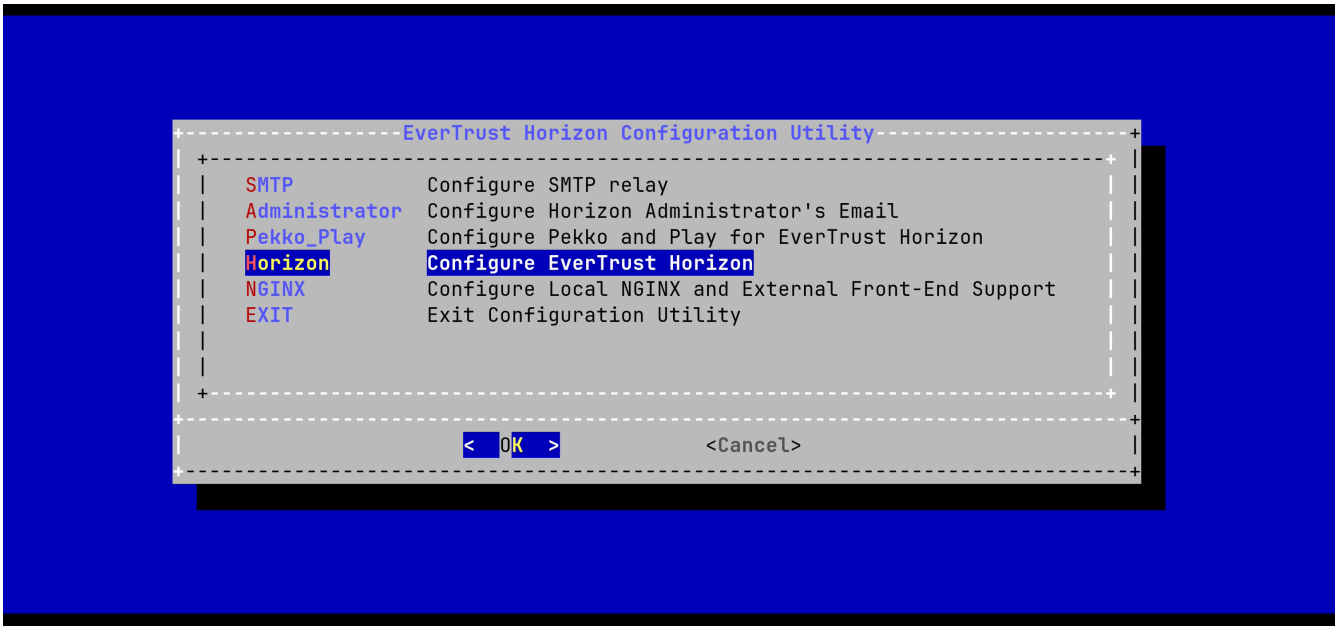
Upload the `horizon.lic` file through SCP under `/tmp/horizon.lic`;

Connect to the server with an account with administrative privileges;

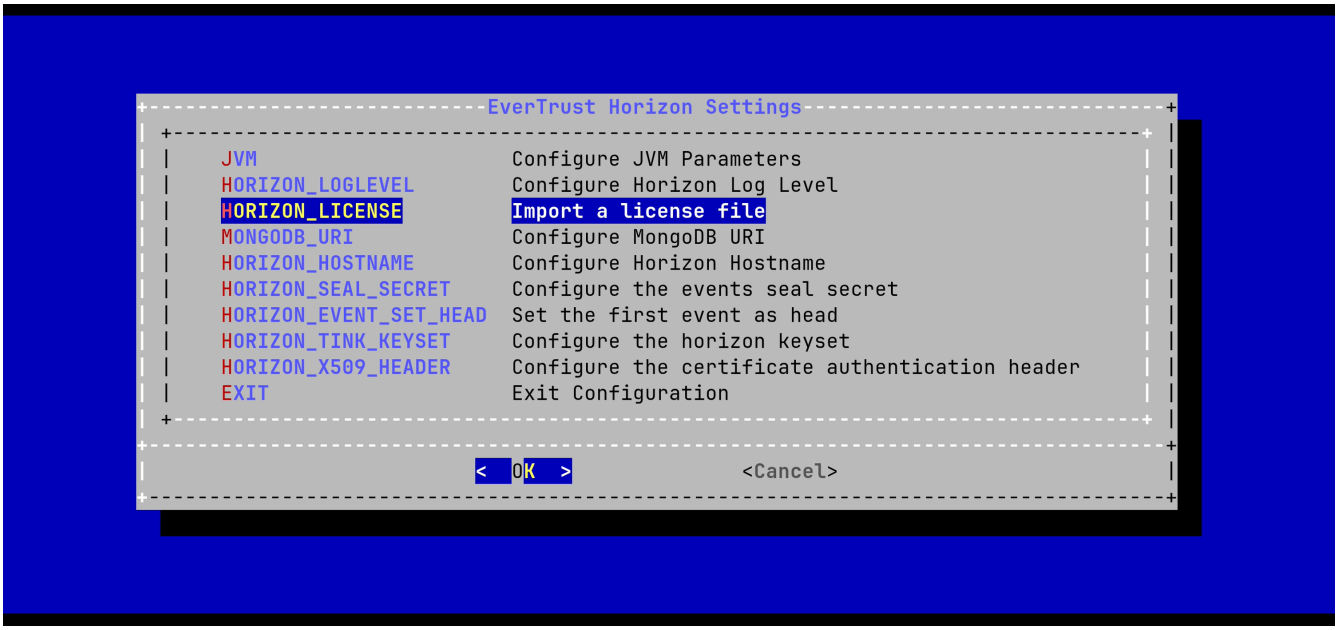
Start the Horizon configuration utility by running:

```
# /opt/horizon/sbin/horizon-config
```

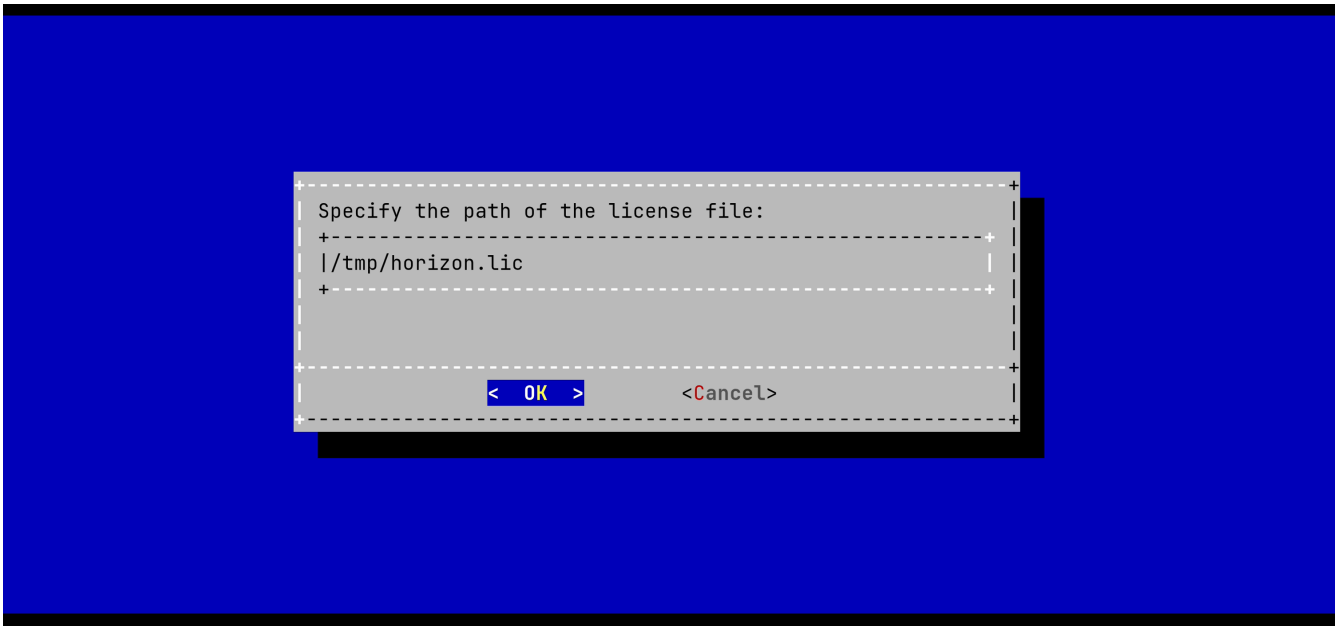
In the main menu, select '**Horizon**':



In the Horizon configuration menu, Select '**HORIZON_LICENSE**':



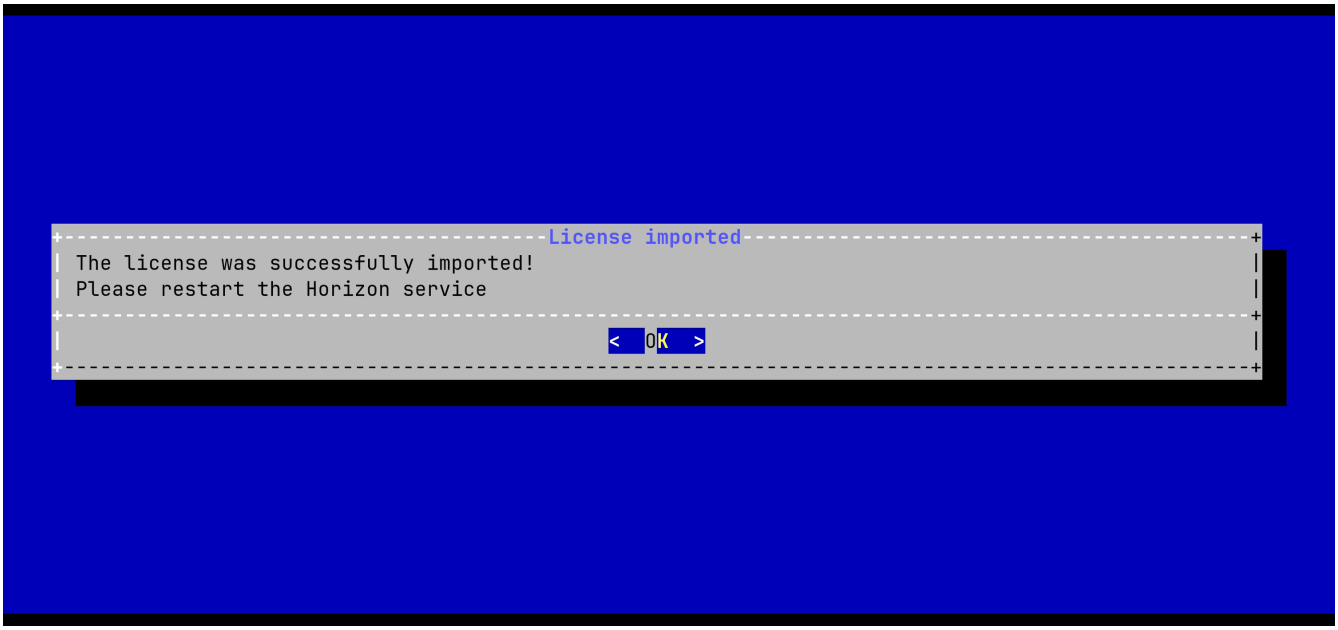
Specify the path `/tmp/horizon.lic` and validate:



The information of the license should be prompted. If everything is good, import the license:



The Horizon License is configured:



For the changes to take effect, you must restart the Horizon service by running:

```
# systemctl restart horizon
```

Generating a Tink keyset

To protect its secrets, Horizon relies on Tink. A Tink keyset can be issued as:

- A plaintext keyset (stored as a file, protected by the filesystem rights and SELinux);
- A GCP keyset (protected by a master key in a GCP KMS);
- An AWS keyset (protected by a master key in an AWS KMS).
- A PKCS#11 keyset (protected by a master key in an HSM).



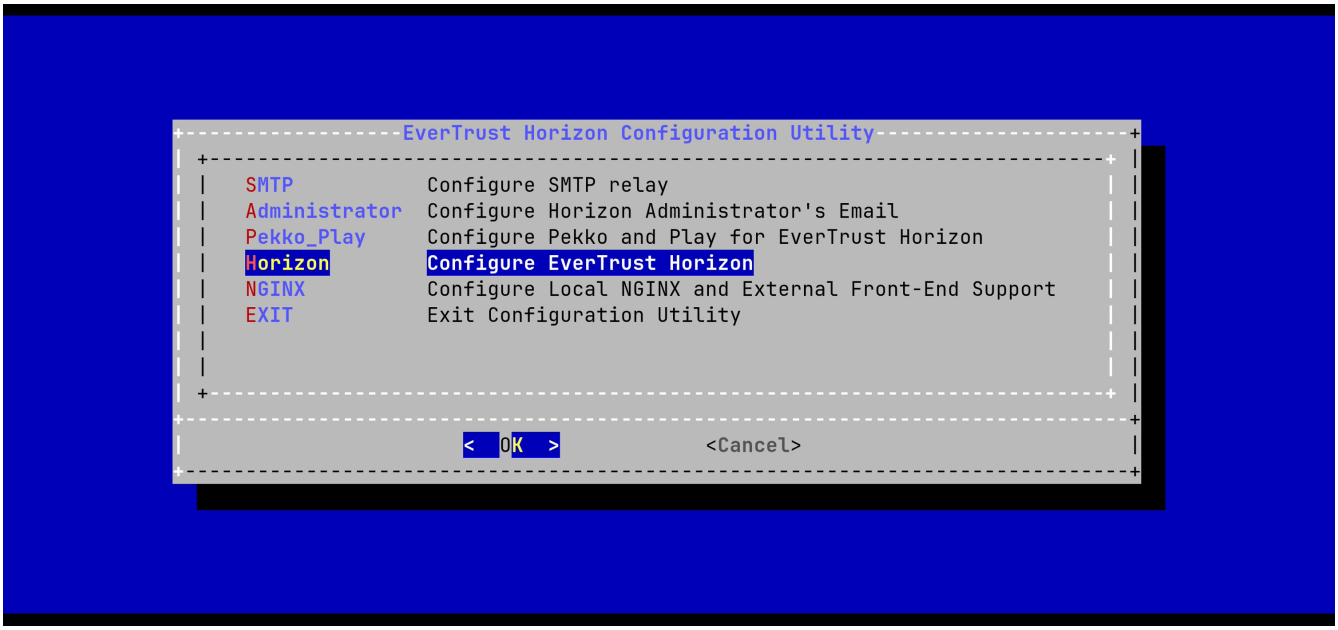
In order to generate a keyset, the Tinkey tool must be installed.

Connect to the server with an account with administrative privileges;

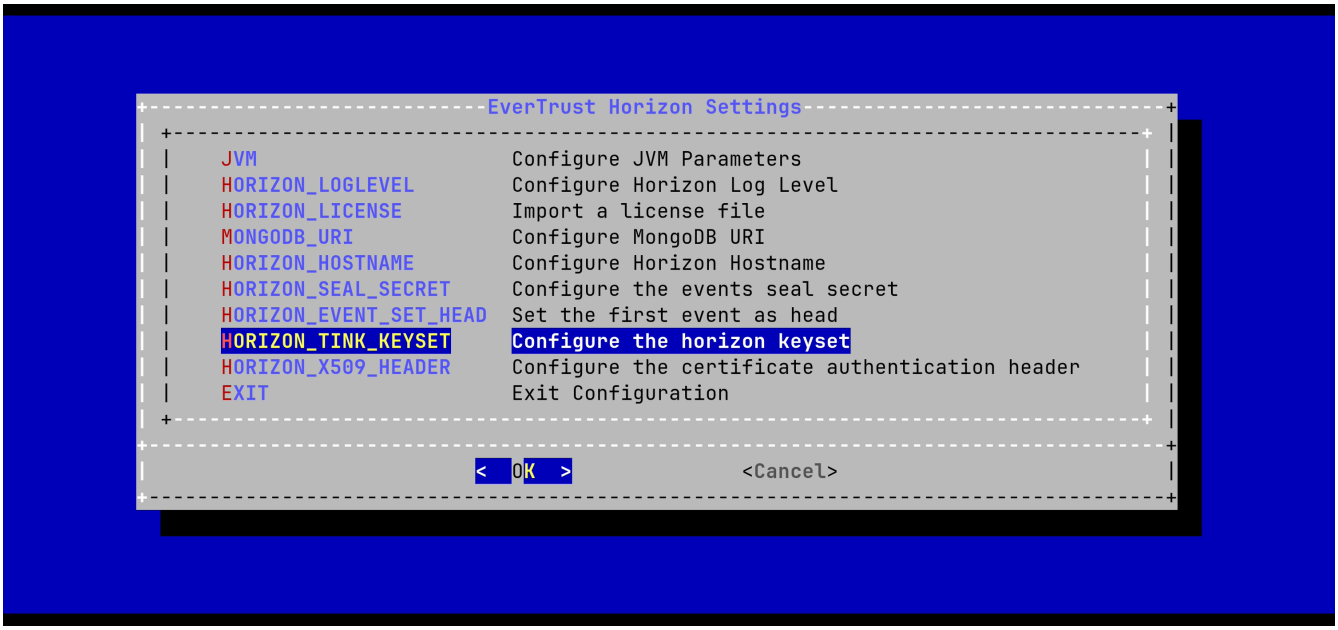
Start the Horizon configuration utility by running:

```
# /opt/horizon/sbin/horizon-config
```

In the main menu, select '**Horizon**':

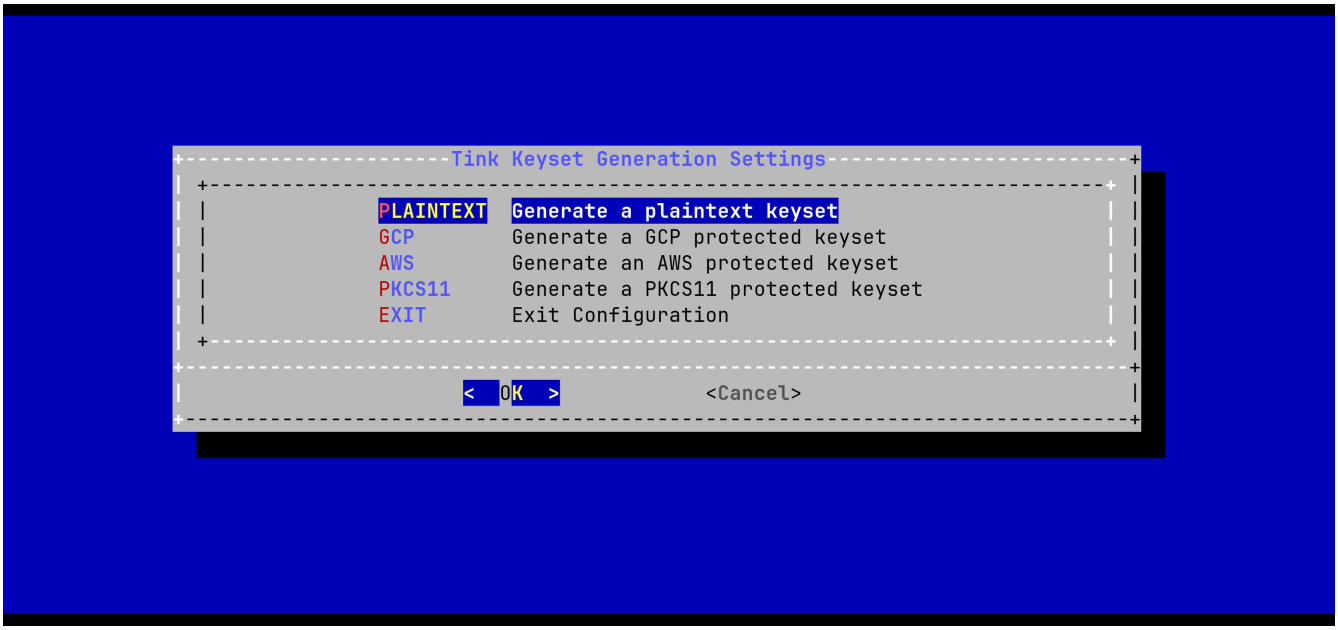


In the Stream menu, select '**HORIZON_TINK_KEYSET**':

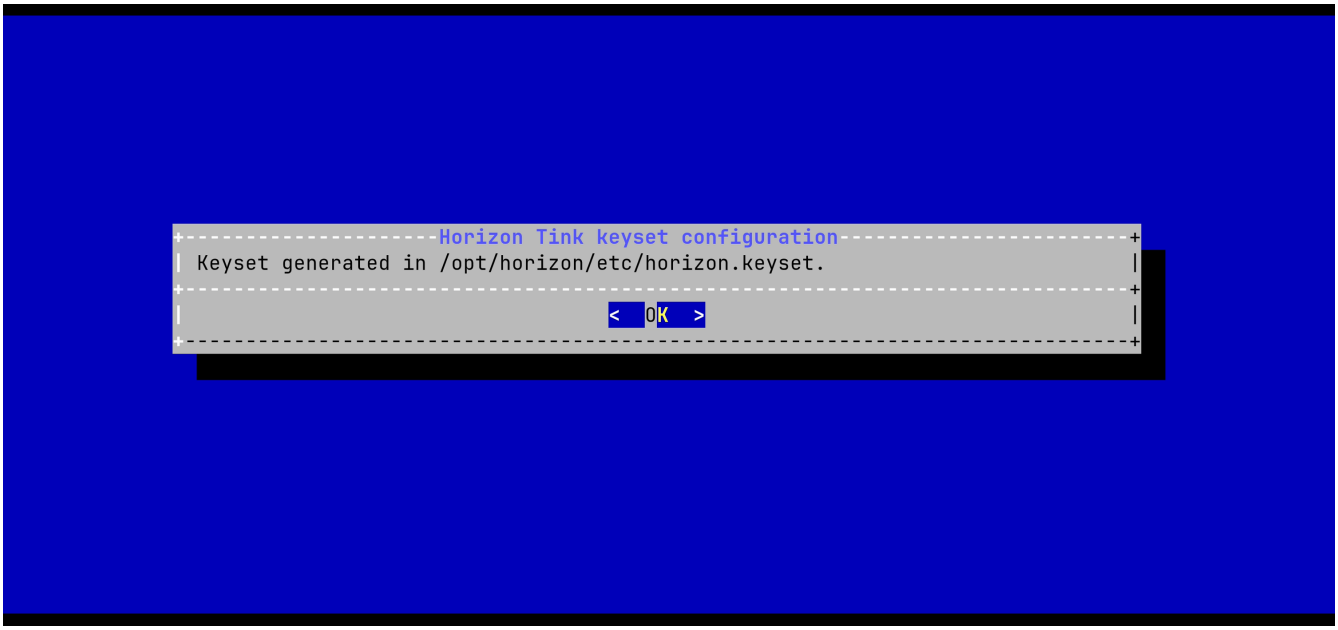


Generating a plaintext keyset

In the Tink Keyset Generation menu, select '**PLAINTEXT**':



The keyset will be generated automatically.

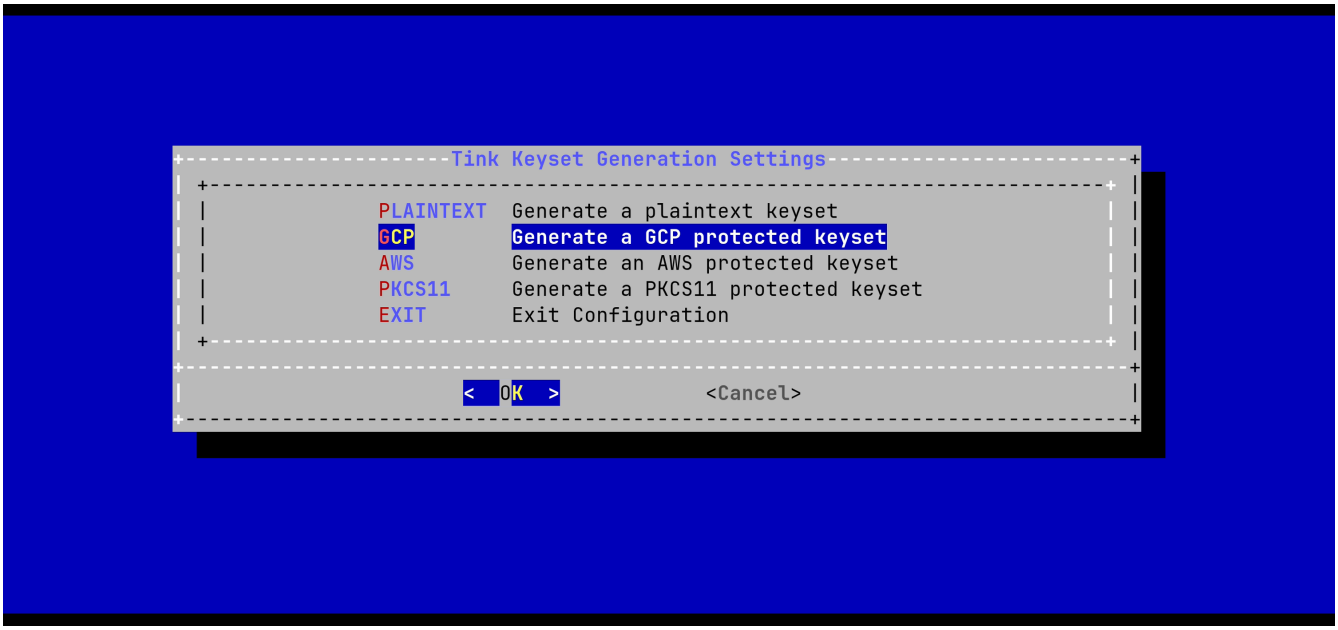


For the changes to take effect, you must restart the Horizon service by running:

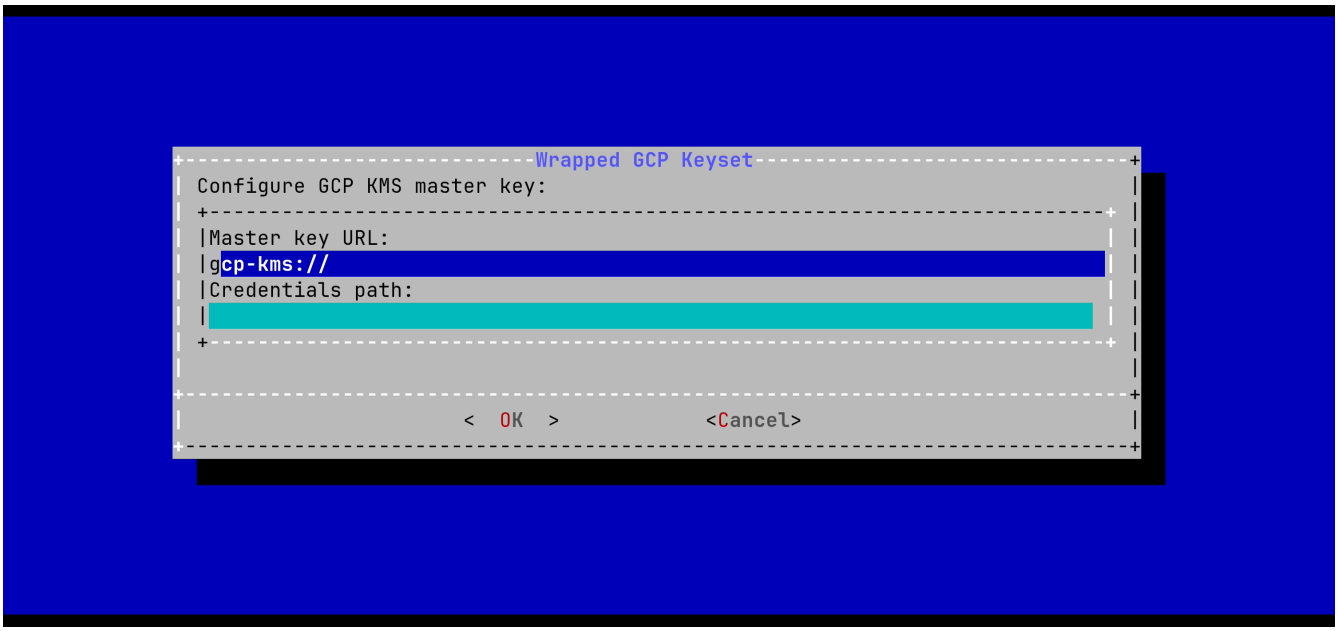
```
# systemctl restart horizon
```

Generating a GCP protected keyset

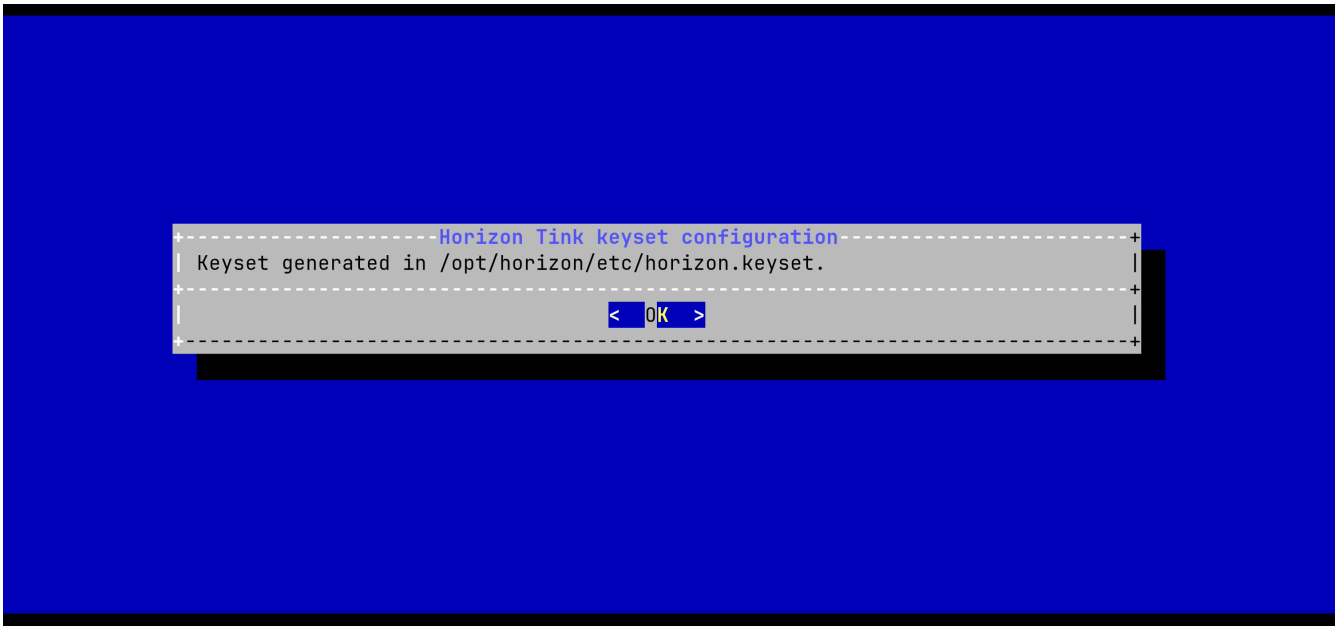
In the Tink Keyset Generation menu, select '**GCP**':



The URL of the GCP master key must be typed in the menu. Path to a credentials file can be specified if not using the default SDK path.



After pressing **OK**, the keyset will be generated automatically.

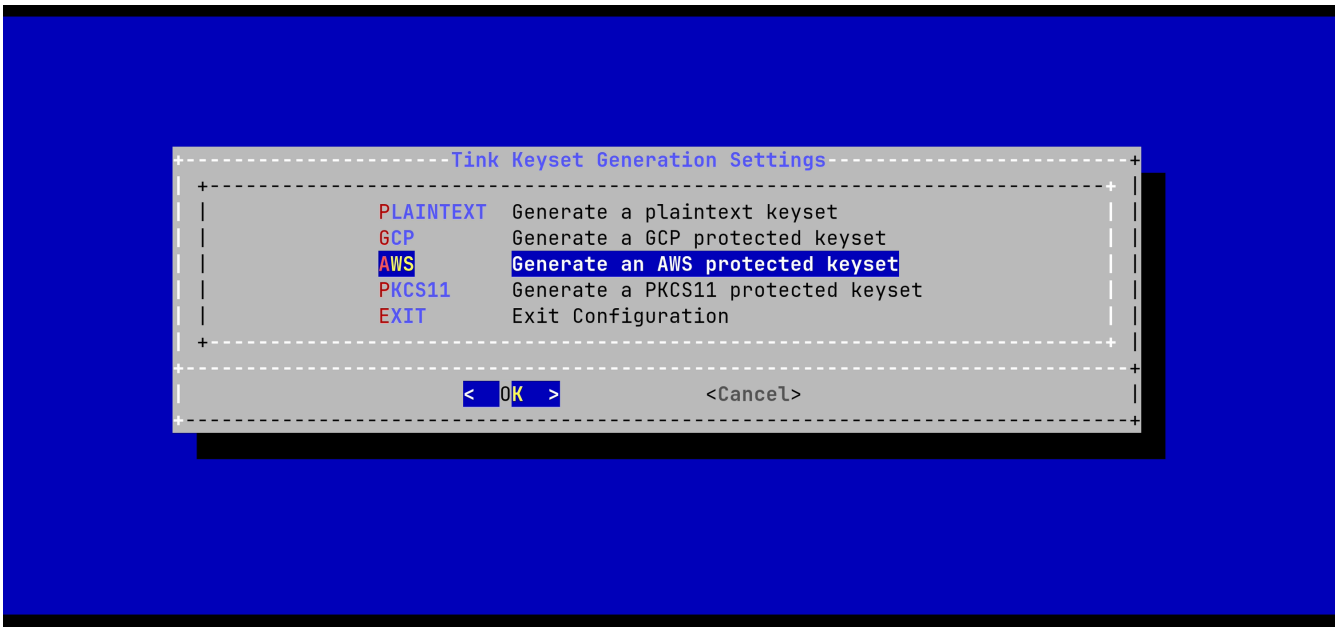


For the changes to take effect, you must restart the Horizon service by running:

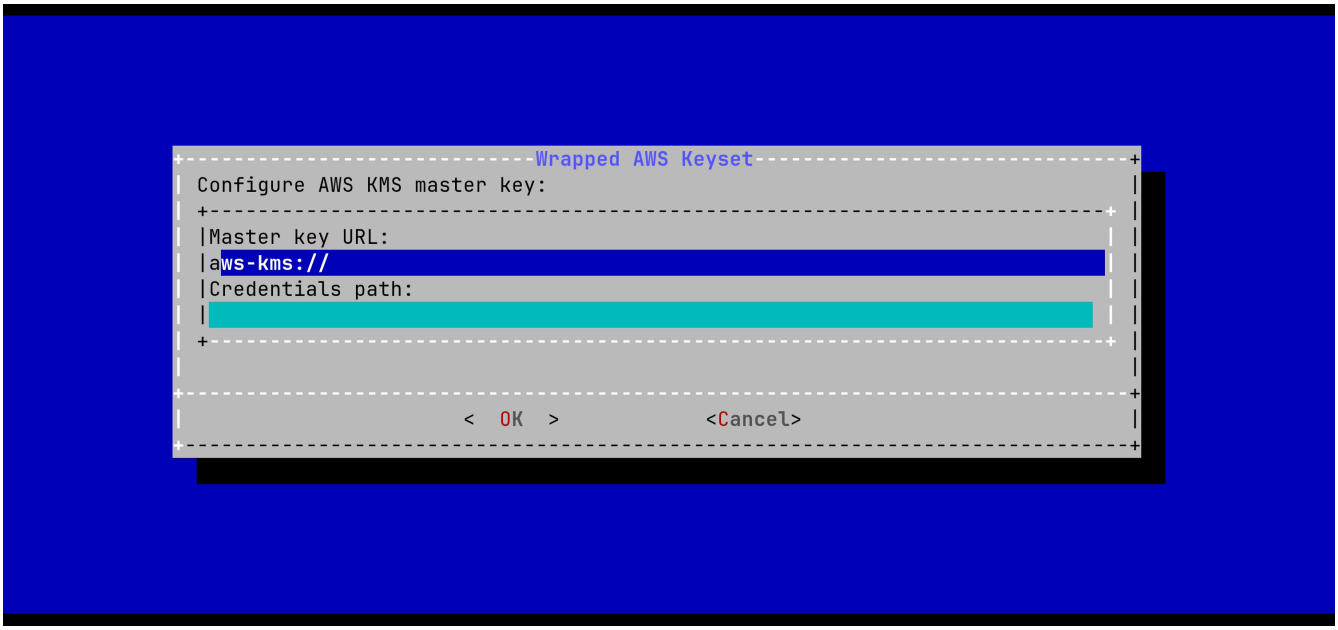
```
# systemctl restart horizon
```

Generating an AWS protected keyset

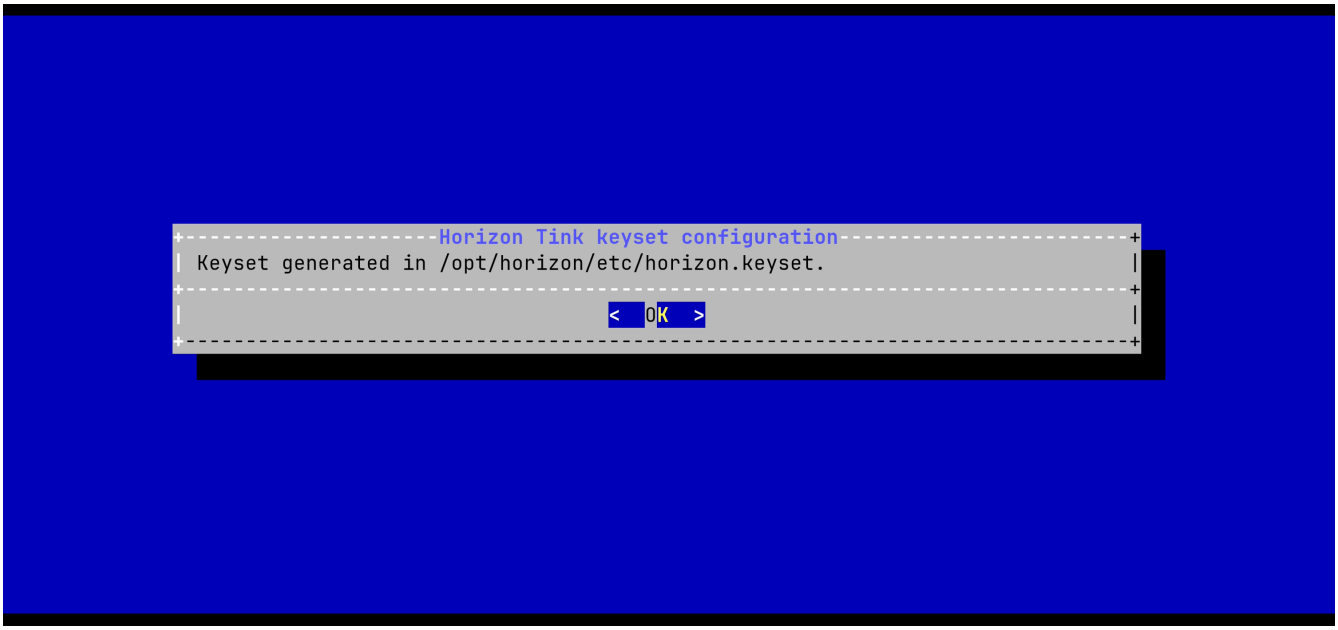
In the Tink Keyset Generation menu, select '**AWS**':



The URL of the AWS master key must be typed in the menu. Path to a credentials file can be specified if not using the default SDK path.



After pressing **OK**, the keyset will be generated automatically.

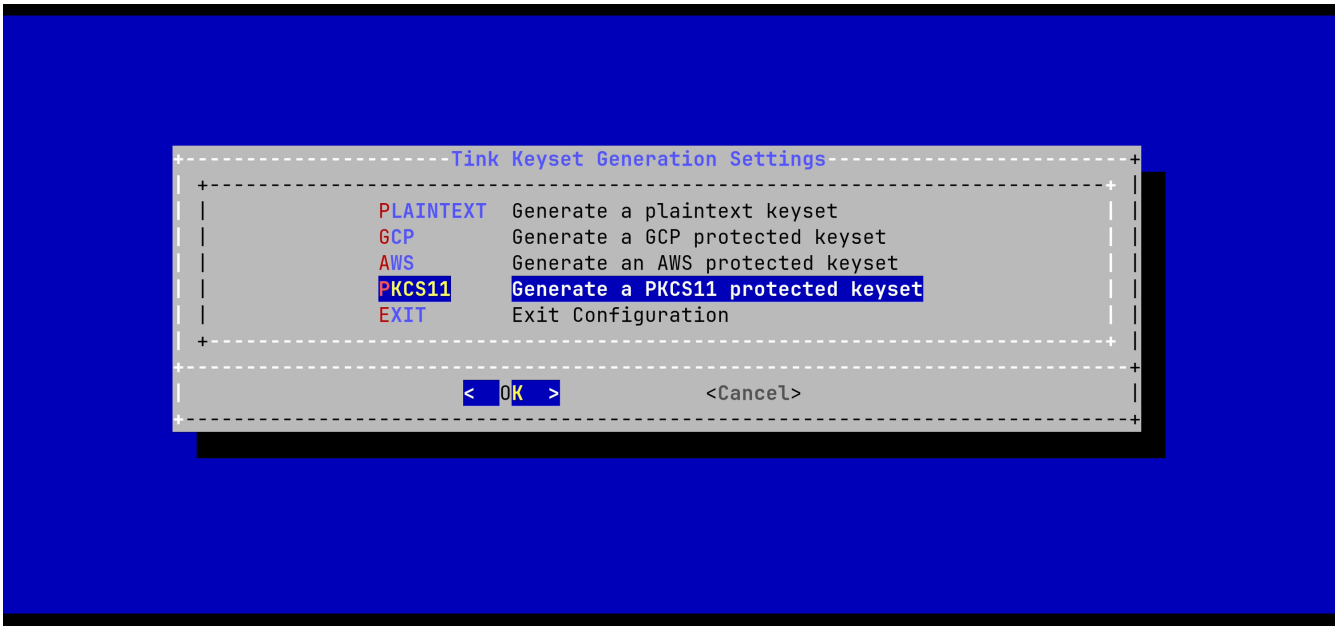


For the changes to take effect, you must restart the Horizon service by running:

```
# systemctl restart horizon
```

Generating a PKCS#11 protected keyset

In the Tink Keyset Generation menu, select '**PKCS11**':



The URL of the PKCS#11 master key must be typed in the menu.



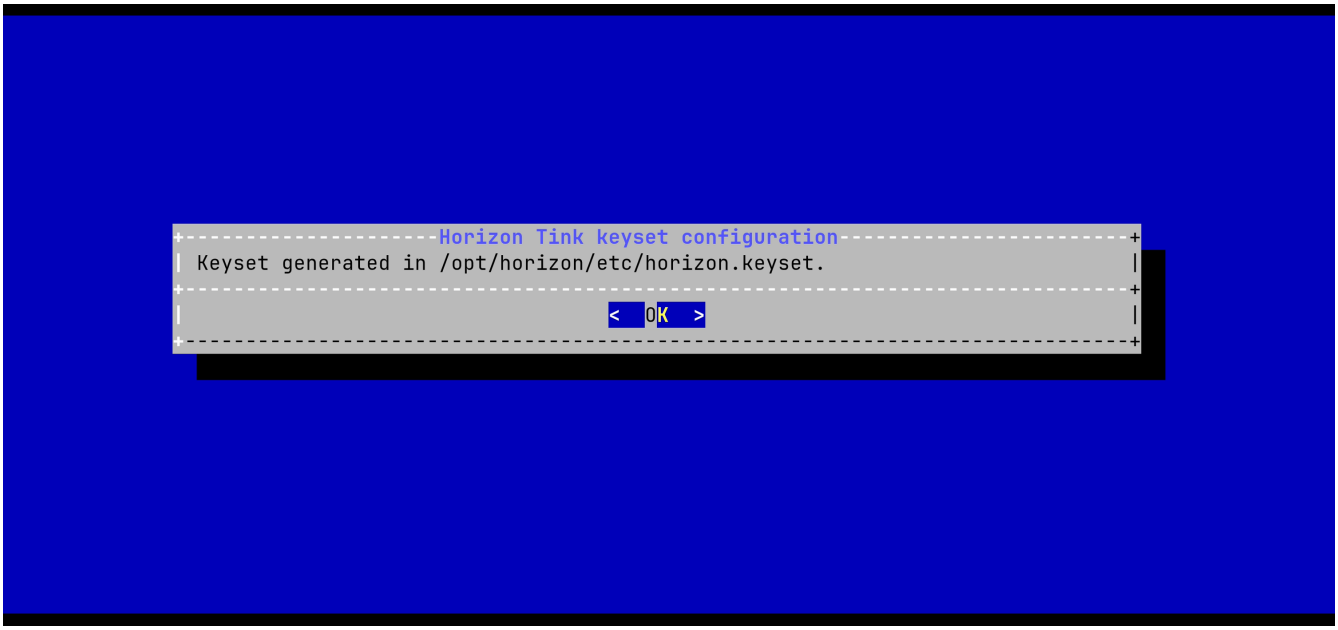
The expected format is:

```
pkcs11://object=<object name>;type=<object type>;slot-id=<slot id>?module-
path=<library path>&pin-value=<pin>;
```

Example:

```
pkcs11://object=kek;type=secret-key;slot-id=-1?module-
path=/usr/lib/softhsm/libsofthsm2.so&pin-value=1234";
```

After pressing **OK**, the keyset will be generated automatically.



For the changes to take effect, you must restart the Horizon service by running:

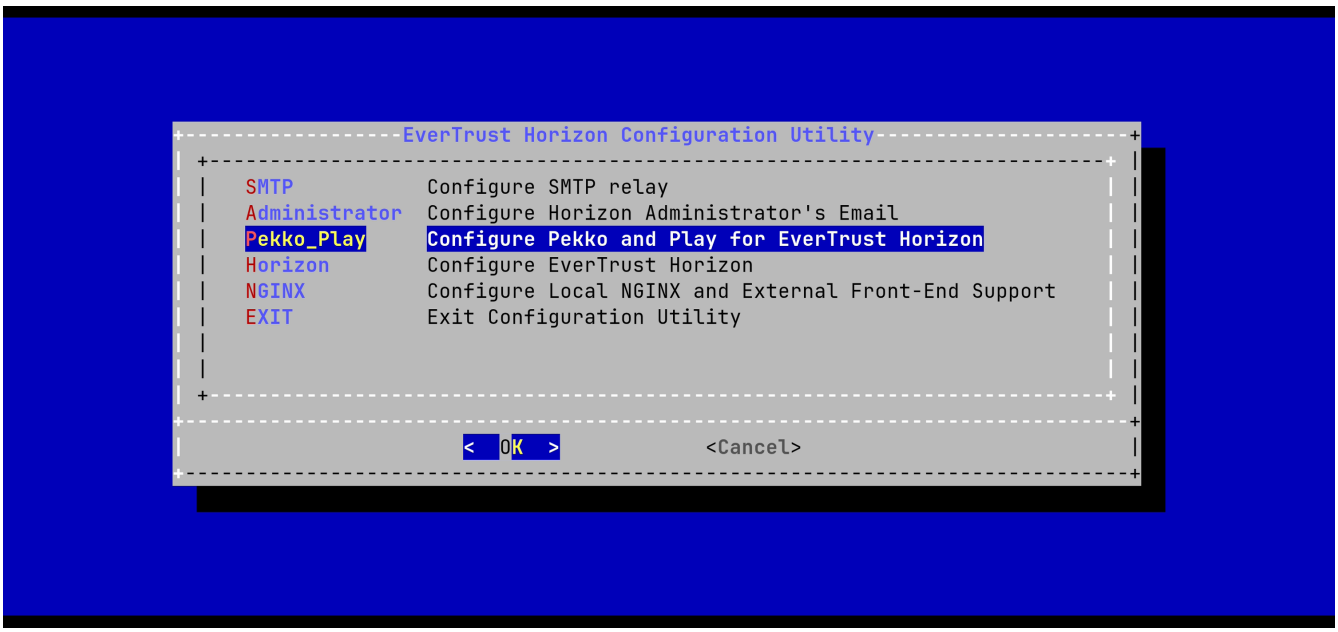
```
# systemctl restart horizon
```

Installing Horizon on a cluster of servers

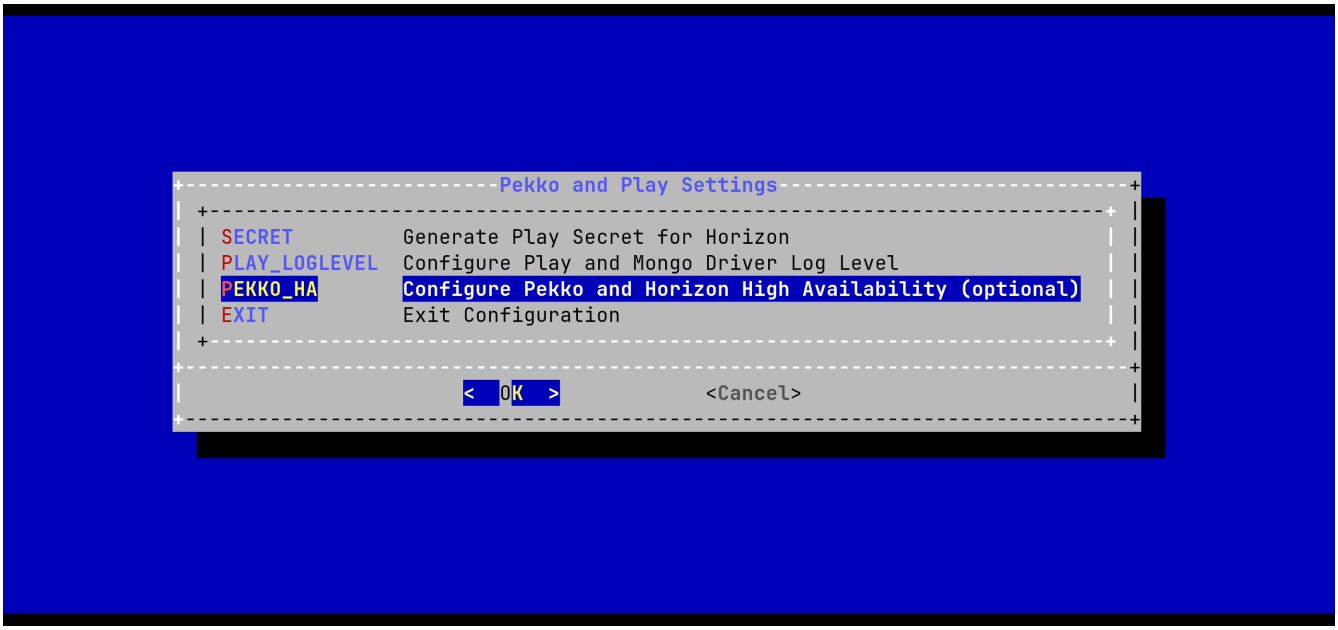


This section must not be followed if you plan on deploying Horizon in standalone mode (vs cluster mode). WARNING: This section does not explain how to install Horizon on a Kubernetes cluster. Please refer to the dedicated section.

In the main menu, select '**Pekko_Play**':



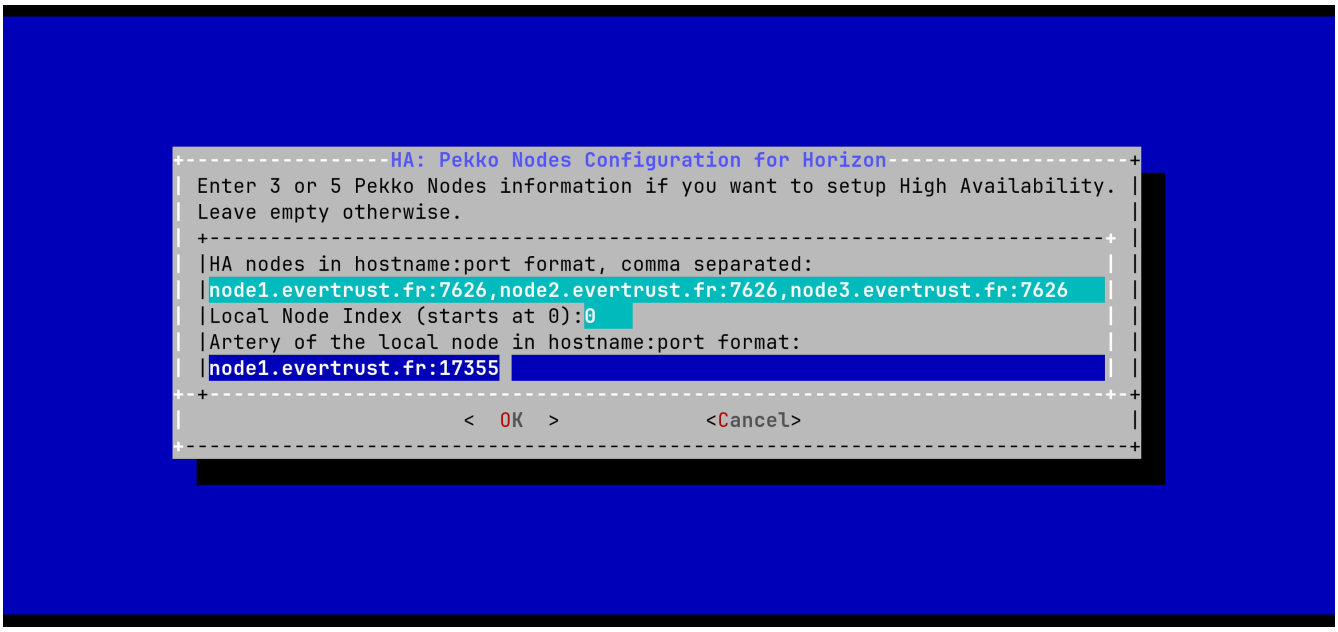
In the Pekko_Play menu, select '**PEKKO_HA**':



In this menu, specify either the IP address or the DNS name for each server that will be running Horizon on this cluster, as well as the local node index (the number of the node that you are configuring at that moment). You must also specify where the port Artery is hosted, usually it should be on the same node with a different port.

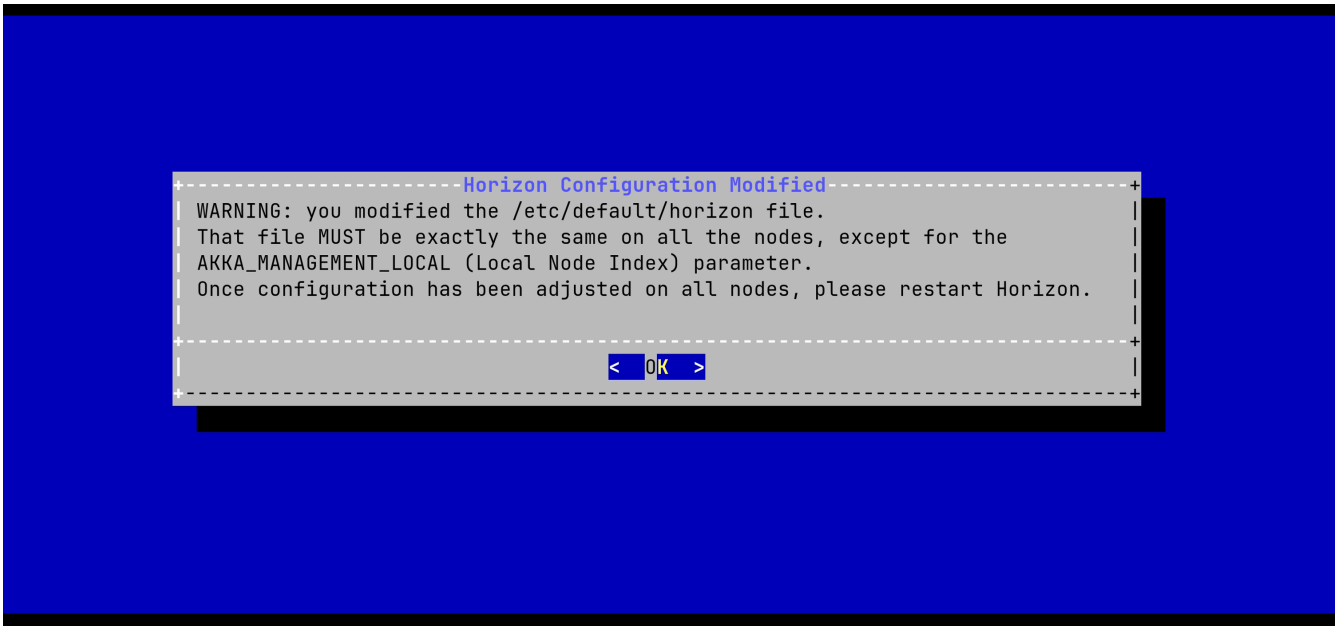


Note that the local node index must match the Node Hostname parameter:



Save your changes from the menu.

The High Availability mode is now configured on the current node:



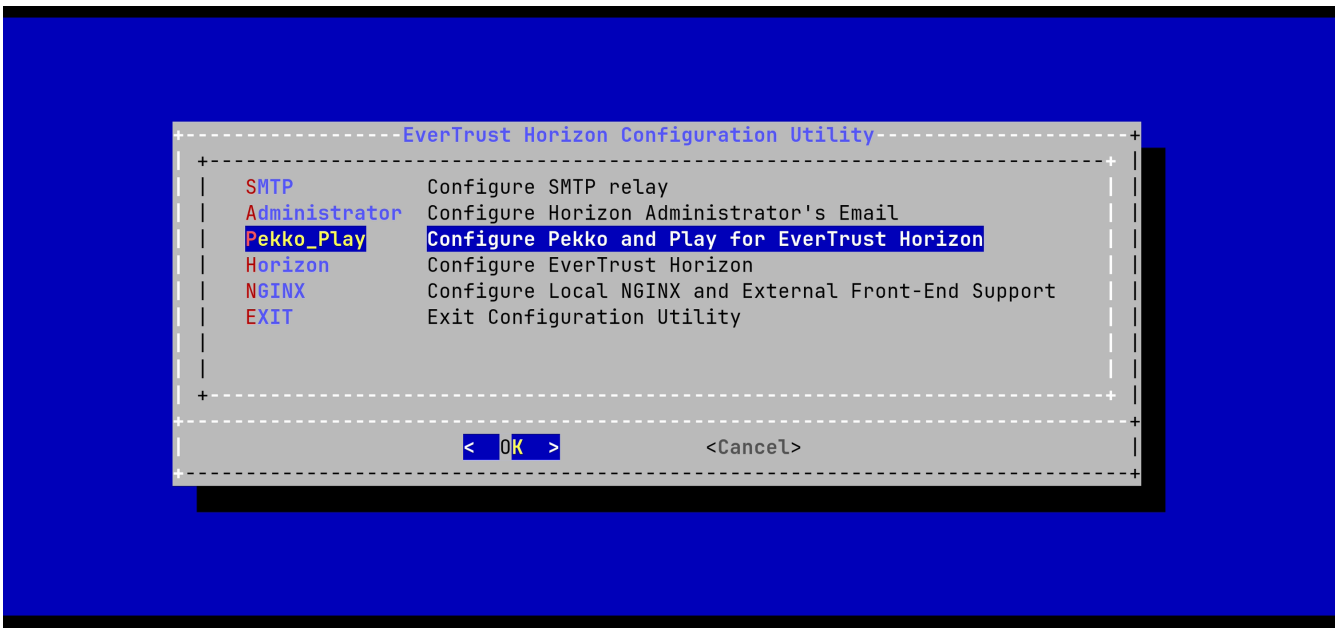
You must now configure your other nodes, but because they belong to the same cluster they need to share the **same secret, the same secret seal event, the same hostname and the same database**. In order to be able to do that, you need to copy the configuration file that was generated by the horizon-config app, named `/etc/default/horizon` and paste it on each one of your nodes;

Then on each other node, run the Horizon Configuration utility:

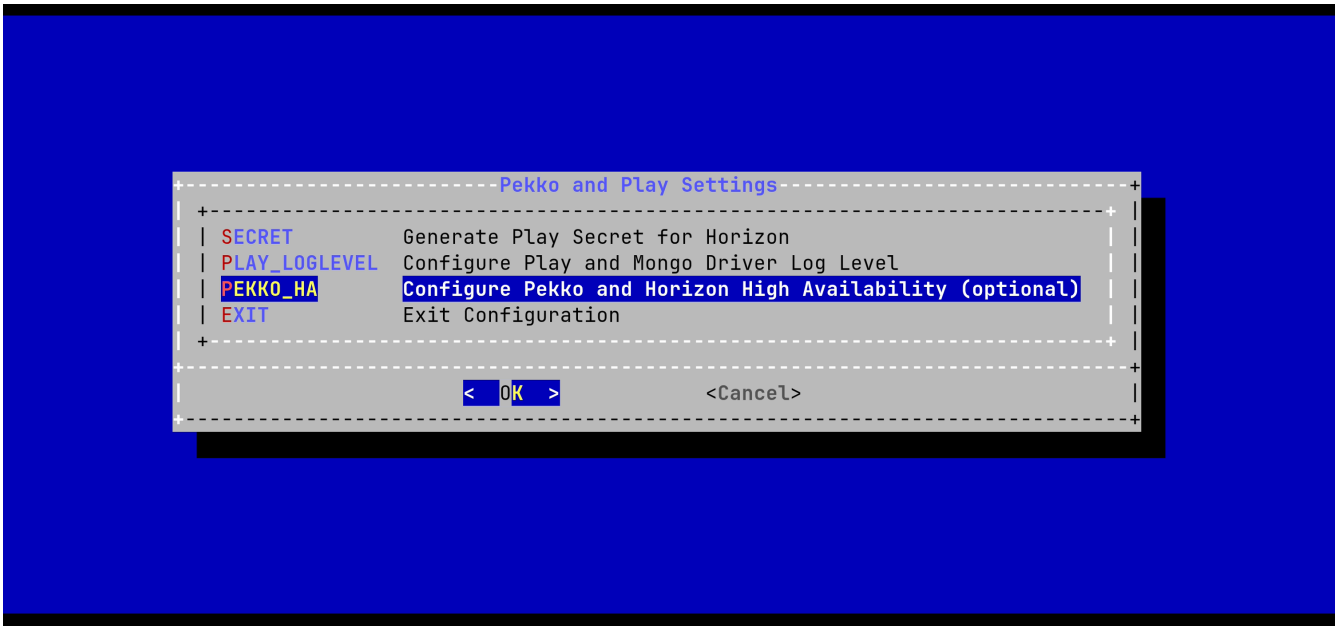
Connect to the server with an account with administrative privileges;

Start the Horizon configuration utility by running:

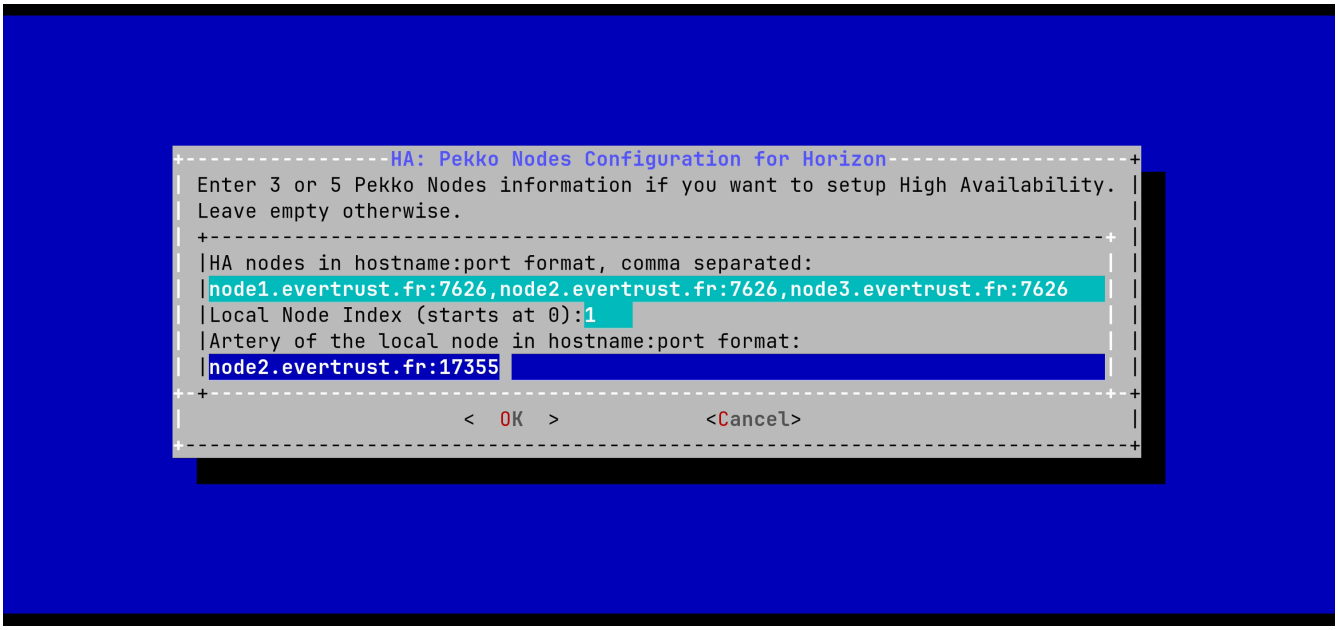
```
# /opt/horizon/sbin/horizon-config
```



In the Pekko_Play menu, select '**PEKKO_HA**':



Here, you need to change the local node index to match the hostname of the node that you are configuring:





You will need to import the Horizon license file on each node manually, following the guidelines of section Installing the Horizon license, as well as copying the keyset in `/opt/horizon/etc/horizon.keyset`.

Additionally, on each node, you will need to open the ports used for Pekko_HA and Pekko_MGMT, which are by default 17355 and 7626:

RHEL

```
$ firewall-cmd --permanent --add-port=17355/tcp
$ firewall-cmd --permanent --add-port=7626/tcp
```

Reload the firewall configuration with:

```
$ systemctl restart firewalld
```

Debian

If you are using a specific firewall, make sure to open these ports.

For the changes to take effect, you must restart the Horizon service by running:

```
# systemctl restart horizon
```

Enabling the lease

To allow for High Availability even when a minority of nodes are up, the following configuration should be added (reference).

```
pekko.cluster.split-brain-resolver {
    active-strategy = "lease-majority"
    lease-majority {
      lease-implementation = "lease.mongo"
    }
}
```

Server Authentication Certificate

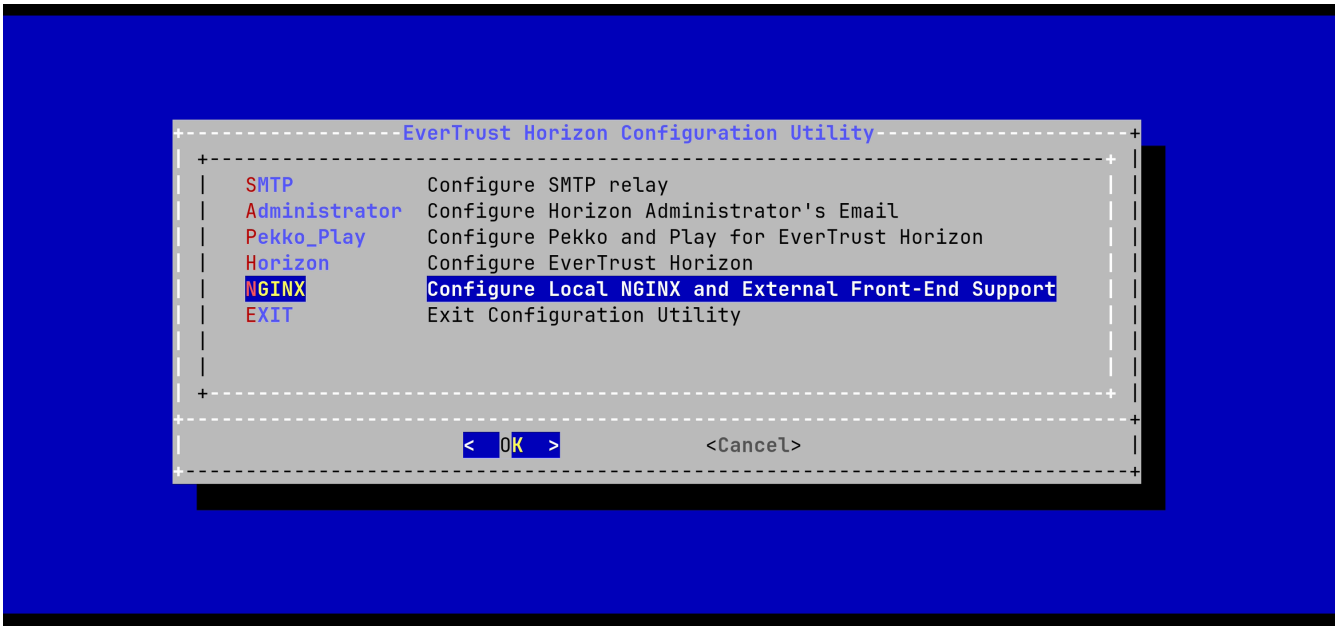
Issuing a Certificate Request (PKCS#10)

Connect to the server with an account with administrative privileges;

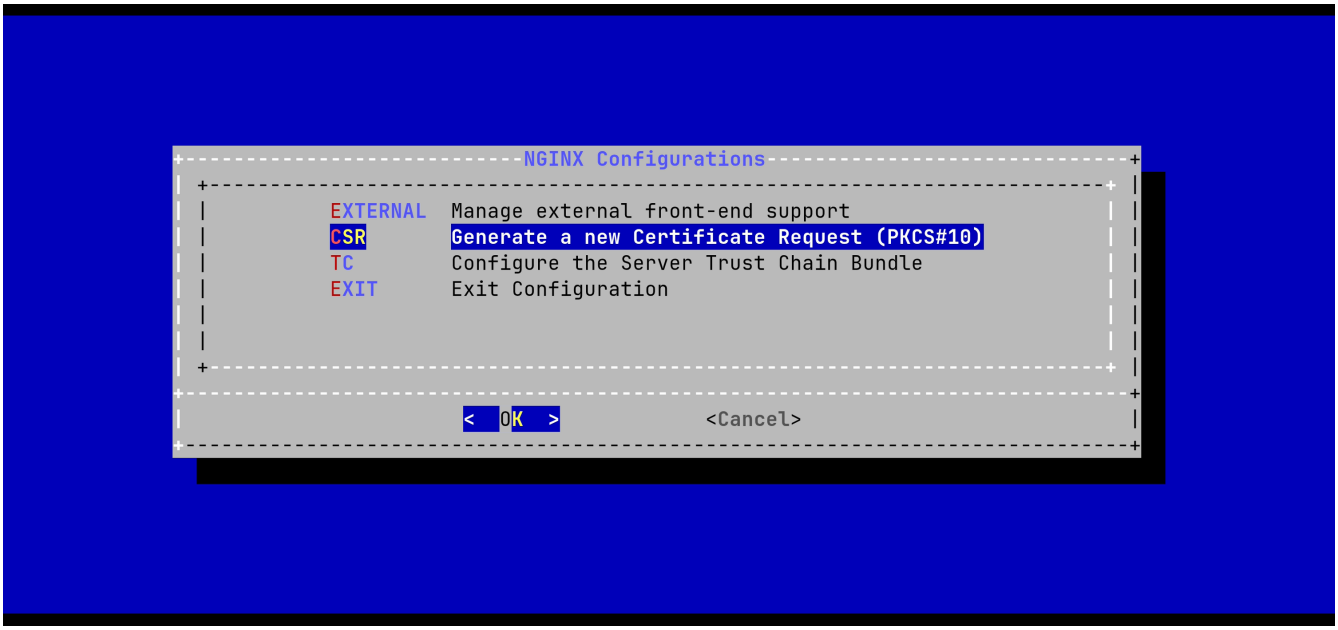
Start the Horizon configuration utility by running:

```
# /opt/horizon/sbin/horizon-config
```

In the main menu, select '**NGINX**':



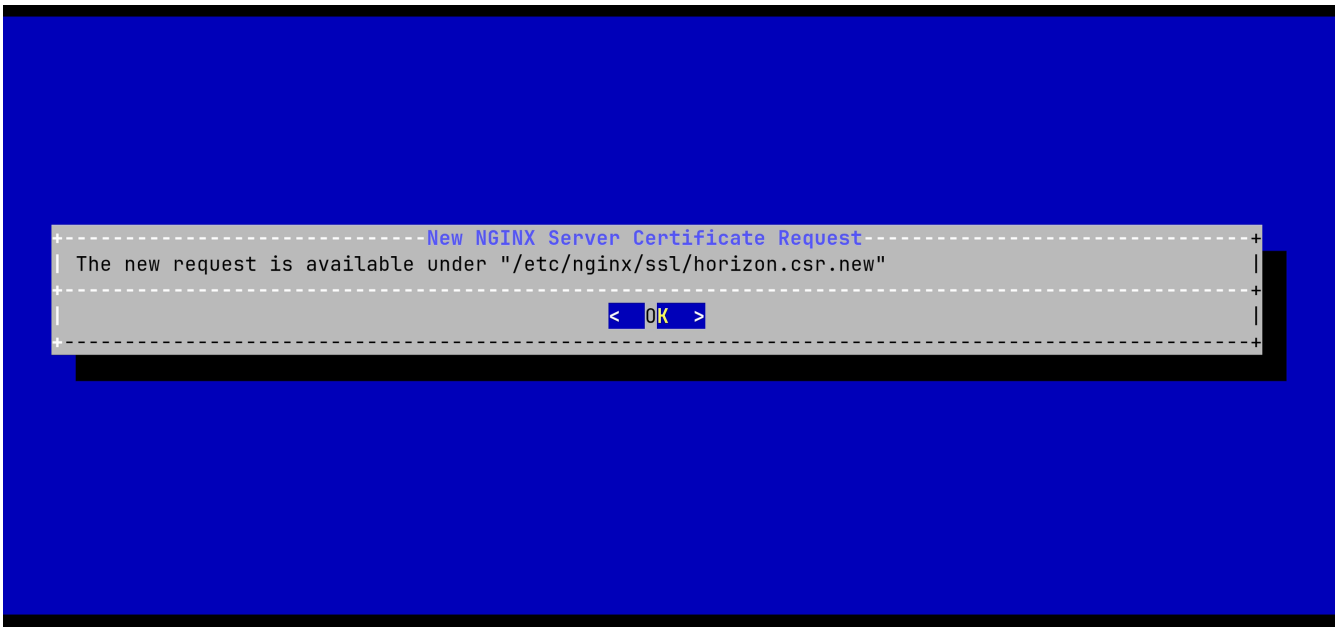
In the NGINX menu, select '**CSR**':



Specify the DNS Name of the Horizon server (by default, the config script takes the Horizon hostname if defined or the local machine hostname otherwise):



The certificate request is generated and available under `/etc/nginx/ssl/horizon.csr.new`:

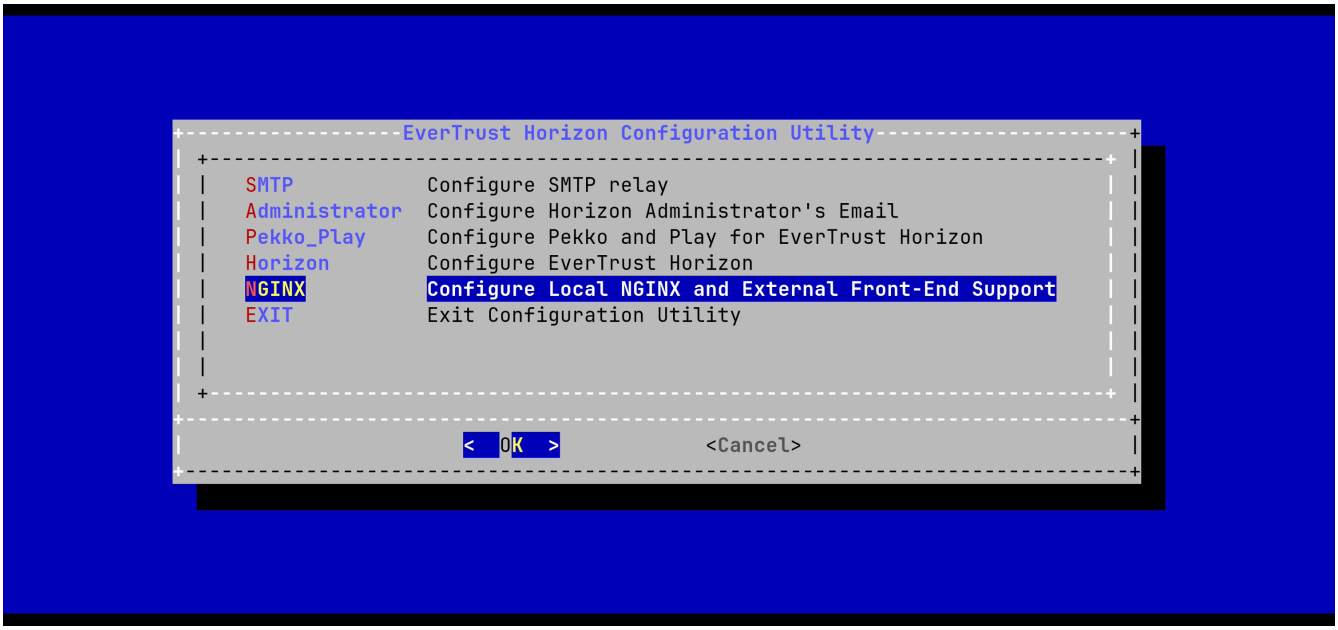


Sign the certificate request using your PKI.

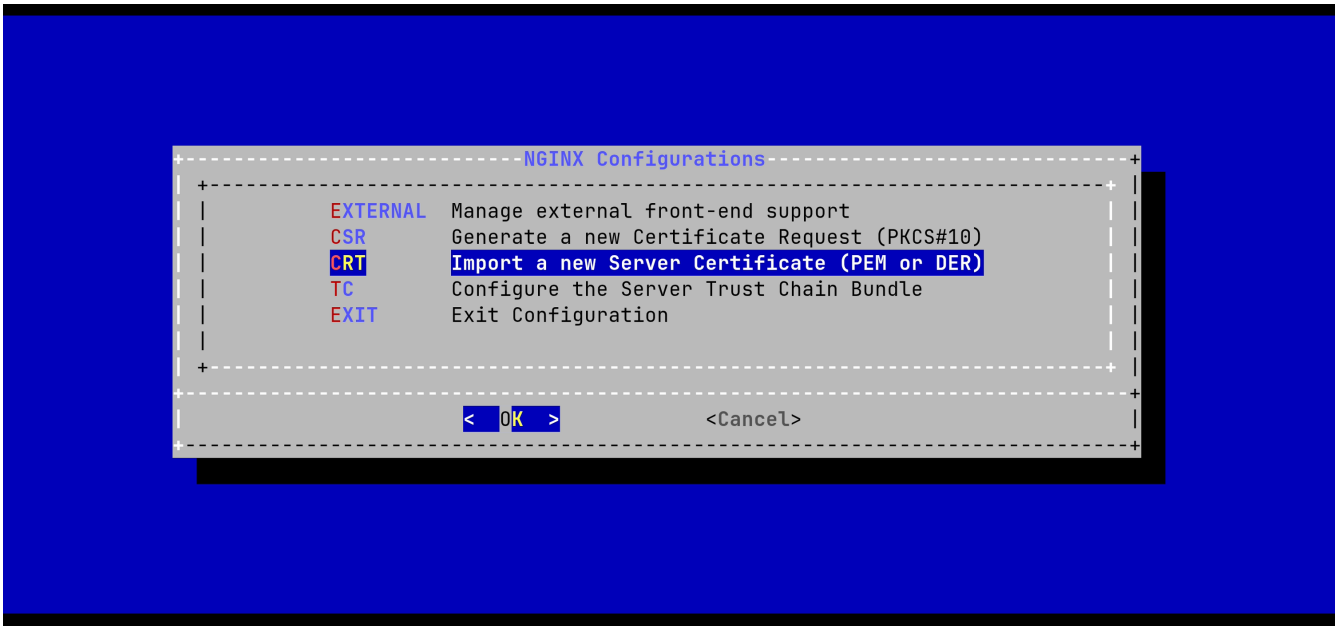
Installing a Server Certificate

Upload the generated server certificate on the Horizon server under `/tmp/horizon.pem` through SCP;

In the main menu, select '**NGINX**':



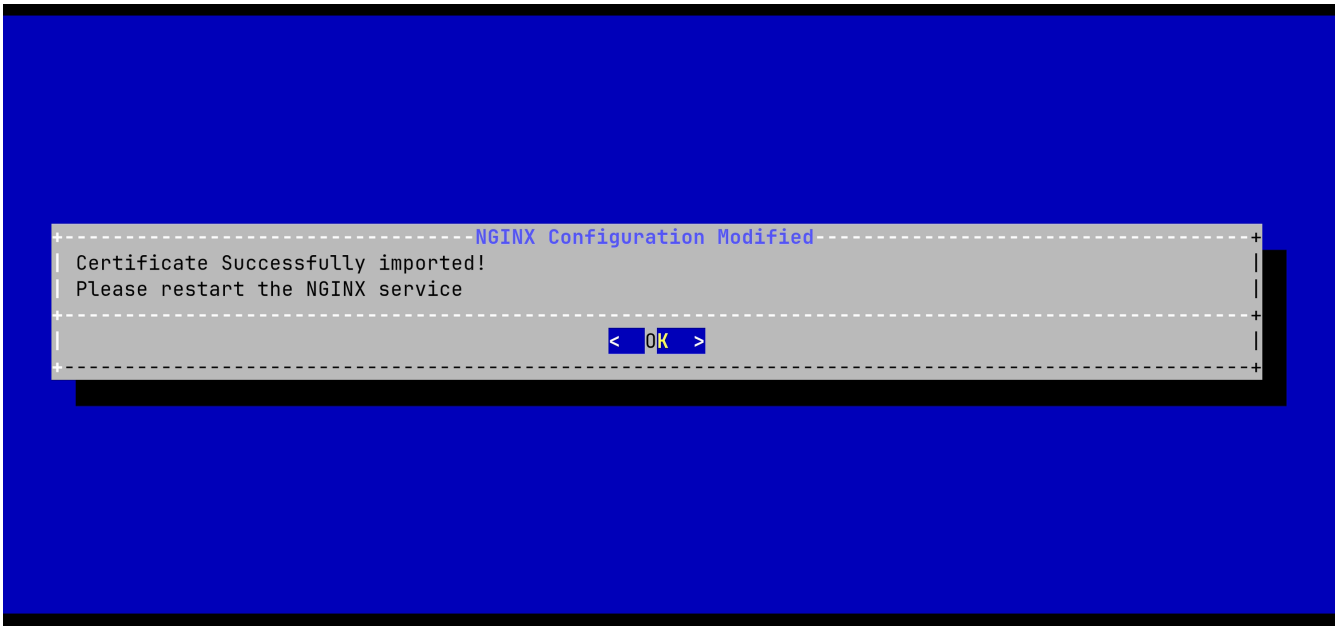
In the NGINX configuration menu, select '**CRT**':



Specify the path `/tmp/horizon.pem` and validate:


Specify the path of the new server certificate:
/tmp/horizon.pem
< OK >
<Cancel>

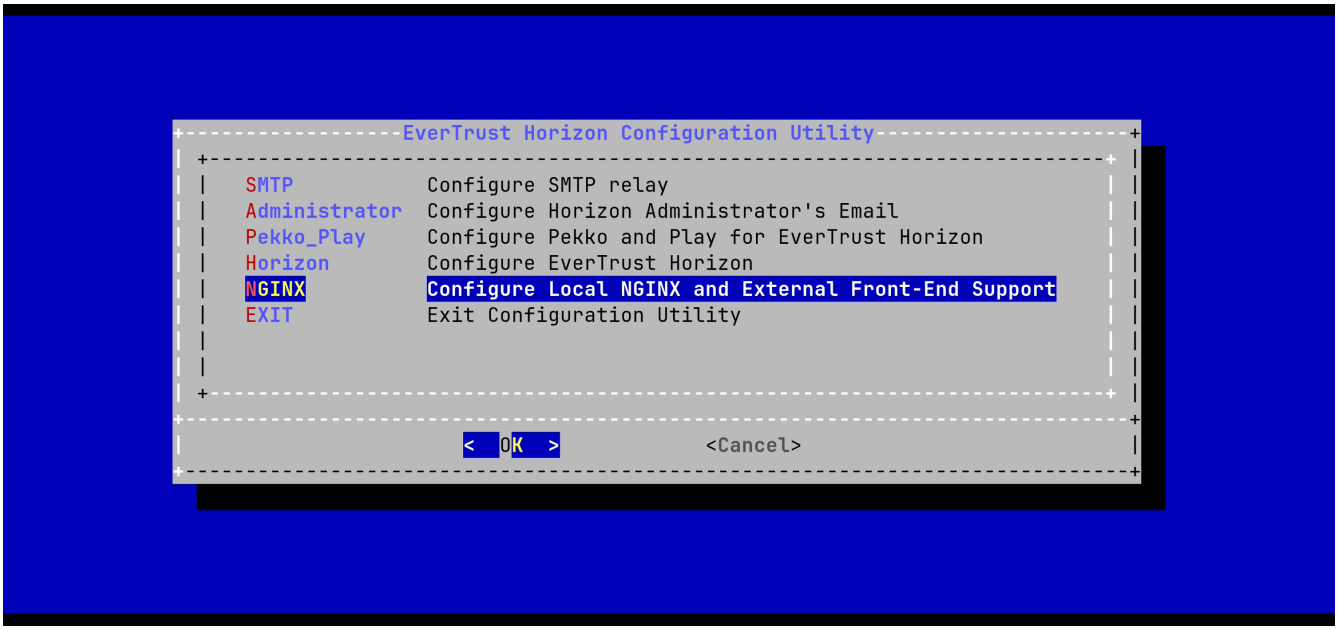
The server certificate is successfully installed:



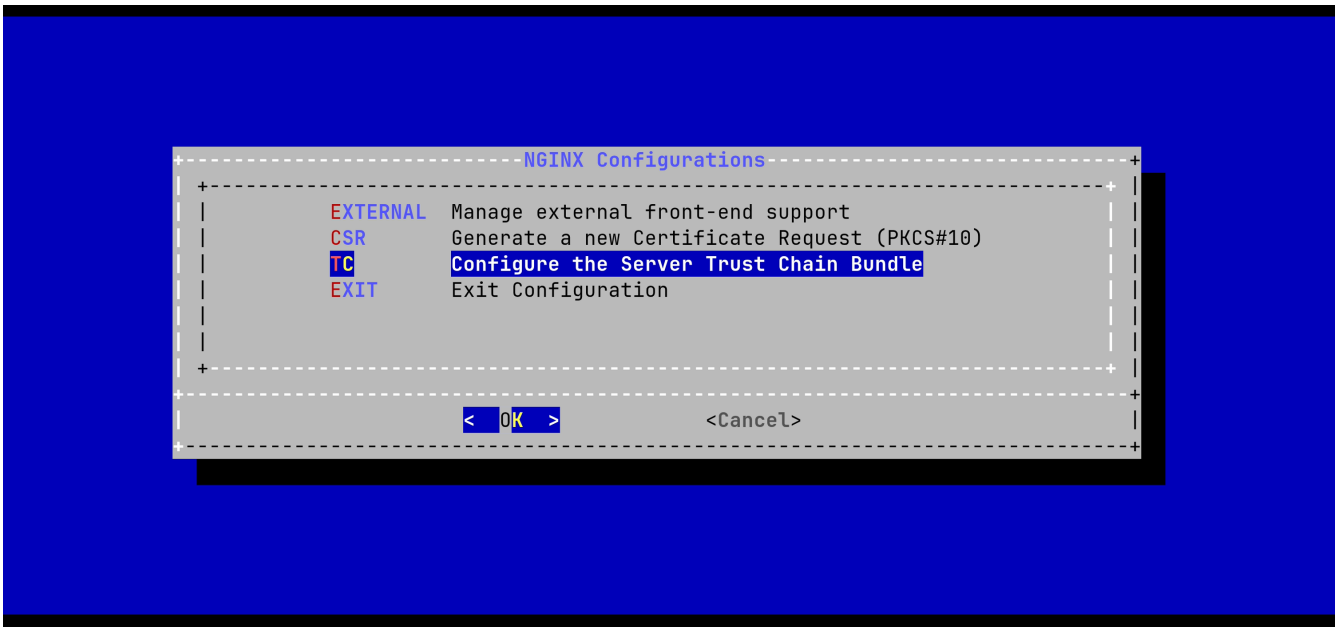
Installing the Server Certificate Trust Chain

Upload the server certificate trust chain (the concatenation of the Certificate Authority certificates in PEM format) on the Horizon server under `/tmp/server.bundle` through SCP;

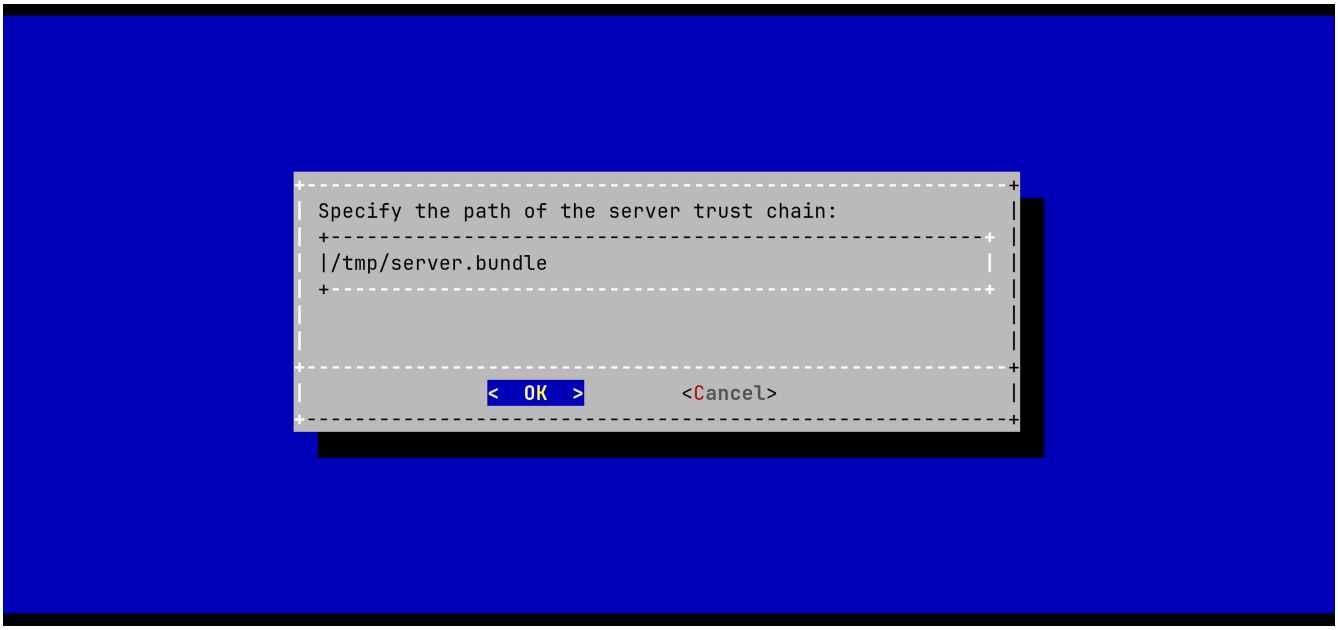
In the main menu, select '**NGINX**':



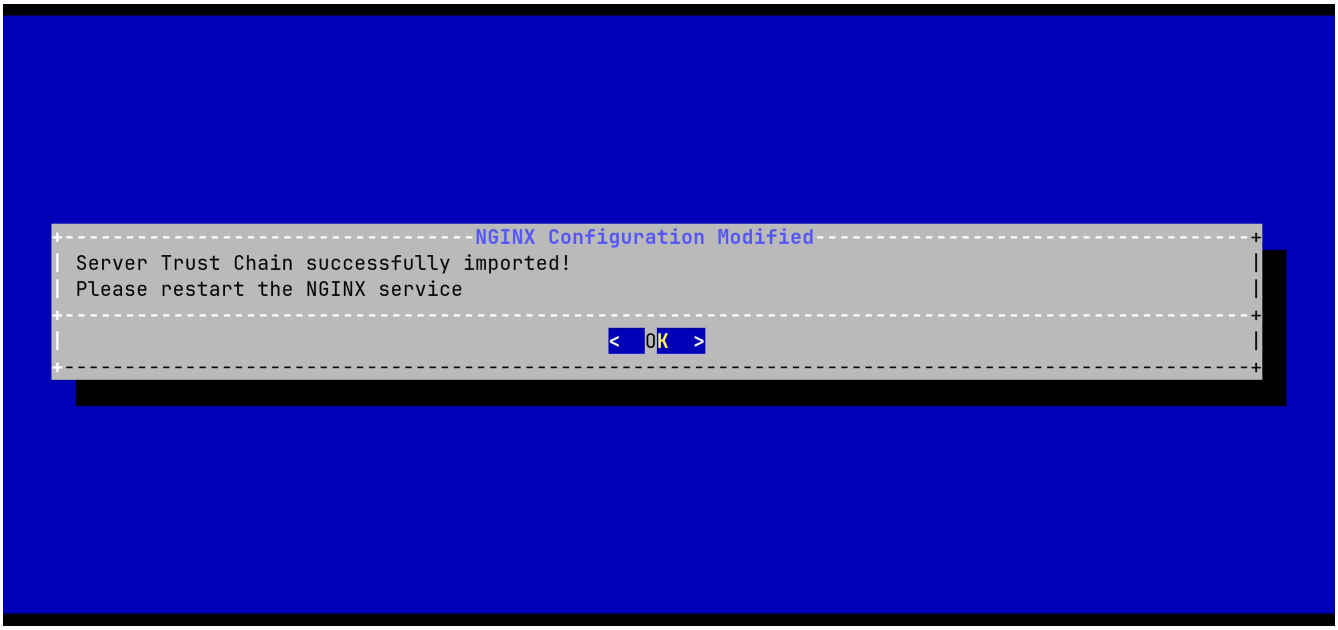
In the NGINX configuration menu, select '**TC**':



Specify the path `/tmp/server.bundle` and validate:


Specify the path of the server trust chain:
/tmp/server.bundle
< OK >
<Cancel>

The server bundle is successfully installed:



Verify the NGINX configuration with the following command:

```
$ nginx -t
```

Restart the NGINX service with the following command:

```
$ systemctl restart nginx
```

1.1.4. Startup & login

Starting the Horizon services

1. Access the server through SSH with an account with administrative privileges;
2. Start the horizon service with the following command:

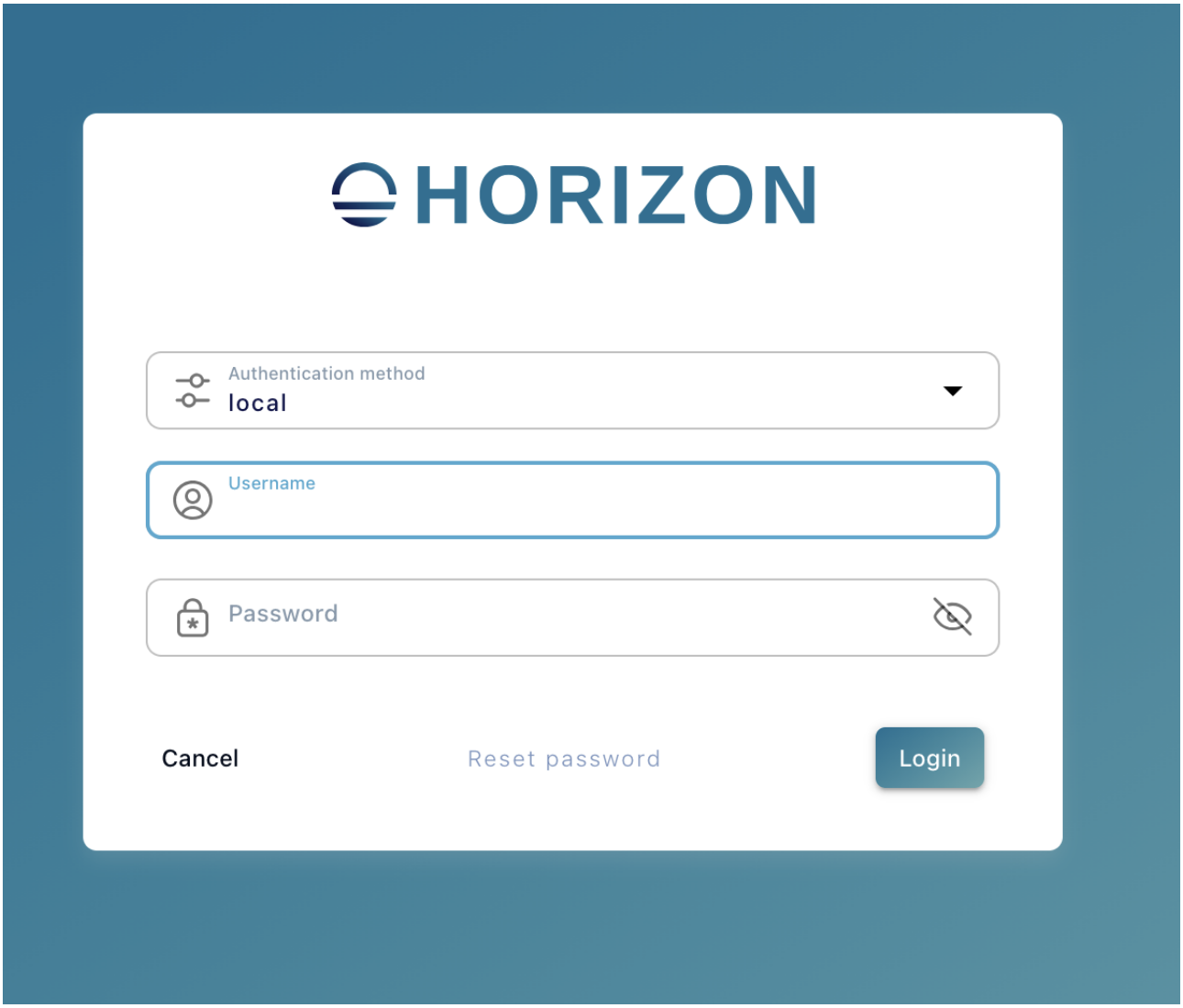
   ```
   $ systemctl start horizon
   ```

3. Start the nginx service with the following command:

   ```
   $ systemctl start nginx
   ```

Accessing the web UI

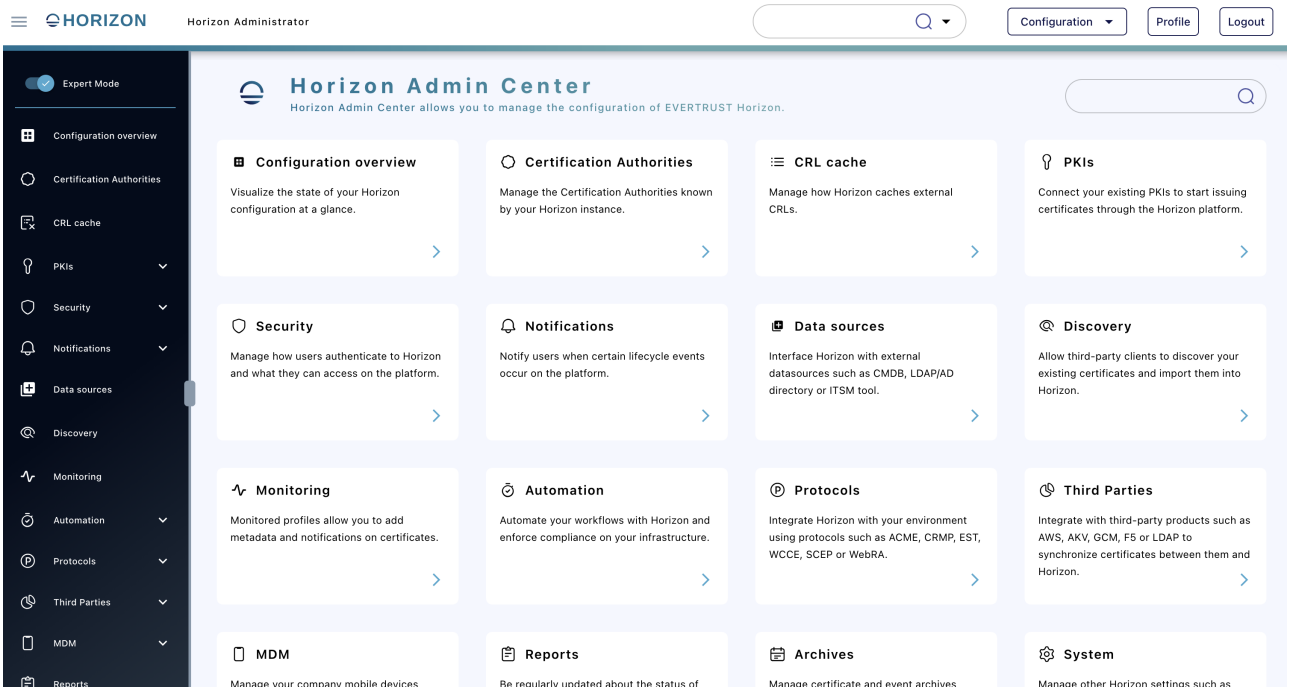
1. Launch a web browser;
2. Browse to `https://[Horizon IP or FQDN]`:





Upon first boot, a random administrator password will be generated. To retrieve it, open the `/opt/horizon/var/run/adminPassword` file. The default administration login is `administrator`.

3. Specify the default administration credentials and hit the '**Login**' button:





It is **highly recommended** to create a dedicated administration account and delete the default one, or at least modify the default administrator password.

1.1.5. Backup and Restore

This section details how to back-up and restore Horizon. Back-up and restore operation can be performed using the back-up and restore tool available under `/opt/horizon/sbin/horizon-backup`. It is designed to be used only in Linux-Based deployments.

For Docker or Kubernetes based deployments, the configuration should be managed by the Docker/Kubernetes management platform, and the database should be backed-up using MongoDB tools.

Backup Procedure

This section details how to back up Horizon configuration elements.

Several elements can be backed up:

- The Horizon configuration files.
- The Horizon MongoDB.

The backup tool allows backing up these elements independently.

```
$ /opt/horizon/sbin/horizon-backup --help
 usage: horizon-backup [-cdho:qs]
    -c | --conf          Backup the configuration files
    -d | --db            Backup the MongoDB database
    -h | --help          Display the 'horizon-backup' help
    -o | --output [path] Specify the backup output folder (default:
'/opt/horizon/var/backup')
```

```
    -q | --quiet         Quiet mode
```

To back up the configuration files, run the following command:

```
$ /opt/horizon/sbin/horizon-backup -c
```

The configuration files backup consists of a compressed archive (`.tar.gz`) located under `/opt/horizon/var/backup/`.

To back up the MongoDB database, run the following command:

```
$ /opt/horizon/sbin/horizon-backup -d
```

The MongoDB database backup consists of a compress file (`.gz`) located under `/opt/horizon/var/backup/`.

To run a complete backup, execute the following command:

```
$ /opt/horizon/sbin/horizon-backup -c -d
```



- The backup output folder can be overridden using the `-o | --output` parameter
- The backup tool can operate in quiet mode (when scheduled in a cron job) using the `-q | --quiet` parameter

Restoration Procedure

This section details how to restore horizon configuration elements.



This restore procedure only applies to the exact same application version as the backup file.

Restoration operation should be performed while the Horizon service is not running. Stop the Horizon service with the following command:

```
$ systemctl stop horizon
```

To restore a configuration backup, run the following command:

```
$ tar xzpvf [horizon configuration backup archive path] -C/
```

To restore the MongoDB database, run the following command:

```
$ mongorestore --uri="[MongoDB URI]" --drop --gzip --archive=[horizon MongoDB backup
archive path]
```



The MongoDB URI can be retrieved from the `/etc/default/horizon/_*` configuration file, as `MONGODB_URI` parameter.

The Horizon service can now be started with the following command:

```
$ systemctl start horizon
```

1.2. Installing on Kubernetes

1.2.1. Installation

Concepts overview

In Kubernetes, applications are deployed onto **Pods**, which represents a running version of a containerized application.Pods are grouped by **Deployments**, which represent a set of Pods running the same application. For instance, should you need to run Horizon in high availability mode, your deployment will contain 3 pods or more.Applications running in Pods are made accessible by a **Service**, which grants a set of Pods an IP address (which can either be internal to the cluster or accessible on the public Internet through a Load Balancer).

The recommended way of installing on Horizon is through the Horizon's Helm Chart. Helm is a package manager for Kubernetes that will generate Kubernetes resources necessary to deploy

Horizon onto your cluster. The official Helm Chart will generate a deployment of one or more Pods running Horizon on your cluster.

Prerequisites

Before you start, make sure you have the following prerequisites:

- The `kubectl` command line tool installed and configured to access the destination cluster : installation.
- The `helm` command line tool installed and configured to access the destination cluster: installation.
- A working knowledge of Kubernetes and Helm. If you are new to Kubernetes, we recommend you read the Kubernetes Basics tutorial. If you are new to Helm, we recommend you read the Helm Quickstart tutorial.
- A cluster that can pull images from the EVERTRUST container registry. If this is not possible through Internet, see Running behind a container registry proxy for more information on how to set up a private registry mirror.
- A license file for your Horizon installation. This file is usually named `horizon.lic` and should have been provided to you by EVERTRUST.
- A set of credentials to access the EVERTRUST container repository. You should have received them from EVERTRUST.

Configuring the namespace

For isolation purposes, we strongly recommend that you create a dedicated namespace for **Horizon**:

```
$ kubectl create namespace horizon
```

The namespace should be empty. In order to run Horizon, you'll need to create two secrets in that namespace:

- A license secret containing your Horizon license file
- An image pull secret, allowing Kubernetes to authenticate to the EVERTRUST's container repository

Creating the application secrets

Licence

You should have a license file for your Horizon installation, most probably named `horizon.lic`. To convert this file to a Kubernetes secret, run:

```
$ kubectl create secret generic horizon-license \
 --from-file=license="<path to your license file>" \
```

```
 --namespace horizon
```

Keyset

Horizon requires a keyset in order to work. It replaces the SSV/SHV secret that were used in the previous versions. Follow the steps in the Tinkey page to setup Tinkey, create your Keyset, and make it available to Horizon.

Creating the image pull secret

Next, you should configure Kubernetes to authenticate to the EVERTRUST repository using your credentials. They are necessary to pull the Horizon docker image, you should have received them upon purchase. Get your username and password and create the secret:

```
$ kubectl create secret docker-registry evertrust-registry \
 --docker-server=registry.evertrust.io \
 --docker-username="<your username>" \
 --docker-password="<your password>" \
 --namespace horizon
```

Setting up Helm repository

Now that the application secrets are configured, add the **EVERTRUST Helm repository** to your machine:

```
$ helm repo add evertrust https://repo.evertrust.io/repository/charts
```

Verify that you have access to the Chart:

```
$ helm search repo evertrust/horizon
NAME                        CHART VERSION   APP VERSION DESCRIPTION
evertrust/horizon           2.2.2           2.12.0          EverTrust Horizon Helm
chart
```

Configuring the chart

You'll next need to override the defaults `values.yaml` file of the Helm Chart to reference the secrets that we've created. We'll provide a minimal configuration for demonstration purposes, but please do follow our production setup guide before deploying for production.

Create a `override-values.yaml` file somewhere and paste this into the file:

```
image:
  pullSecrets:
    - evertrust-registry
```

```
license:
  secretName: horizon-license
  secretKey: license
```



If your environment is subject to FIPS constraints, use the FIPS flavor of the Horizon image and of the Horizon Migration Tool image (used by the upgrade jobs), whose tags are suffixed with `-fips`. Add the following to your `override-values.yaml` file:

```
image:
  flavor: fips ①

upgrade:
  image:
    tag: <migration tool version>-fips ②
```

① The chart appends `-fips` to the Horizon image tag, for instance `registry.evertrust.io/horizon:2.12.0-fips`.

② The tag of the Horizon Migration Tool image, for instance `registry.evertrust.io/horizon-migration:<version>-fips`.

To finish Horizon's installation, simply run the following command:

```
$ helm install horizon evertrust/horizon -f override-values.yaml -n horizon
```

Please allow a few minutes for the Horizon instance to boot up. You are now ready to go on with the Startup & Login. This instance will allow you to test out if Horizon is working correctly on your cluster. However, this installation is not production-ready. Follow our Production Checklist to make sure your instance is fit to run in your production environment.

1.2.2. Production checklist

Even though the Helm Chart makes installing Horizon a breeze, you'll still have to set up a few things to make Horizon resilient enough to operate in a production environment.

Operating the database

All persistent data used by Horizon is stored in the underlying MongoDB database. Therefore, the database should be operated securely and backed up regularly.

When installing the chart, you face multiple options regarding your database:

- By default, a temporary MongoDB instance will be spawned in your cluster, using the `mongo` image. No additional configuration is required but it is not production ready out of the box.
- If you want to use an existing MongoDB instance, provide the `externalDatabase.uri` value. The

URI should be treated as a secret as it must include credentials:

```
externalDatabase:
  uri:
    valueFrom:
      secretKeyRef:
        name: <secret name>
        key: <secret key>
```

The chart doesn't manage the database. You are still in charge of making sure that the database is correctly backed up. You could either back up manually using `mongodump` or use a managed service such as MongoDB Atlas, which will take care of the backups for you.

Managing secrets

Storing secrets is a crucial part of your Horizon installation. On cloud-native installations like on Kubernetes, we recommend using SSV (Secure Software Vault) to encrypt sensitive data: a master passphrase will be used to encrypt and decrypt data before they enter the database. Alongside with other application secrets like your MongoDB URI (containing your credentials or certificate). We recommend that you create Kubernetes secrets beforehand or inject them directly into the pod.

Values that should be treated as secrets in this chart are:

Name	Description	Impact on loss
`vaults.*.master_password`	SSV password used to encrypt sensitive data in database.	Highest impact: database would be unusable
`events.secret`	Secret used to sign and chain events.	Moderate impact: events integrity would be unverifiable
`externalDatabase.uri`	External database URI, containing a username and password.	Low impact: reset the MongoDB password
`appSecret`	Application secret use to encrypt session data.	Low impact: sessions would be reset
`mailer.password`	SMTP server password	Low impact: reset the SMTP password

For each of these values, either:

- leave the field empty, so that a secret will be automatically generated.
- derive the secret value from an existing Kubernetes secret:

```
appSecret:
  valueFrom:
    secretKeyRef:
      name: <secret name>
```

```
      key: <secret key>
```



Always store auto-generated secrets in a safe place after they're generated. If you ever uninstall your Helm chart, the deletion of the SSV secret will lead to the impossibility of recovering most of your data.

High availability

By default, the chart will configure a single-pod deployment. This deployment method is fine for testing but not ready for production as a single failure could take down the entire application. Instead, we recommend that you set up a Horizon cluster using at least 3 pods.

In order to do that, configure an `horizontalAutoscaler` in your `override-values.yaml` file:

```
horizontalAutoscaler:
  enabled: true
  minReplicas: 3
  maxReplicas: 3
```



Use `nodeAffinity` to spread your Horizon cluster Pods among multiple nodes in different availability zones to reduce the risk of Single Point of Failure.

If your cluster setup requires specific configurations (that could be due to network or configuration constraints), we encourage you to check out the Networking overview section of the documentation.

Configuring ingresses

The recommended way to access Horizon is behind a reverse proxy, known in the Kubernetes world as "ingress controllers". However, Horizon requires that the reverse proxy in front of it (that also terminates the TLS connection) requests certificate client authentication (also known as mTLS).

To create an ingress upon installation, simply set the following keys in your `override-values.yaml` file:

```
ingress:
  enabled: true
  hostname: horizon.lab
  tls: true
```

Identify CAs which will require certificate authentication

You'll need to gather a list of CAs that will emit certificates which will be able to authenticate to Horizon. To identify them, ask yourself whether the certificates signed by these CAs will:

- renew using the EST protocol (used by the Horizon Client)

- be used to authenticate users to Horizon (either through API or via the UI)
- be used to authenticate the WinHorizon component (in an Active Directory environment)

Other use-cases might also require you to authenticate with a client certificate.

Configure your ingress to require a client certificate

Configuration for mTLS depends on the ingress controller that you use. The following ingress controllers are officially supported by EVERTRUST, and we strongly advise to use one of them with Horizon. However, almost any ingress controller can be configured to correctly request client certificates manually.

ingress-nginx

The Horizon Helm Chart supports autoconfiguring `ingress-nginx`. To enable client certificate authentication, simply set the following values in the `values-override.yaml` file:

```
ingress:
  enabled: true
  type: nginx
  clientCertificateAuth: true
  hostname: horizon.lab
  tls: true
```

Skip to the Ensure certificate authentication is effective section to test your configuration.



`ingress-nginx` doesn't require a list of CAs trusted for client authentication, so any certificate may be submitted by a connecting client. If you wish to specify a list of CAs, disable autoconfiguration and manually configure your ingress using annotations following the ingress-nginx documentation.

Traefik

The Horizon Helm Chart supports autoconfiguring Traefik. To enable client certificate authentication, simply set the following values in the `values-override.yaml` file:

```
ingress:
  enabled: true
  type: traefik
  clientCertificateAuth: true
  hostname: horizon.lab
  tls: true
```

Skip to the Ensure certificate authentication is effective section to test your configuration.



Traefik doesn't require a list of CAs trusted for client authentication, so any certificate may be submitted by a connecting client. If you wish to specify a list of

CAs, disable autoconfiguration and manually configure your ingress using annotations following the Traefik documentation.

Other ingress controllers

If you do not wish or cannot use autoconfiguration, you should ensure your ingress controller is correctly configured to enable all Horizon features.

- When requiring client certificates for authentication, the web server should not perform checks to validate that the certificate is signed by a trusted CA. Instead, the certificate should be sent to Horizon through a request header, base64-encoded. The header name used can be controlled using the `clientCertificateHeader`.
- Some endpoints should not be server over HTTPS, in particular those used for SCEP enrollment. You may want to create an HTTP-only ingress for serving paths prefixed by `/scep` and `/certsrv`, and prevent those from redirecting to HTTPS.



The cert-auth-proxy component, maintained by EverTrust, can be used to add client certificate authentication to any ingress controller which supports passthrough TLS.

Ensure certificate authentication is effective

To ensure that Horizon can properly decode certificates being sent by clients, get a certificate from a CA configured for client authentication in a `cert.pem` file and its associated key in a `key.pem` file.

Then, run the following `curl` command :

```
$ curl -k --cert cert.pem --key key.pem https://<Horizon
URL>/api/v1/security/principals/self
```

If Horizon returns an error, or states that the principal is not authenticated (through a 204 HTTP code), then certificate authentication is incorrectly configured.

Instead, information about the certificate should be returned in the principal key :

```
{
  "identity": {
    "identifier": "CN=User, O=EVERTRUST, C=FR", ①
    "name": "User",
    "identityProviderType": "X509", ②
    "identityProviderName": "EVERTRUST CA"
  },
  "permissions": [],
  "roles": null,
  "teams": null,
  "preferences": null,
  "customDashboards": null
```

```
}
```

① The DN of the certificate is used as the principal identifier.

② The identity provider is of type X509.

1.2.3. Startup & login

Accessing Horizon

Once the Horizon deployment is up and running, you can expose it to access the web UI and start configuring the instance.



By default, Horizon will expose a plain HTTP endpoint on port 9000 and an HTTPS endpoint on port 9443 (serving a self-signed certificate, unless configured otherwise).

Expose locally with a port forward

Recommended for testing and debugging, this is the fastest way to connect to your Horizon instance. The idea is to map a local port of your host computer to the remote port of the Horizon container.

To do so, run:

```
kubectl port-forward <horizon pod name> 9000:9000
```

Horizon will then be available on `http://localhost:9000`. A more in-depth tutorial on port forwarding can be found here.

Expose through an ingress controller

When an ingress controller is configured in your cluster, this is the proper way to access Horizon. To deploy an ingress alongside Horizon, set the `ingress.enabled` key to true in the Helm Chart's values override.

Logging in for the first time

Upon the first startup, an administrator account will be generated for you to log in. This account has the `administrator` username and a random password stored on disk, on the master Horizon pod.

To find out the randomly generated password, run:

```
kubectl exec $(kubectl get pods -n <namespace> -l "app.kubernetes.io/name=horizon"
--sort-by={.status.podIP} -o jsonpath="{.items[0].metadata.name}") -n <namespace> --
/bin/sh -c "cat /tmp/tmp.*/adminPassword"
```



It is **highly recommended** to create a dedicated administration account and delete the default one, or at least modify the default administrator password.

Set a default password

As mentioned above, when Horizon first boots in an empty database, it generates a random password for the default `administrator` account. If you wish to deterministically set the password for this account, you can do so by setting the `initialAdminPassword` key in the Helm Chart's values override, or set the `HRZ_ADMIN_PWD_HASH` environment variable.

Horizon expects a SHA-crypt hash, which can be generated using our toolbox image:

```
docker run --rm -it quay.io/evertrust/toolbox:latest generate-password sample_password
```

1.2.4. Advanced usage

Some edge use-cases might not have been included in the previous installation documentation, for clarity purposes. You may find some of them below.

Running behind a container registry proxy

If your installation environment requires you to whitelist images that can be pulled by the Kubernetes cluster, you must whitelist the `registry.evertrust.io/horizon` and `registry.evertrust.io/horizon-upgrade` images. It is then possible to override the images being pulled by setting the `global.imageRegistry` key in your `values.yaml` file to point to your private registry:

```
global:
  imageRegistry: <YOUR-PRIVATE-REGISTRY>
```

Leases

To ensure clustering issues get resolved as fast as possible, Horizon can use Kubernetes leases. We strongly recommend that you use this safety mechanism. However, the feature can be disabled by setting the `leases.enabled` key to `false`.

Trusting custom CAs

When your application needs to establish secure connections with services that use certificates signed by custom Certificate Authorities, you need to import these CA certificates into your system's trust store. This documentation shows how to accomplish this using Kubernetes ConfigMaps or Secrets.

1. Import Your Custom CA Certificate

Choose one of the following methods to store your CA certificate:

A. Using a ConfigMap

```
extraObjects:
  - apiVersion: v1
    kind: ConfigMap
    metadata:
      name: custom-ca-certificates
    data:
      company-internal-ca.pem: |
        -----BEGIN CERTIFICATE-----
        MIIGfjCCBGagAwIBAg...
        -----END CERTIFICATE-----
```

B. Using a Secret

```
extraObjects:
  - apiVersion: v1
    kind: Secret
    metadata:
      name: custom-ca-certificates
    data:
      company-internal-ca.pem: <YOUR_CA_CERTIFICATE_BASE64_ENCODED>
```



To encode your certificate in base64, use: `cat your-ca.pem | base64 -w 0`

2. Configure Your Application to Trust the CA Certificate

After creating the ConfigMap or Secret with your CA certificate, configure your application to use it by setting the appropriate environment variable:

A. Loading from ConfigMap

```
environment:
  - name: SYSTEM_CA_TRUST
    valueFrom:
      configMapKeyRef:
        name: custom-ca-certificates
        key: company-internal-ca.pem
```

B. Loading from Secret

```
environment:
  - name: SYSTEM_CA_TRUST
    valueFrom:
      secretKeyRef:
        name: custom-ca-certificates
```

```
        key: company-internal-ca.pem
```

Notes

- The `SYSTEM_CA_TRUST` environment variable is used by the application to add the provided certificate to the system's trusted certificate store.
- You can provide multiple CA certificates by concatenating multiple certificates under a single key (ensure each certificate begins with -----BEGIN CERTIFICATE----- and ends with -----END CERTIFICATE-----)

Example Complete Deployment using Horizon Helm Chart

```
environment:
  - name: SYSTEM_CA_TRUST
    valueFrom:
      configMapKeyRef:
        name: custom-ca-certificates
        key: company-internal-ca.pem

extraObjects:
  - apiVersion: v1
    kind: ConfigMap
    metadata:
      name: custom-ca-certificates
    data:
      company-internal-ca.pem: |
        -----BEGIN CERTIFICATE-----
        MIIGfjCCBGagAwIBAg...
        -----END CERTIFICATE-----
```

Custom startup scripts

Sometimes, you'll want to run scripts each time the container starts up in order to configure files in the container or set environment variables. To do so, you'll need to mount shell scripts into the `/docker-entrypoint.d/` directory in the container.

Networking overview

When installed in HA, Horizon sends messages to other running instances in its cluster. To form the cluster and set up networking between nodes, Horizon is relying on Pekko, a framework for building clusterized applications. Understanding how clustering works is important when building deployments with highly specific needs or when preparing a disaster recovery plan.

When deployed on multiple nodes inside a Kubernetes cluster, the following steps are followed:

1. **Discovery**: the discovery process locates all nodes that will be used to form a cluster. It relies on a third-party to give that information, such as a DNS record or the Kubernetes API (which is the

default when deploying using the Helm Chart). For documentation, see Pekko Discovery.

2. **Bootstrap**: once each node in the cluster has the address of every other node, nodes start to contact each other. This is done though Pekko Management, a tool for helping nodes coordinate. For documentation, see Pekko Management.
3. **Remoting**: the cluster is now formed, nodes can communicate with each other. This uses Pekko Remoting, a higher level protocol for serializing data over multiple transports. Typically, TCP is used. For documentation, see Pekko Remoting.

This clustering process can be summarized by the below diagram:

Sequence diagram of the cluster management of Horizon

```
sequenceDiagram
    autonumber
    rect rgb(191, 223, 255)
    Pod1 ->> Kubernetes API: Discovery request
    destroy Kubernetes API
    Kubernetes API ->> Pod1: Returns other pods addresses
    end
    Note right of Pod2: 1-2: Discovery process


    rect rgb(156, 250, 152)
    Pod1 ->> Pod2: Contact Pekko Management
    Pod2 ->> Pod1: Returns already contacted nodes

    break when an existing cluster is found
    Pod1 ->> Pod2: Joins the existing cluster
    end

    break when no existing cluster is found
    Pod1 ->> Pod1: Self-joins and create cluster
    Pod2 ->> Pod1: Joins the created cluster
    end
    end

    Note over Pod1,Pod2: Leader election is performed at this point

    Note right of Pod2: 3-7: Bootstrap process

    rect rgb(250, 148, 142)
    Pod1 ->> Pod2: Exchanges actor messages
    Pod2 ->> Pod1: Exchanges actor messages
    end

    Note right of Pod2: 8-9: Remoting
```

Traffic between different nodes is described in the below table:

Table 1. Traffic detail for Horizon clustering

Traffic type	Diagram color	Protocol	Port
Kubernetes API	Blue	HTTP	443
Pekko Management	Green	HTTP	7626 (by default)
Pekko Remote	Red	TCP (by default)	17355

1.2.5. Chart reference

The Helm chart aims to offer easy config settings through values for most used Kubernetes or Horizon features. If something you need is not yet covered, you can also use:

- `extraConfig` to directly edit Horizon config;
- `extraObjects` to add related Kubernetes resources.

The chart source is also freely available to investigate unwanted behaviors.

global

Global Docker image parameters. Please note that this will override the image parameters, including dependencies, configured to use the global value.

imageRegistry

Global Docker image registry

```
global.imageRegistry: ""
```

imagePullSecrets

Global Docker registry secret names as an array

```
global.imagePullSecrets: []
```

Example

```
imagePullSecrets:
  - myRegistryKeySecretName
```

kubeVersion

Force target Kubernetes version (using Helm capabilities if not set)

```
kubeVersion: ""
```

nameOverride

String to partially override horizon.fullname

```
nameOverride: ""
```

fullnameOverride

String to fully override horizon.fullname

```
fullnameOverride: ""
```

imageRegistry

String to override the image registry for all containers

```
imageRegistry: ""
```

commonLabels

Labels to add to all deployed objects

```
commonLabels: {}
```

commonAnnotations

Annotations to add to all deployed objects

```
commonAnnotations: {}
```

image

By default, we fetch the Horizon image from the Evertrust registry. If the tag is null or unset, the default value will be set to the chart appVersion. As the official Evertrust registry is not in open-access, you should specify an image pull secret that has access to Horizon images.

Kubernetes Reference

registry

Horizon image registry

```
image.registry: "registry.evertrust.io"
```

repository

Horizon image repository

```
image.repository: "horizon"
```

tag

Horizon image tag (immutable tags are recommended)

```
image.tag: "2.9.2"
```

flavor

Horizon image flavor (for HSM compatible images)

```
image.flavor: ""
```

pullPolicy

Horizon image pull policy

```
image.pullPolicy: "IfNotPresent"
```

pullSecrets

Horizon image pull secrets

```
image.pullSecrets: []
```

updateStrategy

Kubernetes Reference

Example

```
updateStrategy:
 type: RollingUpdate
 rollingUpdate:
   maxSurge: 25%
```

```
    maxUnavailable: 25%
```

type

Horizon deployment strategy type

```
updateStrategy.type: "RollingUpdate"
```

rollingUpdate

Rolling update spec

```
updateStrategy.rollingUpdate:
  maxUnavailable: 1
```

deploymentAnnotations

Annotations to add to the deployment object

```
deploymentAnnotations: {}
```

deploymentLabels

Annotations to add to the deployment object

```
deploymentLabels: {}
```

priorityClassName

Horizon pod priority class name

```
priorityClassName: ""
```

hostAliases

Horizon pod host aliases

Kubernetes Reference

```
hostAliases: []
```

extraVolumes

Optionally specify extra list of additional volumes for Horizon pods

```
extraVolumes: []
```

Example

```
extraVolumes:
  - name: extra-volume-name
    configMap:
      name: example-configmap
```

extraVolumeMounts

Optionally specify extra list of additional volumeMounts for Horizon container(s)

```
extraVolumeMounts: []
```

Example

```
extraVolumeMounts:
  - name: extra-volume-name
    mountPath: /mnt/extra-volume
```

sidecars

Add additional sidecar containers to the Horizon pod

```
sidecars: []
```

Example

```
sidecars:
  - name: your-image-name
    image: your-image
    imagePullPolicy: Always
    ports:
      - name: portname
        containerPort: 1234
```

initContainers

Add additional init containers to the Horizon pod

```
initContainers: []
```

lifecycleHooks

Add lifecycle hooks to the Horizon deployment

```
lifecycleHooks: {}
```

podLabels

Extra labels for Horizon pods

Kubernetes Reference

```
podLabels: {}
```

podAnnotations

Annotations for Horizon pods

Kubernetes Reference

```
podAnnotations: {}
```

podAffinityPreset

Pod affinity preset. Ignored if `affinity` is set. Allowed values: `soft` or `hard`

Kubernetes Reference

```
podAffinityPreset: ""
```

podAntiAffinityPreset

Pod anti-affinity preset. Ignored if `affinity` is set. Allowed values: `soft` or `hard`

Kubernetes Reference

```
podAntiAffinityPreset: "soft"
```

nodeAffinityPreset

Node affinity preset

Kubernetes Reference

type

Node affinity preset type. Ignored if `affinity` is set. Allowed values: `soft` or `hard`

```
nodeAffinityPreset.type: ""
```

key

Node label key to match. Ignored if `affinity` is set

```
nodeAffinityPreset.key: ""
```

values

Node label values to match. Ignored if `affinity` is set

```
nodeAffinityPreset.values: []
```

Example

```
values:
  - e2e-az1
  - e2e-az2
```

revisionHistoryLimit

Number of controller revisions to keep

```
revisionHistoryLimit: 3
```

replicas

Replica count when no autoscaler is configured

```
replicas: 1
```

affinity

Affinity for pod assignment



podAffinityPreset, podAntiAffinityPreset, and nodeAffinityPreset will be ignored when it's set.

Kubernetes Reference

```
affinity: {}
```

nodeSelector

Node labels for pod assignment

Kubernetes Reference

```
nodeSelector: {}
```

tolerations

Tolerations for pod assignment

Kubernetes Reference

```
tolerations: []
```

topologySpreadConstraints

Spread Constraints for pod assignment

Kubernetes Reference

```
topologySpreadConstraints: []
```

Example

```
topologySpreadConstraints:
  - maxSkew: 1
    topologyKey: node
    whenUnsatisfiable: DoNotSchedule
```

resources

Horizon containers' resource requests and limits. The JVM will automatically adapt the memory allocation pool to the container allocated resources.

Kubernetes Reference

limits

The resources limits for the Horizon container

```
resources.limits: {}
```

requests

The requested resources for the Horizon container

```
resources.requests:
  memory: 512Mi
  cpu: 300m
```

podSecurityContext

Configure Pods Security Context

Kubernetes Reference

enabled

Enabled Horizon pods' Security Context

```
podSecurityContext.enabled: true
```

fsGroup

Set Horizon pod’s Security Context fsGroup

```
podSecurityContext.fsGroup: 1001
```

containerSecurityContext

Configure Container Security Context (only main container)

Kubernetes Reference

enabled

Enabled Horizon containers' Security Context

```
containerSecurityContext.enabled: true
```

runAsUser

Set Horizon container’s Security Context runAsUser

```
containerSecurityContext.runAsUser: 1001
```

runAsNonRoot

Set Horizon container’s Security Context runAsNonRoot

```
containerSecurityContext.runAsNonRoot: true
```

livenessProbe

Configure extra options for Horizon containers' liveness probe.

Kubernetes Reference

enabled

Enable livenessProbe

```
livenessProbe.enabled: true
```

initialDelaySeconds

Initial delay seconds for livenessProbe

```
livenessProbe.initialDelaySeconds: 0
```

periodSeconds

Period seconds for livenessProbe

```
livenessProbe.periodSeconds: 10
```

timeoutSeconds

Timeout seconds for livenessProbe

```
livenessProbe.timeoutSeconds: 5
```

successThreshold

Success threshold for livenessProbe

```
livenessProbe.successThreshold: 1
```

failureThreshold

Failure threshold for livenessProbe

```
livenessProbe.failureThreshold: 3
```

startupProbe

A startup probe allows us to define a shorter period to improve Horizon time-to-liveliness time while preserving the Horizon pod from a restart loop when it is slow to start.

Kubernetes Reference

enabled

Enable startupProbe. Since Horizon is slow to start, this is highly recommended.

```
startupProbe.enabled: true
```

failureThreshold

Failure threshold for startupProbe

```
startupProbe.failureThreshold: 60
```

periodSeconds

Period seconds for startupProbe

```
startupProbe.periodSeconds: 3
```

readinessProbe

Kubernetes Reference

enabled

Enable readinessProbe

```
readinessProbe.enabled: true
```

initialDelaySeconds

Initial delay seconds for readinessProbe

```
readinessProbe.initialDelaySeconds: 0
```

periodSeconds

Period seconds for readinessProbe

```
readinessProbe.periodSeconds: 5
```

timeoutSeconds

Timeout seconds for readinessProbe

```
readinessProbe.timeoutSeconds: 3
```

successThreshold

Success threshold for readinessProbe

```
readinessProbe.successThreshold: 1
```

failureThreshold

Failure threshold for readinessProbe

```
readinessProbe.failureThreshold: 3
```

horizontalAutoscaler

Kubernetes Reference

enabled

Enable Horizontal POD autoscaling for Horizon

```
horizontalAutoscaler.enabled: false
```

minReplicas

Minimum number of Horizon replicas

```
horizontalAutoscaler.minReplicas: 1
```

maxReplicas

Maximum number of Horizon replicas

```
horizontalAutoscaler.maxReplicas: 3
```

targetCPU

Target CPU utilization percentage

```
horizontalAutoscaler.targetCPU: 50
```

targetMemory

Target Memory utilization percentage

```
horizontalAutoscaler.targetMemory: 50
```

disruptionBudget

Kubernetes Reference

enabled

Created a PodDisruptionBudget

```
disruptionBudget.enabled: false
```

minAvailable

Min number of pods that must still be available after the eviction

```
disruptionBudget.minAvailable: 1
```

maxUnavailable

Max number of pods that can be unavailable after the eviction

```
disruptionBudget.maxUnavailable: 0
```

environment

Configure environment variable injections into Horizon's pods. This is the way you should inject secrets into the app if you wish to use the Kubernetes secrets implementation.

Kubernetes Reference

```
environment: []
```

Example

```
environment:
  - name: KEY
    value: VALUE
```

dnsConfig

This value is useful if you need to resolve your custom domain for ACME challenges.

Kubernetes Reference

```
dnsConfig: {}
```

Example

```
nameservers:
  - 1.2.3.4
searches:
  - ns1.svc.cluster-domain.example
  - my.dns.search.suffix
options:
  - name: ndots
```

```
    value: "2"
```

dnsPolicy

Kubernetes Reference

```
dnsPolicy: ""
```

service

Service configuration

type

Kubernetes service type

```
service.type: "ClusterIP"
```

clusterIP

Horizon service clusterIP IP

```
service.clusterIP: ""
```

Example

```
clusterIP: None
```

loadBalancerIP

Load balancer IP for the Horizon Service (optional, cloud specific)

Kubernetes Reference

```
service.loadBalancerIP: ""
```

loadBalancerSourceRanges

Address that are allowed when service is LoadBalancer

Kubernetes Reference

```
service.loadBalancerSourceRanges: []
```

Example

```
loadBalancerSourceRanges:
  - 10.10.10.0/24
```

externalTrafficPolicy

Enable client source IP preservation

Kubernetes Reference

```
service.externalTrafficPolicy: "Cluster"
```

extraPorts

Extra port to expose on Horizon service

```
service.extraPorts: []
```

annotations

Annotations for Horizon service

```
service.annotations: {}
```

ingress

Ingress configuration

Kubernetes Reference

enabled

Set to true to enable ingress record generation

```
ingress.enabled: false
```

type

Ingress type

Automatically configure your ingress for an ingress controller. Accepted values are nginx, traefik. This will override the clientCertificateHeader if set, and generate annotations, resources, and ingresses resources to ensure Horizon works correctly.

```
ingress.type: ""
```

clientCertificateAuth

Client certificate authentication

When ingress.type is set, determines whether the ingress controller should request client certificates.

```
ingress.clientCertificateAuth: false
```

clientCertificateCASecrets

Client certificate CA secrets

If set, the ingress controller will only request client certificates signed by these CAs. Each secret should contain a `ca.crt` key containing the PEM-encoded AC certificate.

```
ingress.clientCertificateCASecrets: []
```

scepCompatibilityMode

SCEP compatibility mode

Adds a secondary ingress for SCEP support over HTTP.

```
ingress.scepCompatibilityMode: false
```

ingressClassName

IngressClass that will be used to implement the Ingress (Kubernetes 1.18+)

```
ingress.ingressClassName: ""
```

hostname

Default host for the ingress resource

```
ingress.hostname: ""
```

Example

```
hostname: "horizon.local"
```

path

Default path for the ingress record



You may need to set this to '/*' in order to use this with ALB ingress controllers.

```
ingress.path: "/"
```

pathType

Ingress path type

```
ingress.pathType: "Prefix"
```

annotations

Additional annotations for the Ingress resource

To enable certificate autogeneration, place here your cert-manager annotations.

```
ingress.annotations: {}
```

Example

```
annotations:
  cert-manager.io/cluster-issuer: cluster-issuer-name
```

tls

Enable TLS configuration for the hostname defined at ingress.hostname parameter

TLS certificates will be retrieved from a TLS secret with name: {{- printf "%s-tls" .Values.ingress.hostname }} You can use the ingress.secrets parameter to create this TLS secret, relay on cert-manager to create it, or let the chart create self-signed certificates for you.

```
ingress.tls: false
```

extraHosts

The list of additional hostnames to be covered with this ingress record

Most likely the hostname above will be enough, but in the event more hosts are needed, this is an array.

```
ingress.extraHosts: []
```

Example

```
extraHosts:
- name: horizon.local
  path: /
```

extraPaths

An array with additional arbitrary paths that may need to be added to the ingress under the main host

```
ingress.extraPaths: []
```

Example

```
extraPaths:
- path: /*
  backend:
    serviceName: ssl-redirect
    servicePort: use-annotation
```

extraTls

The tls configuration for additional hostnames to be covered with this ingress record

Kubernetes Reference

```
ingress.extraTls: []
```

Example

```
extraTls:
- hosts:
    - horizon.local
  secretName: horizon.local-tls
```

extraRules

Additional rules to be covered with this ingress record

Kubernetes Reference

```
ingress.extraRules: []
```

Example

```
extraRules:
- host: horizon.local
    http:
      path: /
      backend:
        service:
          name: horizon
          port:
            name: http
```

monitoring

Prometheus monitor configuration

enabled

Enable the creation of a ServiceMonitor object for Horizon if the cluster has the monitoring.coreos.com/v1 capability

```
monitoring.enabled: true
```

appSecret

Configure the Play secret for the Horizon instance. As this is used for cryptographic purposes, it should be fetched from the environment.

Play Framework Application Secret

```
appSecret: {}
```

Example

```
appSecret:
  valueFrom:
    secretKeyRef:
      name: horizon-secret
      key: appSecret
```

license

A valid Horizon license is required for the software to run. You should store it (base64-encoded) in a Kubernetes secret and specify the secret details here.

README.md[README.md]

secretName

Existing secret name where the Horizon license is stored

```
license.secretName: ""
```

secretKey

Existing secret key where the Horizon license is stored

```
license.secretKey: ""
```

initialAdminHashPassword

Set up initial admin user.

enabled

Whether to enable the initial admin user

```
initialAdminHashPassword.enabled: false
```

secretName

Existing secret name where the initial admin password is stored

```
initialAdminHashPassword.secretName: ""
```

secretKey

Existing secret key where the initial admin password is stored

```
initialAdminHashPassword.secretKey: ""
```

defaultVault

Horizon Default vault configuration

keyset

Keyset

A reference to a secret that contains the keyset. You should store it (base64-encoded) in a Kubernetes secret and specify the secret details here.

README.md[README.md]

```
defaultVault.keyset: {}
```

Example

```
keyset:
 secretName: ""
 secretKey: ""
```

legacySsvPassword

Horizon legacy SSV password (Optional). Should be set if upgrading from Horizon 2.7 or earlier to Horizon 2.8 or later.

```
legacySsvPassword: {}
```

Example

```
legacySsvPassword:
  secretName: horizon-legacy-ssv-password
  secretKey: legacySsvPassword
```

allowedHosts

Additional allowed hosts that are whitelisted to access the Horizon UI. Configured ingresses will automatically be added to the list, this should only be used when port forwarding or when an ingress is created manually.

```
allowedHosts:
  - localhost:9000
```

Example

```
allowedHosts:
  - localhost:9000
  - demo.example.org
```

trustedProxies

Depending on your Kubernetes environment, Ingress IPs may be unpredictable. In that case, you should whitelist every IP in your local addressing space.

```
trustedProxies:
  - 0.0.0.0/0
  - '::/0'
```

Example

```
trustedProxies:
  - 0.0.0.0/0
  - ::/0
```

events

Configuration for Horizon events

chainsign

Whether Horizon events should be signed and chained using the event seal secret

```
events.chainsign: true
```

secret

Secret used to sign and chain events

Can be a reference to a Kubernetes secret.

```
events.secret: {}
```

Example

```
secret:
  valueFrom:
    secretKeyRef:
      name: horizon-secret
```

```
      key: eventSealSecret
```

ttl

Duration during which events are kept in database

```
events.ttl: ""
```

discoveryTtl

Duration during which discovery events are kept in database

```
events.discoveryTtl: ""
```

logFormat

Format in which logs will be outputted. Can be set either to "console" or "json" for structured logging.

```
logFormat: "console"
```

tls

TLS configuration

enabled

Whether to use the HTTPS port by default on ingresses and other services

```
tls.enabled: false
```

secretName

Existing secret name where a PKCS12 certificate is stored

```
tls.secretName: ""
```

secretKey

Existing secret key where the PKCS12 certificate is stored

```
tls.secretKey: ""
```

mailer

Configuration for the Horizon mailer. You should configure this if you want your Horizon instance to be able to send emails. You should fetch credentials from the environment if they are required.

host

SMTP host

```
mailer.host: ""
```

port

SMTP host port

```
mailer.port: ""
```

tls

Enable TLS for this SMTP host

```
mailer.tls: ""
```

ssl

Enable SSL for this SMTP host

```
mailer.ssl: ""
```

user

Authentication username for this SMTP host

```
mailer.user: ""
```

password

Authentication password for this SMTP host

Can be a reference to a Kubernetes secret.

```
mailer.password: {}
```

Example

```
password:
  valueFrom:
    secretKeyRef:
      name: horizon-secret
      key: mailerPassword
```

logback

Configure the logger for this Horizon instance. Sensible defaults are set, but you may need a more verbose logging experience when debugging the application.

Play Framework Logging

level

Global logging level for all loggers

```
logback.level: "info"
```

pattern

Log messages pattern

```
logback.pattern: "%date{yyyy-MM-dd HH:mm:ss} - [%logger] - [%traceID] - [%level] -
%message%n%xException{full}"
```

loggers

Logging level overrides for specific loggers

```
logback.loggers:
  - name: events
    level: warn
  - name: json_events
    level: info
```

You might want to use the following loggers to gather more info about your Horizon instance

```
loggers:
  - name: actors
    level: debug
  - name: actions
    level: debug
  - name: controllers
```

```
    level: debug
  - name: filters
    level: debug
  - name: models
    level: debug
  - name: modules
    level: debug
  - name: pki-connectors
    level: debug
  - name: racs-connectors
    level: debug
```

serviceAccount

Kubernetes Reference

create

Enable the creation of a ServiceAccount for Horizon pods

```
serviceAccount.create: true
```

name

Name of the created ServiceAccount

If not set and create is true, a name is generated using the horizon.fullname template.

```
serviceAccount.name: ""
```

annotations

Annotations for Horizon Service Account

```
serviceAccount.annotations:
  helm.sh/hook: pre-install, pre-upgrade, pre-rollback
  helm.sh/hook-delete-policy: before-hook-creation
  helm.sh/hook-weight: '0'
```

automountServiceAccountToken

Automount service account token for the server service account

```
serviceAccount.automountServiceAccountToken: true
```

clientCertificateHeader

Indicates to Horizon in which header the client certificate will be passed. Will be automatically set by the ingress.clientCertificateAuth value if set.

```
clientCertificateHeader: ""
```

podsDirectConnect

Whether Horizon pods should connect to each other directly via IP, or through a DNS record generated by a Kubernetes DNS server. Useful if the kube-dns server is configured with "pods disabled" or if you use GKE Cloud DNS. NOTE: This is not supported by Istio.

```
podsDirectConnect: false
```

extraConfig

Additional configuration for Horizon. Injecting arbitrary config could result in unexpected behavior. Proceed with caution.

Horizon Configuration Parameters

```
extraConfig: ""
```

Example

```
extraConfig: |
  horizon {
    notification.mail.attachment.extension.der = "der"
  }
```

upgrade

Upgrade job

enabled

If true, an upgrade job will be run when upgrading the release, modifying your database schema. This works even if `mongodb.enabled` is set to false.

```
upgrade.enabled: true
```

force

If true, an upgrade job will be run every time the Chart is installed or upgraded.

```
upgrade.force: false
```

annotations

Extra annotations to add to the upgrade job

```
upgrade.annotations:
  helm.sh/hook: post-upgrade
  helm.sh/hook-delete-policy: before-hook-creation
  helm.sh/hook-weight: '0'
```

image

Upgrade image

registry

Horizon Migration image registry

```
upgrade.image.registry: "registry.evertrust.io"
```

repository

Horizon Migration image repository

```
upgrade.image.repository: "horizon-migration"
```

tag

Horizon Migration image tag (immutable tags are recommended)

```
upgrade.image.tag: "1.17.0"
```

pullPolicy

Horizon Migration image pull policy

```
upgrade.image.pullPolicy: "IfNotPresent"
```

pullSecrets

Horizon Migration image pull secrets

```
upgrade.image.pullSecrets: []
```

resources

horizon-migration container resources

Kubernetes Reference

```
upgrade.resources:
  limits:
    memory: 512Mi
    cpu: 500m
  requests:
    memory: 512Mi
    cpu: 500m
```

from

Sets the version you're upgrading from. If empty, the chart will try to infer the version from the database.

```
upgrade.from: ""
```

to

Sets the version you're upgrading to. If empty, the chart will use Chart.AppVersion.

```
upgrade.to: ""
```

nodeSelector

Node labels for upgrade pod assignment

Kubernetes Reference

```
upgrade.nodeSelector: {}
```

tolerations

Tolerations for upgrade pod assignment

Kubernetes Reference

```
upgrade.tolerations: []
```

podSecurityContext

Configure Pods Security Context

Kubernetes Reference

enabled

Enabled upgrade pod Security Context

```
upgrade.podSecurityContext.enabled: true
```

fsGroup

Set upgrade pod Security Context fsGroup

```
upgrade.podSecurityContext.fsGroup: 1001
```

containerSecurityContext

Configure Container Security Context

Kubernetes Reference

enabled

Enabled upgrade container Security Context

```
upgrade.containerSecurityContext.enabled: true
```

runAsUser

Set upgrade container Security Context runAsUser

```
upgrade.containerSecurityContext.runAsUser: 1001
```

runAsNonRoot

Set upgrade container Security Context runAsNonRoot

```
upgrade.containerSecurityContext.runAsNonRoot: true
```

extraVolumes

Optionally specify extra list of additional volumes for the upgrade pod

```
upgrade.extraVolumes: []
```

Example

```
extraVolumes:
  - name: extra-volume-name
    configMap:
      name: example-configmap
```

extraVolumeMounts

Optionally specify extra list of additional volumeMounts for the upgrade container

```
upgrade.extraVolumeMounts: []
```

Example

```
extraVolumeMounts:
  - name: extra-volume-name
    mountPath: /mnt/extra-volume
```

ignoreEmptyFrom

Ignore empty from

```
upgrade.ignoreEmptyFrom: false
```

externalDatabase

Configuration for an Horizon external database

Refer to the Horizon installation guide to configure the installation correctly.

uri

External MongoDB URI

For an external database to be used, `mongodb.enabled` must be set to `false`. Can be a reference to a

Kubernetes secret.

```
externalDatabase.uri: {}
```

Example

```
uri:
  valueFrom:
    secretKeyRef:
      name: horizon-secret
      key: mongoDBUri
```

extraObjects

Create dynamic manifests via values

```
extraObjects: []
```

Example

```
extraObjects:
  - apiVersion: "kubernetes-client.io/v1"
    kind: ExternalSecret
    metadata:
      name: horizon-secrets
    spec:
      backendType: gcpSecretsManager
      data:
        - key: horizon-secret-key
          name: horizon-secret-name
```

metrics

Enable Prometheus metrics

enabled

Whether to enable Prometheus metrics

```
metrics.enabled: false
```

port

Prometheus metrics port

```
metrics.port: 9095
```

analytics

Enable analytics engine

Refer to the Horizon installation guide to configure the installation correctly.

Analytics guide

Analytics for docker guide

enabled

Whether to enable analytics

```
analytics.enabled: false
```

persistence

Enable Persistence

When enabled, Horizon will be deployed as a statefulset.

enabled

Whether to enable persistence

```
persistence.enabled: false
```

persistentVolumeClaimRetentionPolicy

Persistent Volume Claim Retention Policy

Kubernetes Reference

```
persistence.persistentVolumeClaimRetentionPolicy:
  whenDeleted: Delete
  whenScaled: Retain
```

1.3. Installing on Openshift

Installing Horizon on Openshift is very similar to installing on Kubernetes. The main difference is that you need to use the `oc` command instead of `kubectl`. For that reason, you should follow the Kubernetes installation procedure.

This page details the differences expected between Kubernetes and Openshift.

Security contexts

The default Horizon Helm chart uses the `1001` user to avoid running as root inside the container. However, on OpenShift, this results in the `anyuid` SCC being required to run the container. Since a random non-root UID will be assigned by OpenShift to the container upon startup, this security measure is unnecessary. It can be safely disabled by adding the following YAML to your `values-override.yaml` file:

```
podSecurityContext:
  enabled: false

containerSecurityContext:
  enabled: false
```

If you're using the built-in database for test purposes, you'll also need to disable the security context for the database container:

```
mongodb:
  podSecurityContext:
    enabled: false

  containerSecurityContext:
    enabled: false
```

Leases

In a large cluster, chances are that CRDs cannot be installed by a regular user. However, Horizon can be configured to rely on leases that are CRDs for clustering. See the dedicated documentation section for more information on how leases work.

Leases can be safely disabled without having a large impact on Horizon reliability. They mostly help in case of a network partition across multiple datacenters or availability zones.

To disable leases, add the following YAML to your `values-override.yaml` file:

```
leases:
  enabled: false
```

Then, when installing the helm chart, add the `--skip-crds` option to ensure that the leases CRD is not installed.

Router configuration

When exposing Horizon through the OpenShift router, you need to provide Horizon with a way to authenticate client certificates. You have two options to do so:

- Install the `cert-auth-proxy` component as a sidecar of the Horizon pod and use a passthrough route to forward traffic to Horizon. (**recommended**)
- Configure the router to ask for client certificates and forward traffic to Horizon.

Using `cert-auth-proxy`

The `cert-auth-proxy` component is a small proxy that can be used to authenticate client certificates. It is installed as a sidecar container to Horizon, and then referenced in place of Horizon in the OpenShift route or ingress. To install it, add the following YAML to your `values-override.yaml` file:

```yaml
clientCertificateHeader: "X-Forwarded-Tls-Client-Cert"

sidecars:
  - name: cert-auth-proxy
    image: registry.evertrust.io/cert-auth-proxy:latest
    imagePullPolicy: Never
    ports:
      - name: https-proxy
        containerPort: 8443
    env:
      - name: UPSTREAM
        value: localhost:9000
    volumeMounts:
      - name: horizon-local-tls
        # This mountPath will enable the certificate for the "horizon.local" route
        mountPath: /var/cert-auth-proxy/certificates/horizon.local

extraVolumes:
  - name: horizon-local-tls
    secret:
      # This secret must contain a valid TLS certificate for route hostname.
      secretName: horizon.local-tls

service:
  extraPorts:
    - name: https-proxy
      protocol: TCP
      port: 8443
      targetPort: https-proxy
```

Then, you can either use the following extra values to `override-values.yaml` to generate an ingress with a passthrough route:

```yaml
ingress:
  enabled: true
  annotations:
    route.openshift.io/termination: "passthrough"
  extraRules:
    - host: "horizon.local"
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: horizon
                port:
                  name: https-proxy
  extraTls:
    - hosts:
        - "horizon.local"
      secretName: horizon.local-tls
```

If you wish to use the `Route` resource instead, disable the ingress by setting `ingress.enabled` to `false` and manually create the route:

```
$ oc create route passthrough horizon --service=horizon --port=https-proxy
--hostname=horizon.local
```

Using the router mTLS configuration



This method is no longer recommended since it requires deploying a specific ingress controller for Horizon purposes. Changing mTLS settings on an ingress controller affects all routes served by this ingress controller.

Follow the Kubernetes ingress controller configuration procedure. Gather all ACs identified in the previous step and create a bundle file containing all of them, called `ca-bundle.pem`.

Then, follow the Openshift documentation to configure the ingress controller serving Horizon requests to ask for client certificates signed by any of these ACs:

Upload the ACs to the OpenShift cluster in a configmap

```
$ oc create configmap router-ca-certs-default --from-file=ca-bundle.pem=ca-bundle.pem
-n openshift-config
```

Tell the ingress controller to ask for client certificates

```
$ oc edit IngressController default -n openshift-ingress-operator
```

And set the following values:

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
    name: default
namespace: openshift-ingress-operator
spec:
    clientTLS:
      clientCertificatePolicy: Optional
      clientCA:
        name: router-ca-certs-default
```

Then, when installing Horizon through the Chart, set the `clientCertificateDefaultParsingType` key to the value `haproxy` (which is what the Openshift ingress controller is based on).



As of 4.14, Openshift will only download CRLs from the certificates in the `ca-bundle.pem` chain (inferred from their CRLDPs). This can lead to a TLS handshake failure when authenticating using a client certificate. Introducing a dummy entity certificate in the chain might be required to ensure that the operational CAs CRLs are downloaded by the Openshift ingress controller. See this issue for more information.

Skip to the Ensure certificate authentication is effective section to test your configuration.

1.4. Running with Docker/Compose

If you just want to try out Horizon, one way of doing so could be to directly run Horizon from Docker. For resiliency reasons, this is obviously not recommended for production usage.

We provide a Docker image that's entirely configurable through environment variables. All Docker examples require that you login to our Docker repository beforehand :

```
$ docker login registry.evertrust.io
```



If you're looking to try out Horizon's features, take a look at the EVERTRUST Playground. It is a Docker Compose project bundled with demo values to get you started swiftly.

Before using Horizon, please note that a Tink keyset is required to encrypt secrets. Refer to the documentation on how to generate one.

Docker Compose example

The simplest way to spin up an Horizon instance is to let Docker Compose manage the required components :

- the database,
- the Horizon instance
- and (optionally) the reverse proxy.

Copy the following `docker-compose.yaml` file and tweak it to match your needs :

```
version: "3.1"
services:
  horizon:
    image: registry.evertrust.io/horizon:2.12.x
    ports:
      - "9000:9000"
    networks:
      - horizon
    environment:
      LICENSE: MI...
      APPLICATION_SECRET: tobechanged
      EVENT_SEAL_SECRET: tobechanged
      KEYSET: '{"tink keyset": "..."}'
      HOSTS_ALLOWED.0: .
      MONGODB_URI: mongodb://mongo:27017/horizon
    depends_on:
      - mongo
    healthcheck:
      test: [ "CMD", "curl", "-f", "http://localhost:7626/ready" ]
      interval: 10s
      timeout: 60s
      retries: 10
  mongo:
    image: mongo:7
    restart: always
    volumes:
      - database:/data/db
    networks:
      - horizon
volumes:
  database: {}
networks:
  horizon: {}
```

You then only need to run the following in the directory where you created the previous file :

```
$ docker compose up
```

Horizon should quickly become available on `http://localhost:9000`.

Vanilla Docker example

Pull the latest Horizon image:

```
$ docker pull registry.evertrust.io/horizon:2.12.x
```

The Horizon Docker image ships with sensible configuration defaults. Most can be configured by injecting environment variables when running the container, like so:

```
$ docker run \
 -e LICENSE="MI..." \
 -e APPLICATION_SECRET="tobechanged" \
 -e EVENT_SEAL_SECRET="tobechanged" \
 -e KEYSET="{\"tink keyset\": \"...\"}" \
 -e HOSTS_ALLOWED.0="." \
 -e MONGODB_URI="" \
 -p [port]:9000 \
 registry.evertrust.io/horizon:2.12.x
```

Environment variables

General configuration

Variable	Type	Description	Default
LICENSE	`string`	A valid Horizon license string, base64-encoded. Can be used if `LICENSE_PATH` is empty.	
LICENSE_PATH	`path`	Path where an Horizon license file is mounted inside the container. Can be used if the license is not passed directly through `LICENSE`.	
APPLICATION_SECRET	`string`	Application secret used by Horizon	
MONGODB_URI	`string`	A valid MongoDB URI. See MongoDB URI Configuration.	

Variable	Type	Description	Default
HOSTS_ALLOWED	`array`	Array of hosts. Append the array index after a dot (the nth allowed host variable name would be HOSTS_ALLOWED.n).	



Your license usually contains newline characters, that you must replace by '\n' when setting it through the environment.

Configure the secrets vault

Variable	Type	Description	Default
KEYSET	`string`	The raw Tink keyset to use to encrypt secret. It should look like `{"primaryKeyId":1, ⋯}`. Can be used if `VAULT_TINK_KEYSET_PATH` is empty.	
VAULT_TINK_KEYSET_PATH	`string`	Path to a mounted file containing the Tink keyset.	
VAULT_TINK_MASTER_KEY_URI	`string`	The Master Key URI that wraps the Tink keyset.	
VAULT_TINK_CREDENTIALS_PATH	`string`	Path to a mounted file containing Tink Master Key URI credentials if it is not at the standard path for the SDK.	

Configuring HTTPS

In production, it is strongly recommended to ensure all requests go through a layer of encryption. Configuring TLS for Horizon will allow your reverse proxy to request Horizon data using TLS.



If all settings are left empty, Horizon will generate a self-signed certificate upon startup and still expose its HTTPS endpoint on

Variable	Type	Description	Default
HTTP_PORT	`port`	Port of the HTTP server	`9000`

Variable	Type	Description	Default
HTTPS_PORT	`port`	Port of the HTTPS server	`9443`
HTTPS_KEYSTORE_PATH	`string`	Location where the keystore containing a server certificate is located.	
HTTPS_KEYSTORE_PASSWORD	`string`	Password for the given keystore, if required by the keystore type	
HTTPS_KEYSTORE_TYPE	`string`	Format in which the keystore is. Can be either `pkcs12`, `jks` or `pem` (a base64-encoded DER certificate)	`pkcs12`
HTTPS_KEYSTORE_ALGORITHM	`string`	The key store algorithm	Platform default algorithm

Mailer configuration

Variable	Type	Description	Default
SMTP_HOST	`string`	SMTP host	
SMTP_PORT	`string`	SMTP port	
SMTP_SSL	`boolean`	Whether SSL should be used	
SMTP_TLS	`boolean`	Whether TLS should be used	
SMTP_USER	`string`	SMTP user	
SMTP_PASSWORD	`string`	SMTP password	

Events configuration

Variable	Type	Description	Default
EVENT_CHAINSIGN	`boolean`	Whether to sign events to verify their integrity	`true`
EVENT_TTL	`duration`	Event time to live in database	

Variable	Type	Description	Default
EVENT_DISCOVERY_TTL	`duration`	Discovery events time to live. Can be shorter in case a large number of discovery events are logged.	

Analytics parameters

Variable	Type	Description	Default
ANALYTICS_URL	`string`	The absolute path of the analytics database file	
ANALYTICS	`boolean`	Enable all analytics: certificate, event and discovery event. A database file must be provided through the `ANALYTICS_URL` environment variable	`false`
CERTIFICATE_ANALYTICS	`boolean`	Enable certificate analytics only. A database file must be provided through the `ANALYTICS_URL` environment variable	`false`
EVENT_ANALYTICS	`boolean`	Enable event analytics only. A database file must be provided through the `ANALYTICS_URL` environment variable	`false`
EVENT_DISCOVERY_ANALYTICS	`boolean`	Enable discovery event analytics only. A database file must be provided through the `ANALYTICS_URL` environment variable	`false`

Advanced parameters

Variable	Type	Description	Default
AKKA_ACTOR_SYSTEM	`string`	Name of the actor system used by Pekko. Useful if you need to run multiple instances of Horizon in the same Kubernetes namespace. Due to compatibility reasons, the variable is still called Akka.	`horizon`
SESSION_MAXAGE	`string`	Log in session duration.	`15 minutes`
HTTP_CERTIFICATE_HEADER	`string`	Header name in which the client certificate should be sent when using mTLS.	

1.5. FIPS mode

Starting with version 2.12, Horizon is distributed in two flavors:

- the **standard** flavor, used by default;
- the **FIPS** flavor, in which all cryptographic operations are performed by the BouncyCastle FIPS provider (`bc-fips`) running in approved-only mode, and TLS is handled by the BouncyCastle JSSE provider in FIPS mode.

The FIPS flavor is intended for environments subject to FIPS constraints. Both flavors are configured the same way, and the FIPS flavor is selected by installing its dedicated artifacts.

Artifacts

The FIPS flavor of each component is published under its own name, suffixed with `-fips`:

Component	Standard	FIPS
Horizon	RPM / DEB package: `horizon`	RPM / DEB package: `horizon-fips`
	Docker image: `registry.evertrust.io/horizon:<version>`	Docker image: `registry.evertrust.io/horizon:<version>-fips`
	Helm chart: default values	Helm chart: `image.flavor: fips`
Tinkey	RPM / DEB package: `tinkey`	RPM / DEB package: `tinkey-fips`
	Docker image: `registry.evertrust.io/tinkey:latest`	Docker image: `registry.evertrust.io/tinkey:latest-fips`

Component	Standard	FIPS
Horizon Migration Tool	RPM / DEB package: `horizon-migration`	RPM / DEB package: `horizon-migration-fips`
	Docker image: `registry.evertrust.io/horizon-migration:latest`	Docker image: `registry.evertrust.io/horizon-migration:latest-fips`

The FIPS packages are published in the same repositories as the standard ones, and install to the same paths (for instance `/opt/horizon`). Only the package name differs.



Each FIPS artifact can be used on its own. If your environment is subject to FIPS constraints, use the FIPS flavor of every component that falls under these constraints.

Installation

RPM

Follow the installation procedure and install the `horizon-fips` package instead of `horizon`:

```
# yum install horizon-fips
```

Debian

Follow the installation procedure and install the `horizon-fips` package instead of `horizon`:

```
# sudo apt install horizon-fips
```

Kubernetes

Set the image flavor in your `values.yaml` file. The chart appends it to the image tag:

```
image:
  flavor: fips
```

The upgrade jobs use the Horizon Migration Tool image, whose tag must also carry the `-fips` suffix:

```
upgrade:
  image:
    tag: <migration tool version>-fips
```

Docker

Use the image tag suffixed with `-fips`:

```
$ docker pull registry.evertrust.io/horizon:<version>-fips
```

Specific behavior

In FIPS mode, the following applies:

- PKCS#12 files must use the `AES_STRONG` encryption level (AES-256-CBC with PBKDF2 HMAC-SHA256) and be integrity protected with PBMAC1.
- Key store passwords must be at least 14 characters long.
- SCEP uses OAEP encryption instead of PKCS 1.5
- By default, `bc-fips` refuses to use the same RSA key both for encryption and for signature. When a SCEP RA key is used to both decrypt requests and sign responses, add `-Dorg.bouncycastle.rsa.allow_multi_use=true` to the JVM options (`JAVA_OPTS`).
- CRMP profiles cannot be created since they require legacy encryption.
- RSA 1024 and SHA-1 are disabled.
- No PQC support outside of discovery.

1.6. Analytics

Analytics can be enabled on Horizon to speed search and dashboards on certificates, events and discovery events. It will create an embedded analytics database on the filesystem on each of the Horizon nodes to store a copy of those objects.

This will increase RAM and CPU consumption on the Horizon server itself, but will reduce load on the database. You should consider enabling the analytics if you have slow interfaces due to a large number of certificates or events



The analytics database is only used for research; all other operations are done directly to the mongo database

Configuring the analytics

Analytics is an opt-in feature that can be enabled through configuration:

RPM

Follow the Advanced configuration guide to add the following key to set up the analytics database file:

```
horizon.analytics.url = "jdbc:duckdb:/opt/horizon/var/run/analytics.db"
```

The following configuration keys are used to enable the analytics on certificate, event

and/or discovery event.

```
horizon.event.analytics.enabled = true
horizon.discovery.event.analytics.enabled = true
horizon.certificate.analytics.enabled = true
```

The following configuration keys are additional parameters for advanced configuration:

```
horizon.analytics.pool-size = 10
horizon.analytics.memory-limit = "1GB"
```

Debian

Follow the Advanced configuration guide to add the following key to set up the analytics database file:

```
horizon.analytics.url = "jdbc:duckdb:/opt/horizon/var/run/analytics.db"
```

The following configuration keys are used to enable the analytics on certificate, event and/or discovery event.

```
horizon.event.analytics.enabled = true
horizon.discovery.event.analytics.enabled = true
horizon.certificate.analytics.enabled = true
```

The following configuration keys are additional parameters for advanced configuration:

```
horizon.analytics.pool-size = 10
horizon.analytics.memory-limit = "1GB"
```

Kubernetes

While the analytics data store on disk is not critical for the core application functionality, we recommend enabling data persistence to reduce synchronization overhead. When persistence is enabled in the Helm chart, Horizon will be deployed as a StatefulSet instead of a standard Deployment.

This configuration utilizes a Persistent Volume Claim (PVC) to ensure reliable storage and retention of analytics data across pod restarts and rescheduling.

```
analytics:
  enabled: true

updateStrategy:
  type: RollingUpdate
```

```
persistence:
  enabled: true
  volumeClaimTemplates:
    analytics:
      storageClass: <YOUR-CLUSTER-STORAGE-CLASS>
      size: "1Gi"
```



Updating the application with a `StatefulSet` will require additional considerations.

Docker

Environment variables are available to configure the analytics see the docker analytics parameters.

1.7. Tinkey

This part describes how to generate a Tink Keyset and configure it in Horizon

Installation

RHEL



Tinkey requires java 17 or 21 installed

Installation from the EverTrust repository

Create a `/etc/yum.repos.d/tinkey.repo` file containing the EverTrust repository info:

```
[tinkey]
enabled=1
name=Tinkey Repository
baseurl=https://repo.evertrust.io/repository/tinkey-rpm/
gpgcheck=1
gpgkey=https://evertrust.io/.well-known/rpm/gpg.pub
username=<username>
password=<password>
```

Replace `<username>` and `<password>` with the credentials you were provided.

Make sure the Evertrust GPG key is trusted:

```
# rpm --import https://evertrust.io/.well-known/rpm/gpg.pub
```

You can then run the following to install the latest Tinkey version:

```
# yum install tinkey
```

To prevent unattended upgrades when running yum update, you should pin the Tinkey version by adding

```
exclude=tinkey
```

at the end of the `/etc/yum.repos.d/tinkey.repo` file after installing Tinkey.

Installing from RPM

Download the latest RPM for Tinkey on the Official EVERTRUST repository.

Upload the file `tinkey-<latest>.noarch.rpm` to the server;

Access the server with an account with administrative privileges;

Install the Tinkey package with the following command:

```
# yum localinstall /root/tinkey-<latest>.noarch.rpm
```

If you wish to verify the signature of the RPM package, the EVERTRUST key can be added to your trusted keys using the following command:

```
# rpm --import https://evertrust.io/.well-known/rpm/gpg.pub
```

The signature can then be verified using the following command:

```
# rpm -K /root/tinkey-<latest>.noarch.rpm
```

Debian



Tinkey requires java 17 or 21 installed

Installation from the EverTrust repository

If you haven't already, to add the EVERTRUST repository to your APT repositories, run the following commands:

1. Install the required tools (`gpg`)

   ```
   # sudo apt install gnupg
   ```

2. Download and install the EVERTRUST GPG key

```
# curl https://evertrust.io/.well-known/apt/gpg.pub | sudo gpg -o
/usr/share/keyrings/evertrust.gpg --dearmor
```

3. Add the repository

```
# echo "deb [ arch=all,amd64 signed-by=/usr/share/keyrings/evertrust.gpg ]
https://repo.evertrust.io/repository/apt all main" | sudo tee
/etc/apt/sources.list.d/evertrust.list
```

Once the repository has been added, authentication to it must be provided. To do so, edit the `/etc/apt/auth.conf` file and add the following lines:

```
machine repo.evertrust.io
login <your EVERTRUST login>
password <your EVERTRUST password>
```

Once the repository has been added, run the following command to update the APT repository list.

```
# sudo apt update
```

You can then run the following command to install the latest Tinkey version:

```
# sudo apt install tinkey
```

To prevent unattended upgrades when running `apt upgrade`, you should pin the Tinkey version by creating a `/etc/apt/preferences.d/tinkey` file:

```
Package: tinkey
Pin: version <installed-version>
Pin-Priority: 1001
```

Installing from DEB

Download the latest DEB for Tinkey on the Official EVERTRUST repository.

Upload the file `tinkey_<latest>_all.deb` to the server;

Access the server with an account with administrative privileges;

Install the Tinkey package with the following command:

```
# apt install /root/tinkey_<latest>_all.deb
```

Docker

The EVERTRUST Tinkey utility is available at:

```
registry.evertrust.io/tinkey
```

The FIPS flavor is tagged `latest-fips`:

```
registry.evertrust.io/tinkey:latest-fips
```



If your environment is subject to FIPS constraints, use the FIPS flavor of Tinkey:

- RPM / DEB package: `tinkey-fips`
- Docker image: `registry.evertrust.io/tinkey:latest-fips`

Usage

PlainText keyset

In this mode, the keyset directly contains an AES key without additional encryption. When Horizon starts, the keyset is loaded into memory and used for all encryption and decryption operations.

RHEL

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES256_GCM
--out=horizon.keyset
```

Debian

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES256_GCM
--out=horizon.keyset
```

Docker

```
docker run registry.evertrust.io/tinkey create-keyset --key-template AES256_GCM
--out=horizon.keyset
```



This is the equivalent to previous Horizon **SSV** encryption level.

PKCS#11 keyset

PKCS#11 templates enable integration with Hardware Security Modules (HSM) for enhanced security. Horizon supports three different PKCS#11 modes depending on your security

requirements.

The following parameters are commonly used when generating the key-uri:

Parameter	Description
object	The label of the symmetric key. **The key must already exist on the HSM**.
type	The type of key (typically "secret-key")
slot-id	The HSM slot identifier to use
module-path	The path to the .so library required for HSM interaction
pin-value	The PIN required to authenticate with the HSM

Wrapped mode

In this mode, a software-based AES key is encrypted (wrapped) with a master key stored in the HSM. The wrapped key is stored in the keyset file, while the master key remains securely in the HSM. When Horizon starts, the keyset is decrypted using the HSM's master key, then loaded into memory and used for all subsequent encryption and decryption operations. This approach balances security with performance by minimizing HSM communication while keeping the master key protected in hardware.

To create a wrapped keyset using the GCM algorithm:

RHEL

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES128_GCM --out
horizon.keyset --master-key-uri pkcs11://object=AES1;type=secret-key;slot-id=-
1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

Debian

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES128_GCM --out
horizon.keyset --master-key-uri pkcs11://object=AES1;type=secret-key;slot-id=-
1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

Docker

```
docker run registry.evertrust.io/tinkey create-keyset --key-template AES128_GCM
--out horizon.keyset --master-key-uri pkcs11://object=AES1;type=secret-
key;slot-id=-1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

To use the CBC algorithm instead, replace `pkcs11://` with `pkcs11-aes-cbc://` in the master-key-uri:

RHEL

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES128_GCM --out
horizon.keyset --master-key-uri pkcs11-aes-cbc://object=AES1;type=secret-
key;slot-id=-1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

Debian

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES128_GCM --out
horizon.keyset --master-key-uri pkcs11-aes-cbc://object=AES1;type=secret-
key;slot-id=-1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

Docker

```
docker run registry.evertrust.io/tinkey create-keyset --key-template AES128_GCM
--out horizon.keyset --master-key-uri pkcs11-aes-cbc://object=AES1;type=secret-
key;slot-id=-1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```



When using wrapped mode, Horizon needs the master-key-uri to decrypt the keyset at startup. Please follow the configuration steps below.

Hardware protected mode

In this mode, the encryption key is stored directly in the HSM. All encryption and decryption operations are performed by the HSM itself, ensuring the key never leaves the hardware security boundary. This provides a high level of security but requires HSM communication for every cryptographic operation.

To create the keyset using the GCM algorithm:

RHEL

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template PKCS11_AES_GCM
--out keyset --key-uri pkcs11://object=AES1;type=secret-key;slot-id=-1?module-
path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

Debian

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template PKCS11_AES_GCM
--out keyset --key-uri pkcs11://object=AES1;type=secret-key;slot-id=-1?module-
path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

Docker

```
docker run registry.evertrust.io/tinkey create-keyset --key-template
```

```
PKCS11_AES_GCM --out keyset --key-uri pkcs11://object=AES1;type=secret-
key;slot-id=-1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

To use the CBC algorithm instead, change the `--key-template` parameter to `PKCS11_AES_CBC`:

RHEL

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template PKCS11_AES_CBC
--out keyset --key-uri pkcs11://object=AES1;type=secret-key;slot-id=-1?module-
path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

Debian

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template PKCS11_AES_CBC
--out keyset --key-uri pkcs11://object=AES1;type=secret-key;slot-id=-1?module-
path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

Docker

```
docker run registry.evertrust.io/tinkey create-keyset --key-template
PKCS11_AES_CBC --out keyset --key-uri pkcs11://object=AES1;type=secret-
key;slot-id=-1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

Derived hardware protected mode

In this mode, a master key stored in the HSM is used to derive encryption keys on demand. Each time Horizon needs to encrypt or decrypt data, it derives a unique key from the master key in the HSM using a random seed. This seed is then stored alongside the encrypted data. This provides the highest level of security but requires HSM communication for every cryptographic operation.

To create the keyset using the GCM algorithm:

RHEL

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template
PKCS11_AES_GCM_DERIVED --out keyset --key-uri pkcs11://object=AES1;type=secret-
key;slot-id=-1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

Debian

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template
PKCS11_AES_GCM_DERIVED --out keyset --key-uri pkcs11://object=AES1;type=secret-
key;slot-id=-1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

Docker

```
docker run registry.evertrust.io/tinkey create-keyset --key-template
PKCS11_AES_GCM_DERIVED --out keyset --key-uri pkcs11://object=AES1;type=secret-
key;slot-id=-1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

To use the CBC algorithm instead, change the `--key-template` parameter to `PKCS11_AES_CBC_DERIVED`:

RHEL

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template
PKCS11_AES_CBC_DERIVED --out keyset --key-uri pkcs11://object=AES1;type=secret-
key;slot-id=-1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

Debian

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template
PKCS11_AES_CBC_DERIVED --out keyset --key-uri pkcs11://object=AES1;type=secret-
key;slot-id=-1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```

Docker

```
docker run registry.evertrust.io/tinkey create-keyset --key-template
PKCS11_AES_CBC_DERIVED --out keyset --key-uri pkcs11://object=AES1;type=secret-
key;slot-id=-1?module-path=/usr/lib64/pkcs11/libsofthsm2.so&pin-value=1234
```



The `PKCS11_AES_CBC_DERIVED` parameter is the equivalent to previous Horizon **SHV** encryption level.

AWS KMS keyset

AWS KMS (Key Management Service) enables secure key management using Amazon's cloud infrastructure. In this mode, a software-based AES key is encrypted (wrapped) with a master key stored in AWS KMS. When Horizon starts, the keyset is decrypted using the KMS master key, then loaded into memory and used for all subsequent encryption and decryption operations.

The following parameters are required when generating the AWS KMS key-uri:

Parameter	Description
key-uri	The AWS KMS key ARN (Amazon Resource Name) or alias
credentials	Path to AWS credentials file (optional if using IAM roles or environment variables)

To create an AWS KMS wrapped keyset:

RHEL

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES256_GCM --out
horizon.keyset --master-key-uri aws-kms://arn:aws:kms:us-east-
1:123456789012:key/12345678-1234-1234-1234-123456789012
```

Debian

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES256_GCM --out
horizon.keyset --master-key-uri aws-kms://arn:aws:kms:us-east-
1:123456789012:key/12345678-1234-1234-1234-123456789012
```

Docker

```
docker run registry.evertrust.io/tinkey create-keyset --key-template AES256_GCM
--out horizon.keyset --master-key-uri aws-kms://arn:aws:kms:us-east-
1:123456789012:key/12345678-1234-1234-1234-123456789012
```

If you need to specify AWS credentials explicitly:

RHEL

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES256_GCM --out
horizon.keyset --master-key-uri aws-kms://arn:aws:kms:us-east-
1:123456789012:key/12345678-1234-1234-1234-123456789012 --credentials
/path/to/credentials.json
```

Debian

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES256_GCM --out
horizon.keyset --master-key-uri aws-kms://arn:aws:kms:us-east-
1:123456789012:key/12345678-1234-1234-1234-123456789012 --credentials
/path/to/credentials.json
```

Docker

```
docker run registry.evertrust.io/tinkey create-keyset --key-template AES256_GCM
--out horizon.keyset --master-key-uri aws-kms://arn:aws:kms:us-east-
1:123456789012:key/12345678-1234-1234-1234-123456789012 --credentials
/path/to/credentials.json
```



When using AWS KMS keyset, Horizon needs the master-key-uri to decrypt the

keyset at startup. Please follow the configuration steps below. Additionally, ensure your AWS credentials are properly configured (via service account, application default credentials, or credentials file).

GCP KMS keyset

GCP KMS (Google Cloud Key Management Service) provides cloud-based key management using Google Cloud Platform. In this mode, a software-based AES key is encrypted (wrapped) with a master key stored in GCP KMS. When Horizon starts, the keyset is decrypted using the KMS master key, then loaded into memory and used for all subsequent encryption and decryption operations.

The following parameters are required when generating the GCP KMS key-uri:

Parameter	Description
key-uri	The GCP KMS key resource name in the format: `projects/PROJECT_ID/locations/LOCATION/keyRings/KEY_RING/cryptoKeys/KEY_NAME`
credentials	Path to GCP service account credentials JSON file (optional if using default application credentials)

To create a GCP KMS wrapped keyset:

RHEL

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES256_GCM --out
horizon.keyset --master-key-uri gcp-kms://projects/my-project/locations/us-
east1/keyRings/my-keyring/cryptoKeys/my-key
```

Debian

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES256_GCM --out
horizon.keyset --master-key-uri gcp-kms://projects/my-project/locations/us-
east1/keyRings/my-keyring/cryptoKeys/my-key
```

Docker

```
docker run registry.evertrust.io/tinkey create-keyset --key-template AES256_GCM
--out horizon.keyset --master-key-uri gcp-kms://projects/my-
project/locations/us-east1/keyRings/my-keyring/cryptoKeys/my-key
```

If you need to specify GCP credentials explicitly:

RHEL

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES256_GCM --out
```

```
horizon.keyset --master-key-uri gcp-kms://projects/my-project/locations/us-
east1/keyRings/my-keyring/cryptoKeys/my-key --credentials /path/to/service-
account.json
```

Debian

```
/opt/evertrust/tinkey/bin/tinkey create-keyset --key-template AES256_GCM --out
horizon.keyset --master-key-uri gcp-kms://projects/my-project/locations/us-
east1/keyRings/my-keyring/cryptoKeys/my-key --credentials /path/to/service-
account.json
```

Docker

```
docker run registry.evertrust.io/tinkey create-keyset --key-template AES256_GCM
--out horizon.keyset --master-key-uri gcp-kms://projects/my-
project/locations/us-east1/keyRings/my-keyring/cryptoKeys/my-key --credentials
/path/to/service-account.json
```



When using GCP KMS keyset, Horizon needs the master-key-uri to decrypt the keyset at startup. Please follow the configuration steps below. Additionally, ensure your GCP credentials are properly configured (via service account, application default credentials, or credentials file).

Horizon Configuration

Keyset

Once the keyset is created, it must be made available to Horizon:

RPM

Move it to the expected location on the Horizon server:

```
mv horizon.keyset /opt/horizon/etc/horizon.keyset
```

Then set the appropriate ownership and permissions:

```
chown horizon:horizon /opt/horizon/etc/horizon.keyset
chmod 660 /opt/horizon/etc/horizon.keyset
```

Debian

Move it to the expected location on the Horizon server:

```
mv horizon.keyset /opt/horizon/etc/horizon.keyset
```

Then set the appropriate ownership and permissions:

```
chown horizon:horizon /opt/horizon/etc/horizon.keyset
chmod 660 /opt/horizon/etc/horizon.keyset
```

Kubernetes

Create a Kubernetes secret containing the keyset on the Horizon namespace :

```
$ kubectl create secret generic horizon-keyset \
 --from-file=keyset="<path to your keyset file>" \
 --namespace horizon
```

Then reference the keyset in your `values.yaml`:

```
defaultVault:
  keyset:
    secretName: horizon-keyset
    secretKey: keyset
```

Master Key URI

If the keyset is wrapped, the master key URI must be made available to Horizon:

RPM

Edit your `/etc/default/horizon` file and add the variable `HORIZON_TINK_MASTER_KEY_URI` with your master-key-uri value.

```
HORIZON_TINK_MASTER_KEY_URI=<master key URI>
```

Debian

Edit your `/etc/default/horizon` file and add the variable `HORIZON_TINK_MASTER_KEY_URI` with your master-key-uri value.

```
HORIZON_TINK_MASTER_KEY_URI=<master key URI>
```

Kubernetes

Set the `defaultVault.masterKeyURI` key in your Helm `values.yaml` file:

```
defaultVault:
  masterKeyURI: "<master key URI>"
```

Credentials

If the keyset is wrapped using a KMS, the master key uses credentials that must be made available to Horizon. If your credentials are at the standard path for the KMS SDK, it will be found automatically. Otherwise, follow the steps below:

RPM

Edit your `/etc/default/horizon` file and add the variable `HORIZON_TINK_CREDENTIALS_PATH` with the path to your credentials file.

```
HORIZON_TINK_CREDENTIALS_PATH=<path to credentials>
```

Then set the appropriate ownership and permissions:

```
chown horizon:horizon <path to credentials>
chmod 660 <path to credentials>
```

Debian

Edit your `/etc/default/horizon` file and add the variable `HORIZON_TINK_CREDENTIALS_PATH` with the path to your credentials file.

```
HORIZON_TINK_CREDENTIALS_PATH=<path to credentials>
```

Then set the appropriate ownership and permissions:

```
chown horizon:horizon <path to credentials>
chmod 660 <path to credentials>
```

Kubernetes

Specify the path to the credentials file (which should be mounted into the container) in your `values.yaml` file:

```
environment:
  - name: VAULT_TINK_CREDENTIALS_PATH
    value: /mnt/sample/path
```

1.8. Monitoring

Healthchecks

Liveness check

The liveness check is available on the `/alive` route of the pekko management port (7626 by default).

It checks that the pekko cluster is operational and performs a ping on the mongo database.

Readiness check

The readiness check is available on the `/ready` route of the pekko management port (7626 by default).

It checks that the pekko cluster is operational and verifies that the instance has been bootstrapped.



For RPM/DEB configuration, this check is proxied by the default NGINX configuration, and available on `/ready`

Metrics

Horizon can expose Prometheus metrics to monitor key KPIs and health infos about the instance.

Enabling metrics

RPM

To enable basic Prometheus metrics on port 9095, the following configuration must be applied (following this guide):

```
kamon {
  modules {
    prometheus-reporter.enabled = yes
    apm-reporter.enabled = no
    host-metrics.enabled = no
    jvm-metrics.enabled = no
  }

  prometheus {
    include-environment-tags = true
    embedded-server {
      hostname = 0.0.0.0
      port = 9095
    }
  }
}
```

```
horizon {
  metrics.enabled = true
}
```

Debian

To enable basic Prometheus metrics on port 9095, the following configuration must be applied (following this guide):

```
kamon {
  modules {
    prometheus-reporter.enabled = yes
    apm-reporter.enabled = no
    host-metrics.enabled = no
    jvm-metrics.enabled = no
  }

  prometheus {
    include-environment-tags = true
    embedded-server {
      hostname = 0.0.0.0
      port = 9095
    }
  }
}

horizon {
  metrics.enabled = true
}
```

Kubernetes

To enable metrics on port 9095, add the following to your `values.yaml` file:

```
metrics:
  enabled: true
  port: 9095
```

Exposed metrics

Exposed metrics are basic clustering metrics, and Horizon-specific metrics such as:

- License expiration information
- License usage information
- License type
- Horizon version

- Scala version
- PKI Queue size
- PKI Connector status
- Credentials expiration
- Last user activity
- Analytics database status



Additional metrics configuration such as refresh intervals can be found on the configuration reference page.

1.9. Troubleshooting

Horizon Doctor



Horizon Doctor is currently only available for deployments on Linux. To troubleshoot deployments on Kubernetes, use built-in tools like events and logs.

Horizon doctor is a tool that performs checks on your Horizon installation as well as its required dependencies to ensure that everything is configured properly. The tool is targeted towards troubleshooting during installation or update procedures. Note that the tool requires root permissions to run.

Performed checks

At the moment, Horizon Doctor checks for:

OS checks

- Checks for installed Horizon version, MongoDB version, Java version, Nginx version, OS Version.
  - If the OS is a RedHat distribution, checks if the RedHat subscription is active
  - If Mongo is not installed locally, it notices it as an information log
- Checks for **SELinux**'s configuration: throws a warning if it is enabled, says ok if it is on permissive or disabled
- Checks for the status of the necessary services: `postfix`, `mongod`, `nginx` and `horizon`.
  - If the **postfix** service is running, tries to connect via a TCP SYN on the port 25 of the **relayhost** specified in the */etc/postfix/main.cf* file and throws an error if it can't.
- Checks how long the **Horizon** service has been running for.
- Checks if there is an **NTP service** active on the machine and checks if the system clock is synchronized with the NTP service.

Config checks

- Checks for existence and permissions of the **configuration** file: the permissions are expected to be at least 640 and the file is supposed to belong to horizon:horizon
- Checks for existence and permissions of the **licence** file: the permissions are expected to be at least 640 and the file is supposed to belong to horizon:horizon.
- Checks for existence and permissions of the **vault** file: the permissions are expected to be at least 640 and the file is supposed to belong to horizon:horizon.
- Checks for the permission of the Horizon directory (default: /opt/horizon): the permission is expected to be at least 755.
- Checks for the existence of the **symbolic link** for **nginx configuration** and runs an **nginx -t** test.
- Retrieves the **Java heap size parameters** that were set for Horizon and throws a warning if the default ones are used (min = 2048 and max = 3072).
- Retrieves the **Horizon DNS hostname** and stores it for a later test (throws an error if it has not been set).
- Checks for the **Horizon Play Secret** and **Horizon Event Seal Secret**: these are the Horizon application secrets and should be different from default value thus Horizon Doctor throws an error if either of them is equal to the default value (*changeme*).
- Retrieves the **MongoDB URI** (throws a warning if MongoDB is running on localhost; throws an error if MongoDB is running on an external instance but the *authSource=admin* parameter is missing from the URI).
- Parses the **Horizon license file** to retrieve its expiration date as well as the license details (number of holders per category).

Network checks

- Runs a **MongoDB ping** on the URI, then checks for the database used in the URI (throws a warning if the database used is not called *horizon*; throws an error if no database is specified in the URI).
- Checks for **PEKKO High Availability** settings: if no node hostname is set up, skips the remaining HA checks. If 2 nodes are set up, retrieves which node is running the doctor and checks for the other node. If 3 nodes are set up, retrieves which node is running the doctor and checks for the other 2 nodes. The check runs as:
  - if *curl* is installed, runs a *curl* request on the Node hostname at *alive* on the management port (default is 7626), and if alive runs another *curl* request on the Node hostname at */ready* on the management port. Both requests should return HTTP/200 if ok, 000 otherwise.
  - if *curl* is not installed, uses the built-in Linux TCP socket to run TCP SYN checks on both the HA communication port (default is 17355) and the management port (default is 7626) on the Node hostname.
- Checks for **firewall configuration**. Currently only supports *firewalld* (RHEL) and a netstat test.
  - The **netstat part** will run a *netstat* command to check if the JVM listening socket is active (listening on port 9000). If *netstat* is not installed, it will skip this test.

    - The **firewalld part** will check if the HTTP and HTTPS services are opened in the firewall and if it detected a HA configuration, it will check if the HA ports (both of them) are allowed through the firewalld. If *firewalld* is not installed or not active, it will skip this test.
- Checks if **IPv6** is active in every network interface and throws a warning if it is the case (specifying the interface with IPv6 turned on).

TLS checks

- Checks for existence and permissions of the **Horizon server certificate** file: the permissions are expected to be at least 640 and the file is supposed to belong to the nginx group.
- Parses the **Horizon server certificate** file: it should be constituted of the actual TLS server certificate first, then of every certificate of the trust chain (order being leaf to root). It throws a warning if the certificate is self-signed or raises an error if the trust chain has not been imported. It otherwise tries to reconstitute the certificate trust chain via the *openssl verify* command, and throws an error if it cannot.
- Parses the **Horizon server certificate** file and checks if the **Horizon hostname** is present in the **SAN DNS names** of the certificate, throws an error if it is not there.

Log packing option

If the Horizon doctor is launched with the *-l option*, it will pack the logs of the last 7 days (in */opt/horizon/var/log*) as well as the startup logs (the */var/log/horizon/horizon.log* file) and create a tar archive.

The *-l option* accepts an optional parameter that should be an integer (1-99) and will pack the logs of the last n days instead, as well as the startup logs.

Note that the **Horizon doctor** will still perform all of its check; the log packing is done at the very end of the program.

Example of call to pack the logs of the last 7 days:

```
$  horizon-doctor -l
```

Example of call to pack the logs of the last 30 days:

```
$ horizon-doctor -l 30
```

Saving the doctor's output

If the Horizon doctor is launched with the *-o option*, it will perform all of its checks and save the output in the specified file instead of displaying it into the stdout (default is the command line interface).

If you use the option, you must provide a filepath in a writable directory.

Example of call to save the output in a file named `horizon-doctor.out` instead of the stdout:

```
$ horizon-doctor -o horizon-doctor.out
```

Help menu

To display Horizon doctor's help menu, use the `-h` option.

Additional checks

- Ensure that you are using an up-to-date web browser when trying to access the Horizon web interface.
- Ensure that Javascript in turned on in your web browser.
- Ensure that your user machine can access the server where Horizon was installed.
- If several hostnames have been set up for the Horizon interface, ensure that every single one of them is present in the TLS certificate SAN DNS names.

1.10. Logging

Horizon emits logs of two main categories:

- technical logs, which are emitted on the default log location. They are used for troubleshooting the app and monitored for bugs;
- events, which are stored in database and can be viewed in-app. They allow auditing actions on the platform.

Default log location

The default log location varies depending on your deployment mode:

RPM

Horizon defines a default rolling file appender named **RUN**. This appender keeps the technical logs for 30 days into files with the following naming convention:

```
horizon.log-<yyyy-MM-dd>.log
```

Those files are available under the `/opt/horizon/var/log` directory.

Debian

Horizon defines a default rolling file appender named **RUN**. This appender keeps the technical logs for 30 days into files with the following naming convention:

```
horizon.log-<yyyy-MM-dd>.log
```

Those files are available under the `/opt/horizon/var/log` directory.

Kubernetes

By default, Horizon logs are written to `stdout`. It is currently not possible to write logs to any other destination.

Log format

By default, Horizon logs are formatted to be human readable using the following format:

```
%date{yyyy-MM-dd HH:mm:ss} - [%logger] - [%traceID] - [%level] -
%message%n%xException{full}
```

This format can be customized:

RPM

In the `/opt/horizon/etc/horizon-logback.xml` file, update the appender's `<encoder>` key:

```
<encoder>
    <pattern>%date{yyyy-MM-dd HH:mm:ss ZZZZ} | %message</pattern>
</encoder>
```

Debian

In the `/opt/horizon/etc/horizon-logback.xml` file, update the appender's `<encoder>` key:

```
<encoder>
    <pattern>%date{yyyy-MM-dd HH:mm:ss ZZZZ} | %message</pattern>
</encoder>
```

Kubernetes

Add the following keys to your `values.yaml` file:

```
logback:
  pattern: "%date{yyyy-MM-dd HH:mm:ss ZZZZ} | %message"
```



All the available patterns can be found in the logback docs.

It's also possible to configure Horizon to emit JSON structured logs, which can be easier to parse by machines. To do so:

RPM

Edit the `/opt/horizon/etc/horizon-logback.xml` file to either:

- send the logs to a syslog server:

  An example for a syslog server on 192.168.1.2 and the logs processed by the LOCAL6 facility

```
<conversionRule conversionWord="syslogStart"
converterClass="ch.qos.logback.classic.pattern.SyslogStartConverter"/>
<appender name="JSON_SYSLOG"
class="net.logstash.logback.appender.LogstashUdpSocketAppender">
    <host>192.168.1.2</host>
    <port>514</port>
    <layout class="net.logstash.logback.layout.LogstashLayout">
      <prefix class="ch.qos.logback.classic.PatternLayout">
        <pattern>%syslogStart{LOCAL6}</pattern>
      </prefix>
      <fieldNames>
        <timestamp>time</timestamp>
```

```
        <logger>logger</logger>
        <thread>thread</thread>
        <level>severity</level>
        <stackTrace>exception</stackTrace>
      </fieldNames>
      <customFields>{"app":"horizon",
"hostname":"${HOSTNAME}"}</customFields>
    </layout>
</appender>
```

- or send the logs to the local console:

```
<appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
    <encoder class="net.logstash.logback.encoder.LogstashEncoder">
        <fieldNames>
          <timestamp>time</timestamp>
          <logger>logger</logger>
          <thread>thread</thread>
          <level>severity</level>
          <stackTrace>exception</stackTrace>
        </fieldNames>
        <customFields>{"app":"horizon",
"hostname":"${HOSTNAME}"}</customFields>
    </encoder>
  </appender>
```

Then, update any logger with the appender ref and ensure that the log level is not OFF:

```
<logger name="event" level="INFO">
    <appender-ref ref="JSON_SYSLOG"/>
</logger>
```

Debian

Edit the `/opt/horizon/etc/horizon-logback.xml` file to either:

- send the logs to a syslog server:

  An example for a syslog server on 192.168.1.2 and the logs processed by the LOCAL6 facility

```
<conversionRule conversionWord="syslogStart"
converterClass="ch.qos.logback.classic.pattern.SyslogStartConverter"/>
<appender name="JSON_SYSLOG"
class="net.logstash.logback.appender.LogstashUdpSocketAppender">
    <host>192.168.1.2</host>
    <port>514</port>
    <layout class="net.logstash.logback.layout.LogstashLayout">
      <prefix class="ch.qos.logback.classic.PatternLayout">
        <pattern>%syslogStart{LOCAL6}</pattern>
```

```
      </prefix>
      <fieldNames>
        <timestamp>time</timestamp>
        <logger>logger</logger>
        <thread>thread</thread>
        <level>severity</level>
        <stackTrace>exception</stackTrace>
      </fieldNames>
      <customFields>{"app":"horizon",
"hostname":"${HOSTNAME}"}</customFields>
    </layout>
</appender>
```

- or send the logs to the local console:

```
<appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
    <encoder class="net.logstash.logback.encoder.LogstashEncoder">
        <fieldNames>
          <timestamp>time</timestamp>
          <logger>logger</logger>
          <thread>thread</thread>
          <level>severity</level>
          <stackTrace>exception</stackTrace>
        </fieldNames>
        <customFields>{"app":"horizon",
"hostname":"${HOSTNAME}"}</customFields>
    </encoder>
  </appender>
```

Then, update any logger with the appender ref and ensure that the log level is not OFF:

```
<logger name="event" level="INFO">
    <appender-ref ref="JSON_SYSLOG"/>
</logger>
```

Kubernetes

Update the `values.yaml` file to set the log format:

```
logFormat: json
```

Horizon should now start producing JSON logs such as :

```
{
    "time": "2023-08-16T16:12:54.481+02:00",
    "@version": "1",
```

```
    "message": "[Actor pkimanager] - Registering PKI Queue 'slowed-queue' (cluster
wide: 'false')",
    "logger": "actors.pki.PKIManagerActor",
    "thread": "application-blocking-io-dispatcher-43",
    "severity": "INFO",
    "level_value": 20000,
    "HOSTNAME": "horizon.evertrust",
    "application.home": "/opt/horizon",
    "kamonSpanId": "c5a74b959971c7ee",
    "kamonTraceId": "b1ccb54c9eb7e493",
    "kamonSpanName": "/ui",
    "app": "horizon",
    "hostname": "horizon.evertrust"
}
```


Additional loggers

Sometimes, for debugging purposes, you'll be asked to enable a specific logger or change the logging level of an existing one. To do so:

RPM

Edit the `/opt/horizon/etc/horizon-logback.xml` file, and add or edit the logger you wish to change:

```
<logger name="<logger name>" level="<log level>">
    <appender-ref ref="<appender name>"/>
</logger>
```

Debian

Edit the `/opt/horizon/etc/horizon-logback.xml` file, and add or edit the logger you wish to change:

```
<logger name="<logger name>" level="<log level>">
    <appender-ref ref="<appender name>"/>
</logger>
```

Kubernetes

Override the `logback.loggers` array in the `values.yaml` file:

```
logback:
  loggers:
    - name: <logger name>
      level: <log level>
```

Log events to the default log location

Events are produced by Horizon and typically stored in database. For compliance reasons (for example when sending logs to an external processor), you might want to also log events to the default log location.

RPM

In the `/opt/horizon/etc/horizon-logback.xml` file, change the level of the `events` logger from `OFF` to `INFO`:

```
<logger name="events" level="INFO">
    <appender-ref ref="<appender name>"/>
</logger>
```

Debian

In the `/opt/horizon/etc/horizon-logback.xml` file, change the level of the `events` logger from `OFF` to `INFO`:

```
<logger name="events" level="INFO">
    <appender-ref ref="<appender name>"/>
</logger>
```

Kubernetes

Override the `logback.loggers` array in the `values.yaml` file to add :

```
logback:
  loggers:
    - name: json_events
      level: info
```

Sending logs to an external processor

Horizon logs can be sent to an external source (such as a SIEM) using logback.

RPM

In the `/opt/horizon/etc/horizon-logback.xml` file, edit the appender named `SYSLOG` to change the IP address for the syslogHost to redirect to your own syslog server. As an example, if your syslog server is on 192.168.1.2 and the Horizon logs must be processed by the LOCAL6 facility, the syslog appender should look like this:

```
<appender name="SYSLOG" class="ch.qos.logback.classic.net.SyslogAppender">
    <syslogHost>192.168.1.2</syslogHost>
```

```
    <facility>LOCAL6</facility>
    <suffixPattern>%msg%n</suffixPattern>
</appender>
```

Then, update any logger with the SYSLOG appender ref and ensure that the log level is set to "INFO":

```
<logger name="event" level="INFO">
    <appender-ref ref="SYSLOG"/>
</logger>
```

Debian

In the `/opt/horizon/etc/horizon-logback.xml` file, edit the appender named `SYSLOG` to change the IP address for the syslogHost to redirect to your own syslog server. As an example, if your syslog server is on 192.168.1.2 and the Horizon logs must be processed by the LOCAL6 facility, the syslog appender should look like this:

```
<appender name="SYSLOG" class="ch.qos.logback.classic.net.SyslogAppender">
    <syslogHost>192.168.1.2</syslogHost>
    <facility>LOCAL6</facility>
    <suffixPattern>%msg%n</suffixPattern>
</appender>
```

Then, update any logger with the SYSLOG appender ref and ensure that the log level is set to "INFO":

```
<logger name="event" level="INFO">
    <appender-ref ref="SYSLOG"/>
</logger>
```

Kubernetes

On Kubernetes, logging should done by containers to `stdout` and managed at the cluster level by a log collector such as Grafana Alloy or Vector.

As an example of a mature and battle-tested log processing pipeline, take a look at our Cloud log management document.

1.11. Multi-Tenancy

Activation

In order to activate multi-tenancy on an instance, a specific license must be used. Contact the Evertrust team for access.



Multi-tenancy must be enabled when provisioning a new instance, and cannot be disabled afterwards. This means that switching an instance between multi-tenancy modes is not possible.

Tenant isolation

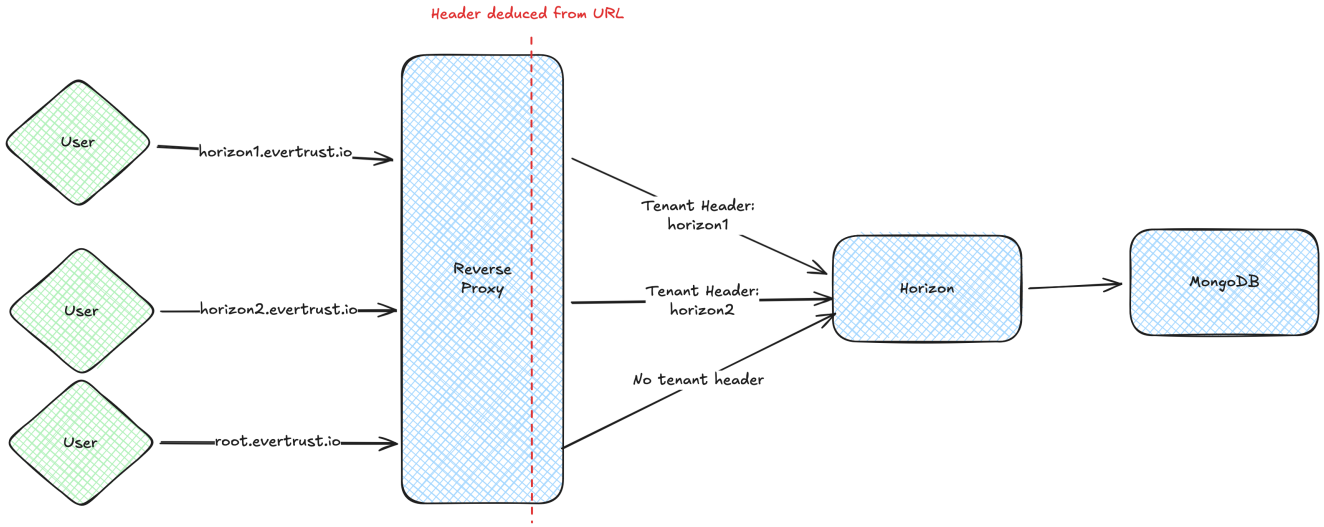
Horizon uses logical isolation, meaning that a single database is used for all tenants of an instance.

To discriminate between tenants, Horizon uses the `x-tenant` header. Using a header allows for a highly customizable tenant access model. It should be enabled on the reverse proxy.



No `x-tenant` header means that Horizon will load the root tenant context

Here is an example architecture:



In this architecture, the content of the `x-tenant` header is set by the reverse proxy based on the prefix before the `evertrust.io` domain, with a specific rule for the root domain.

Root tenant

The root tenant is the base tenant of a multi-tenant instance. It has limited capabilities, mostly linked to user and tenant management.



The root tenant cannot access any of its tenant data, and can only manage tenants, not their data.

Tenant management

For tenant management, please refer to the Administration Guide

Specific configuration

On a multi-tenant instance, some system queues are shared across tenants. If the instance sees

heavy usage, the following parameters should be altered:

- The CRL cache synchronization parallelism and size
- The CRL database synchronization parallelism and size
- The default PKI queue parallelism and size

Secrets

On a multi-tenant instance, secrets for a tenant (credentials, private keys, ...) are encrypted using its own Tink keyset. This keyset is itself encrypted with the instance keyset. This ensures secrets are completely isolated from one tenant to another.

Logging

On a multi-tenant instance, the tenant information can be added to logs.

RPM

The tenant information is already added to standard logs if you are using the JSON or SYSLOG encoder.

If you are using the standard line format, you can add the tenant information using the `%contextTag{tenant}` keyword anywhere in the pattern.

To edit the pattern, follow the standard steps.

Debian

The tenant information is already added to standard logs if you are using the JSON or SYSLOG encoder.

If you are using the standard line format, you can add the tenant information using the `%contextTag{tenant}` keyword anywhere in the pattern.

To edit the pattern, follow the standard steps.

Kubernetes

The tenant information is already added to standard logs if you are using the `json` format.

However, if you are using the `json_events` logger, the tenant information is not added by default.

To add it, one must modify the contents of the `/horizon/etc/logback.xml` file.

The following modification should be applied to the contents of the `json_events` appender:

```
<pattern>
    {
        "event": "#asJson{%message}",
        "tenant": "%contextTag{tenant}", ①
```

```
        "type": "event"
    }
</pattern>
```

① This line adds the `tenant` key to the JSON event logs, and outputs an empty string when no tenant is linked to this event.

1.12. Endpoint configuration

Basic configuration

The basic configuration sets allowed hosts for all protocols using the `horizon-config` utility in RPM mode, and using the `ALLOWED_HOSTS` helm parameters

Advanced configuration

Endpoints can be configured to only allow certain capabilities using the `horizon.endpoints` config parameter.

The format is the following:

```
horizon.endpoints = [{
    # Hostname to allow
    host = "host.evertrust"
    # Allow configuration endpoints - default: false
    configuration = true
    # Allow event endpoints - default: false
    events = true
    # Allow discovery feed endpoints - default: false
    discovery = true
    # Allow WebRA endpoints - default: false
    webra = true
    # Allow WCCE endpoints - default: false
    wcce = true
    # Allow ACME endpoints - default: false
    acme = true
    # Allow EST endpoints - default: false
    est = true
    # Allow SCEP endpoints - default: false
    scep = true
    # Allow SCIM endpoints - default: false
    scim = true
    # Allow JAMF and Intune endpoints - default: false
    mdm = true
    # Allow DCV lifecycle endpoints - default: false
    dcv = true
}, ...]
```



When `horizon.endpoints` is set, the hosts allowed with basic configuration are ignored

The following details each route that is authorized by the above capabilities.

EST

```
GET    /.well-known/est/:profile/cacerts
POST   /.well-known/est/:profile/simpleenroll
POST   /.well-known/est/:profile/simplereenroll
```

SCEP

```
GET    /certsrv/:profile/mscep_admin
GET    /certsrv/:profile/mscep_admin/*restUri
GET    /certsrv/:profile/mscep
POST   /certsrv/:profile/mscep
GET    /certsrv/:profile/mscep/*restUri
POST   /certsrv/:profile/mscep/*restUri
GET    /certSrv/:profile/mscep_admin
GET    /certSrv/:profile/mscep_admin/*restUri
GET    /certSrv/:profile/mscep
POST   /certSrv/:profile/mscep
GET    /certSrv/:profile/mscep/*restUri
POST   /certSrv/:profile/mscep/*restUri
GET    /scep/:profile/pkiclient.exe
POST   /scep/:profile/pkiclient.exe
GET    /scep/:profile/scepRA
```

MDM

```
GET    /intune/:profile/pkiclient.exe
POST   /intune/:profile/pkiclient.exe
GET    /intune/:profile/scepRA
GET    /jamf/:profile/mscep_admin
GET    /jamf/:profile/mscep_admin/*restUri
GET    /jamf/:profile/mscep
POST   /jamf/:profile/mscep
GET    /jamf/:profile/mscep/*restUri
POST   /jamf/:profile/mscep/*restUri
GET    /jamf/:profile/scepRA
```

ACME

```
GET    /acme/:profile/directory
```

```
GET    /acme/:profile/new-nonce
HEAD   /acme/:profile/new-nonce
POST   /acme/:profile/new-acct
POST   /acme/acct/:profile/:accountId
POST   /acme/:profile/key-change
POST   /acme/:profile/new-order
GET    /acme/acct/:profile/:accountId/order/:orderId/finalize
GET    /acme/order/:profile/:orderId
POST   /acme/order/:profile/:orderId
POST   /acme/acct/:profile/:accountId/order/:orderId
POST   /acme/acct/:profile/:accountId/order/:orderId/finalize
GET    /acme/acct/:profile/:accountId/orders
POST   /acme/acct/:profile/:accountId/orders
POST   /acme/acct/:profile/:accountId/*restUri
GET    /acme/authz/:profile/:id/:index
POST   /acme/authz/:profile/:id/:index
POST   /acme/authz/:profile/:id/:index/:challengeType
GET    /acme/authz/:profile/:id/:index/:challengeType
GET    /acme/cert/:profile/:orderId
POST   /acme/cert/:profile/:orderId
POST   /acme/:profile/revoke-cert
```

WCCE

```
POST   /api/v1/wcce/enroll
GET    /api/v1/wcce/exchanges/:profile
```

WEBRA

```
GET    /api/v1/certificates/$id<[0-9a-fA-F]{24}>
GET    /api/v1/certificates/:pem
POST   /api/v1/certificates/find
POST   /api/v1/certificates/
POST   /api/v1/certificates/aggregate
POST   /api/v1/certificates/csv
POST   /api/v1/certificates/search
GET    /api/v1/certificates/search/dictionary
PATCH  /api/v1/certificates/run/$id<[0-9a-fA-F]{24}>/:triggerName/:event
POST   /api/v1/challenge/submit
POST   /api/v1/requests/aggregate
POST   /api/v1/requests/approve
POST   /api/v1/requests/cancel
POST   /api/v1/requests/csv
POST   /api/v1/requests/deny
GET    /api/v1/requests/profiles
POST   /api/v1/requests/search
POST   /api/v1/requests/submit
POST   /api/v1/requests/template
```

```
GET    /api/v1/requests/:id
GET    /api/v1/requests/search/dictionary
GET    /api/v1/rfc5280/crl/:pem
POST   /api/v1/rfc5280/crl
GET    /api/v1/rfc5280/pkcs10/:pem
POST   /api/v1/rfc5280/pkcs10
POST   /api/v1/rfc5280/pkcs12
GET    /api/v1/rfc5280/x509/:pem
POST   /api/v1/rfc5280/x509
GET    /api/v1/rfc5280/tc/:pem
POST   /api/v1/rfc5280/tc
POST   /api/v1/crypto/detect
GET    /api/v1/openssh/:base64
POST   /api/v1/openssh/
GET    /api/v1/rfc3161/:base64
POST   /api/v1/rfc3161/
GET    /api/v1/rfc6960/:base64
POST   /api/v1/rfc6960/
PATCH  /api/v1/security/identity/locals/
POST   /api/v1/security/identity/locals/password
GET    /api/v1/security/identity/locals/password/:identifier
POST   /api/v1/security/identity/locals/generate-password
GET    /api/v1/security/identity/providers/dynamic/enabled
GET    /api/v1/security/passwordpolicies/:name/generate
GET    /api/v1/security/principals/self
GET    /api/v1/security/principals/authenticate
GET    /api/v1/security/principals/logout
GET    /api/v1/security/principals/queries
GET    /api/v1/security/principals/queries/:name
POST   /api/v1/security/principals/queries
DELETE /api/v1/security/principals/queries/:name
GET    /api/v1/security/principals/dashboards
GET    /api/v1/security/principals/dashboards/:name
POST   /api/v1/security/principals/dashboards
PUT    /api/v1/security/principals/dashboards
DELETE /api/v1/security/principals/dashboards/:name
POST   /api/v1/security/principals/preferences
GET    /api/v1/security/principals/dictionary
GET    /api/v1/trustchains/
GET    /api/v1/trustchains/:anchor
GET    /api/v1/ui/
GET    /api/v1/endpoints/
GET    /api/v1/teams/:name/members
POST   /api/v1/teams/:name/members
DELETE /api/v1/teams/:name/members
GET    /reports/:uuid
GET    /api/v1/reports/:reportName
```

EVENTS

```
POST   /api/v1/discovery/events/search
POST   /api/v1/discovery/events/csv
GET    /api/v1/discovery/events/:id
GET    /api/v1/discovery/events/search/dictionary
GET    /api/v1/events/integrity/run
GET    /api/v1/events/integrity/
POST   /api/v1/events/search
POST   /api/v1/events/csv
GET    /api/v1/events/:id
GET    /api/v1/events/search/dictionary
GET    /api/v1/rfc5280/crl/:pem
POST   /api/v1/rfc5280/crl
GET    /api/v1/rfc5280/pkcs10/:pem
POST   /api/v1/rfc5280/pkcs10
POST   /api/v1/rfc5280/pkcs12
GET    /api/v1/rfc5280/x509/:pem
POST   /api/v1/rfc5280/x509
GET    /api/v1/rfc5280/tc/:pem
POST   /api/v1/rfc5280/tc
POST   /api/v1/crypto/detect
GET    /api/v1/openssh/:base64
POST   /api/v1/openssh/
GET    /api/v1/rfc3161/:base64
POST   /api/v1/rfc3161/
GET    /api/v1/rfc6960/:base64
POST   /api/v1/rfc6960/
PATCH  /api/v1/security/identity/locals/
POST   /api/v1/security/identity/locals/password
GET    /api/v1/security/identity/locals/password/:identifier
POST   /api/v1/security/identity/locals/generate-password
GET    /api/v1/security/identity/providers/dynamic/enabled
GET    /api/v1/security/passwordpolicies/:name/generate
GET    /api/v1/security/principals/self
GET    /api/v1/security/principals/authenticate
GET    /api/v1/security/principals/logout
GET    /api/v1/security/principals/queries
GET    /api/v1/security/principals/queries/:name
POST   /api/v1/security/principals/queries
DELETE /api/v1/security/principals/queries/:name
POST   /api/v1/security/principals/preferences
GET    /api/v1/security/principals/dictionary
GET    /api/v1/ui/
GET    /api/v1/endpoints/
```

CONFIGURATION

```
DELETE /api/v1/acme/eab/:name
```

```
POST   /api/v1/acme/eab
POST   /api/v1/acme/eab/search
POST   /api/v1/acme/eab/:name/renew
POST   /api/v1/acme/eab/:eabName/status
GET    /api/v1/acme/eab/:name
PUT    /api/v1/acme/eab
DELETE /api/v1/acme/accounts/:accountId
GET    /api/v1/acme/accounts/:accountId
POST   /api/v1/acme/accounts/:accountId/status
POST   /api/v1/acme/accounts/search
GET    /api/v1/acme/orders/:orderId
POST   /api/v1/acme/orders/account/:accountId
POST   /api/v1/acme/eab-policies/
POST   /api/v1/acme/eab-policies/list
PUT    /api/v1/acme/eab-policies/
GET    /api/v1/acme/eab-policies/:name
DELETE /api/v1/acme/eab-policies/:name
GET    /api/v1/adoc/
GET    /api/v1/automation/executions/
GET    /api/v1/automation/executions/:name
POST   /api/v1/automation/executions/
PUT    /api/v1/automation/executions/
DELETE /api/v1/automation/executions/:name
GET    /api/v1/automation/policies/
GET    /api/v1/automation/policies/:name
POST   /api/v1/automation/policies/
PUT    /api/v1/automation/policies/
DELETE /api/v1/automation/policies/:name
GET    /api/v1/cas/
GET    /api/v1/cas/:name
POST   /api/v1/cas/
PUT    /api/v1/cas/
DELETE /api/v1/cas/:name
GET    /api/v1/caches/crls
GET    /api/v1/caches/crls/:ca
GET    /api/v1/certificate/grading/policies/
GET    /api/v1/certificate/grading/policies/:name
GET    /api/v1/certificate/grading/policies/:policy/explain/:input
POST   /api/v1/certificate/grading/policies/:policy/explain
GET    /api/v1/certificate/grading/policies/:policy/run
GET    /api/v1/certificate/grading/rulesets/
GET    /api/v1/certificate/grading/rulesets/:name
GET    /api/v1/certificate/grading/rulesets/:ruleset/explain/:input
POST   /api/v1/certificate/grading/rulesets/:ruleset/explain
GET    /api/v1/certificate/labels/
GET    /api/v1/certificate/labels/:name
POST   /api/v1/certificate/labels/
PUT    /api/v1/certificate/labels/
DELETE /api/v1/certificate/labels/:name
GET    /api/v1/certificate/profiles/
GET    /api/v1/certificate/profiles/:name
```

```
POST   /api/v1/certificate/profiles/
PUT    /api/v1/certificate/profiles/
DELETE /api/v1/certificate/profiles/:name
GET    /api/v1/datasources/
GET    /api/v1/datasources/:name
POST   /api/v1/datasources/
PUT    /api/v1/datasources/
DELETE /api/v1/datasources/:name
PATCH  /api/v1/datasources/
POST   /api/v1/datasource/flows/
POST   /api/v1/datasource/flows/template
POST   /api/v1/templatestring/playground
GET    /api/v1/discovery/campaigns/
GET    /api/v1/discovery/campaigns/:name
POST   /api/v1/discovery/campaigns/
PUT    /api/v1/discovery/campaigns/
DELETE /api/v1/discovery/campaigns/:name
PATCH  /api/v1/discovery/campaigns/:name
GET    /api/v1/pki/queues/
GET    /api/v1/pki/queues/:name
POST   /api/v1/pki/queues/
PUT    /api/v1/pki/queues/
DELETE /api/v1/pki/queues/:name
GET    /api/v1/pki/connectors/
GET    /api/v1/pki/connectors/:name
POST   /api/v1/pki/connectors/
PUT    /api/v1/pki/connectors/
DELETE /api/v1/pki/connectors/:name
PATCH  /api/v1/pki/connectors/connect
PATCH  /api/v1/pki/connectors/materials
GET    /api/v1/proxy/httpproxies/
GET    /api/v1/proxy/httpproxies/:name
POST   /api/v1/proxy/httpproxies/
PUT    /api/v1/proxy/httpproxies/
DELETE /api/v1/proxy/httpproxies/:name
GET    /api/v1/rfc5280/crl/:pem
POST   /api/v1/rfc5280/crl
GET    /api/v1/rfc5280/pkcs10/:pem
POST   /api/v1/rfc5280/pkcs10
POST   /api/v1/rfc5280/pkcs12
GET    /api/v1/rfc5280/x509/:pem
POST   /api/v1/rfc5280/x509
GET    /api/v1/rfc5280/tc/:pem
POST   /api/v1/rfc5280/tc
POST   /api/v1/crypto/detect
GET    /api/v1/openssh/:base64
POST   /api/v1/openssh/
GET    /api/v1/rfc3161/:base64
POST   /api/v1/rfc3161/
GET    /api/v1/rfc6960/:base64
POST   /api/v1/rfc6960/
```

```
GET    /api/v1/security/identity/locals/
GET    /api/v1/security/identity/locals/:identifier
POST   /api/v1/security/identity/locals/
PUT    /api/v1/security/identity/locals/
DELETE /api/v1/security/identity/locals/:identifier
PATCH  /api/v1/security/identity/locals/
POST   /api/v1/security/identity/locals/password
GET    /api/v1/security/identity/locals/password/:identifier
POST   /api/v1/security/identity/locals/generate-password
GET    /api/v1/security/identity/providers/
GET    /api/v1/security/identity/providers/:name
POST   /api/v1/security/identity/providers/
PUT    /api/v1/security/identity/providers/
DELETE /api/v1/security/identity/providers/:name
GET    /api/v1/security/identity/providers/dynamic/enabled
POST   /api/v1/security/identity/providers/search
GET    /api/v1/security/passwordpolicies/
GET    /api/v1/security/passwordpolicies/:name
GET    /api/v1/security/passwordpolicies/:name/generate
POST   /api/v1/security/passwordpolicies/
PUT    /api/v1/security/passwordpolicies/
DELETE /api/v1/security/passwordpolicies/:name
GET    /api/v1/security/principals/self
GET    /api/v1/security/principals/authenticate
GET    /api/v1/security/principals/logout
GET    /api/v1/security/principals/queries
GET    /api/v1/security/principals/queries/:name
POST   /api/v1/security/principals/queries
DELETE /api/v1/security/principals/queries/:name
POST   /api/v1/security/principals/preferences
GET    /api/v1/security/principals/dictionary
GET    /api/v1/security/principalinfos/:identifier
POST   /api/v1/security/principalinfos/
PUT    /api/v1/security/principalinfos/
DELETE /api/v1/security/principalinfos/:identifier
POST   /api/v1/security/principalinfos/search
GET    /api/v1/security/roles/
GET    /api/v1/security/roles/:name
POST   /api/v1/security/roles/
PUT    /api/v1/security/roles/
DELETE /api/v1/security/roles/:name
GET    /api/v1/security/roles/:name/members
POST   /api/v1/security/roles/:name/members
DELETE /api/v1/security/roles/:name/members
GET    /api/v1/security/scim/profiles/
GET    /api/v1/security/scim/profiles/:name
PUT    /api/v1/security/scim/profiles/
DELETE /api/v1/security/scim/profiles/:name
POST   /api/v1/security/scim/profiles/
GET    /api/v1/security/teams/
GET    /api/v1/security/teams/:name
```

```
POST   /api/v1/security/teams/
PUT    /api/v1/security/teams/
DELETE /api/v1/security/teams/:name
PATCH  /api/v1/security/teams/:previousTeam/:newTeam
GET    /api/v1/security/teams/:name/members
POST   /api/v1/security/teams/:name/members
DELETE /api/v1/security/teams/:name/members
GET    /api/v1/security/tenants/
GET    /api/v1/security/tenants/:name
GET    /api/v1/security/tenants/:name/licenses
POST   /api/v1/security/tenants/:name/reset
POST   /api/v1/security/tenants/:name/restore
POST   /api/v1/security/tenants/
PUT    /api/v1/security/tenants/
DELETE /api/v1/security/tenants/:name
GET    /api/v1/security/credentials/
GET    /api/v1/security/credentials/:name
POST   /api/v1/security/credentials/
PUT    /api/v1/security/credentials/
DELETE /api/v1/security/credentials/:name
GET    /api/v1/security/service-accounts/
POST   /api/v1/security/service-accounts/
PUT    /api/v1/security/service-accounts/
GET    /api/v1/security/service-accounts/:name
DELETE /api/v1/security/service-accounts/:name
GET    /api/v1/scheduler/tasks/
GET    /api/v1/scheduler/tasks/:name/run
GET    /api/v1/scheduler/tasks/:name
POST   /api/v1/scheduler/tasks/
PUT    /api/v1/scheduler/tasks/
DELETE /api/v1/scheduler/tasks/:name
GET    /api/v1/thirdparty/connectors/
GET    /api/v1/thirdparty/connectors/:name
POST   /api/v1/thirdparty/connectors/
PUT    /api/v1/thirdparty/connectors/
DELETE /api/v1/thirdparty/connectors/:name
PATCH  /api/v1/thirdparty/connectors/:name
GET    /api/v1/triggers/
GET    /api/v1/triggers/:name
POST   /api/v1/triggers/
PUT    /api/v1/triggers/
DELETE /api/v1/triggers/:name
PATCH  /api/v1/triggers/
GET    /api/v1/wcce/forests
GET    /api/v1/wcce/forests/:name
POST   /api/v1/wcce/forests
PUT    /api/v1/wcce/forests
DELETE /api/v1/wcce/forests/:name
GET    /api/v1/system/configuration/
GET    /api/v1/system/configuration/:type
PUT    /api/v1/system/configuration/
```

```
GET    /api/v1/system/configurations/export
POST   /api/v1/system/configurations/export
POST   /api/v1/system/configurations/import
GET    /api/v1/system/storages/
GET    /api/v1/system/storages/:name
POST   /api/v1/system/storages/
PUT    /api/v1/system/storages/
DELETE /api/v1/system/storages/:name
GET    /api/v1/ui/
POST   /api/v1/ui/cr/format
GET    /api/v1/endpoints/
GET    /api/v1/analytics/certificates
PATCH  /api/v1/analytics/certificates
DELETE /api/v1/analytics/certificates
GET    /api/v1/analytics/events
PATCH  /api/v1/analytics/events
DELETE /api/v1/analytics/events
GET    /api/v1/analytics/discovery/events
PATCH  /api/v1/analytics/discovery/events
DELETE /api/v1/analytics/discovery/events
GET    /api/v1/dcv/policies/
GET    /api/v1/dcv/policies/:name
POST   /api/v1/dcv/policies/
PUT    /api/v1/dcv/policies/
DELETE /api/v1/dcv/policies/:name
GET    /api/v1/dcv/providers/
GET    /api/v1/dcv/providers/:name
POST   /api/v1/dcv/providers/
PUT    /api/v1/dcv/providers/
DELETE /api/v1/dcv/providers/:name
GET    /api/v1/dcv/provisioners/
GET    /api/v1/dcv/provisioners/:name
POST   /api/v1/dcv/provisioners/
PUT    /api/v1/dcv/provisioners/
DELETE /api/v1/dcv/provisioners/:name
DELETE /api/v1/reports/:uuid
GET    /api/v1/reports/
GET    /api/v1/archives/
GET    /api/v1/archives/:name
GET    /api/v1/archives/:name/download
GET    /api/v1/archives/:name/retry
GET    /api/v1/archives/:name/cancel
POST   /api/v1/archives/
POST   /api/v1/archives/count
DELETE /api/v1/archives/:name
GET    /api/v1/system/terms-of-services/
GET    /api/v1/system/terms-of-services/:name
POST   /api/v1/system/terms-of-services/
PUT    /api/v1/system/terms-of-services/
DELETE /api/v1/system/terms-of-services/:name
```

SCIM

```
GET    /security/scim/:scimProfile/ServiceProviderConfig
GET    /security/scim/:scimProfile/ResourceTypes
GET    /security/scim/:scimProfile/Schemas
GET    /security/scim/:scimProfile/Schemas/:schema
GET    /security/scim/:scimProfile/Users
GET    /security/scim/:scimProfile/Users/:identifier
POST   /security/scim/:scimProfile/Users
PATCH  /security/scim/:scimProfile/Users/:userName
PUT    /security/scim/:scimProfile/Users/:identifier
DELETE /security/scim/:scimProfile/Users/:identifier
GET    /security/scim/:scimProfile/Groups/:groupName
GET    /security/scim/:scimProfile/Groups
PATCH  /security/scim/:scimProfileName/Groups/:GroupName
```

DCV

```
GET    /api/v1/dcv/lifecycle/policies/
GET    /api/v1/dcv/lifecycle/policies/:name
POST   /api/v1/dcv/lifecycle/policies/:name/run
POST   /api/v1/dcv/lifecycle/policies/:name/run/:domain
POST   /api/v1/dcv/lifecycle/policies/:name/cancel
POST   /api/v1/dcv/lifecycle/events/:policy
POST   /api/v1/dcv/lifecycle/events/:policy/:domain
```

DISCOVERY

```
POST   /api/v1/discovery/feed/
GET    /api/v1/discovery/feed/:name
PUT    /api/v1/discovery/feed/
DELETE /api/v1/discovery/feed/:campaign/:id
```

1.13. Advanced configuration

Some technical configurations can be applied to an instance directly in its configuration file. This should be used carefully as it may cause things to break.

Injecting advanced configuration

RPM

On VMs, you have access to the `/opt/horizon/etc/conf.d/horizon-extra.conf` file. For each parameter you wish to override, create a newline and use the following syntax:

```
<parameter>=<value>
```

As an example, if you want to modify the file extension that DER certificates will have when sent as email attachments and set it to CRT, you need to add:

```
horizon.notification.mail.attachment.extension.der="crt"
```

After modifying the file, restart the Horizon service:

```
$ systemctl restart horizon
```



One added line means one modified option, you need to add as many lines at the end of the file as there are values that you want to override.

Debian

On VMs, you have access to the `/opt/horizon/etc/conf.d/horizon-extra.conf` file. For each parameter you wish to override, create a newline and use the following syntax:

```
<parameter>=<value>
```

As an example, if you want to modify the file extension that DER certificates will have when sent as email attachments and set it to CRT, you need to add:

```
horizon.notification.mail.attachment.extension.der="crt"
```

After modifying the file, restart the Horizon service:

```
$ systemctl restart horizon
```



One added line means one modified option, you need to add as many lines at the end of the file as there are values that you want to override.

Kubernetes

The Horizon container provides a bundled `application.conf` file that is mostly configured through environment variables. To modify low-level behavior of Horizon that are not accessible through an environment variable, use the `extraConfig` value in your `values.yaml` file to update specific settings:

```
extraConfig: |
    horizon {
      notification.mail.attachment.extension.der = "der"
```

```
}
```

Extra configurations are appended at the end of the config file, overriding any previously set config value.

Docker

The Horizon container provides a bundled `application.conf` file that is mostly configured through environment variables. To modify low-level behavior of Horizon that are not accessible through an environment variable, you can mount custom configuration files, giving you full control over how Horizon behaves.

The mounted folder :

- MUST contain an `pekko.conf` file configuring the Pekko cluster. See the reference config to get an idea over what's configurable.
- CAN contain a `application.conf` file containing any extra config options unrelated to clustering.

A typical Docker command would then be :

```
$ docker run \
   -v [configurationPath]:/opt/horizon/etc/:rw \
   ...
   registry.evertrust.io/horizon:2.12.x
```

Startup scripts

Sometimes, you'll want to run scripts each time the container starts up in order to configure files in the container or set environment variables. To do so, you'll need to mount shell scripts into the `/docker-entrypoint.d/` directory in the container :

Kubernetes

Using the Helm chart, this can be achieved easily using the following `values.yaml` overrides:

```
extraVolumes:
  - name: horizon-entrypoint-scripts
    configMap:
      name: horizon-entrypoint-scripts

extraVolumeMounts:
  - name: horizon-entrypoint-scripts
    mountPath: /docker-entrypoint.d/
```

Given you've previously create a `ConfigMap` called `horizon-entrypoint-scripts`:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: horizon-entrypoint-scripts
data:
  run-on-startup.sh: |
    echo "Hello World !"
```

Docker

```
$ docker run \ -v [scriptsPath]:/docker-entrypoint.d/ \
   ...
   registry.evertrust.io/horizon:2.12.x
```

Where `scriptsPath` is a directory containing one or multiple shell scripts that will be sourced before running Horizon.



By design, Horizon is configured to run as an unprivileged user inside the container to follow industry best practices. This means that your scripts won't be able to perform privileged operations on the container, such as trusting custom CAs. If you do want to overcome this problem, you can run the container as `root`, even though it is generally discouraged.

Available settings

ACME Configuration

horizon.acme.url.default-scheme

```
horizon.acme.url.default-scheme = "https"
```

Protocol to use to calculate the ACME base URL if there isn't any X-Forwarded-Proto nor X-Forwarded-Host in the header of the request

horizon.acme.url.prefix

```
horizon.acme.url.prefix = "/acme"
```

Prefix used to calculate the ACME base URL

horizon.acme.maximum.timeout

```
horizon.acme.maximum.timeout = "5m"
```

Maximum configurable timeout in the ACME profiles

horizon.acme.maximum.retry.delay

```
horizon.acme.maximum.retry.delay = "1h"
```

Maximum configurable delay in the ACME profiles

horizon.acme.maximum.retry.count

```
horizon.acme.maximum.retry.count = 15
```

Maximum configurable retry count in the ACME profiles

horizon.acme.order.updater.worker

```
horizon.acme.order.updater.worker = 5
```

Number of instances that will be started for each Horizon node to perform the ACME validation

horizon.acme.order.ttl

```
horizon.acme.order.ttl = "1m"
```

Order time to live

horizon.acme.challenge.entropy

```
horizon.acme.challenge.entropy = 32
```

Acme challenge size

horizon.acme.challenge.queue.size

```
horizon.acme.challenge.queue.size = 1000
```

Number of challenges that can be buffered before being processing

horizon.acme.challenge.parallelism

```
horizon.acme.challenge.parallelism = 100
```

Number of challenges that can be processed at the same time

horizon.acme.http.json-prettify

```
horizon.acme.http.json-prettify = true
```

Http response as sent as prettyfied JSON

ACME Pki connector configuration

horizon.pki.acme.order.interval

```
horizon.pki.acme.order.interval = "5s"
```

Interval at which order status is checked against the ACME server

horizon.pki.acme.order.max-delay

```
horizon.pki.acme.order.max-delay = "30s"
```

Max delay before order retrieval against the ACME server is abandoned

Analytics Configuration

horizon.event.analytics.actor.timeout

```
horizon.event.analytics.actor.timeout = "1m"
```

The timeout for requests to the event analytics actor

horizon.event.analytics.actor.interval

```
horizon.event.analytics.actor.interval = "5s"
```

Interval at which the events are synchronized

horizon.event.analytics.enabled

```
horizon.event.analytics.enabled = false
```

Enable event analytics

horizon.discovery.event.analytics.actor.timeout

```
horizon.discovery.event.analytics.actor.timeout = "1m"
```

The timeout for requests to the discovery event analytics actor

horizon.discovery.event.analytics.actor.interval

```
horizon.discovery.event.analytics.actor.interval = "5s"
```

Interval at which the discovery events are synchronized

horizon.discovery.event.analytics.enabled

```
horizon.discovery.event.analytics.enabled = false
```

Enable discovery event analytics

horizon.certificate.analytics.actor.timeout

```
horizon.certificate.analytics.actor.timeout = "1m"
```

The timeout for requests to the certificate analytics actor

horizon.certificate.analytics.enabled

```
horizon.certificate.analytics.enabled = false
```

Enable certificate analytics

horizon.analytics.url

```
horizon.analytics.url = "jdbc:duckdb:"
```

The url to the analytics database. Should start with jdbc:duckdb: followed by the absolute path of the file.

horizon.analytics.pool-size

```
horizon.analytics.pool-size = 10
```

The thread pool size for the analytics operations. Should be equal to ((physical_core_count * 2) + effective_spindle_count)

horizon.analytics.memory-limit

```
horizon.analytics.memory-limit = "1GB"
```

The memory limit to set to the duck db analytics database

Archive Configuration

horizon.archive.certificate.batch-size

```
horizon.archive.certificate.batch-size = 1000
```

Batch size for certificate archive creation

horizon.archive.certificate.grace-period

```
horizon.archive.certificate.grace-period = "7d"
```

Grace period of certificate archives before they can be deleted. Decreasing this value means less time to securely download the archive before it is available for deletion

horizon.archive.event.batch-size

```
horizon.archive.event.batch-size = 1000
```

Batch size for event archive creation

horizon.archive.event.grace-period

```
horizon.archive.event.grace-period = "7d"
```

Grace period of event archives before they can be deleted. Decreasing this value means less time to securely download the archive before it is available for deletion

horizon.archive.event.minimum-retention

```
horizon.archive.event.minimum-retention = "90d"
```

Time frame after which event archiving is allowed

horizon.archive.parquet.buffer-size

```
horizon.archive.parquet.buffer-size = "8M"
```

Buffer size for the parquet writer. If using s3 backend, this should be no more than 2 times the 'storage.multipart-upload-buffer-size' to optimize performance

Async Enrollment Configuration

horizon.async-enrollment.interval

```
horizon.async-enrollment.interval = "5m"
```

Interval at which the background process polls in-progress enrollment requests

horizon.async-enrollment.retry.increment

```
horizon.async-enrollment.retry.increment = "5m"
```

Delay increment between consecutive polls for the same request (5, 10, 15, ... minutes)

horizon.async-enrollment.retry.max-delay

```
horizon.async-enrollment.retry.max-delay = "60m"
```

Upper bound for the poll delay

horizon.async-enrollment.parallelism

```
horizon.async-enrollment.parallelism = 3
```

Parallelism for concurrent request polling

Auto Renew Configuration

horizon.auto-renew.interval

```
horizon.auto-renew.interval = "6h"
```

Interval at which the auto renewal process will check certificates to renew

horizon.auto-renew.parallelism.renewal

```
horizon.auto-renew.parallelism.renewal = 3
```

Parallelism for concurrent certificate renewals on auto renew

horizon.auto-renew.parallelism.profile

```
horizon.auto-renew.parallelism.profile = 1
```

Parallelism for concurrent profile auto renewal execution. Bumping this parameter will have a multiply effect on the `horizon.auto-renew.parallelism.renewal` parameter

Bootstrap Configuration

horizon.bootstrap.administrator.name

```
horizon.bootstrap.administrator.name = "administrator"
```

How long the authentication cache lasts

Default administrator account name

horizon.bootstrap.administrator.display-name

```
horizon.bootstrap.administrator.display-name = "Horizon Administrator"
```

Default administrator account display name

horizon.bootstrap.administrator.password.path

```
horizon.bootstrap.administrator.password.path = "var/run/adminPassword"
```

Relative path of the file where the initial admin password should be stored into

horizon.bootstrap.administrator.password.length

```
horizon.bootstrap.administrator.password.length = 24
```

Length (in bytes) of the initial admin password

horizon.bootstrap.local.identity.provider

```
horizon.bootstrap.local.identity.provider = "local"
```

Default administrator account identity provider to use

horizon.bootstrap.timeout

```
horizon.bootstrap.timeout = "1m"
```

Duration after which the bootstrap of Horizon times out

CA Configuration

horizon.ca.manager.default-cache-idletime

```
horizon.ca.manager.default-cache-idletime = "30d"
```

Default idle time after which a CA crl is removed from cache

horizon.ca.manager.timeout

```
horizon.ca.manager.timeout = "1m"
```

Duration that the CA manager actor will wait to retrieve information about certificates (trust status, trust chain, ...)

horizon.ca.maximum.timeout

```
horizon.ca.maximum.timeout = "5m"
```

Maximum configurable timeout for CRL/OCSP request for a CA

horizon.ca.maximum.refresh

```
horizon.ca.maximum.refresh = "7d"
```

Maximum configurable refresh for a CA's CRL

horizon.ca.exporters

```
horizon.ca.exporters = []
```

List of exporters to export client authenticated CAs to

horizon.ca.exporters.[].enabled

```
horizon.ca.exporters.[].enabled = true
```

Whether this exporter should export client CAs

horizon.ca.exporters.[].type

```
horizon.ca.exporters.[].type = "k8s-configmap|k8s-secret|file"
```

Type of exporter. One of `k8s-configmap`, `k8s-secret`, `file`. The corresponding configuration parameters should be filled

horizon.ca.exporters.[].format

```
horizon.ca.exporters.[].format = "pem-blocks"
```

How to format the CA certificates. One of `pem-blocks`

horizon.ca.exporters.[].file.path

```
horizon.ca.exporters.[].file.path = "<path to file>"
```

Path to the file to write the CAs to. Mandatory if type is `file`

horizon.ca.exporters.[].k8s-configmap.name

```
horizon.ca.exporters.[].k8s-configmap.name = "<configmap name>"
```

Name of the configmap. Mandatory if type is `k8s-configmap`

horizon.ca.exporters.[].k8s-configmap.map-key

```
horizon.ca.exporters.[].k8s-configmap.map-key = "<key in the configmap data map>"
```

Key of the element in the configmap

horizon.ca.exporters.[].k8s-secret.name

```
horizon.ca.exporters.[].k8s-secret.name = "<secret name>"
```

Name of the secret. Mandatory if type is `k8s-secret`

horizon.ca.exporters.[].k8s-secret.secret-key

```
horizon.ca.exporters.[].k8s-secret.secret-key = "<key in the secret data map>"
```

Key of the element in the secret

CRL Configuration

horizon.crl.updater.batch

```
horizon.crl.updater.batch = 500
```

Number of certificates per batch when Horizon synchronizes the database with the CRL or updates the cached entries



This parameter replaces `horizon.crl.updater.parallelism`. Please modify your configuration accordingly

horizon.crl.refresh.queue.size

```
horizon.crl.refresh.queue.size = 100
```

The maximum number of CAs awaiting CRL synchronization for the certificate status cache. When modifying this value, the change should be reflected on the `crl-task-dispatcher.thread-pool-executor.fixed-pool-size` value

horizon.crl.refresh.queue.parallelism

```
horizon.crl.refresh.queue.parallelism = 5
```

Number of CRLs to synchronize in parallel for the certificate status cache

horizon.crl.synchronizer.timeout

```
horizon.crl.synchronizer.timeout = "30s"
```

Timeout for the synchronizer actor during DB synchronization

horizon.crl.synchronizer.queue.parallelism

```
horizon.crl.synchronizer.queue.parallelism = 1
```

Number of CRLs to synchronize in parallel for DB synchronization. When modifying this value, the change should be reflected on the `crl-task-dispatcher.thread-pool-executor.fixed-pool-size` value

horizon.crl.synchronizer.queue.size

```
horizon.crl.synchronizer.queue.size = 100
```

The maximum number of CAs awaiting CRL synchronization for DB synchronization

CSV Configuration

horizon.request.search.csv.delimiter

```
horizon.request.search.csv.delimiter = ";"
```

The CSV delimiter to use when exporting an HRQL query result to a CSV file

horizon.event.search.csv.delimiter

```
horizon.event.search.csv.delimiter = ";"
```

The CSV delimiter to use when exporting an HEQL query result to a CSV file

horizon.discovery.event.search.csv.delimiter

```
horizon.discovery.event.search.csv.delimiter = ";"
```

The CSV delimiter to use when exporting an HDQL query result to a CSV file

horizon.certificate.search.item.attribute.separator

```
horizon.certificate.search.item.attribute.separator = ":"
```

The CSV item attribute separator to use when exporting an HCQL query result to a CSV file

horizon.certificate.search.item.separator

```
horizon.certificate.search.item.separator = "\n"
```

The CSV item separator to use when exporting an HCQL query result to a CSV file

horizon.certificate.search.csv.delimiter

```
horizon.certificate.search.csv.delimiter = ";"
```

The CSV delimiter to use when exporting an HCQL query result to a CSV file

Certificate authentication

horizon.security.http.headers.certificate

```
horizon.security.http.headers.certificate = null
```

Name of the HTTP header containing the certificate

DCV Configuration

horizon.dcv.manager.timeout

```
horizon.dcv.manager.timeout = "1m"
```

Duration after which the DCV manager actor times out when retrieving DCV configuration in the database

horizon.dcv.manager.queue.size

```
horizon.dcv.manager.queue.size = 100
```

Number of DCV policy that can be queued

horizon.dcv.manager.validation.initial-delay

```
horizon.dcv.manager.validation.initial-delay = "30s"
```

Delay between DNS record creation and first validation attempt, to allow DNS propagation

horizon.dcv.validation.throttle.elements

```
horizon.dcv.validation.throttle.elements = 1
```

Maximum number of domains submitted for validation within the time window defined by throttle.per for each DCV policy

horizon.dcv.validation.throttle.per

```
horizon.dcv.validation.throttle.per = "1s"
```

Time window for the throttle

horizon.dcv.validation.worker-count

```
horizon.dcv.validation.worker-count = 1
```

Number of DCVWorkerActor instances to start per cluster node

horizon.dcv.search.page.default-size

```
horizon.dcv.search.page.default-size = 50
```

Default number of DCV lifecycle events returned per page

horizon.dcv.search.page.max-size

```
horizon.dcv.search.page.max-size = null
```

Maximum number of DCV lifecycle events that can be requested per page (null for unlimited)

horizon.dcv.search.timeout

```
horizon.dcv.search.timeout = "30s"
```

Timeout for DCV lifecycle event search queries

horizon.dcv.event.retention

```
horizon.dcv.event.retention = "30d"
```

Retention of DCV lifecycle event before they are removed from the database. This cannot be less than 30 days

Database Configuration

horizon.security.principal.search.timeout

```
horizon.security.principal.search.timeout = "0s"
```

Maximum time allowed for security principals search operations. For infinite timeout, use 0s

horizon.request.search.timeout

```
horizon.request.search.timeout = "0s"
```

Maximum time allowed for request search and aggregate operations. For infinite timeout, use 0s

horizon.event.search.timeout

```
horizon.event.search.timeout = "30s"
```

Maximum time allowed for event search operations. For infinite timeout, use 0s

horizon.discovery.event.search.timeout

```
horizon.discovery.event.search.timeout = "30s"
```

Maximum time allowed for discovery event search and aggregate operations. For infinite timeout, use 0s

horizon.certificate.search.timeout

```
horizon.certificate.search.timeout = "30s"
```

Maximum time allowed for certificate search and aggregate operations. For infinite timeout, use 0s

Discovery Event Configuration

horizon.discovery.event.ttl

```
horizon.discovery.event.ttl = null
```

Time to live of the discovery events. If not set, events never expire

Event Configuration

horizon.event.ttl

```
horizon.event.ttl = null
```

Time to live of the events. If not set, events never expire

horizon.event.chainsign

```
horizon.event.chainsign = true
```

Specify whether to chain and sign the Horizon events to ensure they haven't been tampered with

horizon.event.seal.algorithm

```
horizon.event.seal.algorithm = "HS512"
```

Algorithm to use to hash the signature of the events in Horizon (other possible values are "HS384" and "HS256")

horizon.event.seal.secret

```
horizon.event.seal.secret = null
```

Secret to seal the events with

horizon.event.ignore-unsealed-pending

```
horizon.event.ignore-unsealed-pending = false
```

Do not throw an error if pending events are unsealed

horizon.event.timeout

```
horizon.event.timeout = "30s"
```

Duration after which the event manager times out when trying to retrieve the last signed event in the database

horizon.event.manager.interval

```
horizon.event.manager.interval = "5s"
```

How often will the Event Manager actor check in the database if new a new event appeared to sign it and display it in the "Events" section of Horizon

General

horizon.security.pop.iat.future

```
horizon.security.pop.iat.future = "5s"
```

Difference of time allowed between the "Issued At Time" and the validation time (or the server time) (in the future only)

horizon.security.pop.iat.past

```
horizon.security.pop.iat.past = "5s"
```

Difference of time allowed between the "Issued At Time" and the validation time (or the server time) (in the past only)

horizon.security.pop.iat.clock-skew

```
horizon.security.pop.iat.clock-skew = "30s"
```

Difference of time allowed between the client time and the server time

horizon.security.identity.local.password-reset.duration

```
horizon.security.identity.local.password-reset.duration = "2m"
```

Time to live of a password reset request (from the login prompt)

horizon.security.trustmanager.enforce-serverauth

```
horizon.security.trustmanager.enforce-serverauth = false
```

If set to true, enforces the use of the serverAuth EKU in the server authentication certificates (when Horizon accesses a service through TLS)

horizon.security.manager.timeout

```
horizon.security.manager.timeout = "1m"
```

Duration after which the security manager times out when trying to authenticate a principal with its session

horizon.acme.compromise.interval

```
horizon.acme.compromise.interval = "1d"
```

Interval between compromise checks (recommended: 1 day)

horizon.acme.compromise.parallelism

```
horizon.acme.compromise.parallelism = 5
```

Maximum number of accounts to compromise simultaneously

horizon.acme.compromise.queue.size

```
horizon.acme.compromise.queue.size = 100
```

Number of accounts buffered before processing

horizon.request.default.grace-period

```
horizon.request.default.grace-period = "30d"
```

Default grace period for all requests

horizon.request.default.duration

```
horizon.request.default.duration = "7d"
```

Default duration for all requests

horizon.request.failed.duration

```
horizon.request.failed.duration = "7d"
```

Default duration for failed requests

horizon.intune.revocation.max-requests

```
horizon.intune.revocation.max-requests = 250
```

Number of revocation requests downloaded from Intune

Limited to 500 max

horizon.datasource.default-timeout

```
horizon.datasource.default-timeout = "5s"
```

Default timeout for REST requests for the REST datasource

horizon.scheduler.manager.timeout

```
horizon.scheduler.manager.timeout = "1m"
```

Duration after which the Scheduler manager actor times out when retrieving scheduled tasks in the database

horizon.notification.mail.attachment.extension.der

```
horizon.notification.mail.attachment.extension.der = "der"
```

File extension that DER certificates sent as email attachments (through the notifications feature) will be given

horizon.notification.mail.attachment.extension.p7b

```
horizon.notification.mail.attachment.extension.p7b = "p7b"
```

File extension that PKCS#7 certificates sent as email attachments (through the notifications feature) will be given

horizon.notification.mail.attachment.extension.pem

```
horizon.notification.mail.attachment.extension.pem = "pem"
```

File extension that PEM certificates sent as email attachments (through the notifications feature) will be given

horizon.hql.max-recursion-depth

```
horizon.hql.max-recursion-depth = 5
```

Maximum recursion allowed for the HQL queries

horizon.system.monitor.timeout

```
horizon.system.monitor.timeout = "30s"
```

Timeout for the system monitor loading

horizon.thirdparty.manager.timeout

```
horizon.thirdparty.manager.timeout = "1m"
```

Timeout for thirdparty synchronization requests

horizon.pki.manager.maximum.timeout

```
horizon.pki.manager.maximum.timeout = "5m"
```

Maximum configurable timeout on the PKI connectors

horizon.pki.manager.timeout

```
horizon.pki.manager.timeout = "1m"
```

Duration after which the PKI Manager times out when trying to enroll or revoke a certificate

horizon.pki.manager.queue.parallelism

```
horizon.pki.manager.queue.parallelism = 5
```

Number of parallel certificate requests (enrollment, revocation...) on the default queue

horizon.pki.manager.queue.size

```
horizon.pki.manager.queue.size = 100
```

Number of certificate requests (enrollment, revocation) that can be queued on the default queue

horizon.pki.manager.healthcheck.interval

```
horizon.pki.manager.healthcheck.interval = "5m"
```

Interval at which the PKI connectors statuses are checked

horizon.show-banner

```
horizon.show-banner = true
```

Hide the start-up banner

horizon.est.store-encryption-type

```
horizon.est.store-encryption-type = "AES_STRONG"
```

Default store encryption type to use when sending centralized EST responses

horizon.scim.discovery-endpoints.authenticated

```
horizon.scim.discovery-endpoints.authenticated = true
```

Choose whether or not scim discovery endpoints are authenticated

horizon.automation-policy.default.keytype

```
horizon.automation-policy.default.keytype = "rsa-2048"
```

Default key type used for automation when none are specified in the profile

horizon.endpoints

```
horizon.endpoints = null
```

Custom endpoint configuration

Global constraints Configuration

horizon.default.constraints.allowed.domains

```
horizon.default.constraints.allowed.domains = null
```

Default allowed domains: a regular expression that the dns or email domains should match

horizon.default.constraints.allowed.email.domains

```
horizon.default.constraints.allowed.email.domains = null
```

Default allowed email domains: a regular expression that the email domains should match (after the @)

horizon.default.constraints.allowed.dns.domains

```
horizon.default.constraints.allowed.dns.domains = null
```

Default allowed dns domains: a regular expression that the dns domains should match

Grading Configuration

horizon.grading.manager.timeout

```
horizon.grading.manager.timeout = "30s"
```

Duration after which the grading manager times out when retrieving the grading configuration from the database

horizon.grading.manager.queue.size

```
horizon.grading.manager.queue.size = 100
```

How large can the grading manager queue can get before it discards new grading requests

horizon.grading.timeout

```
horizon.grading.timeout = "30s"
```

Duration after which the grading actor times out when grading a certificate (upon enrolment)

HTTP Headers Configuration

horizon.security.http.headers.enforce-connection-close

```
horizon.security.http.headers.enforce-connection-close = true
```

Defines whether HTTP connections should remain open

horizon.security.http.headers.real-ip

```
horizon.security.http.headers.real-ip = "X-Real-IP"
```

Name of the HTTP header to use as Real IP

horizon.security.http.headers.scheme

```
horizon.security.http.headers.scheme = "X-Forwarded-Proto"
```

Name of the HTTP header containing the scheme requested - used for ACME

horizon.security.http.headers.host

```
horizon.security.http.headers.host = "X-Forwarded-Host"
```

Name of the HTTP header containing the host requested - used for ACME

Kubernetes Configuration

horizon.kubernetes.namespace-path

```
horizon.kubernetes.namespace-path =
"/var/run/secrets/kubernetes.io/serviceaccount/namespace"
```

Path to the file containing the namespace identifier of the Kubernetes cluster

horizon.kubernetes.namespace

```
horizon.kubernetes.namespace = null
```

Namespace of the Kubernetes cluster. If defined, contents of the `horizon.kubernetes.namespace-path` will be ignored

horizon.kubernetes.api-ca-path

```
horizon.kubernetes.api-ca-path =
"/var/run/secrets/kubernetes.io/serviceaccount/ca.crt"
```

Path to the file containing the certificate authorities of the Kubernetes cluster

horizon.kubernetes.api-token-path

```
horizon.kubernetes.api-token-path =
"/var/run/secrets/kubernetes.io/serviceaccount/token"
```

Path to the file containing the token to authenticate on the API of the Kubernetes cluster

horizon.kubernetes.api-service.host

```
horizon.kubernetes.api-service.host = "localhost"
```

Host on which to join the Kubernetes API

horizon.kubernetes.api-service.port

```
horizon.kubernetes.api-service.port = 8080
```

Port on which to join the Kubernetes API

horizon.kubernetes.api-service.request-timeout

```
horizon.kubernetes.api-service.request-timeout = "5s"
```

Timeout for requests to the Kubernetes API

horizon.kubernetes.api-service.secure

```
horizon.kubernetes.api-service.secure = true
```

Whether to join the Kubernetes API with http or https

horizon.kubernetes.api-service.tls-version

```
horizon.kubernetes.api-service.tls-version = "TLSv1.2"
```

TLS Version to use while connecting to the Kubernetes API

horizon.kubernetes.token-rotation-retry.max-attempts

```
horizon.kubernetes.token-rotation-retry.max-attempts = 5
```

Number of retries in case of 401 response from Kubernetes API (can occur on token rotation)

horizon.kubernetes.token-rotation-retry.min-backoff

```
horizon.kubernetes.token-rotation-retry.min-backoff = 10 ms
```

Minimum backoff for retries in case of 401 response from Kubernetes API (can occur on token rotation)

horizon.kubernetes.token-rotation-retry.max-backoff

```
horizon.kubernetes.token-rotation-retry.max-backoff = 1 minute
```

Maximum backoff for retries in case of 401 response from Kubernetes API (can occur on token rotation)

horizon.kubernetes.token-rotation-retry.random-factor

```
horizon.kubernetes.token-rotation-retry.random-factor = 0.3
```

Random factor for retries in case of 401 response from Kubernetes API (can occur on token rotation)

Maintenance Configuration

horizon.maintenance.filter.timeout

```
horizon.maintenance.filter.timeout = "3s"
```

Timeout to check for maintenance state on each request

Metrics Configuration

horizon.metrics.enabled

```
horizon.metrics.enabled = false
```

Enable advanced metrics for collection

horizon.metrics.intervals.short

```
horizon.metrics.intervals.short = "30s"
```

Interval at which short lived metrics are computed

horizon.metrics.intervals.long

```
horizon.metrics.intervals.long = "5m"
```

Interval at which background metrics are computed

Nonce Configuration

horizon.automation.nonce.size

```
horizon.automation.nonce.size = 32
```

Size of the nonce value used for the JWT authentication token

horizon.automation.nonce.ttl

```
horizon.automation.nonce.ttl = "5s"
```

Time to live of the nonce used to validate the JWT authentication token

horizon.acme.nonce.size

```
horizon.acme.nonce.size = 32
```

Size (in bytes) of the challenge stored in the nonce

horizon.acme.nonce.ttl

```
horizon.acme.nonce.ttl = "5s"
```

Duration for which a nonce stays in Horizon before being removed

horizon.openid.nonce.size

```
horizon.openid.nonce.size = 32
```

Size (in bytes) of the challenge stored in the nonce

horizon.openid.nonce.ttl

```
horizon.openid.nonce.ttl = "5s"
```

Duration for which a nonce stays in Horizon before being removed

horizon.request.nonce.size

```
horizon.request.nonce.size = 32
```

Size (in bytes) of the challenge stored in the nonce

horizon.request.nonce.ttl

```
horizon.request.nonce.ttl = "5s"
```

Duration for which a nonce stays in Horizon before being removed

OpenID Configuration

horizon.openid.state-separator

```
horizon.openid.state-separator = "#"
```

Separator character of the OpenID state

horizon.openid.callback-scheme

```
horizon.openid.callback-scheme = "https"
```

Scheme to use for the callback



This parameter replaces `horizon.openid.https-callback`. Please modify your configuration accordingly

horizon.openid.token-auth-method-order

```
horizon.openid.token-auth-method-order = ["client_secret_post", "client_secret_basic"]
```

Selection order for the authentication method on the access token retrieval endpoint

Report Configuration

horizon.report.link.refresh-interval

```
horizon.report.link.refresh-interval = "5m"
```

The refresh interval between the deletion of expired reports with link email

horizon.report.link.maxStored

```
horizon.report.link.maxStored = "3"
```

Max stored number of CSV for a given report. If the field is negative or equal to zero, then the retention will be deactivated.

Search Configuration

horizon.security.principal.search.page.default-size

```
horizon.security.principal.search.page.default-size = 50
```

How many elements to retrieve in a security principals search query if no pageSize has been specified

horizon.security.principal.search.page.max-size

```
horizon.security.principal.search.page.max-size = null
```

How big can the pageSize parameter be in a security principals search query (Must be a positive integer)

horizon.acme.search.eab.page.default-size

```
horizon.acme.search.eab.page.default-size = 50
```

How many elements to retrieve in a EAB search query if no pageSize has been specified

horizon.acme.search.eab.page.max-size

```
horizon.acme.search.eab.page.max-size = null
```

How big can the pageSize parameter be in a EAB search query (Must be a positive integer)

horizon.acme.search.eab.timeout

```
horizon.acme.search.eab.timeout = "0s"
```

Maximum time allowed for EAB search operations. For infinite timeout, use 0s

horizon.acme.search.account.page.default-size

```
horizon.acme.search.account.page.default-size = 50
```

How many elements to retrieve in a account search query if no pageSize has been specified

horizon.acme.search.account.page.max-size

```
horizon.acme.search.account.page.max-size = null
```

How big can the pageSize parameter be in a account search query (Must be a positive integer)

horizon.acme.search.account.timeout

```
horizon.acme.search.account.timeout = "0s"
```

Maximum time allowed for account search operations. For infinite timeout, use 0s

horizon.acme.search.order.page.default-size

```
horizon.acme.search.order.page.default-size = 50
```

How many elements to retrieve in a order search query if no pageSize has been specified

horizon.acme.search.order.page.max-size

```
horizon.acme.search.order.page.max-size = null
```

How big can the pageSize parameter be in a order search query (Must be a positive integer)

horizon.acme.search.order.timeout

```
horizon.acme.search.order.timeout = "0s"
```

Maximum time allowed for order search operations. For infinite timeout, use 0s

horizon.request.search.page.default-size

```
horizon.request.search.page.default-size = 50
```

How many elements to retrieve in a request search query if no pageSize has been specified

horizon.request.search.page.max-size

```
horizon.request.search.page.max-size = null
```

How big can the pageSize parameter be in a request search query (Must be a positive integer)

horizon.event.search.page.default-size

```
horizon.event.search.page.default-size = 50
```

How many elements to retrieve in an event search query if no pageSize has been specified

horizon.event.search.page.max-size

```
horizon.event.search.page.max-size = null
```

How big can the pageSize parameter be in an event search query (Must be a positive integer)

horizon.discovery.event.search.page.default-size

```
horizon.discovery.event.search.page.default-size = 50
```

How many elements to retrieve in a request search query if no pageSize has been specified

horizon.discovery.event.search.page.max-size

```
horizon.discovery.event.search.page.max-size = null
```

How big can the pageSize parameter be in a request search query (Must be a positive integer)

horizon.certificate.search.page.default-size

```
horizon.certificate.search.page.default-size = 50
```

How many elements to retrieve in a request search query if no pageSize has been specified

horizon.certificate.search.page.max-size

```
horizon.certificate.search.page.max-size = null
```

How big can the pageSize parameter be in a request search query (Must be a positive integer)

Service Account Configuration

horizon.service-account.validated-token.cache

```
horizon.service-account.validated-token.cache = "5m"
```

Default idle time after which a token validation is no longer valid

horizon.service-account.refresh.passive

```
horizon.service-account.refresh.passive = "1d"
```

Default interval for JWKS updates

horizon.service-account.refresh.active

```
horizon.service-account.refresh.active = "1h"
```

Default cooldown after a JWKS update

horizon.service-account.refresh.timeout

```
horizon.service-account.refresh.timeout = "3m"
```

Timeout for the JWKS endpoint HTTP call during a passive refresh

horizon.service-account.refresh-timeout

```
horizon.service-account.refresh-timeout = "10s"
```

Default timeout for HTTP client requests when fetching JWKS endpoints

Storage Configuration

horizon.storage.whitelist

```
horizon.storage.whitelist = null
```

Whitelist of allowed storage types for heavy files

Tenancy Configuration

horizon.tenancy.timeout

```
horizon.tenancy.timeout = "1m"
```

Timeout for the tenancy cache actor initialization

horizon.tenancy.grace-period

```
horizon.tenancy.grace-period = "30d"
```

Grace period for the tenant when deleted before effective deletion can occur

Transient Key Configuration

horizon.transient-keys.lifetime

```
horizon.transient-keys.lifetime = "7d"
```

Lifetime for transient (non escrowed) keys

horizon.transient-keys.removal-interval

```
horizon.transient-keys.removal-interval = "30m"
```

Interval at which expired keys are removed from the database

Trigger Configuration

horizon.trigger.retry.initial-delay

```
horizon.trigger.retry.initial-delay = "5m"
```

How long must a trigger that fails for the first time wait before retrying

horizon.trigger.retry.max-attempts

```
horizon.trigger.retry.max-attempts = 15
```

Maximum amount of failed attempts that a trigger can have before canceling

horizon.trigger.manager.timeout

```
horizon.trigger.manager.timeout = "1m"
```

Trigger manager timeout

horizon.trigger.manager.interval

```
horizon.trigger.manager.interval = "5m"
```

How often does the trigger manager check for triggers to run

Validation Rules Configuration

horizon.validation.dns.timeout

```
horizon.validation.dns.timeout = "10s"
```

Timeout to fetch dns records when using validation rules

Vault Configuration

horizon.vault.escrow

```
horizon.vault.escrow = null
```

The name of the escrow vault

horizon.vault.configuration

```
horizon.vault.configuration = null
```

The name of the configuration vault

horizon.vault.manager.timeout

```
horizon.vault.manager.timeout = "1m"
```

Timeout for encryption requests

1.14. Upgrade

Before the upgrade

- Before upgrading Horizon, you should always take a full database backup. Remind that upgrading is a one-way operation, and that you'll need to restore to an older database state if you wish to rollback the upgrade.
- Check the release notes for the version you're upgrading to for known defects that might impact you. Also, carefully read the Specific upgrade instructions for any particular instructions for the version you're upgrading to.
- Upgrades should follow the below order to ensure no errors when using High Availability



This new migration workflow is only available when upgrading from a 2.7.x instance. Please refer to 2.7.x documentation to upgrade to this version, and then use this page.

Installing the migration tool

The migration tool should be updated to the latest available version before each migration.

RPM



In order to install Horizon Migration Tool, the server must have access to a repository (mirror, ISO file, ...) of the linux distribution you are using in order to be able to install the dependencies of the software. Horizon Migration Tool package has the following dependencies:

- `java-17-openjdk-headless`

Please note that these packages may have their own dependencies.

Installation from the EverTrust repository

Create a `/etc/yum.repos.d/horizon-migration.repo` file containing the EverTrust repository info:

```
[horizon-migration]
enabled=1
name=Horizon Migration Tool Repository
baseurl=https://repo.evertrust.io/repository/horizon-migration-rpm/
gpgcheck=1
gpgkey=https://evertrust.io/.well-known/rpm/gpg.pub
username=<username>
password=<password>
```

Replace `<username>` and `<password>` with the credentials you were provided.

Make sure the Evertrust GPG key is trusted:

```
# rpm --import https://evertrust.io/.well-known/rpm/gpg.pub
```

You can then run the following to install the latest Horizon Migration Tool version:

```
# yum install horizon-migration
```

To prevent unattended upgrades when running yum update, you should pin the Horizon Migration Tool version by adding

```
exclude=horizon-migration
```

at the end of the `/etc/yum.repos.d/horizon-migration.repo` file after installing Horizon Migration Tool.

Installing from RPM

Download the latest RPM for Horizon Migration Tool on the Official EVERTRUST repository.

Upload the file `horizon-migration-<latest>.noarch.rpm` to the server;

Access the server with an account with administrative privileges;

Install the Horizon Migration Tool package with the following command:

```
# yum localinstall /root/horizon-migration-<latest>.noarch.rpm
```

If you wish to verify the signature of the RPM package, the EVERTRUST key can be added to your trusted keys using the following command:

```
# rpm --import https://evertrust.io/.well-known/rpm/gpg.pub
```

The signature can then be verified using the following command:

```
# rpm -K /root/horizon-migration-<latest>.noarch.rpm
```

Debian



In order to install Horizon Migration Tool, the server must have access to a repository (mirror, ISO file, ...) of the linux distribution you are using in order to be able to install the dependencies of the software. Horizon Migration Tool package has the following dependencies:

- `openjdk-17-jre-headless`

Please note that these packages may have their own dependencies.

Installation from the EverTrust repository

If you haven't already, to add the EVERTRUST repository to your APT repositories, run the following commands:

1. Install the required tools (`gpg`)

```
# sudo apt install gnupg
```

2. Download and install the EVERTRUST GPG key

```
# curl https://evertrust.io/.well-known/apt/gpg.pub | sudo gpg -o
/usr/share/keyrings/evertrust.gpg --dearmor
```

3. Add the repository

```
# echo "deb [ arch=all,amd64 signed-by=/usr/share/keyrings/evertrust.gpg ]
https://repo.evertrust.io/repository/apt all main" | sudo tee
/etc/apt/sources.list.d/evertrust.list
```

Once the repository has been added, authentication to it must be provided. To do so, edit the `/etc/apt/auth.conf` file and add the following lines:

```
machine repo.evertrust.io
login <your EVERTRUST login>
password <your EVERTRUST password>
```

Once the repository has been added, run the following command to update the APT repository list.

```
# sudo apt update
```

You can then run the following command to install the latest Horizon Migration Tool version:

```
# sudo apt install horizon-migration
```

To prevent unattended upgrades when running `apt upgrade`, you should pin the Horizon Migration Tool version by creating a `/etc/apt/preferences.d/horizon-migration` file:

```
Package: horizon-migration
Pin: version <installed-version>
Pin-Priority: 1001
```

Installing from DEB

Download the latest DEB for Horizon Migration Tool on the Official EVERTRUST repository.

Upload the file `horizon-migration_<latest>_all.deb` to the server;

Access the server with an account with administrative privileges;

Install the Horizon Migration Tool package with the following command:

```
# apt install /root/horizon-migration_<latest>_all.deb
```

Kubernetes

No installation is required as the helm chart will pull the appropriate image.

If your environment is subject to FIPS constraints, set the `upgrade.image.tag` key of your `values.yaml` file to the FIPS tag of the migration tool (suffixed with `-fips`).

Docker

The docker image is available on the registry:

```
$ docker pull registry.evertrust.io/horizon-migration:latest
```

The FIPS flavor is tagged `latest-fips`:

```
$ docker pull registry.evertrust.io/horizon-migration:latest-fips
```



If your environment is subject to FIPS constraints, use the FIPS flavor of the migration tool:

- RPM / DEB package: `horizon-migration-fips`
- Docker image: `registry.evertrust.io/horizon-migration:latest-fips`

In the commands and manifests below, replace `horizon-migration:latest` with `horizon-migration:latest-fips`.

Pre-migration checks

Before upgrading, some checks should be run to ensure compatibility with the newer version:

RPM

Before planning an upgrade, the migration tool should be upgraded to the latest version. It is recommended to install it on one of the Horizon machine for a seamless experience.

Once upgraded, run the following command to check the database compatibility:

```
$ /opt/evertrust/horizon-migration/bin/horizon-migration check \
  --to <target version>
```

In most cases, `horizon-migration` can detect the version you're upgrading from by checking the database. However, the source version can be specified using `--from` flag:

```
$ /opt/evertrust/horizon-migration/bin/horizon-migration check \
  --to <target version> \
  --from <source version>
```

If you wish to run the migration from another machine that the one Horizon is installed on, or if you wish to migrate another database, the `--mongo-uri` flag should be added to access to MongoDB.

```
$ /opt/evertrust/horizon-migration/bin/horizon-migration check \
  --to <target version> \
  --mongo-uri <mongo uri>
```

Debian

Before planning an upgrade, the migration tool should be upgraded to the latest version. It is recommended to install it on one of the Horizon machine for a seamless experience.

Once upgraded, run the following command to check the database compatibility:

```
$ /opt/evertrust/horizon-migration/bin/horizon-migration check \
```

```
  --to <target version>
```

In most cases, `horizon-migration` can detect the version you're upgrading from by checking the database. However, the source version can be specified using `--from` flag:

```
$ /opt/evertrust/horizon-migration/bin/horizon-migration check \
  --to <target version> \
  --from <source version>
```

If you wish to run the migration from another machine that the one Horizon is installed on, or if you wish to migrate another database, the `--mongo-uri` flag should be added to access to MongoDB.

```
$ /opt/evertrust/horizon-migration/bin/horizon-migration check \
  --to <target version> \
  --mongo-uri <mongo uri>
```

Kubernetes

By default, the chart will automatically create a `Job` that checks database compatibility before an upgrade. This job will be assigned the `helm.sh/hook: pre-upgrade` annotation, which means they'll run before the upgrade is actually performed. If the job fails, the upgrade will also fail before proceeding with altering the database.

Should you wish to disable the automatic upgrade mechanism, just set the `upgradeCompatibilityCheck.enabled` key to `false`. If you still wish to manually run the pre-migration script, you can use this Kubernetes manifest as a starting point:

```
apiVersion: batch/v1
kind: Job
metadata:
  name: horizon-migration
spec:
  template:
    spec:
      containers:
        - name: horizon-migration
          image: registry.evertrust.io/horizon-migration:latest
          imagePullPolicy: IfNotPresent
          args: [
              "check",
              "-y",
              "--mongo-uri", "$(MONGODB_URI)",
              "--ignore-empty-from",
              "--to", "<target-version>" ①
          ]
          env:
            - name: MONGODB_URI
```

```
              valueFrom:
                secretKeyRef:
                  name: horizon ②
                  key: mongoUri
      restartPolicy: Never
  backoffLimit: 0
```

① The target version should be updated to match the desired upgrade path. The source version will be inferred from database.

② The secret name and key should match where you store the Horizon MongoDB URI, so it will be injected as an environment variable to the Pod.

Docker

You can run the migration tool on any computer that has access to your MongoDB database, provided that it has Docker installed:

```
$ docker run -it --rm registry.evertrust.io/horizon-migration:latest check \
  --mongo-uri <mongo uri> \
  --to <target version> \
  --from <source version> # This is required when upgrading from an older
version of Horizon
```

Upgrade Horizon

RPM

Follow the installation procedure again to retrieve the new version.



For High Availability instances, nodes can be updated one by one. Upgraded nodes will be idle and serve a maintenance page until the database migration (next step) has ended.

Debian

Follow the installation procedure again to retrieve the new version.



For High Availability instances, nodes can be updated one by one. Upgraded nodes will be idle and serve a maintenance page until the database migration (next step) has ended.

Kubernetes

When upgrading Horizon, you'll need to pull the latest version of the chart:

```
$ helm repo update evertrust
```

Verify that you now have the latest version of Horizon (through the App version column):

```
$ helm search repo evertrust/horizon
NAME                        CHART VERSION   APP VERSION DESCRIPTION
evertrust/horizon           2.2.2           2.12.0          EverTrust Horizon
Helm chart
```

Launch an upgrade by specifying the new version of the chart through the `--version` flag in your command:

```
$ helm upgrade <horizon> evertrust/horizon \
 --values override-values.yaml \
 --version 0.9.3
```



We recommended that you only change values you need to customize in your `values.yml` file to ensure smooth upgrading.

Docker

Pull the new image from the repository.

Migrate the database

RPM

Using the horizon-migration tool used at the first step, the migration can now be run using the following command:

```
$ /opt/evertrust/horizon-migration/bin/horizon-migration migrate --to <target
version>
```

Just like the check command, additional flags should be added when running on another machine.

Debian

Using the horizon-migration tool used at the first step, the migration can now be run using the following command:

```
$ /opt/evertrust/horizon-migration/bin/horizon-migration migrate --to <target
version>
```

Just like the check command, additional flags should be added when running on another machine.

Kubernetes

By default, a chart upgrade will dispatch a migration job annotated with `helm.sh/hook:`

`post-upgrade`, meaning it will run once all nodes have been upgraded to the target version. While the rollout and the migration script are running, nodes running a newer version will be put in maintenance mode. Maintenance mode will automatically exit once the upgrade script finishes running.

Should you wish to disable the automatic upgrade mechanism, just set the `upgrade.enabled` key to `false`. If you still wish to manually run the migration script, you can use this Kubernetes manifest as a starting point:

```
apiVersion: batch/v1
kind: Job
metadata:
  name: horizon-migration
spec:
  template:
    spec:
      containers:
        - name: horizon-migration
          image: registry.evertrust.io/horizon-migration:latest
          imagePullPolicy: IfNotPresent
          args: [
              "migrate",
              "-y",
              "--mongo-uri", "$(MONGODB_URI)",
              "--ignore-empty-from",
              "--to", "<target-version>" ①
          ]
          env:
            - name: MONGODB_URI
              valueFrom:
                secretKeyRef:
                  name: horizon ②
                  key: mongoUri
      restartPolicy: Never
  backoffLimit: 0
```

① The target version should be updated to match the desired upgrade path. The source version will be inferred from database.

② The secret name and key should match where you store the Horizon MongoDB URI, so it will be injected as an environment variable to the Pod.

Docker

You can run the script on any computer that has access to your MongoDB database, provided that it has Docker installed:

```
$ docker run -it --rm registry.evertrust.io/horizon-migration:latest migrate \
   --mongo-uri <mongo uri> \
   --to <target version> \
```

```
--from <source version>
```

Specific upgrade instructions

These steps should be followed in addition to the common upgrade procedure found in the above standard upgrade instructions.

Upgrading from a version prior to 2.8.0

When upgrading from a version prior to 2.8.0, you must manually create a new keyset using Tinkey.



The migration tool will fail if the keyset is not generated before running the upgrade.

Different keyset templates are available depending on your security requirements.To maintain the same encryption level as your current installation, use the following mapping:

- SSV → PlainText
- SHV → Derived PKCS#11 keys with CBC



If you are unsure about which mode to use, contact the EVERTRUST support team.

Follow the steps in the Tinkey page to setup Tinkey, create your Keyset and make it available for Horizon.

Once your keyset has been created, migration can occur as specified by the steps above. However for the 2.8 migration, specific information must be provided to the migration tool to handle the encryption of secrets. This depends on your current encryption mechanism.

SSV

If you are running the migration tool on the same server as the Horizon installation, and using the same user, the tool will pick up the current configuration automatically, and requires no additional configuration.

If you are using the Helm Chart, upgrading it should also handle the migration automatically.

If the migration tool does not run in the same context, the required configuration can be provided using environment variables:

Variable	Description
`HORIZON_MIG_280_SSV_PASSWORD`	The SSV password
`HORIZON_MIG_280_KEYSET`	The raw Tink keyset to use. To define if `HORIZON_MIG_280_KEYSET_PATH` is not defined

Variable	Description
HORIZON_MIG_280_KEYSET_PATH	Path to the Tink keyset file. To define if HORIZON_MIG_280_KEYSET is not defined
HORIZON_MIG_280_MASTER_KEY_URI	Tink Master key URI if the keyset is wrapped
HORIZON_MIG_280_CREDENTIALS_PATH	Path to the credentials file to combine with the Master key if using a KMS

This will migrate all vaults to the tink keyset provided.

If you wish to use other vaults configuration, the path to a full configuration file for the vault migration can be provided.

The migration file must be provided in the HORIZON_MIG_280_VAULT_CONF_PATH environment variable. This will override all other behaviors.

This file defines all the source and destination vaults, and how to modify them. Here is an example file that describes migrating to an SHV vault to a keyset (that can contain any key, for example an equivalent PKCS#11 CBC Derived key).

```
vault {
①
  source {
    configuration = "source_default"
    escrow = "source_default"
    transient = "source_default"
  }
②
  destination {
    configuration = "dest_default"
    escrow = "dest_default"
  }
}

③
vaults {
  source_default {
    module_path = "/usr/lib64/pkcs11/p11.so"
    slot_id = "12345678"
    pin = "1234"
    label = "horizon_masterkey"
    allow_master_key_gen = false
    vault_type = "shv"
  }
  dest_default {
    vault_type = "tink"
    path = "/opt/horizon/etc/horizon.keyset"
  }
}
```

① The previous vault types and the associated vault to use.

② The target vault types. The `transient` vault is now considered part of the `configuration` vault.

③ The definition of the vaults (how to encrypt the data.)

Others

If your current vault configuration does not only use `SSV`, the migration tool must be configured using a configuration file.

The migration file must be provided in the `HORIZON_MIG_280_VAULT_CONF_PATH` environment variable. This will override all other behaviors.

This file defines all the source and destination vaults, and how to modify them. Here is an example file that describes migrating to an `SHV` vault to a keyset (that can contain any key, for example an equivalent `PKCS#11 CBC Derived` key).

```
vault {
①
  source {
    configuration = "source_default"
    escrow = "source_default"
    transient = "source_default"
  }
②
  destination {
    configuration = "dest_default"
    escrow = "dest_default"
  }
}

③
vaults {
  source_default {
    module_path = "/usr/lib64/pkcs11/p11.so"
    slot_id = "12345678"
    pin = "1234"
    label = "horizon_masterkey"
    allow_master_key_gen = false
    vault_type = "shv"
  }
  dest_default {
    vault_type = "tink"
    path = "/opt/horizon/etc/horizon.keyset"
  }
}
```

① The previous vault types and the associated vault to use.

② The target vault types. The `transient` vault is now considered part of the `configuration`

vault.

③ The definition of the vaults (how to encrypt the data.)

Architecture-specific instructions

RPM

No specific instructions.

Debian

No specific instructions.

Kubernetes

Upgrading with a StatefulSet

When using a `StatefulSet` in Kubernetes (for instance when Analytics are configured with persistence), you'll need to take extra steps before launching the upgrade to ensure that pods get shutdown while the migration script is running:

- the `StatefulSet` strategy should be set to `RollingUpdate`;
- the `StatefulSet` resource should be scaled down to 1 replica:

```
$ kubectl scale --replicas=1 rs/{release-name}
```



If not scaled down to one replica, multiple versions of Horizon could write to the database and corrupt data.

Upgrading to 0.3.0

- Loggers are now configured with an array instead of a dictionary. Check the `values.yaml` format and update your override `values.yaml` accordingly.
- The init database parameters (`initDatabase`, `initUsername` and `initPassword`) have been renamed and moved to `mongodb.horizon`.

Upgrading to 0.5.0 - The ingress definition has changed. The `rules` and `tls` keys have been removed in favor of a more user-friendly `hostname` that will autoconfigure the ingress rules, and a boolean `tls` key that will enable TLS on that ingress. Check the Ingress section.

Upgrading to 0.9.0 - `clientCertificateDefaultParsingType` has been removed and is no longer supported by Horizon. Explicitly set the `clientCertificateHeader` or use ingress autoconfiguration to continue using client certificate authentication. - `ingress.type` will now be strictly validated. It may fail if you use an unsupported value. - `mailer.port`, `mailer.tls` and `mailer.ssl` are no longer set by default. You must now explicitly set if you want to use them.

Upgrading to 0.11.0 - New Lease CRD is added. - `akka.conf` has been replaced with

`pekko.conf`. It may fail if you use custom configuration otherwise it will be handled by the helm chart.

Upgrade to 0.16.0 - Switching to native Kubernetes leases implementation. CRDs leases aren't used anymore.

Upgrade to 1.0.0

- This version drops support for the Bitnami MongoDB subchart. Instead, a new `temporaryDatabase` key controls whether a temporary MongoDB instance should be created for the duration of the upgrade. To migrate from the Bitnami MongoDB subchart to a temporary instance or an external MongoDB database, you can use the `mongodump` and `mongorestore` utilities.

Upgrade to 2.0.0

- `vaults` section is no longer supported.
- `legacySsvPassword` must be set to allow migration of existing secrets.
- `defaultVault` must be configured with the correct values.
- `defaultVault.keyset` must be generated, stored as a secret, and properly referenced before performing the upgrade.

Docker

No specific instructions.

1.15. Uninstallation

Before uninstalling

Before uninstalling, please make sure that you have a **proper backup of the Horizon component**. That includes:

- the database contents
- a copy of your application secrets (in particular the SSV secret). Without it, you won't be able to decrypt your database and it will become useless.

If not, once uninstalled, all the Horizon data will be **irremediably lost**!

Uninstallation procedure

RPM

First, uninstall Horizon with the following commands:

```
$ systemctl stop horizon
$ yum remove horizon
```

```
$ rm -rf /opt/horizon
$ rm -rf /var/log/horizon
$ rm -f /etc/default/horizon
```

If NGINX was installed alongside with Horizon, you can remove it with the following commands:

```
$ systemctl stop nginx
$ yum remove nginx
$ rm -rf /etc/nginx
$ rm -rf /var/log/nginx
```

The same cleanup operation apply to MongoDB, which can be removed with the following commands:

```
$ systemctl stop mongod
$ rpm -qa | grep -i mongo | xargs rpm -e
$ rm -rf /var/log/mongodb
$ rm -rf /var/lib/mongodb
```

Debian

First, uninstall Horizon with the following commands:

```
$ systemctl stop horizon
$ apt remove horizon
$ rm -rf /opt/horizon
$ rm -rf /var/log/horizon
$ rm -f /etc/default/horizon
```

If NGINX was installed alongside with Horizon, you can remove it with the following commands:

```
$ systemctl stop nginx
$ apt remove nginx
$ rm -rf /etc/nginx
$ rm -rf /var/log/nginx
```

The same cleanup operation apply to MongoDB, which can be removed with the following commands:

```
$ systemctl stop mongod
$ apt-get purge "mongodb-org*"
$ rm -rf /var/log/mongodb
$ rm -rf /var/lib/mongodb
```

Kubernetes

To uninstall Horizon from your cluster, simply run:

```
$ helm uninstall horizon -n horizon
```

This will uninstall Horizon. If you installed a local MongoDB instance through the Horizon's chart, it will also be uninstalled, meaning you'll lose all data from the instance.

Chapter 2. Admin guide

Description

The admin guide describes configuration options for the Horizon product.

Prerequisites

To administrate Horizon, you need the following prerequisites:

- an up and running Horizon instance, which you can access through your web browser;
- an administrator account (or any account with configuration permissions), see Startup & login.

2.1. User Information

This is the section where to find all your profile information (identifier, email, name, authentication type, role and permissions), your preferences and change your account password (local account authentication only).

Profile access

1. Log in to Horizon.

2. Access your profile from the header by clicking on your account name.

How to change your password

1. Profile access

2. Fill your local password and confirm it.

3. Click on the 'Change Password' button.



Changing your password is only available if you are using a local account.

How to change your preferences

1. Profile access

2. Change your preferences:

- Appearance (light/dark mode)
- Horizon default language

3. Click on the 'Save' button.

2.2. Certification Authorities

This section details how to configure the Certification Authorities known by EverTrust Horizon.

Prerequisites

Certification Authorities will be needed beforehand, in one of these formats:

- Certificate file (PEM or DER).
- Certificate string (PEM).

You might also need the URL of the CRL issued by the CA, and/or the URL of the OCSP Responder for that CA.

How to configure a Certification Authority

1. Log in to Horizon Administration Interface.

2. Access Certification Authorities from the drawer or card: **Certification Authorities**.

3. Click on  .

Certificate Tab:

4. Either

- Fill in the certificate section with certificate string (PEM) OR
- Import the certificate file (PEM or DER).

Then click on the next button.

Details Tab:

5. Check the information from your CA certificate. Then click on the next button.

Configuration Tab:

6. Fill in the information you want to add.

- **Name*** *(string input)*:
  Enter a meaningful certificate authority name. It must be unique for each certificate authority.
- **OCSP responder URL** *(string)*:
  URL to request an OCSP responder.
- **CRL URL** *(string)*:
  URL to download the CA CRL.
- **Outdated Revocation Status Policy** *(option)*:
  Select "Revoked" if you want all certificates to be handled as revoked if the CRL/OCSP are unavailable. Select "Last available status" if you want Horizon to use the last available

revocation status for the certificates.

- **Refresh Period** *(finite duration)*:
  CRL refresh period. Must be a valid finite duration.
- **Proxy** *(string select)*:
  The HTTP/HTTPS proxy to use to reach the CRL or the OCSP Responder, if any.
- **Timeout** *(finite duration)*:
  Connection timeout when reaching CRL or OCSP. Must be a valid finite duration.
- **Trusted for server authentication** *(boolean)*:
  Tells whether the CA should be trusted for server authentication, aka SSL/TLS server trust. The default value is set to false.
- **Trusted for client authentication** *(boolean)*:
  Tells whether the CA should be trusted for client authentication. The default value is set to false.
- **Exposed on Registration Authority** *(boolean)*:
  Display the CA in the Trust chains view on the RA side. The default value is set to false.
- **Downloadable on Registration Authority** *(boolean)*:
  Allow the CA certificate to be downloaded from the Trust chains view on the RA side. The default value is set to false.

When **Trusted for client authentication** is enabled and the expert mode is active, three template-string fields are displayed to control how Horizon derives the authenticated identity from a client certificate. They use the standard template string syntax, with access to the raw certificate dictionary entries.

- **Identifier mapping** *(template string)*:
  Template used to compute the unique identifier of the authenticated principal. Default: `{{certificate.dn}}`.
- **Display name mapping** *(template string)*:
  Template used to compute the display name of the authenticated principal. Default: `{{certificate.subject.cn.1}}`.
- **Email mapping** *(template string)*:
  Template used to compute the email of the authenticated principal. Default: `{{certificate.san.rfc822name.1}}`.

7. Click on the import button.

You can edit , download  or delete  the Certification Authorities.



You will not be able to delete a Certification Authority if it is referenced in any other configuration element. Pay also attention that the CA might be used (e.g. for TLS trust chain building), even if it is not explicitly referenced in configuration items.

Exporting client authentication CAs

Horizon requires a reverse proxy for client certificate authentication (`mTLS`). To let the reverse proxy advertise the list of CAs that Horizon trusts for client authentication, Horizon can export the bundle of all CAs flagged as **Trusted for client authentication** to one or more destinations.

Exporters are declared in the HOCON configuration. Each exporter writes the bundle of all client authentication CAs in a configurable format, and Horizon refreshes the destination whenever a CA is added, updated or removed.

Supported destinations:

- **file**: a file on the local filesystem, suitable for nginx running on a VM (see `ssl_client_certificate`).
- **k8s-configmap**: a Kubernetes `ConfigMap`, suitable for the OpenShift Router which expects a `ca-bundle.pem` key.
- **k8s-secret**: a Kubernetes `Secret`, suitable for `ingress-nginx` which expects a `ca.crt` key referenced through the `nginx.ingress.kubernetes.io/auth-tls-secret` annotation.

The exporter is configured under `horizon.ca.exporters`:

```
horizon {
  ca.exporters = [
    {
      enabled = true

      type = "file"          # file | k8s-configmap | k8s-secret
      format = "pem-blocks"  # concatenated PEM blocks

      file {
        path = "/etc/nginx/horizon-client-cas.pem"
      }

      k8s-configmap {
        name = "horizon-client-cas"
        key  = "ca-bundle.pem"
      }

      k8s-secret {
        name = "horizon-client-cas"
        key  = "ca.crt"
      }
    }
  ]
}
```



Kubernetes destinations require the cluster API access configuration described under `horizon.kubernetes` (namespace, API host, service account token, and CA

file). Full configuration reference



If the targeted `ConfigMap` or `Secret` does not exist, Horizon creates it the first time the bundle is exported. Make sure the service account used by Horizon has the corresponding `create`, `update` and `get` permissions on the target object.

2.3. PKIs

2.3.1. PKI Queue

This section details how to configure a PKI Queue.PKI Queue are used to limit the PKI requests (enrollment, revocation)

PKI Queue Configuration

1. Log in to Horizon Administration Interface.

2. Access PKI Queues from the drawer or card: **PKI › PKI Queues**.

3. Click on ⊕ .

4. Fill in the fields:

- **Name*** *(string input)*:
  Choose a meaningful queue name. It must be unique.
- **Description** *(string input)*:
  The description for the PKI Queue.
- **Throttle Parallelism** *(int input)*:
  Number of requests processed at the same time.
- **Throttle Duration** *(finite duration)*:
  Maximum requests processed at the same time in a given duration. Parallelism must be set.
- **Max Size*** *(int input)*:
  Maximum requests stored in the queue
- **Cluster Wide** *(boolean)*:
  If not enabled, then the `throttleParallelism` and `throttleDuration` will be the same for all nodes in the cluster. If enabled, then the `throttleParallelism` and `throttleDuration` is generalized for all clusters.



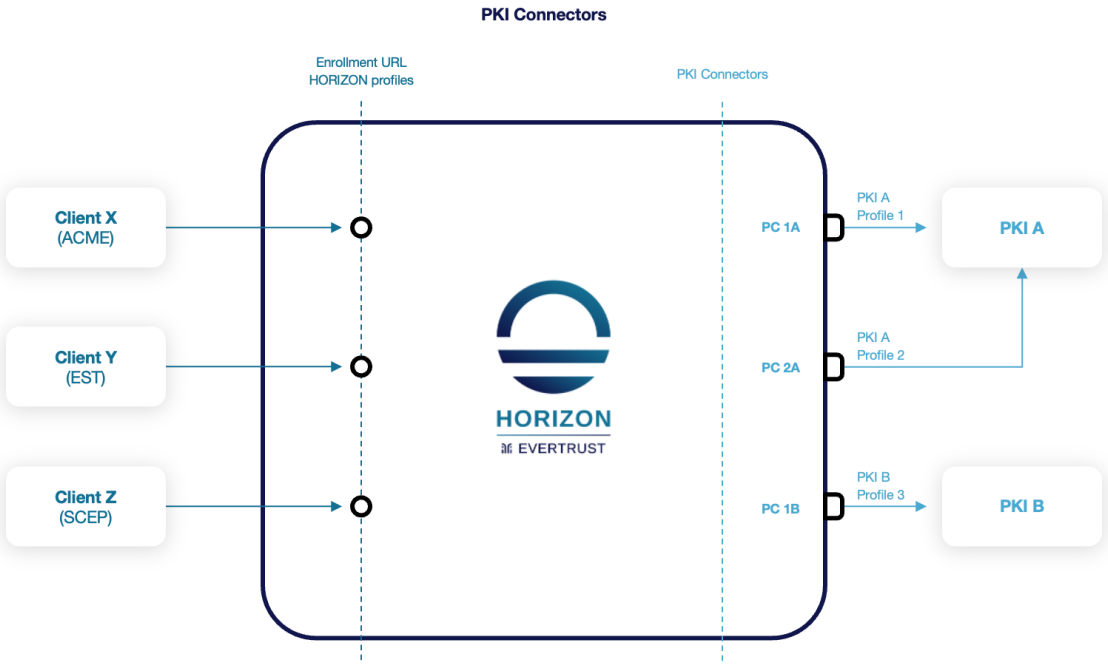
If the queue is full every new request will be discarded.

2.3.2. PKI Connectors

General Information

Description

A "PKI Connector" is a configuration piece that allows to establish the communication with any supported PKI. Additionally, it enables to map a specific certificate profile within the connected PKI.



Common prerequisites

To grant "Horizon" proper access to a given PKI, three categories of requirements must be gathered:

- **Credentials**: It could be either certificates (PKCS#12 format) or technical accounts (login/password) allowing to authenticate against the PKI API.
- **Permissions**: The credentials must be granted with the proper permissions on the PKI in order to be able to manage certificate lifecycle (enroll, revoke, renew).
- **Profile/Certificate information**: This information is used to map certificate types and/or certificate fields.

Asynchronous certificate lifecycle

Some PKIs cannot return a certificate within Horizon's synchronous request timeout, typically because they require an extended validation (manual review, email or phone confirmation, ...). Connectors that support asynchronous issuance let Horizon return a non-final request status, then resume the lifecycle in the background.

When a request has been submitted, but no certificate has yet been delivered, it is `In Progress`. The request is polled periodically by a background process until the certificate is available or the order is canceled. As the requester, an `in_progress` request can be canceled. As an approver, it can be denied. Either action cancels the underlying order on the PKI.



Asynchronous behavior is only used by enrollment workflows that opt in to it (currently WebRA enroll and renew). Protocol-driven workflows (ACME, SCEP, EST, ...) still run in synchronous mode to preserve their existing semantics.

At enrollment time, Horizon first performs a short synchronous polling cycle within the connector's configured timeout. If a certificate is returned during that window, the request completes as usual.

Otherwise, the request is switched to `in_progress` and a background process takes over.

Asynchronous Enrollment decision flow

```
flowchart LR
    st{{"Certificate enrollment"}}
    standardEnroll{{"Classic enroll"}}
    setStatus{{"Return the metadata (request 'in_progress')"}}
    request{{"Request enroll"}}
    waitTimeout{{"Waiting for PKI connector maximum timeout
(horizon.pki.manager.timeout)"}}
    e{{"Return the certificate (request 'completed')"}}

    asyncSupport{"Asynchronous enrollment requested and supported"}
    certificateReady{"Certificate ready"}

    st --> asyncSupport

    asyncSupport --"NO"--> standardEnroll
    asyncSupport --"YES"--> request

    request --> waitTimeout
    waitTimeout --> certificateReady

    certificateReady --"YES"--> e
    certificateReady --"NO"--> setStatus

    standardEnroll --> e
```

Once a request is `in_progress`, a background actor polls the PKI on the schedule defined by `horizon.async-poll.interval` (default: every 5 minutes). Retry intervals grow in 5-minute increments up to one hour. A successful poll updates the certificate; a transient error logs an event and reschedules the next poll; a fatal error switches the request to `failed`.

Public PKI reissue

To accommodate the gap between order lifetime (typically one year) and the reduced certificate lifetime imposed by the CA/Browser Forum (200 days, decreasing toward 47 days), public PKI connectors that support it now reissue a certificate on the same order during renewal instead of creating a new order, as long as the order is still valid.

The behavior is automatic and transparent:

- on renewal, if the connector reports an order expiration metadata (`PKI_ORDER_EXPIRATION`) and the order is not in its renewal period, Horizon issues a `reissue` against the existing order and inherits the original order expiration;
- if the order is in its renewal period, or if the connector does not expose an order expiration, Horizon falls back to a classic enroll, which creates a new order on the PKI.



Nameshield fetches the `PKI_ORDER_EXPIRATION` metadata on renewal even if it is not present on the certificate being renewed.

This guarantees that the certificate issued by a renewal is never already in its renewal period, and that subscriptions are consumed through to their end.



This feature is only available for certificates enrolled after version 2.10.0. Certificates enrolled before this version will not see their order reissued even if they still have validity time available.

Reissue vs. classic enroll decision flow

```
flowchart LR
    st{{"Certificate enrollment"}}
    standardEnroll{{"Classic enroll"}}
    reissue{{"Reissue"}}
    e{{"Return the certificate"}}

    inRenewalPeriod{"Order expiration is in certificate's renewal period"}
    renewMetadata{"Certificate to enroll is a renewal"}

    st --> renewMetadata

    renewMetadata --"YES"--> inRenewalPeriod
    renewMetadata --"NO"--> standardEnroll

    inRenewalPeriod --"NO"--> reissue
    inRenewalPeriod --"YES"--> standardEnroll

    reissue --> e
    standardEnroll --> e
```



Connectors with reissue support are:

- GlobalSign MSSL
- Nameshield
- DigiCert

AWS

Prerequisites

- You need to create a user using AWS IAM, and give it the `AWSCertificateManagerPrivateCAUser` right.
- You need to retrieve the Private CA ARN from ACM Private CA console.



Refer to the editor's documentation to configure the PKI side here.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on ⊕ .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **AWS Region*** *(string input)*:
  AWS region to use.
- **AWS PCA ARN*** *(string input)*:
  Amazon Resource Name (ARN) is a file naming convention used to identify a particular resource in AWS public cloud. To be retrieved from AWS ACM Console.
- **AWS PCA Template ARN** *(string input)*:
  A template is a declaration of the AWS resources that make up a stack. The default value is set to: `arn:aws:acm-pca:::template/EndEntityCertificate/V1`.
- **AWS PCA Role ARN** *(string input)*
- **Certificate Policy OID** *(string input)*:
  An identifying number, in the form of an "object identifier" that is included in the `certificatePolicies` field of a certificate.

- **Certificate signing hash** *(select)*:
  Select the hash function that will be used.
- **Certificate Usage** *(select)*:
  Select the certificate usage.
- **Number of valid days** *(finite duration)*:
  Certificate validity duration in days. Must be a valid finite duration. The default value is set to 365 days.
- **Retry Interval** *(finite duration)*:
  Predefined interval of time before retrying to retrieve the certificate from AWS. Must be a valid finite duration. The default value is set to 3 seconds.

9. Click on the next button

Authentication tab

10. Fill in the PKI-authentication fields:

- **AWS Access Credentials*** *(select)*:
  Select `Login` credentials containing the AWS user access key ID and the AWS user secret key (see the AWS Account and Access Keys documentation).

11. Click on the save button.

You can edit , duplicate or delete the AWS PKI connector.

CertEurope

Prerequisites

- A technical account should be created.
- This technical account must have permissions to enroll and revoke SSL certificates on the desired domain(s).

Limitations

- Only the following fields are managed: `commonName` and `subjectAltName DNS`.
- For multi-valued fields (SAN DNS), if more data items are provided than configured in CCS for the given "Offer Identifier", the exceeding items will be ignored.
- All limitations induced by the use of the CCS REST Connector.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on  .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **Endpoint URL to the CSS partner API*** *(string input)*:
  URL to access the CertEurope web service API.
- **Technical account credentials*** *(select)*:
  Select `Login` credentials containing your technical account created in CCS. The login is usually an email address.
- **CCS offer identifier*** *(string input)*:
  The identifier of the offer within CCS.
- **Organization ID*** *(string input)*:
  Customer organization ID. For French companies, it's usually the "SIREN".
- **Revocation reason** *(string select)*:
  Select from the drop down the default revocation reason.
- **Interval before retrying to retrieve certificate** *(finite duration)*:
  The default value is set to 21 seconds.

9. Click on the next button.

Authentication tab

10. Fill in the PKI-authentication fields:

- **Authentication Credentials*** *(select)*:
  Select `Certificate` credentials containing the authentication certificate used to connect to the PKI.

11. Click on the save button.

You can edit , duplicate  or delete  the CertEurope PKI connector.

CS-Novidy's TrustKey

Prerequisites

- A technical account should be created.
- This technical account must have permissions to enroll and revoke SSL certificates on the desired certificate profiles.
- An authentication and a signature certificate must be issued under as PKCS#12 files for this account.

Limitations

- Only the following fields are managed: `commonName (as mail_lastname)`, `contactEmail (as mail_email)`, `OU (as org_unit)`, `O (as corp_company)`, `C (as country)`, `UID (as employeeID)`, `subjectAltNames DNS` and `msUPN`.
- For multi-valued fields (SAN DNS), if more data items are provided than configured in TrustyKey for the given `PGC`, the exceeding items will be ignored.
- All limitations induced by the use of the TrustyKey CMP Connector.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.

- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **API endpoint URL*** *(string input)*:
  URL to access the CS-Novidy's TrustyKey web service.
- **PGC*** *(string input)*:
  Enter name of the `PGC` to be used.
- **TrustyKey PKI server DN*** *(string input)*:
  Enter the DN of the TrustyKey PKI server, starting from the CN.
- **TrustyKey PKI server Certificate*** *(string input)*:
  Enter the PEM representing the certificate of the CA issuing the certificates.
- **CN mapping** *(string input)*:
  Enter a CN to be mapped.
- **Email mapping** *(string input)*: Enter an email address or domain to be mapped.
- **SAN DNS mapping** *(string input)*:
  Enter a SAN DNS to be mapped.
- **Profile mapping** *(string input)*:
  Enter a profile to be mapped.
- **Issuer mapping** *(string input)*:
  Enter an issuer to be mapped.
- **Legacy CMP Style** *(boolean)*
  Chose whether to use the legacy CMP style.

9. Click on the next button.

Authentication tab

10. Fill in the PKI-authentication fields:

- **Authentication Credentials*** *(select)*:
  Select `Certificate` credentials containing the authentication certificate used to connect to the PKI.

- **Signer Credentials*** *(select)*:
  Select `Certificate` credentials containing the signature certificate used to sign the CMP messages.

11. Click on the save button.

You can edit , duplicate or delete the CS-Novidy's TrustyKey PKI connector.

Digicert CertCentral

Prerequisites

- You need to validate the domain(s) for which you will issue certificates prior to their issuance. This can be done in DigiCert CertCentral in the Certificates > Domains menu.
- You need to retrieve the `organizationId` from DigiCert CertCentral in the Certificates > Organizations menu.
- You need to generate an API Key in DigiCert CertCentral using the Account > Account Access menu.

Limitations

- Only the following fields are managed: `commonName` and `subjectAltName DNS` and `RFC822Name`.
- For multi-valued fields (SAN DNS and `RFC822Name`), if more data items are provided than configured in DigiCert CertCentral for the given type of certificate, the exceeding items will be ignored.
- All limitations induced by the use of the DigiCert CertCentral REST Connector.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:

If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.

- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **DigiCert CertCentral API baseUrl*** *(string input or select)*:
  Base url to access DigiCert CertCentral API along with the certificate type to issue the possible values are listed here. To do so you can select from the drop down menu or type in your "certificate offer" value, then press "Enter" the corresponding URL will be automatically fetched.
- **DigiCert CertCentral API productId*** *(string input or select)*:
  The type of certificate enrolled on the PKI. An exhaustive list is available here.
- **DigiCert CertCentral Customer Organization ID*** *(int)*:
  Enter customer organization ID.
- **DigiCert CertCentral CA Cert ID** *(int)*:
  Enter CA Cert ID, to be used for private CA only.
- **Interval before retrying to retrieve certificate** *(finite duration)*:
  Use for private CA only. The default value is set to 9 seconds.
- **Skip Approval** *(boolean)*:
  The default value is set to false.

9. Click on the next button.

Custom tab

10. Click on  if custom data mapping is needed.

11. Fill in the PKI-custom data mapping:

- **Custom data field*** *(string input)*:
- **Label field*** *(select)*:
  Any existing Horizon Label

12. Click on the next button.

Authentication tab

13. Fill in the PKI-authentication fields:

- **DigiCert CertCentral API Key*** *(select)*:
  Select `API Token` credentials containing the API Key.

14. Click on the save button.

You can edit , duplicate or delete the DigiCert CertCentral PKI connector.

Digicert ONE

Prerequisites

- The DigiCert ONE certificate profile identifier (UUID).
- A DigiCert ONE service user with access to Trust Lifecycle is required. The connector authenticates with its API key. See DigiCert ONE authentication.
  - On DigiCert ONE certificate profile, it needs the following Trust Lifecycle permissions:
    - `CREATE_EM_CERTIFICATE` to enroll certificates.
    - `RENEW_EM_CERTIFICATE` to renew certificates.
    - `REVOKE_EM_CERTIFICATE` to revoke certificates.
    - `MANAGE_EM_SEAT` to create seats, only with a dynamic seat.
- Seat:
  - Static seat: the seat must already exist in DigiCert ONE. Fetch its technical id (UUID).
  - Dynamic seat: fetch the technical id (UUID) of the business unit in which the seats are created.
- Identify the DigiCert ONE base URL matching the target environment. The available base URLs are listed here.

Limitations

- Only the following subject elements are sent to DigiCert ONE: CN, O, OU, L, ST, C and E.
- Only the following SAN types are sent to DigiCert ONE: DNS names, IP addresses, emails (RFC822), URIs, UPNs and registered IDs.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **DigiCert ONE API baseUrl*** *(string input)*:
  Base URL of the DigiCert ONE API. Example: `https://one.digicert.com/`
- **Profile ID*** *(string input)*:
  Identifier (UUID) of the DigiCert ONE certificate profile used to issue certificates.
- **Interval before retrying to retrieve certificate*** *(finite duration)*:
  Interval at which pending requests are polled during the enrollment process. See Asynchronous certificate lifecycle.
- **Validity** *(finite duration)*:
  Overrides the validity of the certificates issued through this connector. When not set, the validity defined by the DigiCert ONE certificate profile is used. The profile must allow overriding its default validity. The value is sent to DigiCert ONE in minutes, so any precision below the minute is truncated.
- **Seat mode*** *(select)*:
  Every DigiCert ONE enrollment references a seat, the licensed identity (server, device, user) a certificate is issued for. Select one of the following modes:
  - `Static`: all certificates are enrolled on a single existing seat. This is the recommended mode.
    - **Seat ID*** *(string input)*:
      Technical identifier (UUID) of the existing DigiCert ONE seat. Example: `3f8c2a1e-7b4d-4e9a-9c6f-1d2e5a8b0c47`
  - `Dynamic`: the connector creates a seat for each certificate, named after the requested common name, and reuses it if it already exists.
    - **Business unit ID*** *(string input)*:

Technical identifier (UUID) of the DigiCert ONE business unit in which the seats are created. It must be the business unit the certificate profile belongs to, otherwise enrollment and revocation fail. Example: `3f8c2a1e-7b4d-4e9a-9c6f-1d2e5a8b0c47`

- **Seat type*** *(select)*:
  Type of the created seats: `USER_SEAT`, `DEVICE_SEAT`, `CERTIFICATE_MANAGEMENT_SEAT`, `ORGANIZATION_SEAT` or `SERVER_SEAT`.

9. Click on the next button.

Authentication tab

10. Fill in the PKI-authentication fields:

- **DigiCert ONE API Key*** *(select)*:
  Select `API Token` credentials containing the API key of the DigiCert ONE service user.

11. Click on the save button.

You can edit , duplicate  or delete  the DigiCert ONE PKI connector.

Entrust CA Gateway

Prerequisites

- The base URL of the Entrust CA Gateway, without the `/v1` API version. Example: `https://cagateway.example.com/cagw`
- The identifier of the CA on the Entrust CA Gateway.
- The identifier of the CA Gateway profile used to enroll certificates.
- A client certificate and its private key, authorized on the Entrust CA Gateway. The connector authenticates using mutual TLS.

Limitations

- The subject DN is sent as subject variables, driven by the subject variable requirements of the CA Gateway profile. DN elements that the profile does not require are not sent to Entrust CA Gateway.
- Only the following SAN types are sent to Entrust CA Gateway: DNS names, IP addresses, emails (RFC822), URIs and registered IDs.
- CA specific operations (reissue, renew) are not supported.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on  .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **Endpoint*** *(string input)*:
  Base URL of the Entrust CA Gateway, without the `/v1` API version. Example: `https://cagateway.example.com/cagw`
- **CA ID*** *(string input)*:
  Identifier of the CA on the Entrust CA Gateway.
- **Profile ID*** *(string input)*:
  Identifier of the CA Gateway profile used to enroll certificates.
- **Validity period** *(string input)*:
  Overrides the validity of the certificates issued through this connector, as an ISO 8601 interval or duration accepted by the Entrust CA Gateway. When not set, the validity defined by the CA Gateway profile is used. The profile must allow the validity to be overridden. Accepted forms:
  - a duration only, starting when the certificate is issued: `P1Y3M10DT0H0M`
  - a start date and an expiry date: `2018-07-06T13:00Z/2019-07-06T13:00:00Z`
  - a start date and a duration: `2018-07-06T13:00Z/P1Y3M0DT0H0M`
  - a duration and an expiry date, the start date being computed from them: `P0Y3M0DT0H0M/2018-12-31T00:00Z`

9. Click on the next button.

Custom tab

10. Click on  if custom data mapping is needed. Each mapped label is sent as a property of the enrollment request.

11. Fill in the PKI-custom data mapping:

- **Custom data field*** *(string input)*:
  Name of the property sent to Entrust CA Gateway.
- **Label field*** *(select)*:
  Any existing Horizon Label

12. Click on the next button.

Authentication tab

13. Fill in the PKI-authentication fields:

- **Authentication credentials*** *(select)*:
  Select `Certificate` credentials containing the authentication certificate used to connect to the Entrust CA Gateway.

14. Click on the save button.

You can edit , duplicate  or delete  the Entrust CA Gateway connector.

EJBCA

Prerequisites

- A certificate profile should be created, e.g. reusing the default "SERVER" certificate profile.
- An authentication certificate should be issued for Horizon, and it should be given certificate issuance and revocation permissions on the aforementioned certificate procedure.

Limitations

- Only the following fields are managed: all Subject DN fields and `subjectAltNames` `DNS`, `IPaddress`, `RFC822Name`, `msUPN` and `msGUID`.
- For multi-valued fields (SAN DNS and `RFC822Name`), if more data items are provided than configured in EJBCA for the given *End Entity* profile, the exceeding items will be ignored.
- All limitations induced by the use of the EJBCA RA SOAP Connector.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on  .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **EJBCA RA URL*** *(string input)*:
  Enter SOAP endpoint URL of the EJBCA WebService.
- **EJBCA Certificate Profile Name*** *(string input)*:
  Enter EJBCA Certificate Profile to map for certificate issuance.
- **EJBCA CA Name*** *(string input)*:
  Enter CA to use for certificate issuance.
- **EJBCA End Entity Profile*** *(string input)*:
  Enter EJBCA End Entity profile.

9. Click on the next button.

Authentication tab

10. Fill in the PKI-authentication fields:

- **Authentication Credentials*** *(select)*:
  Select `Certificate` credentials containing the authentication certificate used to connect to the

PKI.

11. Click on the save button.

You can edit , duplicate or delete the EJBCA PKI connector.

Eviden IDCA

Prerequisites

- A certificate profile should be created.
- An authentication certificate should be issued for Horizon, and it should be given certificate issuance and revocation permissions on the aforementioned certificate profile.

Limitations

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **IDCA RA Connector URL*** *(string input)*:
  Must point to the "RA" connector URL.
- **IDCA Certificate template name*** *(string input)*:
  The IDCA certificate template to use.
- **IDCA partition** *(string input)*:
  Specify a partition (if used).

9. Click on the next button.

Authentication tab

10. Fill in the PKI-authentication fields:

- **Authentication Credentials*** *(select)*:
  Select `Certificate` credentials containing the authentication certificate used to connect to the PKI.

11. Click on the save button.

You can edit , duplicate  or delete  the OpenTrust PKI connector.

EverTrust Integrated CA

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:

The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).

- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **Certificate Type*** *(select)*:
  Specify the certificate type to issue.
- **Signing algorithm*** *(select)*:
  Specify the signing algorithm.
- **CA Certificate** *(string input)*:
  Enter CA certificate.
- **CA Key** *(string input)*:
  Enter CA key.
- **CRL save path** *(string input)*:
  Path to save the CRL on the Horizon server.
- **CRL lifetime** *(finite duration)*:
  CRL lifetime in days. Must be a valid finite duration.
- **Certificate Back Date** *(finite duration)*:
  Certificate Back Date. Must be a valid duration.
- **Check Proof of Possession** *(boolean)*

9. Click on the save button.

You can edit , duplicate  or delete  the EverTrust integrated CA PKI connector.

EverTrust Stream CA

Prerequisites

- A certificate template should be created in Stream for Horizon to enroll certificates upon.
- A dedicated Horizon account should be created in Stream and should have all lifecycle permissions on the desired CA. The credentials of this account should be either login and password or a PKCS#12 authentication certificate.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on  .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

8. Fill all mandatory fields:

- **Endpoint*** *(string input)*:
  Fill in the Stream endpoint url.
- **Template*** *(string input)*:
  Fill in the Stream certificate template to enroll upon.
- **CA** *(string input)*:
  Fill in the Stream CA enrolling certificate (internal name).

9. Click on the next button.

Authentication tab

- **Authentication Credentials*** *(select)*:
  Select `Certificate` credentials containing the authentication certificate used to connect to the PKI, or `Credentials` containing the dedicated Horizon account on Stream.

10. Click on the save button.

You can edit  , duplicate  or delete  the Evertrust Stream PKI connector.

FISId

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on  .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **FISId endpoint URL*** *(string input)*:
  URL to access the API.
- **Template ID*** *(int)*:
  Enter the template ID.
- **Default owner ID*** *(string input)*:
  Enter a default owner ID.
- **Authentication domain ID*** *(int)*:
  Enter an authentication domain ID.
- **Owner groups** *(string input)*:
  Enter one or several, separated by commas

- **To delete after revocation** *(boolean)*:
  The default value is set to false.

9. Click on the next button.

Authentication tab

10. Fill in the PKI-authentication fields:

- **API Key*** *(select)*:
  Select `API Token` credentials containing the API Key.

11. Click on the save button.

You can edit , duplicate or delete the FISId PKI connector.

GCP Certificate Authority Service

Prerequisites

- You need a CA pool containing at least one enabled certificate authority, and to retrieve the GCP project ID and the location hosting it. The pool automatically selects an enabled certificate authority at issuance time.
- You need to create a service account using the GCP IAM, and grant it the appropriate permissions on the CA pool, as documented here. Typically, these can be granted through the CA Service Certificate Requester role (`roles/privateca.certificateRequester`) for issuance and the CA Service Certificate Manager role (`roles/privateca.certificateManager`) for revocation, or through the individual following permissions:
  - `privateca.certificates.create`
  - `privateca.certificates.list`
  - `privateca.certificates.update`
- You need to retrieve the JSON key file of that service account, unless Horizon runs on GCP and authenticates through Application Default Credentials.

Refer to the editor's documentation to configure the PKI side here.

The health check of this connector only lists the certificates of the CA pool. It succeeds with an empty list even when the service account lacks the issuance or revocation permissions, so a healthy connector does not guarantee that enrollment will succeed.

Limitations

- Only the following Subject DN fields are managed: `CN`, `C`, `O`, `OU`, `L`, `ST` and `STREET`. Any other DN element is ignored and not sent to GCP.
- Subject DN fields are single-valued on GCP. For multi-valued fields, the exceeding items will be

ignored.

- Only the following `subjectAltNames` are managed: `DNS`, `URI`, `RFC822Name` and `IPaddress`. `msUPN`, `msGUID` and `registeredID` are not supported.
- Certificate extensions and policies are defined by the CA pool or by the certificate template, not by Horizon.
- All limitations induced by the use of the GCP Certificate Authority Service API.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on  .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **Project ID*** *(string input)*:
  ID of the GCP project hosting the CA pool. It may differ from the project of the service account used to authenticate.
- **Location*** *(string input)*:
  Location of the CA pool, for example `europe-west1`.

- **CA pool*** *(string input)*:
  ID of the CA pool to use for certificate issuance.
- **Certificate lifetime*** *(finite duration)*:
  Validity duration applied to every certificate issued through this connector. Must be a valid finite duration. There is no default value.
- **Certificate template** *(string input)*:
  The CAS certificate template to apply. Enter either its short name or its full resource path, in the form `projects/<project>/locations/<location>/certificateTemplates/<template>`.
- **Endpoint** *(string input)*:
  Overrides the endpoint used to reach the API. By default, the standard Google Cloud endpoint is used.

9. Click on the next button.

Authentication tab

10. Fill in the PKI-authentication fields:

- **Credentials** *(select)*:
  Select `API Token` credentials whose token holds the full content of the service account JSON key file. If left empty, Horizon falls back to the Application Default Credentials of the machine it runs on, which covers workload identity.
- **Impersonated service account** *(string input)*:
  Email of the service account to impersonate. Leave empty not to use impersonation.
- **Impersonation lifetime** *(finite duration)*:
  Validity duration of the impersonated token. Mandatory when an impersonated service account is set. Must be a valid finite duration and must not exceed 12 hours.

11. Click on the save button.

You can edit , duplicate or delete the GCP Certificate Authority Service PKI connector.

GlobalSign Atlas

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **Hash Algorithm** *(select)*:
  Select the hash algorithm for the certificate to be issue.
- **API Credentials*** *(select)*:
  Select `Login` credentials containing the key and password that allows to authenticate against GlobalSign Atlas API.
- **Certificate Usage** *(select)*:
  Select a usage from the drop down list.
- **Retry Interval** *(finite duration)*:
  The default value is set to 3 seconds.

9. Click on the next button.

Authentication tab

10. Fill in the PKI-authentication fields:

- **Authentication Credentials*** *(select)*:
  Select `Certificate` credentials containing the authentication certificate used to connect to the PKI.

10. Click on the save button.

You can edit , duplicate  or delete  the GlobalSign Atlas PKI connector.

GlobalSign MSSL

Prerequisites

- A technical account should be created.
- This technical account must have permissions to enroll and revoke SSL certificates on the desired domain.

Limitations

- Only the following fields are managed: `contactEmail` and `subjectAltName DNS`.
- For multi-valued fields (SAN DNS), if more data items are provided than configured in GlobalSign MSSL for the given "Product", the exceeding items will be ignored.
- All limitations induced by the use of the GlobalSign MSSL SOAP Connector.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on  .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **GlobalSign endpoint*** *(string select)*:
  Select from the drop-down list: the value must be "prod" for GlobalSign Production endpoint or "test" for the test environment.
- **GlobalSign profile ID*** *(string input)*:
  To be retrieved from the URL in the GlobalSign MSSL console.
- **GlobalSign domain ID*** *(string input)*:
  The ID of the domain to manage. Displayed in the GlobalSign MSSL console.
- **Certificate validity** *(int input)*:
  Certificate validity in months.
- **Default email address** *(string input)*:
  Choose a default email address.
- **Default phone number** *(string input)*:
  Choose a default phone number.
- **Interval before retrying to retrieve certificate** *(finite duration)*:
  The default value is set to 9 seconds.

9. Click on the next button.

Authentication tab

10. Fill in the PKI-authentication fields:

- **Technical account credentials*** *(select)*:
  Select `Login` credentials containing the login/password of the account created in GlobalSign MSSL.

11. Click on the save button.

You can edit , duplicate  or delete  the GlobalSign MSSL PKI connector.

MetaPKI

Prerequisites

Endpoint issuing CA

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **Endpoint*** *(string input)*:
  The MetaPKI Endpoint.
- **Endpoint Issuing CA*** *(string select)*:
  Select the CA that will be issuing the certificates for this connector (from the imported Horizon CAs)
- **Profile*** *(string input)*:
  Example: Applications_Auth_Client_Serveur_SSL.
- **Profile Cle*** *(string input)*:
  Example: Serveur_SSL
- **Workflow*** *(string input)*:
  Example: S_LOCAL_SOFT
- **Form Porteur Name** *(string input)*
- **Valid Days** *(finite duration)*
  Certificate lifetime in days (must be a valid finite duration).

9. Click on the next button.

Authentication tab

10. Fill in the PKI-authentication fields:

- **Authentication Credentials*** *(select)*:

Select `Certificate` credentials containing the authentication certificate used to connect to the PKI.

11. Click on the save button.

You can edit , duplicate or delete the MetaPKI PKI connector.

MSAD Certificate Services

Setup of the ADCS Connector

ADCS Connector installation guide must be completed prior to the configuration of this connector.

Creating the ADCS PKI Connector in Horizon

The previous steps are considered as pre-requisites to continue the setup. If you haven't yet configured the ADCS Connector on the ADCS side, please refer to the Setup of the ADCS Connector. The rest of this section assumes that the EverTrust ADCS Connector is installed and correctly set-up on the ADCS side.

Limitations

- All limitations induced by the use of ADCS.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment,

revocation).

- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **Endpoint*** *(string input)*:
  URL to access the machine where the ADCS connector is running on port 4443.
- **Active Directory Domain Netbios Name*** *(string input)*:
  The NETBIOS name of the Active Directory domain where to find the technical user and the ADCS server.
- **Profile*** *(string input)*:
  The technical name of the template that you created at step 8 of the Setup of the ADCS Connector section. Example: WebServer
- **CA Config*** *(string input)*:
  The `CaConfig` string, as given out by `certutil -getconfig` for the considered ADCS CA. It's usually in the form `<ADCS Hostname>\<CA CommonName>`

9. Click on the next button.

Authentication tab

10. Fill in the ADCS authentication fields:

- **Enrollment agent certificate*** *(select)*:
  Select `Certificate` credentials containing the PKCS#12 enrollment agent certificate that was exported at step 10 of the Setup of the ADCS Connector section.
- **MS ADCS user account*** *(select)*:
  Select `Login` credentials containing the username and password of the technical account created at step 9 of the Setup of the ADCS Connector section.



Specify only the username of the technical account on the ADCS machine, without the Netbios domain name.
For example, in **PKI\Technical** do not include the **PKI** part.

11. Click on the save button.

You can edit , duplicate  or delete  the Microsoft Active Directory Certificate Services PKI connector.

Nameshield

Prerequisites

- A dedicated Horizon account with enroll and revoke permissions must be set up
- An authentication token must be obtained using Nameshield's procedure

Limitations

- CSR must contain at least a FQDN CN and DNS SAN
- Only DNS SANs can be overriden
- Renewal period of the profiles using this connector should be aligned with the renewal period of the NameShield platform (30d), as renewal will otherwise be blocked on the PKI.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on  .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

8. Fill all mandatory fields:

- **Environment*** *(select)*:
  Fill in the environment of the nameshield instance (Production or Testing).
- **Organization ID*** *(number input)*:
  Fill in the Nameshield Organization ID.
- **Product ID*** *(number input)*:
  Fill in the Nameshield Product ID.
- **Customer ID*** *(number input)*:
  Fill in the Nameshield Customer ID.

9. Click on the next button.

Authentication tab

- **API Key*** *(select)*:
  Select `API Token` credentials containing the authentication token used to connect to Nameshield.

10. Click on the save button.

You can edit , duplicate  or delete  the Nameshield connector.

Nexus Certificate Manager

Prerequisites

- A certificate procedure and a token procedure should be created.
- An authentication certificate should be issued for Horizon, and it should be given certificate issuance and revocation permissions on the aforementioned token procedure.
- Nexus Endpoint CA

Limitations

- Only the following fields are managed: `commonName, UID, OU, O, C` and `subjectAltNames DNS, IPaddress, RFC822Name` and `msUPN`.
- For multi-valued fields (SAN DNS, RFC822Name and IP address), if more data items are provided than configured in Nexus CM Procedure, the exceeding items will be ignored.
- All limitations induced by the use of the Nexus CM SDK.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill all mandatory fields:

- **Nexus CM DNS name*** *(string input)*:
  URL to access the Nexus Certificate Manager. Two modes are available:
  - Direct connection, you can specify the IP:PORT
  - Using PGWY, you will need to specify the PGWY url as following https://<pgwy_url>/sdkproxy
- **Nexus endpoint CA*** *(select)*:
  Select the endpoint CA.
- **Nexus CM Certificate procedure name*** *(string input)*:
  The token procedure name to use.
  Should point to the appropriate certificate procedure, and must be on PKCS#10 format.

9. Click on the next button.

Authentication tab

10. Fill in the PKI-authentication fields:

- **Authentication Credentials*** *(select)*:
  Select `Certificate` credentials containing the authentication certificate used to connect to the PKI.

11. Click on the save button.

You can edit , duplicate or delete the Nexus Certificate Manager PKI connector.

OpenTrust PKI

Prerequisites

- A certificate profile should be created.
- An authentication certificate should be issued for Horizon, and it should be given certificate issuance and revocation permissions on the aforementioned certificate profile.

Limitations

- Only the following fields are managed: `commonName`, `userID`, `serialNumber`, `organizationalUnit`, `organization`, `country`, `adminEmail` or `contactEmail`, `msCertTemplateName` and `subjectAltNames` `DNS`, `IPadress`, `RFC822Name`, `msUPN` and `msGUID`.
- For multi-valued fields (SAN DNS, IP address and RFC822Name), if more data items are provided than configured in OTPKI 'certificate template name', the exceeding items will be ignored.
- All limitations induced by the use of the RA SOAP Connector.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed

"Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **OTPKI RA Connector URL*** *(string input)*:
  Must point to the "RA" connector URL.
- **OTPKI Certificate template name*** *(string input)*:
  The OTPKI certificate template to use.
- **OTPKI zone** *(string input)*:
  Specify a zone (if used).
- **Contact email mapping** *(string input)*:
  Allows to change the default fields names accordingly to certificate profiles.
- **SAN DNS mapping** *(string input)*:
  Allows to change the default fields names accordingly to certificate profiles.
- **SAN Email mapping** *(string input)*:
  Allows to change the default fields names accordingly to certificate profiles.
- **UID mapping** *(string input)*:
  Allows to change the default fields names accordingly to certificate profiles.

9. Click on the next button.

Authentication tab

10. Fill in the PKI-authentication fields:

- **Authentication Credentials*** *(select)*:
  Select `Certificate` credentials containing the authentication certificate used to connect to the PKI.

11. Click on the save button.

You can edit , duplicate or delete the OpenTrust PKI connector.

Sectigo SCM

Prerequisites

- For publicly trusted certificates, you need to validate the domain(s) for which you will issue certificates prior to their issuance.
- You need to retrieve the `customerUri` and the `organizationId` from Sectigo SCM.
- You need to create a technical account with appropriate permissions including the `allow ssl auto approve` permission. You need to set a password for the technical account.

Limitations

- Only the `subjectAltName DNS` field is managed.
- The certificate Subject DN will be set to whatever is specified in the PKCS#10 CSR.
- All limitations induced by the use of the Sectigo SCM REST Connector.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on  .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.
- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

Details tab

8. Fill in all mandatory fields:

- **Customer URI*** *(string input)*:
  Enter the Customer URI. An integer is expected.
- **Organization ID*** *(int input)*:
  Enter the Organization ID.
- **Profile (Certificate Type)*** *(string input)*:
  Enter the Profile (Certificate Type). An integer is expected.

- **Retry interval** *(finite duration)*:
  Predefined interval of time before retrying to retrieve the certificate from Sectigo. Must be a valid finite duration. No default value is set.
- **Valid Days** *(finite duration)*:
  Certificate validity duration in days. Must be a valid finite duration. No default value is set.

9. Click on the next button.

Authentication tab

10. Fill in the PKI-authentication fields:

- **Authentication credentials*** *(select)*:
  Select `Login` credentials containing your Sectigo SCM login and password.

11. Click on the save button.

You can edit , duplicate  or delete  the Sectigo SCM PKI connector.

ACME

Prerequisites

- An ACME directory URL.
- If required by your ACME provider, External Account Binding credentials.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to properly forward the traffic.

- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

8. Fill all mandatory fields:

- **Endpoint*** *(string input)*:
  Fill in the ACME directory url. It often ends in `/directory`.
- **Account Key Type*** *(select)*:
  The key type to use for the ACME account that will be created on the directory. Using rsa or ecdsa is recommended, depending on your ACME provider.
- **Account Email** *(string input)*:
  Fill in the email to associate with the account. It will be used at the ACME provider's discretion, to inform on certificate status.
- **External Account Binding** *(select)*:
  Select `Login` credentials containing the External Account Binding (EAB) Key ID as login and the EAB Key as password if your provider requires EAB.
- **Rotate Account** *(boolean)*:
  Activate this if you wish to recreate the account associated with this connector (not needed if no account was yet created). This allows to rotate the account key if required.
- **DNS Provider*** *(select)*:
  Select the DNS provider that will expose the ACME challenge. The following steps will change according to the selected provider.

9. Click on the next button.

Manual

Domain dictionary configuration

Domain dictionaries are available to configure domain-specific dictionary keys. These will only be available when creating the DNS record to validate the specified domain. For CloudFlare, this should contain the zone id for example.

- **Domain*** *(string input)*:
  Define the domain for which the dictionary is available.
  - **Key*** *(string input)*:
    The dictionary key to use in the REST trigger.
  - **Value*** *(string input)*:
    The value associated with the key.

10. Click on the next button.

DNS Record Creation REST Call

This REST request needs to create the TXT DNS record in your DNS Provider

Available dictionary keys:

- **record**: the expected name of the DNS record
- **digest**: the challenge value (content of the DNS record)
- **domain**: the domain the notification is trying to validate. **This is for informational purpose only (comments, ...)**
- The *domain dictionary* defined above for this domain is also available
- For each call after the first one, the response dictionary is available. It is prefixed with the trigger index and the type. If the `info.comment` is available in the response dictionary, it will be available in all subsequent calls in the `set.1.info.comment` key.

REST Configuration

- **HTTP Method and URL***: *(select & string input)*
  Choose the HTTP method and the destination URL for your notification. The URL is a template string and can contain keys for parametrization.
- **Proxy**: *(select)*
  Define a proxy for this REST API call.
- **Timeout*** *(finite duration)*:
  Connection timeout when executing the REST API call. Must be a valid finite duration.
- **Accepted response HTTP code*** *(multiselect | input)*:
  Response codes meaning the REST call was a success. If another one is received, a failure will be logged.
- **Authentication type and credentials*** *(select & select)*:
  Choose the authentication type and the credentials to perform the authentication. Custom authentication allows the credentials values to be accessible in headers.
- **Headers** *(input string & input string)*:
  Choose the header name and value. Header values are template strings and can contain keys for parametrization.
- **Body*** *(string input)*:
  Enter the REST body. It is a template string and can contain keys for parametrization.

11. Click on the next button.

DNS Record Deletion REST Call

This REST request needs to delete the TXT DNS record if needed

Available dictionary keys:

- The *domain dictionary* defined above for this domain is also available
- For each call, the creation triggers response dictionaries , as well as the previous deletion response are available. It is prefixed with the trigger index and the type. If the `info.comment` is available in the first deletion call response dictionary, it will be available in all subsequent calls in the `unset.1.info.comment` key.

REST Configuration

- **HTTP Method and URL***: *(select & string input)*
  Choose the HTTP method and the destination URL for your notification. The URL is a template string and can contain keys for parametrization.
- **Proxy**: *(select)*
  Define a proxy for this REST API call.
- **Timeout*** *(finite duration)*:
  Connection timeout when executing the REST API call. Must be a valid finite duration.
- **Accepted response HTTP code*** *(multiselect | input)*:
  Response codes meaning the REST call was a success. If another one is received, a failure will be logged.
- **Authentication type and credentials*** *(select & select)*:
  Choose the authentication type and the credentials to perform the authentication. Custom authentication allows the credentials values to be accessible in headers.
- **Headers** *(input string & input string)*:
  Choose the header name and value. Header values are template strings and can contain keys for parametrization.
- **Body*** *(string input)*:
  Enter the REST body. It is a template string and can contain keys for parametrization.

11. Click on the save button.

Response dictionary

When receiving a response, its body is made available in the dictionary, depending on the response type.

If the response is valid JSON, it is parsed and made available. For example if the response was:

```
{
  "id": "dns_id",
  "info": {
    "type": "txt",
    "comment": "some comment"
  }
}
```

The `id`, `info.type` and `info.comment` keys are available.

If the response is not valid JSON, the whole body content is available in the `body` key.

Nameshield

Nameshield DNS configuration

- **Environment*** *(select)*:
  Select the Nameshield environment to target.
- **Nameshield credentials*** *(select)*:
  Select `API Token` credentials containing the API key to authenticate against the Nameshield API.
- **Timeout*** *(finite duration)*:
  Timeout to request the Nameshield APIs. Must be a valid Finite Duration.

10. Click on the save button.



When saving the connector, the account will be created. If the configuration is incorrect, this step could fail.

You can edit , duplicate  or delete  the ACME connector.

SwissSign Managed PKI

Prerequisites

- Your managed PKI credentials, given to you by SwissSign.

Create the PKI connector

1. Log in to Horizon Administration Interface.

2. Access PKI from the drawer or card: **PKI › PKI Connectors**.

3. Click on .

4. Select the correct PKI type.

5. Click on the next button

General tab

6. Fill in the common mandatory fields:

- **Connector Name*** *(string input)*:
  Choose a meaningful connector name allowing to identify the mapping between the PKI and the Certificate Profile. It must be unique and must not contain spaces.
- **Proxy** *(string select)*:
  If the PKI is not directly reachable from Horizon, you can set up an HTTP/HTTPS proxy to

properly forward the traffic.

- **PKI Queue** *(string select)*:
  The [admin-guide:pki_queue:::_pki_queue] used to manage the PKI Requests (enrollment, revocation).
- **Timeout** *(finite duration)*:
  Represents a predefined interval of time without a PKI response, when the time has passed "Horizon" will cease trying to establish the communication. Must be a valid finite duration.

7. Click on the next button

8. Fill all mandatory fields:

- **Endpoint Type*** *(select)*:
  Choose between the production and preproduction environment of the SwissSign API.
- **Login Credentials*** *(select)*:
  Select `Login` credentials containing the mpki identifier as login (ex: `mpkiXXXXXX.XXX`) and the API key as password (ex: `IDJjdznbGDziojDBduzh⋯`).

Then click the connect button to retrieve the available products for your MPKI. Then, fill the product field for this connector:

- **Product*** *(select)*:
  Select the product you wish to link with this connector.

Once a product is selected, you can optionally override its validity:

- **Certificate validity** *(int)*:
  Overrides the default validity of the selected product for the certificates enrolled through this connector. The unit (seconds, minutes, hours, days or months) and the accepted values are defined by the product. When not set, the product default validity is used.

9. Click on the save button.

You can edit , duplicate or delete the SwissSign Managed PKI connector.

2.4. DCV

2.4.1. DCV Introduction

Domain Control Validation (DCV) is Horizon's mechanism for automatically proving that an organization controls its DNS domains before certificates can be issued. Without a valid DCV, a Certificate Authority will refuse to issue a certificate for a domain.

Horizon automates the full DCV lifecycle: it periodically checks which domains are approaching expiration, fetches validation challenges from the CA, publishes those challenges to DNS infrastructure, and triggers the CA to confirm domain control, all without manual intervention.

The following three objects must be configured to enable DCV automation:

- [admin-guide:dcv-providers-introduction:::_dcv_provider]: Define the connection to the Certificate Authority. A Provider holds the credentials and endpoint needed to retrieve domain validation status, fetch challenge tokens, and request CA verification of published challenges.
- [admin-guide:dcv-provisioners-introduction:::_dcv_provisioner]: Define the connection to the DNS infrastructure. A Provisioner receives challenge tokens from the Provider and writes them to the appropriate DNS zone so the CA can look them up.
- DCV Policies: Tie a Provider and a Provisioner together with automation rules. A Policy defines which domains to validate, when to run, and how to handle retries and timeouts. It is the only object that is actively executed.



A single Provider or Provisioner can be reused across multiple Policies.

Limitations

- Horizon only supports **TXT** and **CNAME** DNS record types for DCV challenges. This is determined by the CA, the Provider fetches the challenge type from the CA and the Provisioner publishes the corresponding record. No other DCV challenge type should be selected or configured.

Validation Flow

When a Policy runs (on schedule or triggered manually):

1. Horizon queries the Provider for all known domains, then filters those expiring within the renewal window and matching the optional domain filter.
2. For each selected domain, Horizon asks the Provider for a validation challenge token.
3. The token is sent to the Provisioner, which publishes it to DNS.
4. Horizon asks the Provider to verify the challenge. The CA looks up the DNS record and confirms domain control.
5. If any step fails, Horizon retries from the failing step after the configured retry delay, not from the beginning.
6. When all domains reach a final state, the policy run completes and its lifecycle triggers fire.

2.4.2. DCV Providers

General Information

Description

A DCV Provider is the connection to the Certificate Authority (CA) that issues and verifies domain validation challenges. It holds the credentials and endpoint needed to retrieve domain validation status, fetch challenge tokens, and request CA verification once a challenge has been published to DNS.

A single Provider can be reused across multiple DCV Policies.

DigiCert CertCentral

Prerequisites

- You need a DigiCert CertCentral account with DCV API access enabled.
- You need to generate an API key in DigiCert CertCentral under **Account › Account Access**.
- The API key must have permissions to list domains, retrieve validation tokens, and submit validation requests.

How to configure a DigiCert DCV Provider

1. Log in to Horizon Administration Interface.

2. Access DCV Providers from the drawer or card: **DCV › Providers**.

3. Click on  .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful provider name. It must be unique for each DCV provider. Horizon uses the name to identify the provider.
- **Type*** *(select)*:
  Select `DigiCert`.

Configuration

- **Endpoint*** *(string input)*:
  Enter the DigiCert CertCentral API base URL (e.g. `https://www.digicert.com`).
- **Credentials*** *(select)*:
  Select `API Token` credentials containing the DigiCert CertCentral API key.
- **Timeout*** *(finite duration)*:
  Maximum time Horizon waits for a response from the DigiCert API.
- **Proxy** *(select)*:
  The HTTP/HTTPS proxy to use to reach the DigiCert API, if any.

5. Click on the save button.

You can edit  or delete  the DigiCert DCV Provider.



You cannot delete a DCV Provider that is referenced by an existing DCV Policy.

GlobalSign MSSL

This section details how to configure a DCV Provider for the legacy GlobalSign Certificate Center (GCC) platform, through its Managed SSL (MSSL) API.



This provider only handles domain control validation. Certificates are still enrolled through the existing GlobalSign MSSL PKI connector.

Prerequisites

- You need a GlobalSign Certificate Center account with a Managed SSL (MSSL) profile and API access enabled.
- You need the MSSL profile ID the domains to validate belong to.
- You need an API user (login and password) allowed to list domains, retrieve their validation codes, verify them and renew them.

Supported validation methods

Horizon validates domains through the DNS-based methods offered by GlobalSign MSSL. Domains vetted by email are not supported.

When a domain is already approved on GlobalSign, Horizon requests a domain renewal on GlobalSign to obtain a new challenge, then publishes and verifies it like for a new domain.

How to configure a GlobalSign MSSL DCV Provider

1. Log in to Horizon Administration Interface.

2. Access DCV Providers from the drawer or card: **DCV › Providers**.

3. Click on  .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful provider name. It must be unique for each DCV provider. Horizon uses the name to identify the provider.
- **Type*** *(select)*:
  Select `GlobalSign MSSL`.

Configuration

- **Endpoint*** *(string input)*:
  Enter the GlobalSign MSSL API endpoint URL (e.g. `https://system.globalsign.com/kb/ws/v1/ManagedSSLService`).

- **Profile*** *(string input)*:
  Enter the GlobalSign MSSL profile ID the domains belong to.
- **Credentials*** *(select)*:
  Select `Login` credentials containing the GlobalSign MSSL API username and password.
- **Default email*** *(string input)*:
  Contact email sent to GlobalSign when renewing a domain.
- **Default phone*** *(string input)*:
  Contact phone number sent to GlobalSign when renewing a domain.
- **Timeout*** *(finite duration)*:
  Maximum time Horizon waits for a response from the GlobalSign MSSL API.
- **Proxy** *(select)*:
  The HTTP/HTTPS proxy to use to reach the GlobalSign MSSL API, if any.

5. Click on the save button.

You can edit  or delete  the GlobalSign MSSL DCV Provider.



You cannot delete a DCV Provider that is referenced by an existing DCV Policy.

Sectigo

This section details how to configure a DCV Provider for Sectigo Certificate Manager (SCM).



This provider only handles domain control validation. Certificates are still enrolled through the existing Sectigo PKI connector.

Prerequisites

- You need a Sectigo Certificate Manager account with REST API access.
- You need an API client (client ID and client secret) allowed to list the account domains and to manage their domain control validations.

Supported validation methods

Horizon validates domains with the `CNAME` or `TXT` DNS methods.

- A domain that already holds a validation is re-validated with its own method, regardless of the method configured on the provider.
- The configured method only applies to domains that were never validated, or whose previous validation used a method Horizon cannot drive (e.g. email).
- When a validation order is already in progress on Sectigo, Horizon reuses its challenge instead of starting a new one. If that order uses a method Horizon cannot drive, the validation fails and the order is left untouched.



Sectigo checks pending validations every 5 minutes. Set the [admin-guide:dcv-

policy:::_dcv_policy] execution timeout accordingly.

How to configure a Sectigo DCV Provider

1. Log in to Horizon Administration Interface.

2. Access DCV Providers from the drawer or card: **DCV › Providers**.

3. Click on  .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful provider name. It must be unique for each DCV provider. Horizon uses the name to identify the provider.
- **Type*** *(select)*:
  Select `Sectigo`.

Configuration

- **Endpoint*** *(string input)*:
  Enter the Sectigo Certificate Manager API base URL (e.g. `https://admin.enterprise.sectigo.com`).
- **OAuth token endpoint** *(string input)*:
  URL used to retrieve OAuth access tokens. Defaults to the Sectigo SSO token endpoint.
- **Credentials*** *(select)*:
  Select `Login` credentials containing the API client ID as login and the client secret as password.
- **DCV method*** *(select)*:
  Select the default validation method, `CNAME` or `TXT`. See Supported validation methods.
- **Organization ID** *(string input)*:
  Restrict the domains to those of a Sectigo organization. When empty, all the domains of the account are listed.
- **Timeout*** *(finite duration)*:
  Maximum time Horizon waits for a response from the Sectigo API.
- **Proxy** *(select)*:
  The HTTP/HTTPS proxy to use to reach the Sectigo API, if any.

5. Click on the save button.

You can edit  or delete  the Sectigo DCV Provider.



When **Organization ID** is empty, every never-validated domain of the account is selected by the DCV Policy. Use the policy domain filter to restrict the validated domains and the associated license consumption.



You cannot delete a DCV Provider that is referenced by an existing DCV Policy.

2.4.3. DCV Provisioners

General Information

Description

A DCV Provisioner is the connection to the DNS infrastructure where validation challenges are published. When the Provider issues a challenge token for a domain, the Provisioner writes it into the appropriate DNS zone so the CA can look it up and confirm domain control.

A single Provisioner can be reused across multiple DCV Policies.

Common Prerequisites

All provisioner types require:

- **Credentials**: Authentication to the DNS provider API (varies by type: API token, login/password, or IAM role).
- **Permissions**: The credentials must allow creating and deleting TXT records in the target DNS zones.

Advanced Concepts

Zone ID Mappings

DNS providers organize domains into zones, each identified by a unique zone ID in their API. When publishing a challenge, the Provisioner automatically deduces the correct zone from the domain name, this requires the configured credentials to have permission to list zones on the DNS provider.

Zone ID mappings override this automatic resolution for specific domains. Each mapping is a pair: a zone ID and a regex pattern matching the domain names that belong to that zone. If a domain matches a configured mapping, that zone ID is used directly and no zone listing call is made.

Configure zone ID mappings in two situations:

- Automatic resolution produces incorrect results — for example, when a domain lives in a zone that does not align with its TLD.
- The configured credentials must not have zone listing permissions — explicitly mapping zone IDs removes the need for that permission entirely.



If a domain name matches more than one zone ID mapping regex, the validation fails with an error. Ensure mapping patterns are mutually exclusive.

Delegation Zone

A delegation zone is a DNS subdomain that has been delegated to a separate, dedicated set of nameservers, typically a specialized DCV validation system isolated from the organization's main DNS zones.

When configured, instead of publishing the challenge record directly into the domain's authoritative zone, the Provisioner writes it into the delegated zone. This is transparent to the CA: it follows the delegation and finds the challenge record normally.

This offers two practical benefits:

- **Security**: the Provisioner only needs credentials for the dedicated delegated zone, not the organization's main DNS infrastructure. This limits the blast radius if credentials are compromised.
- **Isolation**: teams using shared or managed DNS services can still perform DCV without requiring API access to their primary zones.



Setting up a delegation zone requires a one-time DNS configuration outside of Horizon: create the delegated subdomain, assign it its own nameservers, and add a CNAME record in the domain's authoritative zone pointing the challenge lookup to the delegated zone.

Cloudflare

Prerequisites

- You need a Cloudflare account with DNS management access over the target zones.
- You need to create an API token in the Cloudflare dashboard with the `Zone:DNS:Edit` permission on the relevant zones.

How to configure a DCV Provisioner

1. Log in to Horizon Administration Interface.

2. Access DCV Provisioners from the drawer or card: **DCV** › **Provisioners**.

3. Click on  .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful provisioner name. It must be unique for each DCV provisioner. Horizon uses the name to identify the provisioner.
- **Type*** *(select)*:
  Select the DNS provider type. Additional configuration fields are displayed depending on the

selected type.

Connection

- **Timeout*** *(finite duration)*:
  Maximum time Horizon waits for a response from the DNS provider.
- **TTL*** *(finite duration)*:
  TTL applied to the DNS TXT record created for the validation challenge.
- **Proxy** *(select)*:
  The HTTP/HTTPS proxy to use to reach the DNS provider, if any.

Cloudflare Configuration

- **Endpoint*** *(string input)*:
  Enter the Cloudflare API base URL (e.g. `https://api.cloudflare.com/client/v4`).
- **Credentials*** *(select)*:
  Select `API Token` credentials containing the Cloudflare API token.

Zone ID Mappings

See [admin-guide:dcv-provisioners-introduction:::_dcv_provisioner] for a full explanation of zone ID mappings.

Click on  .

- **Zone ID*** *(string input)*:
  Cloudflare zone identifier.
- **Domain pattern*** *(regex)*:
  Regex matching domain names that belong to this zone.

You can delete  a mapping.

Delegation Zone

See [admin-guide:dcv-provisioners-introduction:::_dcv_provisioner] for a full explanation of delegation zones.

- **Delegation zone** *(string input)*:
  Optional DNS subdomain delegated to dedicated nameservers for DCV. When set, challenge records are published here instead of the domain's authoritative zone.

5. Click on the save button.

You can edit  or delete  the Cloudflare DCV Provisioner.



You cannot delete a DCV Provisioner that is referenced by an existing DCV Policy.

Azure DNS

Prerequisites

- You need an Azure subscription containing the DNS zones to manage.
- You need either:
  - A service principal (application) with the `DNS Zone Contributor` role on the target resource group or DNS zones, or
  - A managed identity assigned to the Horizon host with the same permissions (no credentials required in that case).
- You need to retrieve the Tenant ID, Subscription ID, and Resource Group name from the Azure portal.

How to configure a DCV Provisioner

1. Log in to Horizon Administration Interface.

2. Access DCV Provisioners from the drawer or card: **DCV › Provisioners**.

3. Click on  .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful provisioner name. It must be unique for each DCV provisioner. Horizon uses the name to identify the provisioner.
- **Type*** *(select)*:
  Select the DNS provider type. Additional configuration fields are displayed depending on the selected type.

Connection

- **Timeout*** *(finite duration)*:
  Maximum time Horizon waits for a response from the DNS provider.
- **TTL*** *(finite duration)*:
  TTL applied to the DNS TXT record created for the validation challenge.
- **Proxy** *(select)*:
  The HTTP/HTTPS proxy to use to reach the DNS provider, if any.

Azure DNS Configuration

- **Tenant ID*** *(string input)*:
  Enter the Azure Active Directory tenant ID.

- **Subscription ID*** *(string input)*:
  Enter the Azure subscription ID containing the DNS zones.
- **Resource Group Name*** *(string input)*:
  Enter the name of the Azure resource group containing the DNS zones.
- **Credentials** *(select)*:
  Select `Login` credentials containing the service principal client ID and secret. If omitted, Horizon uses the value configured via environment variable at startup.
- **Endpoint** *(string input)*:
  Azure DNS API endpoint. If omitted, Horizon uses the value configured via environment variable at startup.
- **Authority Host** *(string input)*:
  Azure authority host URL. If omitted, Horizon uses the value configured via environment variable at startup.

Zone ID Mappings

See [admin-guide:dcv-provisioners-introduction:::_dcv_provisioner] for a full explanation of zone ID mappings.

Click on  .

- **Zone ID*** *(string input)*:
  Azure DNS zone name (e.g. `example.com`).
- **Domain pattern*** *(regex)*:
  Regex matching domain names that belong to this zone.

You can delete  a mapping.

Delegation Zone

See [admin-guide:dcv-provisioners-introduction:::_dcv_provisioner] for a full explanation of delegation zones.

- **Delegation zone** *(string input)*:
  Optional DNS subdomain delegated to dedicated nameservers for DCV. When set, challenge records are published here instead of the domain's authoritative zone.

5. Click on the save button.

You can edit  or delete  the Azure DNS DCV Provisioner.



You cannot delete a DCV Provisioner that is referenced by an existing DCV Policy.

EfficientIP

Prerequisites

- You need an EfficientIP SOLIDserver instance with DNS management access.
- You need a technical account (login/password) with permissions to create and delete TXT records in the target DNS zones.
- You need to know the name of the DNS server as configured within EfficientIP.

How to configure a DCV Provisioner

1. Log in to Horizon Administration Interface.

2. Access DCV Provisioners from the drawer or card: **DCV** › **Provisioners**.

3. Click on ⊕ .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful provisioner name. It must be unique for each DCV provisioner. Horizon uses the name to identify the provisioner.
- **Type*** *(select)*:
  Select the DNS provider type. Additional configuration fields are displayed depending on the selected type.

Connection

- **Timeout*** *(finite duration)*:
  Maximum time Horizon waits for a response from the DNS provider.
- **TTL*** *(finite duration)*:
  TTL applied to the DNS TXT record created for the validation challenge.
- **Proxy** *(select)*:
  The HTTP/HTTPS proxy to use to reach the DNS provider, if any.

EfficientIP Configuration

- **Endpoint*** *(string input)*:
  Enter the EfficientIP SOLIDserver API base URL.
- **Credentials*** *(select)*:
  Select `Login` credentials containing the EfficientIP login and password.
- **DNS Name*** *(string input)*:
  Enter the name of the DNS server within EfficientIP where challenge records will be published.

- **DNS View** *(string input)*:
  Enter the DNS view to use within EfficientIP, if applicable. Leave empty to use the default view.

Zone ID Mappings

See [admin-guide:dcv-provisioners-introduction:::_dcv_provisioner] for a full explanation of zone ID mappings.

Click on  .

- **Zone ID*** *(string input)*:
  Zone identifier as recognized by EfficientIP.
- **Domain pattern*** *(regex)*:
  Regex matching domain names that belong to this zone.

You can delete  a mapping.

Delegation Zone

See [admin-guide:dcv-provisioners-introduction:::_dcv_provisioner] for a full explanation of delegation zones.

- **Delegation zone** *(string input)*:
  Optional DNS subdomain delegated to dedicated nameservers for DCV. When set, challenge records are published here instead of the domain's authoritative zone.

5. Click on the save button.

You can edit  or delete  the EfficientIP DCV Provisioner.



You cannot delete a DCV Provisioner that is referenced by an existing DCV Policy.

PowerDNS

Prerequisites

- You need a PowerDNS Authoritative Server instance with the REST API enabled.
- You need to retrieve the API key configured in the PowerDNS server (`api-key` setting).
- The API key must have write access to the target DNS zones.

How to configure a DCV Provisioner

1. Log in to Horizon Administration Interface.

2. Access DCV Provisioners from the drawer or card: **DCV** › **Provisioners**.

3. Click on  .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful provisioner name. It must be unique for each DCV provisioner. Horizon uses the name to identify the provisioner.
- **Type*** *(select)*:
  Select the DNS provider type. Additional configuration fields are displayed depending on the selected type.

Connection

- **Timeout*** *(finite duration)*:
  Maximum time Horizon waits for a response from the DNS provider.
- **TTL*** *(finite duration)*:
  TTL applied to the DNS TXT record created for the validation challenge.
- **Proxy** *(select)*:
  The HTTP/HTTPS proxy to use to reach the DNS provider, if any.

PowerDNS Configuration

- **Endpoint*** *(string input)*:
  Enter the PowerDNS API base URL (e.g. `http://pdns-host:8081/api/v1`).
- **Credentials*** *(select)*:
  Select `API Token` credentials containing the PowerDNS API key.

Zone ID Mappings

See [admin-guide:dcv-provisioners-introduction:::_dcv_provisioner] for a full explanation of zone ID mappings.

Click on  .

- **Zone ID*** *(string input)*:
  PowerDNS zone identifier (e.g. `example.com.`).
- **Domain pattern*** *(regex)*:
  Regex matching domain names that belong to this zone.

You can delete  a mapping.

Delegation Zone

See [admin-guide:dcv-provisioners-introduction:::_dcv_provisioner] for a full explanation of delegation

zones.

- **Delegation zone** *(string input)*:
  Optional DNS subdomain delegated to dedicated nameservers for DCV. When set, challenge records are published here instead of the domain's authoritative zone.

5. Click on the save button.

You can edit  or delete  the PowerDNS DCV Provisioner.



You cannot delete a DCV Provisioner that is referenced by an existing DCV Policy.

AWS Route 53

Prerequisites

- You need an AWS account with Route 53 hosted zones for the target domains.
- You need either:
  - An IAM user with the `route53:ChangeResourceRecordSets` and `route53:ListHostedZones` permissions, and an access key/secret for that user, or
  - An IAM role attached to the Horizon host with the same permissions (no credentials required in that case).

How to configure a DCV Provisioner

1. Log in to Horizon Administration Interface.

2. Access DCV Provisioners from the drawer or card: **DCV › Provisioners**.

3. Click on  .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful provisioner name. It must be unique for each DCV provisioner. Horizon uses the name to identify the provisioner.
- **Type*** *(select)*:
  Select the DNS provider type. Additional configuration fields are displayed depending on the selected type.

Connection

- **Timeout*** *(finite duration)*:
  Maximum time Horizon waits for a response from the DNS provider.

- **TTL*** *(finite duration)*:
  TTL applied to the DNS TXT record created for the validation challenge.
- **Proxy** *(select)*:
  The HTTP/HTTPS proxy to use to reach the DNS provider, if any.

Route 53 Configuration

- **Credentials** *(select)*:
  Select `Login` credentials containing the AWS access key ID and secret access key. If omitted, Horizon uses the value configured via environment variable at startup.
- **Region** *(string input)*:
  AWS region (e.g. `us-east-1`). If omitted, Horizon uses the value configured via environment variable at startup.
- **Endpoint** *(string input)*:
  Route 53 API endpoint. If omitted, Horizon uses the value configured via environment variable at startup.
- **Role ARN** *(string input)*:
  Enter the ARN of an IAM role for Horizon to assume before publishing challenge records. Used for cross-account DNS zone access.

Zone ID Mappings

See [admin-guide:dcv-provisioners-introduction:::_dcv_provisioner] for a full explanation of zone ID mappings.

Click on  .

- **Zone ID*** *(string input)*:
  Route 53 hosted zone ID (e.g. `Z1D633PJN98FT9`).
- **Domain pattern*** *(regex)*:
  Regex matching domain names that belong to this zone.

You can delete  a mapping.

Delegation Zone

See [admin-guide:dcv-provisioners-introduction:::_dcv_provisioner] for a full explanation of delegation zones.

- **Delegation zone** *(string input)*:
  Optional DNS subdomain delegated to dedicated nameservers for DCV. When set, challenge records are published here instead of the domain's authoritative zone.

5. Click on the save button.

You can edit  or delete  the AWS Route 53 DCV Provisioner.



You cannot delete a DCV Provisioner that is referenced by an existing DCV Policy.

2.4.4. DCV Policy

A DCV Policy ties a Provider and a Provisioner together and defines the automation rules for domain control validation. It answers: which domains to validate, when to run, and how to handle retries and timeouts.

A Policy is the only DCV object that is executed, it drives the full validation workflow using its Provider and Provisioner.

How to configure a DCV Policy

1. Log in to Horizon Administration Interface.

2. Access DCV Policies from the drawer or card: **DCV** › **Policies**.

3. Click on  .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful policy name. It must be unique for each DCV policy. Horizon uses the name to identify the policy.
- **Provider*** *(select)*:
  Select the DCV Provider (CA connection) to use for this policy.
- **Provisioner*** *(select)*:
  Select the DCV Provisioner (DNS connection) to use for this policy.

Schedule

- **Schedule*** *(cron expression)*:
  Define when the policy runs automatically. The policy can also be triggered manually via the API at any time.
- **Renewal window*** *(finite duration)*:
  How far in advance of domain expiration Horizon triggers revalidation. Only domains expiring within this window are selected for the current run.

Filters

- **Domain filter** *(regex)*:
  Optional regex to restrict which domains this policy manages. Only domains whose name matches this pattern are selected for validation. Leave empty to process all domains returned by the Provider.

Retry and Timeout

- **Retry delay*** *(finite duration)*:
  Time Horizon waits before retrying a failed validation step. Horizon resumes from the failing step, not from the beginning of the pipeline.
- **Execution timeout*** *(finite duration)*:
  Maximum duration of a single policy run. When exceeded, the run is terminated and any unfinished domains will be retried on the next scheduled execution.

Triggers

Triggers are lifecycle hooks fired at key points during a policy run. Each trigger field accepts one or more existing [admin-guide:notifications-mail:::_email], [admin-guide:notifications-groupware:::_groupware], or [admin-guide:notifications-rest:::_REST] notifications to call when the corresponding event occurs.

- **On DCV Policy Start** *(multiselect)*:
  Select notification(s) to call when a policy run begins.
- **On DCV Policy End** *(multiselect)*:
  Select notification(s) to call when a policy run completes (all domains reached a final state).
- **On DCV Validation Success** *(multiselect)*:
  Select notification(s) to call when a domain is validated successfully.
- **On DCV Validation Failure** *(multiselect)*:
  Select notification(s) to call when a domain reaches an unrecoverable error state.
- **On DCV Validation Retry** *(multiselect)*:
  Select notification(s) to call each time a domain validation step is retried.

5. Click on the save button.

You can edit  or delete  the DCV Policy.



A policy cannot be updated or deleted while it is actively running. Wait for the current run to complete, or cancel it first.



Only one execution per policy can be active or pending at a time. If a policy is already running or queued when a new trigger arrives (via cron or API), the new trigger is rejected.

License Limit

Horizon enforces a license-based cap on how many domains can be validated concurrently within a single policy run. Domains are dispatched in order at run start up to the available license slots. Domains exceeding the limit are immediately moved to the `Skipped` state and are not processed during that run.

A license slot is held for the full duration of a domain's validation, including retries. Slots freed by

completed domains are not reallocated mid-run. Skipped domains remain in the Provider's list and will be picked up on the next scheduled execution if they still fall within the renewal window.

2.5. Security

2.5.1. Local Accounts

This section details how to configure the EverTrust Horizon local accounts and set their password.



Local accounts are useful to create technical accounts, such as required by `horizon-cli` for some scenarios (e.g. Scan/Discovery)

How to create local accounts

1. Log in to Horizon Administration Interface.

2. Access Local accounts from the drawer or card: **Security › Access Management › Local Accounts**.

3. Click on  .

4. Fill in the mandatory fields.

- **Identifier*** *(string input)*:
  Enter a meaningful identifier for the account holder. It will be used as a login to access to the solution.
- **Name** *(string input)*:
  Enter a meaningful name for the account holder.
- **Email** *(string input)*:
  Enter the account holder email.

5. Click on the save button.

How to set a password to a local account

1. Once a local account is created. Click on  .

2. Fill in the mandatory fields.

- **Password*** *(string input)*:
  Set a password.
- **Confirm password*** *(string input)*:
  Confirm the password.

3. Click on the save button.

You can edit  or delete  a local account. You can manage  a local account password.



You can not delete yourself from local accounts.

2.5.2. Service Accounts

This section details how to configure Horizon Service Accounts authenticated via JSON Web Key Set (JWKS).

Service Accounts allow workloads (CI/CD pipelines, Kubernetes pods, third-party automation tools, ...) to authenticate to Horizon by presenting a JWT signed by an external Identity Provider (IdP). Horizon validates the JWT signature against the JWKS exposed by the IdP and applies the configured claim validation rules. No long-lived credential is stored on the calling workload, and tokens can be rotated by the IdP without any change on the Horizon side.



Service Accounts complement the existing username/password and X.509 client certificate authentication methods. They are best suited for unattended workloads that already obtain short-lived JWTs from a trusted IdP (GitHub Actions, GitLab CI, Kubernetes projected tokens, ...).

How to create a Service Account

1. Log in to Horizon Administration Interface.

2. Access Service Accounts from the drawer or card: **Security** › **Access Management** › **Service Accounts**.

3. Click on  .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful name. This name is used by clients to identify which Service Account they want to impersonate.
- **Identifier mapping** *(string input)*:
  Template string used to derive the identifier of the principal authenticated through this Service Account. Claim names must be declared between `{{` and `}}` characters and follow the same flat-map syntax as validation rules.

  The resulting identifier is always prefixed with `<service-account-name>-<jwt-sha256-hash(16 chars)>` so that token identification can occur on each service account. The evaluated template is appended to that prefix, separated by a `-`. If no mapping is configured, the identifier stays equal to the prefix.

  For example, with a Service Account named `ci-runner` and a `sub` claim of `repo:my-org/my-repo`, setting the identifier mapping to `{{sub}}` produces the principal identifier `ci-runner-52ce1cf4649e807e-repo:my-org/my-repo`.

Authorization

Authorizations that will be given to this account

- **Roles** *(multiselect)*:
  Roles granted to the principal authenticated through this Service Account.
- **Permissions** *(multiselect)*:
  Permissions granted to the principal authenticated through this Service Account.

Trust configuration

The trust configuration defines how Horizon retrieves the public keys used to validate the JWT signature. A Service Account can declare several trust entries; a token is accepted if any of them validates its signature.

- **Type*** *(select)*:
  - **Static JWKS**: the JWKS is provided inline as a JSON document. The JWKS is then used as-is to validate signatures.
  - **Dynamic JWKS**: the JWKS is fetched from a remote URL (typically the IdP's `/.well-known/openid-configuration` or `jwks_uri` endpoint). Horizon refreshes it periodically and on first encounter of an unknown `kid`.
- **JWKS*** *(string input)*:
  For a **Static JWKS**, paste the JSON content. For a **Dynamic JWKS**, enter the URL exposing the JWKS.
- **Proxy** *(string select)*:
  HTTP/HTTPS proxy used to reach the JWKS URL, if any. Only applicable for **Dynamic JWKS**.

Validation rules

Validation rules ensure that an otherwise valid JWT is accepted only if its claims match the expected values. Each rule is evaluated using the standard computation rule syntax against the JWT claims.

JWT claims are exposed as a flat map and can be referenced as `{{<claim>}}` template entries:

- Scalar string claims are exposed by their name, e.g. `{{sub}}`, `{{iss}}`, `{{aud}}`.
- Array-valued claims are expanded into 1-indexed entries, e.g. `{{groups.1}}`, `{{groups.2}}`.
- Nested objects use escaped dotted paths, e.g. `{{"kubernetes.io".namespace}}`.

▼ *Example: validating a Kubernetes projected token*

A Kubernetes-issued JWT contains claims similar to:

```
{
  "aud": ["https://kubernetes.default.svc.cluster.local"],
  "iss": "https://kubernetes.default.svc.cluster.local",
  "sub": "system:serviceaccount:default:my-workload",
  "kubernetes.io": {
```

```
    "namespace": "default",
    "serviceaccount": { "name": "my-workload" }
  }
}
```

Suitable validation rules would be:

- `{{iss}} contains "kubernetes.default.svc.cluster.local"`
- `{{aud}} contains "kubernetes.default.svc.cluster.local"`
- `{{"kubernetes.io".namespace}} equals "default"`
- `{{"kubernetes.io".serviceaccount.name}} equals "my-workload"`

JWT time validation

The following optional settings constrain the time-based claims (`iat`, `nbf`, `exp`) carried by the JWT. They complement signature verification and validation rules.

- **Allowed clock skew** *(finite duration)*:
  Allowed clock skew when validating JWT time-based claims.
- **iat future restriction** *(finite duration)*:
  Maximum duration in the future the JWT `iat` claim is allowed to be. Must be set together with **iat past restriction**.
- **iat past restriction** *(finite duration)*:
  Maximum duration in the past the JWT `iat` claim is allowed to be. Must be set together with **iat future restriction**.

5. Click on the save button.

You can update  or delete  a Service Account.

How to authenticate using a Service Account

Clients present the Service Account name and JWT in the following HTTP headers:

- `X-API-SVA`: Service Account name
- `Authorization`: JWT, using the standard `Bearer` scheme (e.g. `Authorization: Bearer <JWT>`) or in `X-API-TOKEN` header (e.g. `X-API-TOKEN: <JWT>`). The JWT is validated identically whichever header carries it.



Service Account authentication never establishes a session. Each request must carry a valid JWT.

2.5.3. Authorization

This section details how to configure the permissions granted to an account, either directly or through a configured role.

Prerequisites

According to the context, you might need to set up:

- [admin-guide:security-roles:::_roles]
- Local accounts

How to add an authorization manually or from a certificate

1. Log in to Horizon Administration Interface.

2. Access Authorizations from the drawer or card: **Security › Access Management › Authorizations**.

3. Click on  .

4. Click on Add Authorization Manually

5. Fill the mandatory fields.

- Either:
  - Fill in an **Identifier*** *(string input or import)*:
    Enter a meaningful identifier. It can be either a local account identifier or an OpenID Connect identifier (usually email address).
  - Import a certificate by clicking on certificate button  .
- **Contact email** *(string input)*:
  Enter the contact email for the account.

6. Click on add button.

How to add an authorization from a search

1. Log in to Horizon Administration Interface.

2. Access Authorizations from the drawer or card: **Security › Access Management › Authorizations**.

3. Click on  .

4. Click on Search and Add Authorization

5. Fill one of the fields.

- **Identifier*** *(string input)*:
  Enter the identifier of the account to look for.
- **Email*** *(string input)*:
  Enter the email of the account to look for.

6. Click on search button.

7. Choose the identifier you want to add.

8. Click on add button.

You can update  or delete  Authorization.

How to grant a permission

1. Click on .

Role

2. Select a role previously created (if needed).

Team

3. Select a team previously created (if needed).

Configuration

You can build here a configuration permission. The permission follows the pattern: Section / Module / Right.

4. Click on add button.

5. Select a section, then a module, then a submodule if there is, and a right.

6. Click on add button (Don't forget to save).

7. Click on the save button if you are done.

Lifecycle

You can build here a lifecycle permission. The permission follows the pattern: Module / Profile / Right. You can further restrict the permission by adding a filter from the "Horizon Permission Query Language".

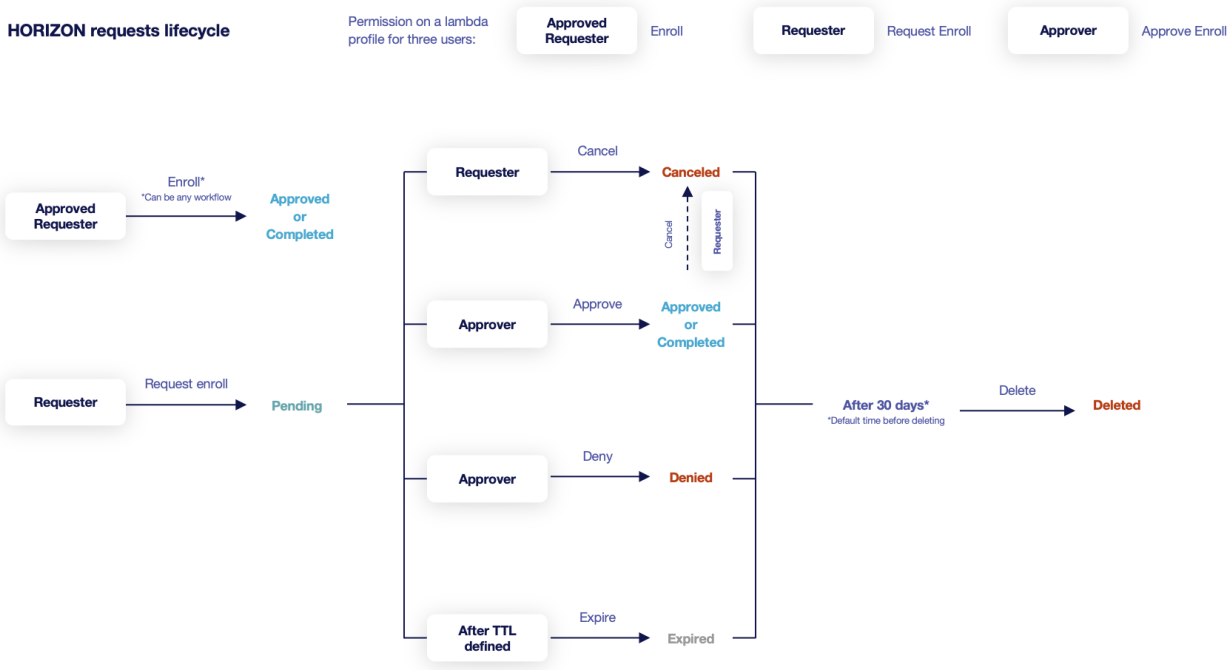
4. Click on add button.

5. Select a module, then a profile, and a right.

6. Click on add button. (don't forget to save).

7. Click on the save button if you are done.

Horizon requests lifecycle:



HPQL

The *Horizon Permission Query Language* allows you to restrict lifecycle permissions on labels and team.

The following keywords are available:

Name	Value
x equals y	true if x 's value equals y
x contains y	true if x 's value contains y
x in y	true if x 's value is contained in y (array)
x matches y	true if x 's value matches y (regex)
x within y	true if x 's value matches a value in y (regex array)

These can be combined with the following keywords:

Name	Value
x and y	true if x and y are true
x or y	true if x or y are true
x not `expression` y	true if x `expression` y is false

Examples

To filter on the `myLabel` label:

```
label.myLabel equals "labelValue"
    => myLabel: label = false
    => myLabel: labelValue =  true
label.myLabel contains "label"
    => myLabel: label = true
    => myLabel: labelValue =  true
label.myLabel within [ "\d+", "other\d?Value" ]
    => myLabel: 12345 = true
    => myLabel: otherValue =  true
```

To filter on the team:

```
team matches "team[A-Z]"
    => team: teamA = true
    => team: bestTeam =  false
team in ["teamA", "teamB"]
    => team: teamA = true
    => team: bestTeam =  false
```

Discovery

You can build here a discovery permission. The permission follows the pattern: Module / Discovery campaign name / Right.

4. Click on add button.

5. Select a module, then a campaign, and a right.

6. Click on add button. (don't forget to save)

7. Click on the save button if you are done.

2.5.4. Roles

This section details how to configure the roles. Roles are groups of permissions that can be configured for authorizations.

How to create a role

1. Log in to Horizon Administration Interface.

2. Access Roles from the drawer or card: **Security** › **Access Management** › **Roles**.

3. Click on  .

4. Fill in at least the mandatory fields.

- **Name*** *(string input)*:

Enter a meaningful name.

- **Description** *(string input)*:
  Enter a description.

6. Configuration permissions

7. Lifecycle permissions

8. Discovery permissions

9. Click on the save button.

You can get the list of members .

You can update  or delete  the Role.

2.5.5. Teams

This section details how to configure teams. Teams are groups of horizon objects owner (certificates, requests) and does not define permissions.

How to create a team

1. Log in to Horizon Administration Interface.

2. Access Teams from the drawer or card: **Security › Access Management › Teams**.

3. Click on .

4. Fill at least the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful name.
- **Description** *(string input)*:
  Enter a description.
- **Contact email** *(string input)*:
  Enter a valid email.
- **Managers** *(string multiple)*:
  Enter valid identifiers of principals. These will be team managers, and will be able to manage team members in the registration authority.
- **Messaging tool** *(select)*:
  Select one of Webhook Messaging tools supported
- **URL** *(string input)*:
  Enter the webhook messaging URL for the team(used by Groupware notifications)

5. Click on the save button.

You can get the list of members .

You can update  or delete  the Team.

2.5.6. Identity Providers Configuration

This section details how to configure Identity Providers. Identity Providers are going to be used by Horizon to verify the identity of an end-user based on the authentication performed by an external authorization server.

How to configure an Identity Provider

1. Log in to Horizon Administration Interface.

2. Access Identity Providers from the drawer or card: **Security › Access Management › Identity Providers**.

3. Click on .

General tab

4. Select an identity provider type. Currently only OpenID is supported

OpenID connect

5. Fill in all mandatory fields:

- **Name*** *(string input)*:
  Enter a meaningful identity provider name.
- **Provider metadata URL*** *(string input)*:
  Enter the OpenID Connect provider metadata URL.
- **OIDC Client Credentials*** *(string select)*:
  Select `Login` credentials with the `OpenID Connect` target, containing the client ID as login and the client secret as password. The client ID and client secret are generated on the OpenID Connect IDP when setting up a new application (Horizon). Credentials can also be created on the go from this field.
- **Scope*** *(string input)*:
  Scope used by Horizon during authentication on the identity provider to authorize access to user's details.
- **Proxy** *(string select)*:
  Proxy used to access Provider metadata URL, if any.
- **Timeout*** *(finite duration)*:
  Timeout used for authentication on the identity provider. Must be a valid finite duration. By default 10 seconds.
- **Identifier Claim*** *(string input)*:

Dynamic expression defining how to construct the identifier from the OpenID Connect claims. Claim names must be declared between `{{` and `}}` characters. For example, if the user identifier is contained in the `login` claim, then the configured value should be `{{login}}`.

- **Email Claim*** *(string input)*:
  Dynamic expression defining how to construct the user email from the OpenID Connect claims. Claim names must be declared between `{{` and `}}` characters. For example, if the user email is contained in the 'email' claim, then the configured value should be `{{email}}`. If the email is not available directly from the claims but can be computed from the 'login' claim by appending a domain, the configured value should be `{{login}}@evertrust.fr`.
- **Name Claim*** *(string input)*:
  Dynamic expression defining how to construct the username from the OpenID Connect claims. Claim names must be declared between `{{` and `}}` characters. For example, if the user name must be constructed as `family name, given name` and family name is available in the `family_name` claim, given name is available in the `given_name` claim, then the configured value should be `{{family_name}}, {{given_name}}`.
- **ID Token Signature Algorithm** *(string select)*:
  Algorithm used by the OpenID Connect provider to sign ID tokens, as defined when registering Horizon on the provider. If not set, Horizon accepts the algorithms advertised in the `id_token_signing_alg_values_supported` provider metadata.
- **User Info Signature Algorithm** *(string select)*:
  Algorithm used by the OpenID Connect provider to sign User Info responses returned as JWT. If not set, Horizon accepts the algorithms advertised in the `userinfo_signing_alg_values_supported` provider metadata.
- **Trust system CAs** *(boolean)*:
  If enabled, the system trust store is used, in addition to the [admin-guide:ca:::certificate_authorities] trusted for server authentication in Horizon, to verify the server certificate of the provider metadata endpoint. Otherwise, only these Certification Authorities are trusted. Enabled by default.
- **Authentication parameters** *(list of parameter → value)*:
  Additional parameters added to the authentication request sent to the OpenID Connect provider. For example, add the `acr_values` parameter with a value supported by the provider to request a higher authentication level.
- **Enable** *(boolean)*:
  Enable/Disable the identity provider.
- **Enable on UI** *(boolean)*:
  Enable/Disable the identity provider on user interface.

Claims mapping tab

This tab allows you to automatically assign roles and teams to users based on claims returned by the OIDC provider (e.g. group memberships). When a user authenticates, Horizon extracts values from the OIDC claims and maps them to pre-existing roles and teams.



When claims mapping is configured, the roles and/or teams selected by the **Synchronization mode** are **replaced** on each login to match the current claim

values. Manual assignments of these fields will be overwritten.

- **Synchronization mode** *(select)*:
  Select which user fields are overridden by the claims mapping on each login:
  - `Roles and teams` (default): both roles and teams are synchronized.
  - `Roles`: only roles are synchronized. Teams assigned in Horizon are kept.
  - `Teams`: only teams are synchronized. Roles assigned in Horizon are kept.
- **Claim extraction** *(Computation rule input)*:
  A computation rule that defines how to extract values from the OIDC claims. For array claims, each element is indexed (e.g. a `groups` claim containing `["admins", "developers"]` is exposed as `groups.1 = admins` and `groups.2 = developers`). Use the `[[admin-guide:security-providers:::groups]]` computation rule syntax to extract all values from such an array.
- **Claim mappings** *(list of claim value → roles and teams)*:
  A list of mappings that associate a specific claim value to one or more roles and/or teams. Each entry must reference at least one role or team. Only claim values that match an entry will result in role/team assignments.

▼ *Example: mapping OIDC groups to Horizon roles and teams*

Suppose your OIDC provider returns a `groups` claim containing the user's group memberships. You want to map these groups to Horizon roles and teams as follows:

- Users in the `admins` group get the `AdminRole` role and the `AdminTeam` team
- Users in the `developers` group get the `DevTeam` team

Configure the claims mapping as:

- **Claim extraction**: `[[admin-guide:security-providers:::groups]]`
- **Claim mappings**:
  - `admins` → Roles: `AdminRole`, Teams: `AdminTeam`
  - `developers` → Teams: `DevTeam`

With this configuration, a user who belongs to both `admins` and `developers` groups will be assigned the `AdminRole` role and both the `AdminTeam` and `DevTeam` teams upon login.

Languages tab

You can add more languages by clicking .

- **Language*** *(select)*:
  Select a language. Supported languages are:
  - **en**: English
  - **fr**: French
- **Display Name** *(string input)*:
  Enter a display name. This will be the localized name of the provider on the login page.

- **Description** *(string input)*:
  Enter a description. This will be displayed in a tooltip when the provider is chosen on the login page.

You can delete the localization.

6. Click on the save button.

You can update or delete the Identity Provider.



You won't be able to delete an Identity Provider if it is referenced in any other configuration element.

2.5.7. Credentials

This section details how to configure credentials. Credentials are where credentials for all integrations are regrouped.

How to create credentials

1. Log in to Horizon Administration Interface.

2. Access Credentials from the drawer or card: **Security › Credentials**.

3. Click on .

4. Fill at least the mandatory fields.

- **Type*** *(select)*:
  Select the credentials type: `Certificate` for certificate based authentication, `Login` for login with password credentials or `API Token` for a single value secret (JSON or other).
- **Name*** *(string input)*:
  Enter a meaningful name.
- **Description** *(string input)*:
  Enter a description.
- **Target(s)*** *(multiselect)*:
  Select the Horizon features allowed to use these credentials. Credentials can only be referenced by a configuration matching one of their targets.
- **Expiration date** *(date input)*:
  Enter an expiration date. This will be taken from the certificate for `Certificate` credentials.
- **Expiration notifications** *(select)*:
  Select [admin-guide:notifications-mail:::_email], [admin-guide:notifications-groupware:::_groupware] or [admin-guide:notifications-rest:::_REST] notifications on event `Credentials expiration` that will run on expiration. Notifications configured here will be sent by the internal monitoring action.

- Certificate:
  - **PKCS#12*** *(file select)*:
    Select your PKCS#12 file containing the authentication certificate and its key.
  - **PKCS#12 Password*** *(string input)*:
    Enter the password of the PKCS#12.
- Credentials:
  - **Login*** *(string input)*:
    Enter the account login.
  - **Password*** *(string input)*:
    Enter the account password.
- JSON Token:
  - **JSON Token*** *(string input)*:
    Enter the token.

5. Click on the save button.

You can update  or delete  the Credentials.

2.5.8. Password Policies

This section details how to configure password policies that will be used by Horizon.

How to configure a Password Policy

1. Log in to Horizon Administration Interface.

2. Access Password Policies from the drawer or card: **Security › Password Policies**.

3. Click on  .

4. Fill in the mandatory fields.

- **Name***:
  Enter a meaningful password policy name;
- **Password range length*** *(int)*:
  Password length (0 is unlimited);
- **Minimum of lowercase** *(int)*:
  Minimum of lowercase characters in the password;
- **Minimum of uppercase** *(int)*:
  Minimum of uppercase characters in the password;
- **Minimum of digit** *(int)*:
  Minimum of digit in the password;
- **Minimum of special character** *(int)*:

Minimum of special characters in the password;

- **Special characters accepted** *(string input)*:
  Whitelist of special characters accepted in the password.

5. Click on the save button.

You can update or delete the Password Policy.



You won't be able to delete a Password Policy if it is referenced in any other configuration element.

2.5.9. Tenants

This section details how to configure tenants. Tenants are only available in Multi-Tenant mode, on the Root Tenant.

How to create a tenant

1. Log in to Horizon Administration Interface.

2. Access Tenants from the drawer or card: **Security** › **Tenants**.

3. Click on .

4. Fill in at least the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful name. This will be the tenant id.
- **Description** *(string input)*:
  Enter a description.
- **License limit*** *(integer)*:
  The custom license limit for this tenant. The instance license must have enough holders to accommodate this limit.
- **License expiration** *(date)*:
  The custom license expiration for this tenant. The custom date must be before the instance's license expiration date.

5. Click on the save button.

Once a tenant is created, its license information usage can be audited , and it can be updated .

The administrator account for this tenant can also be reset , to restore access if the tenant has lost all access to its instance.

Tenant deletion

Tenant deletion occurs in two steps:

- On first deletion , the tenant is completely removed from the application logic. However, its data remains saved for a configured grace period. During this security period, the tenant deletion can be rolled back at any time using the restore function , restoring the tenant into the application logic.
- After the security period has passed, the tenant can be deleted again .



This second deletion is permanent, and all the tenant data is lost

2.5.10. SCIM

SCIM Introduction

This section refers to SCIM 2.0 integration with Horizon, used to provision users and groups in Horizon.

Description

SCIM (System for Cross-domain Identity Management) is an open standard protocol for automating the exchange of groups and users identity information between identity domains and Horizon, users and groups are synchronized between the two systems with a rich but simple set of operations:

- GET
- POST
- PUT
- PATCH
- DELETE

The SCIM protocol is detailed in the following RFCs:

- RFC7642 (System for Cross-domain Identity Management: Definitions, Overview, Concepts, and Requirements)
- RFC7643 (System for Cross-domain Identity Management: Core Schema)
- RFC7644 (System for Cross-domain Identity Management: Protocol)



Horizon does not support the full RFC, Horizon only supports the minimum of the RFC and ensures compatibility with Azure Ad and Okta.

Prerequisites

According to the context, you need:

- An application that is compatible with **SCIM 2.0**.
- Have users or groups configured in your identity manager for provisioning

Authentication with Horizon

- Have a bearer token or basic Auth

To build the bearer token you must encode in base 64 → Login:Password

Endpoint

SCIM 2.0 Base Url corresponds to: `https://<horizonUrl>/security/scim/<scimProfileName>/`

Limitations

Endpoints

List of endpoints supported:

- Users
- Groups
- ServiceProviderConfig
- ResourceTypes

Filters

List of operators supported for filtering:

- eq
- and
- ( )
- [ ]

List of attributes supported for filtering:

- `userName`
- `displayName`

Password

Horizon does not manage the password assignment.

Email

Horizon only have one email for SCIM user, it is the mail type in SCIM Profile.

SCIM User

The id of a SCIM User corresponds to the identifier of a Principal Info.

SCIM Group

Horizon does not support the creation and deletion of SCIM groups.

Supported Attributes

The list of objects and their representations :

SCIM User

- `schemas`
- `userName`
- `id`
- `emails`
- `meta`
- `active`

SCIM Group

- `schemas`
- `id`
- `displayName`
- `members`

Synchronization in Horizon

To synchronize between the SCIM groups and the roles and teams there is an object called a SCIM Profile. This object serves as an intermediary between SCIM and Horizon.

SCIM Profiles

This section details how to configure the SCIM profiles, it allows you to manage SCIM identity in Horizon.

How to create a SCIM Profile

1. Log in to Horizon Administration Interface.

2. Access SCIM Profiles from the drawer or card: **Security › SCIM Profiles**.

3. Click on  .

4. Fill at least the mandatory field.

- **Name*** *(string input)*:
  Enter a meaningful name.

5. Click on the save button.

You can update  or delete  the SCIM Profile.

How to create a SCIM specific parameters

1. Log in to Horizon Administration Interface.

2. Access SCIM Profiles from the drawer or card: **Security › SCIM Profiles**.

3. Click on  .

4. Fill at least the mandatory field.

- **Name*** *(string input)*:
  Enter a meaningful name.

5. Fill the at least the optional field.

- **Mail type** *(string input)*:
  Enter a meaningful mail type. The mail type corresponds to the mail coming from a SCIM provider that must be synchronised in horizon. By default, the mail type is "work".

6. Click on  **Add a mapping**.

The mapping corresponds to the fields allowing synchronization between Horizon and SCIM provider.

7. Fill the SCIM group name, it is referred to the SCIM group coming from the SCIM provider that must be synchronised in horizon.

8. You must choose either a role or a team for the SCIM group, but **you cannot select both**.

You can add more mappings by clicking  .

10. Click on the save button.

You can update  or delete  the SCIM Profile.



You won't be able to choose a role or team if it is referenced in any other Horizon user.

2.6. Notifications

2.6.1. Email

This section details how to configure the email notifications.

How to create an email notification

1. Log in to Horizon Administration Interface.

2. Access emails from the drawer or card: **Notifications** › **Emails**.

3. Click on  .

4. Fill in all mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful email notification name.
- **Event type*** *(select)*:
  Select the event type to notify (certificate or request).
- **Event*** *(select)*:
  Select the event to notify.
- **Retries in case of error** *(int)*:
  Select the number of times Horizon should retry to send the notification in case of error. The default value is set to 10.
- **From***: *(string input)*
  Enter the email address that will appear in the email "From" field.
- **To***: *(select multiple & input multiple)*
  Select one or several recipients. You may also enter an email address.
- **Subject*** *(string input)*:
  Enter the email subject. You may use dynamic attributes, that will be automatically replaced by the appropriate values upon email generation.
- **Body*** *(string input)*:
  Enter the email body. You may use dynamic attributes, that will be automatically replaced by the appropriate values upon email generation.
- **Is HTML** *(boolean)*:
  Sets whether the email body contains HTML code (true) or plain text (false). The default value is set to false.



You can click on the "+" next to "How to use dynamic attributes" in order to get a

range of possibilities from which one or more may be chosen.

*In case you selected a **Request** type event on any **Approval** event, or a **Certificate** event:*

- **Attachments** *(list)*:
  Sets whether to attach the certificate to the email notification and which format to use for the attached certificate (if any).
  - Attach certificate (PEM) attaches the certificate under PEM format
  - Attach bundle (PEM) attaches the certificate as well as the entire trust chain used to sign it in PEM format
  - Attach certificate (PKCS#7) attaches the certificate under PKCS#7 format
  - Attach bundle (PKCS#7) attaches the certificate as well as the entire trust chain used to sign it in PKCS#7 format
  - Attach certificate (DER) attaches the certificate under DER format

*In case you selected **Certificate Expiration**:*

- **Duration before certificate expiration causing the notification*** *(finite duration)*:
  Sets how long before certificate expiration the email notification should be sent. The default value is set to 5 days.
- **Run on renewed** *(boolean)*:
  Sets whether the expiration notification should be sent even though the certificate has been renewed. Default value is set to false (if the certificate has been renewed, the notification will not be sent).

*In case you selected as an Event **Enroll request Approval** or **Renew request Approval** or **Recover request Approval**:*

- **Attach PKCS#12** (set at false) *(boolean)*:
  Sets whether the certificate in PKCS#12 format (certificate + private key encrypted by password) should be attached to the email. The default value is set to false.
- **Send email if** *(select unique)*:
  Select either Always - Centralized (Horizon generates the private key) - Decentralized (a CSR is provided to Horizon). The default value is set to Always.

*In case you selected as an Event **Enroll request Pending** or **Renew request Pending** or **Revoke request Pending** or **Recover request Pending** or **Update request Pending** or **Migrate Request Pending**:*

- **Duration after request submission causing the notification*** *(finite duration)*:
  Duration after request submission causing the notification to be sent, in case the request was not approved in the meantime. The default value is set to 5 days.

6. Click on the save button.

You can edit , duplicate  or delete  the Email Notification .

2.6.2. Groupware

This section details how to configure the groupware notifications.

The supported groupwares are:

- Slack
- Mattermost
- Microsoft Teams

Prerequisites

You will need a webhook URL from the groupware tools in order to send notification:

- Slack
- Mattermost
- Microsoft Teams

How to create a Groupware notification

1. Log in to Horizon Administration Interface.

2. Access Groupware from the drawer or card: **Notifications** › **Groupware**.

3. Click on ⊕ .

4. Fill in all mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful email notification name.
- **Event type*** *(select)*:
  Select the event type to notify (certificate or request).
- **Event*** *(select)*:
  Select the event to notify.
- **Retries in case of error** *(int)*:
  Select the number of times Horizon should retry to send the notification in case of error. The default value is set to 10.
- **Timeout*** *(finite duration)*:
  The time before Horizon stop trying to connect to Webhook or Proxy.
- **Proxy** *(string select)*:
  The HTTP/HTTPS proxy to use to reach the groupware tool, if any.
- **To*** *(select)*:
  Select one of:
  - Static

        - **Groupware*** *(select)*:
          Select the groupware on which to send the message. Supported options are:
            - Slack
            - Mattermost
            - Microsoft Teams
        - **URL*** *(select)*:
          The webhook URL allowing the publication of messages. See the prerequisites to obtain one.
    - [admin-guide:security-teams:::_teams] webhook

- **Title*** *(string input)*:
  Enter the title of the instant message. You may use dynamic attributes, that will be automatically replaced by the appropriate values upon notification generation.
- **Body*** *(string input)*:
  Enter the body of the instant message. You may use dynamic attributes, that will be automatically replaced by the appropriate values upon notification generation.



You can click on the "+" next to "How to use dynamic attributes" in order to get a range of possibilities from which one or more may be chosen.

*In case you selected as an Event **Certificate Expiration**:*

- **Duration before certificate expiration causing the notification*** *(finite duration)*:
  Sets how long before certificate expiration the groupware notification should be sent. The default value is set to 5 days.

*In case you selected as an Event **Enroll request Pending** or **Renew request Pending** or **Revoke request Pending** or **Recover request Pending** or **Update request Pending** or **Migrate request Pending**:*

- **Duration after request submission causing the notification*** *(finite duration)*:
  Duration after request submission causing the groupware notification to be sent, in case the request was not approved in the meantime. The default value is set to 5 days.

6. Click on the save button.

You can edit , duplicate  or delete  the Groupware Notification.

2.6.3. REST

This section details how to configure REST notifications.

How to create a REST notification

1. Log in to Horizon Administration Interface.

2. Access REST from the drawer or card: **Notifications › REST**.

3. Click on  .

4. Fill in all mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful REST notification name.
- **Event type*** *(select)*:
  Select the event type to notify (certificate or request).
- **Event*** *(select)*:
  Select the event to notify.
- **Retries in case of error** *(int)*:
  Select the number of times Horizon should retry to send the notification in case of error. The default value is set to 10.
- **REST requests**:
  Configure multiple REST requests to run to complete this notification.

Request configuration:

- **HTTP Method and URL***: *(select & string input)*
  Choose the HTTP method and the destination URL for your notification. The URL is a template string and can contain keys for parametrization.
- **Proxy**: *(select)*
  Define a proxy for this REST API call.
- **Timeout*** *(finite duration)*:
  Connection timeout when executing the REST API call. Must be a valid finite duration.
- **Accepted response HTTP code*** *(multiselect | input)*:
  Response codes meaning the REST call was a success. If another one is received, a failure will be logged.
- **Authentication type and credentials*** *(select & select)*:
  Choose the authentication type and the credentials to perform the authentication. Custom authentication allows the credentials values to be accessible in headers.
- **Headers** *(input string & input string)*:
  Choose the header name and value. Header values are template strings and can contain keys for parametrization.
- **Body*** *(string input)*:
  Enter the REST body. It is a template string and can contain keys for parametrization.

Response dictionary

When receiving a response to a request, its body is made available in the dictionary, depending on the response type.

If the response is valid JSON, it is parsed and made available. For example if the response to the

first request was:

```
{
  "id": "dns_id",
  "info": {
    "type": "txt",
    "comment": "some comment"
  }
}
```

The `rest.response.1.id`, `rest.response.1.info.type` and `rest.response.1.info.comment` keys are available in subsequent requests.

If the response is not valid JSON, the whole body content is available in the `body` key.



You can click on the "Dynamic attributes" drawer in order to get a range of possibilities from which one or more may be chosen.

6. Click on the save button.

You can edit , duplicate  or delete  the REST Notification.

2.7. Discovery

This section details how to configure Discovery campaigns. An EverTrust Horizon Discovery campaign will contain all certificates discovered on a specific scope.



A discovered certificate can be:

- An unknown certificate.
  > All certificate information will be stored and this certificate will appear as an ' `unmanaged` ' certificate.
- An already discovered certificate (due to another Discovery campaign).
  > Discovery campaign metadata will be added to the existing certificate.
- A managed certificate.
  > Discovery campaign metadata will be added to the existing certificate.

How to create a Discovery Campaign

1. Log in to Horizon Administration Interface.

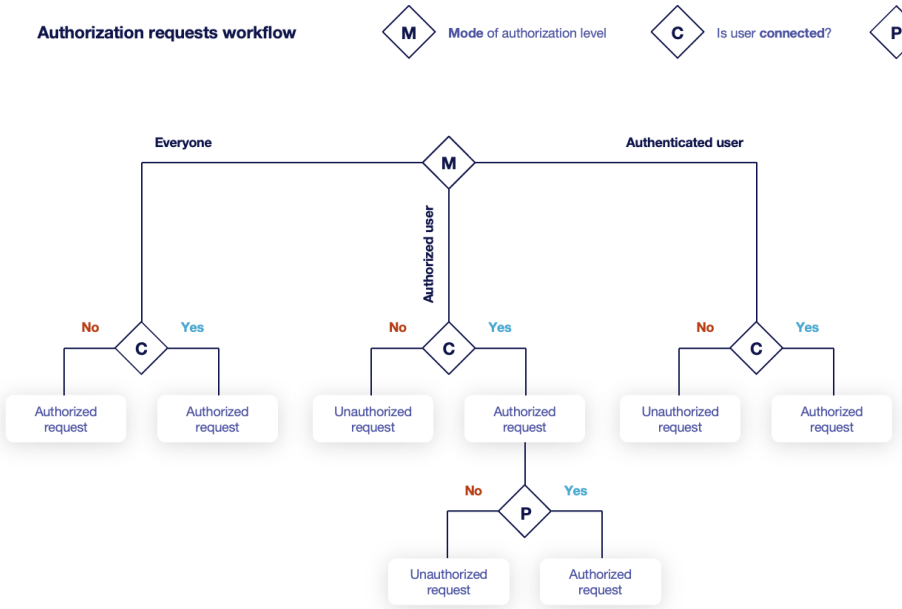
2. Access Discovery from the drawer or card: **Discovery**.

3. Click on  .

4. Fill in all mandatory fields.

General tab

- **Campaign name*** *(string input)*:
  Enter a meaningful Discovery campaign name.
- **Description** *(string input)*:
  Enter Discovery campaign description.
- **Enable** *(boolean)*:
  Enable/Disable this Discovery campaign.
- **Grading policy** *(select)*:
  The grading policy to apply to every discovered certificate on this campaign.
- **Search** *(select)*:
  Select an authorization level to search this Discovery campaign.
- **Feed** *(select)*:
  Select an authorization level to feed this Discovery campaign.



USER A : has workflow permission | USER B : has no workflow permission

Everyone		
USER A	not connected + no permissions	approve
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authenticated		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authorized		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	deny

- **Log event on success*** *(boolean)*:
  Enable/Disable discovery event on success.
- **Log event on failure*** *(boolean)*:
  Enable/Disable discovery event on failure.
- **Log event on warning*** *(boolean)*:
  Enable/Disable discovery event on warning.

Host tab

- **Hosts** *(string input or int)*:
  Specify the target to scan. Can be hostname(s), IP address(es), IP range or CIDR address(es). It is possible to add several hostnames separated by commas.

Port tab

- **Ports** *(string input or int)*:
  Enter the port(s) to scan on hosts. It is possible to add several ports separated by commas or to add a port range separated by an hyphen (ex: 1-1000 to go from 1 to 1000). If no ports are specified, ports 25, 443, 663, 8443 are scanned by the Horizon Client.



Hosts and ports should only be set if you intend to perform a network scan using `horizon-cli` in order to discover the certificates. These parameters are ignored in all other discovery modes (local scan, third party import).

6. Click on the save button.

You can edit , flush  or delete  the Discovery.

How to flush a Discovery Campaign

Flushing a Discovery campaign is the action to remove Discovery campaign reference from all discovered certificates.



There are three different cases:

- If the certificate is not managed by Horizon (only discovered by a Discovery campaign) AND only referenced by the campaign you are willing to flush → The certificate will be removed from the Horizon database.
- If the certificate is not managed by Horizon but is referenced by at least another Discovery campaign → The certificate will NOT be removed from the database and only the Discovery metadata will be removed from the certificate.
- If the certificate is managed by Horizon → Only the Discovery metadata will be removed from the certificate.

1. Log in to Horizon Administration Interface.

2. Access Discovery from the drawer or card: **Discovery**.

3. Click on .

4. Click on the Confirm button to perform the flush.

2.8. Automation

2.8.1. Automation Introduction

Certificate Lifecycle Automation allows your certificates to always be up to date with your security policy without interrupting your services unexpectedly, and without need of tiring manual operations.

The following elements are needed to allow this behavior to take place:

- **horizon-cli**: The horizon client. Installed on your server machine, it will communicate with Horizon to know when to perform the certificate change, and it will install the new certificate automatically. This feature is only available since client version `1.6.0`.

On EverTrust Horizon side:

- **Execution policies**: Define when the interruption of service to switch the certificate should take place.
- **Automation policies**: Define what profile to enroll the certificates on, and also various cryptographic parameters.
- **Profiles**: Define on which protocol and PKI your certificate is enrolled, and its contents. Automation is available on SCEP, ACME and EST profiles.

2.8.2. Execution Policy

Execution policies are a way to define time periods during which execution of automated actions are permitted. This allows you to avoid a service interruption during business hours.

Configure your execution policies

1. Log in to Horizon Administration Interface.

2. Access Execution policy from the drawer or card: **Automation** › **Execution Policy**.

3. Click on ⊕ .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful policy name. It must be unique for each execution policy. Horizon use the name to identify the policy.
- **Description** *(string input)*:
  Enter a description for your policy. It will be displayed in a tooltip on the policy list view.

Authorized periods

The Horizon Client can perform automation operations in the following time frames.

Click on ⊕ .

- **Start Date** *(date: yyyy-mm-dd)*:
  Enter the start date of this period. If no start and no end date are defined, all dates are in this period.
- **End Date** *(date: yyyy-mm-dd)*:
  Enter the end date of this period. If no start and no end date are defined, all dates are in this period.
- **Start Time** *(time: hh:mm:ss)*:
  Enter the start time of this period. If no start and no end time are defined, all times are in this period.
- **End Time** *(time: hh:mm:ss)*:
  Enter the end time of this period. If no start and no end time are defined, all times are in this period.
- **Day selector**:
  Enter the authorized days of the week.



Selecting no weekdays means no weekdays are in this period.

You can delete periods.

Forbidden periods

The Horizon Client cannot perform automation operations in the following time frames.

Click on .

- **Start Date** *(date: yyyy-mm-dd)*:
  Enter the start date of this period. If no start and no end date are defined, all dates are in this period.
- **End Date** *(date: yyyy-mm-dd)*:
  Enter the end date of this period. If no start and no end date are defined, all dates are in this period.
- **Start Time** *(time: hh:mm:ss)*:
  Enter the start time of this period. If no start and no end time are defined, all times are in this period.
- **End Time** *(time: hh:mm:ss)*:
  Enter the end time of this period. If no start and no end time are defined, all times are in this period.
- **Day selector**:
  Enter the authorized days of the week.

  Selecting no weekdays means no weekdays are in this period.

You can delete periods.

5. Click on the save button.

You can edit or delete the policy.

2.8.3. Automation Policy

Automation policies allow you to choose when and how to automate your certificate renewal, while also providing additional security policy parameters.

Configure your automation policies

1. Log in to Horizon Administration Interface.

2. Access Automation policy from the drawer or card: **Automation** › **Automation Policy**.

3. Click on .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful policy name. It must be unique for each automation policy. Horizon use the name to identify the policy.
- **Profile*** *(select)*:
  Select an existing SCEP, EST or ACME profile on which to enroll the certificates.



Cryptographic information, such as the key types for certificate enrollment are taken from the profile Crypto Policy.

- **Execution policy** *(select)*:
  Select a preexisting execution policy. If no policy is selected, renewal actions are always allowed.

Compliance

- **Authorized CAs** *(multiselect)*:
  Select CAs on which the certificate will be considered as compliant if its issuer is in the list. An empty list means all issuing CAs are authorized.
- **Authorized hash algorithms** *(multiselect)*:
  Select algorithms on which the certificate will be considered as compliant if its hash algorithm is in the list. An empty list means all hash algorithms are authorized.
- **Trust chains** *(multiselect)*:
  Select trust chains that will be installed on the machine at the same time as the certificate installation. If no chain is specified, only the one optionally needed by the server will be installed.

5. Click on the save button.

You can edit  or delete  the policy.

2.9. Monitored profiles

Introduction

Monitored certificate profiles apply to certificates whose lifecycle is not managed by Horizon. This means the certificates cannot be renewed or revoked through Horizon. However, you can still enhance them with labels, metadata, and ownership details, as well as configure notifications for supported certificate and request lifecycle events.

In essence, a monitored profile functions like a WebRA profile without a PKI connector configured.

The following lifecycle and request events are supported:

- Expiration

- Update
- Migration
- Recovery (if the private key was escrowed)

Configuring a monitored profile

1. Log in to Horizon Administration Interface.

2. Access Monitored Profiles from the drawer or card: **Monitored profiles**.

3. Click on  .

4. Fill in the mandatory fields.

Profile specific configuration

General

- **Name*** *(string input)*:
  Enter a meaningful profile name, this setting will be the profile identifier. It must be unique for each profile.
- **Enable** *(boolean)*:
  Should the profile be enabled. The default value is set to true.

Crypto Policy

- **Authorized Key Types** *(multiselect)*:
  Key Types that can be used for enrollment. An empty value means no restrictions.
- **Private key escrowing** *(boolean)*:
  Tells whether the private key should be escrowed by Horizon. The default value is set to false.
  - **Show PKCS#12 Password On Recover** *(boolean)*:
    Tells whether the PKCS#12 password should be displayed on recover. The default value is set to false.
  - **Show PKCS#12 On Recover** *(boolean)*:
    Tells whether the PKCS#12 should be displayed on recover. The default value is set to false.
  - **PKCS#12 Password Mode*** *(select)*:
    Select how to generate PKCS#12 password:
    - **manual**: prompt the user to choose its password. This is the default behavior.
    - **random**: have the password generated on Horizon side.
  - **Password policy** *(select)*:
    Select a previously created password policy. It will be enforced on PKCS#12 password for recovery and centralized enrollments.
  - **Store encryption type*** *(select)*:
    Select an encryption algorithm from the list. The PKCS#12 will use this algorithm. The

default value is set to DES_AVERAGE.

Common configuration for profiles

Languages

You can add more languages by clicking  .

- **Language*** *(select)*:
  Select a language. Supported languages are:
  - **en**: English
  - **fr**: French
- **Display Name** *(string input)*:
  Enter a display name. This will be the localized name of this profile.
- **Description** *(string input)*:
  Enter a description. This will be displayed on the list view of the profiles.

You can delete  the localization.

Grading Policies

You can select grading policies that will grade your certificate for a quick overview of its quality. For more information about the inner working of the grading policies in Horizon, please refer to the grading rules page.

Requests time to live

Configure the time your requests have before expiring.



After expiration, requests are stored for an additional 30 days. This can be changed using configuration files.

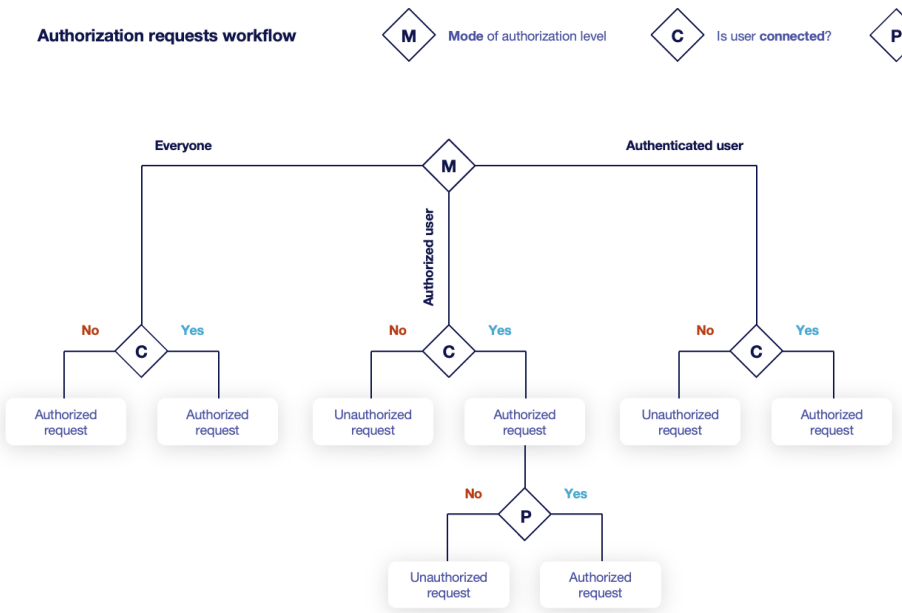
- **Enrollment request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Renewal request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Revocation request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Update request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Migration request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Recover request** *(finite duration)*:
  *Enabled on **escrow**:* Must be a valid finite duration. The default value is set to seven days.

Workflow

Workflows builder

Configure custom rights for actions on this profile.

1. Select an authorization level for each workflow.



USER A : has workflow permission | USER B : has no workflow permission

Everyone		
USER A	not connected + no permissions	approve
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authenticated		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authorized		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	deny

- **Everyone**:
  No authentication is required.
- **Authenticated**:
  User has to be authenticated.
- **Authorized**:
  User has to be authenticated and have an explicit authorizations.

2. Select an access level for identity providers.

You can remove the access level for an identity provider by clicking on 'x'.

Owner-related permissions

These permissions apply to the owners of a certificate (team or owner). An owner can always request the following actions, but this permission allows them to perform the action without validation.

- **Recover** *(boolean)*:
  Grant self recover permission. The default value is set to false.
- **Update** *(boolean)*:
  Grant self update permission. The default value is set to false.
- **Update (pop)** *(boolean)*:

Grant self update permission with owner being determined by Proof of Possession. This is used by horizon-cli. The default value is set to false.

Certificate Metadata

This section details how to define a custom structure for the labels, ownership policy and technical metadata, allowing certificates to hold rich information.

Labels

You can add more labels by clicking  .

- **Name** *(select)*:
  Select a preexisting label.
- **Mandatory** *(boolean)*:
  Should the label be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the label should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the label should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the label.
- **Label value restriction**
  - **Whitelist** *(string input multiple)*:
    The label value will have to be in the whitelist. Open the popup, enter the label value and press "enter" to add this value to the accepted value list. An empty whitelist means no restriction.
  - **Suggestions** *(string input multiple)*:
    Add suggestions that will be displayed to the user. The user will be able to choose one of these values or enter its own. Open the popup, enter your suggestions and press enter to add this value to the suggestions. An empty suggestions list means no restriction.
  - **Regex** *(regex)*:
    The label value will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this label to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can delete  or reorder (drag and drop)  the label template.

Ownership policy

- **Owner**

- **Mandatory** *(boolean)*:
  Specify if the certificate's owner is mandatory when submitting a request.
- **Editable by requester** *(boolean)*:
  Specify if the certificate's owner can be overridden by the requester when submitting a request.
- **Editable by approver** *(boolean)*:
  Specify if the certificate's owner can be overridden by the requester when approving a request.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of the owner to the value of the evaluated computation rule. This value will override any other value including the user input.

- **Contact email**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's contact email is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when approving a request.
  - **Default contact email** *(string input)*:
    Set a default contact email. This value must comply with the contact email restriction.
  - **Contact email restriction**
    - **Whitelist** *(string input multiple)*:
      The contact email will have to be in the whitelist. Open the popup, enter the email and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The contact email will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the contact email to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

- **Team**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's team is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:

Specify if the certificate's team can be overridden by the requester when approving a request.

- **Default team** *(string input)*:
  Set a default team. This value must comply with the team restriction.
- **Team restriction**
  - **Whitelist** *(string input multiple)*:
    The team will have to be in the whitelist. Enter the team and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
  - **Regex** *(regex)*:
    The team will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of the team to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

Metadata policy *(overridable metadata)*



These metadata are technical metadata. They are used by Horizon or Third party connectors, updating them should be done with utmost care.



Metadata edition is not allowed on enroll.



Metadata edition is not available via the User Interface. It must be changed with API, using horizon-cli.

You can allow the override of technical metadata by clicking .

- **Metadata*** *(select)*:
  Select a metadata.
- **Editable by requester** *(boolean)*:
  Tells whether the metadata is editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the metadata is editable by the approver. The default value is set to false.

You can delete  a metadata policy. This will not delete the metadata but will make it non editable.

Third-party policy

- **Synchronize data from discovery** *(boolean)*:
  Enable overriding of third-party data during discovery. When enabled, the following logic applies:
  - If a discovery attempts to set third-party data on a connector that is not referenced in the

profile (by a trigger), the data will not be added.

- If the connector is referenced in the profile, the discovery will override the existing data.

Notifications/Triggers

This section details how to configure notifications and triggers to perform actions on certificate and request lifecycle events.

Certificate lifecycle notifications

Notifications are sent when one of the following event is triggered by a certificate:

Enrollment	Revocation	Expire	Update	Migrate	Renew

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.

Request lifecycle notifications

Notifications are sent when one of the following event is triggered by an Enroll/Revocation/Update/Migrate/Renew request:

Submit	Cancel	Revoke	Approve	Pending

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.



Submit request events are not triggered when the user has the permission to perform the action directly.

5. Click on the save button.

You can edit , duplicate  or delete  the Monitored Profile.



You won't be able to delete a Monitored Profile if it is referenced somewhere else.

2.10. Protocols

2.10.1. ACME

ACME Introduction

This section details how to configure and consume the ACME protocol.

Horizon implements an ACME service respecting the RFC 8555 and more specifically the following lifecycle workflows:

- Enrollment;
- Renewal (which is equivalent to an enrollment);
- Revocation.

The ACME module in Horizon

The ACME configuration is organized around four types of objects, available from the drawer or card **Protocols** › **ACME** in the Horizon Administration Interface:

Object	Role
How to configure ACME Profile	Defines how Horizon serves the ACME protocol: the PKI connector used for issuance, the authorized validation methods and the constraints applied to the requests.
EAB Policy	Defines a named, reusable set of constraints applied to the attached EAB accounts.
EAB Account	Credentials (MAC Key ID and MAC Key) handed over to ACME clients, and required to register an ACME account when the profile's **Require External Account Binding (EAB)** option is enabled.
ACME Account	Accounts registered by ACME clients on the ACME profiles, which can be audited and managed from the Administration Interface.

The ACME protocol exchanges are carried by ACME accounts, registered by ACME clients on an ACME profile as described in RFC 8555, section 7.3. When the profile's **Require External Account Binding (EAB)** option is enabled, the registration must be bound to an EAB account, as described in RFC 8555, section 7.3.4: binding an ACME account to an EAB allows Horizon to restrict who can create ACME accounts, and to enforce additional constraints on the requests carried by the bound accounts.

Managing certificate lifecycle through the ACME protocol involves up to three components:

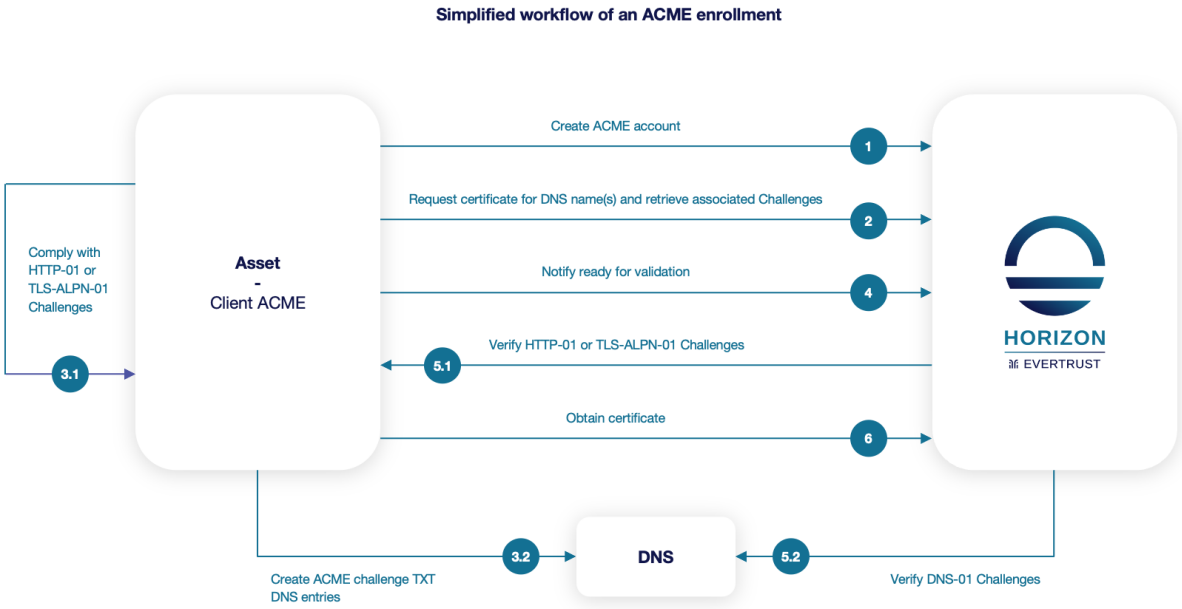
- Horizon as the ACME endpoint;
- An asset executing an ACME client or directly integrating the ACME protocol;
- When the ACME validation method is '`dns-01`', DNS server(s).



ACME validation modes will be detailed later on.

The protocol paradigm can be described as follows: '**if the asset can prove it has authority on the DNS names (called identifiers in ACME) it is requesting for, the certificate should be automatically enrolled / renewed**', which is basically equivalent to a **Domain Validation**.

The following schema is a simplified workflow of an ACME enrollment:



The protocol is based on the notion of challenge and offers three validation modes to actually verify challenges and prove that the asset owns authority on the requested DNS name(s), i.e. ACME identifiers:

- `http-01`: For each requested identifier, Horizon will validate the challenge by connecting back in **HTTP** on the configured `http-01` *validation port* (TCP/80 by default) and retrieve the response to the challenge;
- `tls-alpn-01`: For each requested identifier, Horizon will validate the challenge by connecting back in **HTTPS** on the configured `tls-alpn-01` *validation port* (TCP/443 by default) and extract the response to the challenge from an **ALPN** extension in the asset / client **HTTPS** response;
- `dns-01`: For each requested identifier, Horizon will validate the challenge through a DNS request and look for a specific TXT entry containing the response corresponding to the challenge for the considered identifier.

Therefore, validation modes have the following constraints:

- `http-01 and tls-alpn-01`:
  - Horizon must be able to access the asset on the validation port;
  - The validation port must be available and opened on the asset;
- `dns-01`: the ACME client must be configured with DNS credentials owning the permission to create TXT records on the requested domain(s).



For `http-01` and `tls_alpn-01` validation modes, it is possible to configure an HTTP proxy to proxify the ACME validation tentative(s). Using an HTTP proxy is useful when `http-01` and/or `tls-alpn-01` validation need to be performed on asset(s) hosted within a DMZ where incoming network streams must be limited. In this scenario, an HTTP proxy is configured to relay ACME validations coming from the Horizon nodes within the DMZ and a unique incoming stream needs to be open to

allow communication from Horizon node to the HTTP proxy.

The choice of the validation mode to use mainly depends on the architecture. Here are the EverTrust recommendations:

- If the requester is not the asset, prefer the `dns-01` validation mode;
- If the requester is the asset:
  - If the asset is reachable from Horizon nodes, prefer the `http-01`;
  - If the asset is not reachable from Horizon nodes, prefer the `dns-01`;
- `tls-alpn-01` is the most complicated validation mode to implement and therefore should only be used when no other validation mode is an option.

Qualified ACME clients

EverTrust qualifies the following ACME clients for any release of the Horizon product:

- Linux ACME clients:
  - acme.sh
  - certbot
  - lego
  - Horizon CLI
- Windows ACME client:
  - lego
  - Horizon CLI
- Kubernetes: cert-manager



If an ACME client is not listed above, it does not necessarily mean that the client will not work with Horizon, only that the client is not included in the list of clients tested in Horizon's continuous integration test cases.

ACME Profile

This section details how to configure an ACME Profile.

Prerequisites

PKI Connector

How to configure ACME Profile

1. Log in to Horizon Administration Interface.

2. Access ACME Profile from the drawer or card: **Protocols › ACME**.

3. Click on  .

4. Fill in the mandatory fields.

ACME Profile Specific Configuration

General

- **Name*** *(string input)*:
  Enter a meaningful profile name. It must be unique for each profile. Horizon uses the name to identify the profile. As the name will be part of a URL, it is advisable to use only lower case letters and dashes.
- **Enable*** *(boolean)*:
  Indicates whether the profile is enabled or not. The default value is set to true.
- **PKI Connector** *(string select)*:
  Select a PKI connector previously created.

Validations

- **Validation Methods** *(select)*:
  Select the authorized ACME validation method(s) on the considered profile (`HTTP-01` and/or `TLS-ALPN-01` and/or `DNS-01`).
- `HTTP_01` **validation port** *(int)*:
  HTTP port to perform the `http-01` validation (only if HTTP-01 has been selected). The default value is set to 80.
- `TLS-ALPN_01` **validation port** *(int)*:
  HTTPS port to perform the `tls-alpn-01` validation (only if TLS-ALPN-01 has been selected). The default value is set to 443.
- **Challenge verification attempts*** *(int)*:
  Specify the number of times Horizon should try to validate an ACME challenge. The default value is set to 3.
- **Challenge verification retry delay*** *(finite duration)*:
  Specify the time duration Horizon should wait between two consecutive validations for the same challenge. The default value is set to 3 seconds.
- **Proxy** *(string select)*:
  Specify an HTTP proxy to use when performing `http-01` or `tls-alpn-01` validations.
- **Timeout*** *(finite duration)*:
  Specify the time duration Horizon should wait when performing `http-01`, `tls-alpn-01` or `dns-01` validations.
- **Max ACME accounts allowed by IP** *(int)*:
  Limit the number of ACME accounts that can be created from the same IP address on this profile. When the limit is reached, further ACME account registrations from this IP address are rejected. Only new account registrations are counted: an ACME client re-registering with an existing key retrieves its account without being subject to the limit. Leave empty to disable the

limit.

- **Require External Account Binding (EAB)** *(boolean)*:
  Specify if ACME clients must provide valid EAB credentials (MAC Key ID and MAC Key) when registering an ACME account on this profile. When enabled, account registrations without a valid EAB are rejected, and the requirement is advertised to ACME clients through the `externalAccountRequired` meta field of the ACME directory. The default value is set to false.

Max Certificate per Holder Policy

- **Maximum** *(int)*:
  When specified, define the maximum number of active certificates for a given holder.
- **Behavior** *(select)*:
  What behavior to have when the maximum number is reached:
  - **revoke** the previous certificates.
  - **reject** the current request.



In order to allow renewal in `reject` behavior, one more certificate is allowed when the certificate being renewed is in its renewal period.

- **Revocation reason** *(select)*:
  When the revoke behavior is selected, the revocation reason to revoke the certificate with.

Compatibility options

These options reproduce legacy behaviors of pre-RFC 8555 ACME drafts or implementation-specific quirks. They are disabled by default and should only be enabled to support old or non-compliant ACME clients.

- **Emulate Boulder** *(boolean)*:
  Reproduces certain behaviors of the Let's Encrypt Boulder server in protocol responses: the `Boulder-Requester` header, in-progress challenges returned as `pending`, and additional fields and specific JSON formatting in the account response. The default value is set to false.
- **Legacy GET route** *(boolean)*:
  Allows retrieving an ACME resource via a GET request (without preventing POST-as-GET), inherited from ACME drafts prior to RFC 8555. The default value is set to false.



When this option is enabled, resource ownership is not verified on GET requests.

- **Accept the legacy `newKey` field** *(boolean)*:
  Accepts the legacy `newKey` field, from an early ACME draft, in key change requests sent by old ACME clients. Key change remains possible without this option using the RFC 8555 `oldKey` field. The default value is set to false.
- **Resolve account from order** *(boolean)*:
  Allows requests to be authenticated with a `kid` pointing to an order instead of an account: the account is then resolved through that order. The default value is set to false.
- **Get order from finalize** *(boolean)*:

Allows POST-as-GET requests on the finalize route of the enrollment, returning the pending order's information. The default value is set to false.

- **Allow other identifier types** *(boolean)*:
  Accepts identifiers whose type is neither `DNS` nor `IP`, and validates them as DNS names. The default value is set to false.

Requests management

- **Authorized short name** *(boolean)*:
  Specify if using short name is authorized when requesting certificate. If set to yes, one verifiable FQDN must be requested for each specified short name. The default value is set to false.
- **Authorized empty contact** *(boolean)*:
  Specify if an ACME account can be registered without specifying a contact email address. The default value is set to true.
- **Default contacts email** *(string input multiple)*:
  Specify a list of default contact email addresses when registering an ACME account with no specified contact email address.
- **Max DNS name** *(int)*:
  If specified, enforce the maximum number of requested DNS name(s).
- **IP identifier constraint** *(regex)*:
  Optional regular expression that every IP identifier of an order must match. Orders carrying an IP identifier that does not match the constraint are rejected. Leave empty to accept any IP. IPv4 and IPv6 presets are available to ease the configuration.

Certificate delivery

- **Exclude root CA from certificate chain** *(boolean)*:
  When enabled, the root CA certificate is omitted from the certificate chain returned to ACME clients, as described in RFC 8555, section 7.4.2: the root CA should already be present on the target system. Intermediate CA certificates are unchanged. The default value is set to false.

Meta

- **Is required terms of service** *(boolean)*:
  Specify if explicitly agreeing to the terms of service is required when registering an ACME account. The default value is set to false.
- **Terms of service** *(string input)*:
  Specify an URL identifying the current terms of service.
- **Website** *(string input)*:
  Specify an HTTP or HTTPS URL locating a website providing more information about the ACME server.
- **CAA Identities** *(string input)*:
  The hostnames that the ACME server recognizes as referring to itself for the purposes of CAA record validation as defined in RFC6844.

Crypto policy

- **Default Key Type** *(select)*:
  Key Type that will be used by horizon-cli in certificate enrollment.
- **Authorized Key Types** *(multiselect)*:
  Key Types that can be used for enrollment. An empty value means no restrictions.

Common configuration for profiles

Languages

You can add more languages by clicking  .

- **Language*** *(select)*:
  Select a language. Supported languages are:
  - **en**: English
  - **fr**: French
- **Display Name** *(string input)*:
  Enter a display name. This will be the localized name of this profile.
- **Description** *(string input)*:
  Enter a description. This will be displayed on the list view of the profiles.

You can delete  the localization.

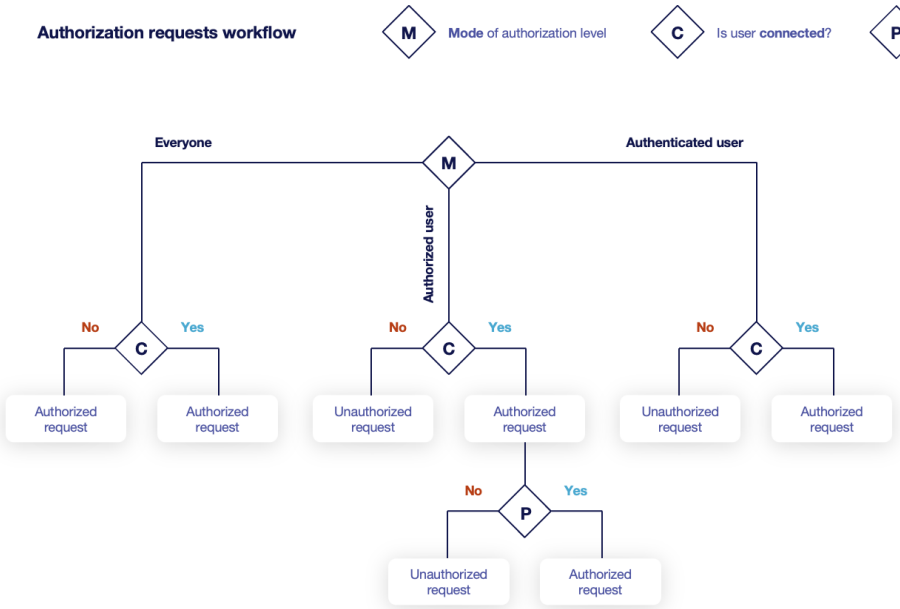
Grading Policies

You can select grading policies that will grade your certificate for a quick overview of its quality. For more information about the inner working of the grading policies in Horizon, please refer to the grading rules page.

Workflows builder

Configure custom rights for actions on this profile.

1. Select an authorization level for each workflow.



USER A : has workflow permission | USER B : has no workflow permission

Everyone		
USER A	not connected + no permissions	approve
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authenticated		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authorized		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	deny

- **Everyone**:
  No authentication is required.
- **Authenticated**:
  User has to be authenticated.
- **Authorized**:
  User has to be authenticated and have an explicit authorizations.

2. Select an access level for identity providers.

You can remove the access level for an identity provider by clicking on 'x'.

Requests time to live

Configure the time your requests have before expiring.



After expiration, requests are stored for an additional 30 days. This can be changed using configuration files.

- **Enrollment request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Renewal request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Revocation request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Update request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Migration request*** *(finite duration)*:

Must be a valid finite duration. The default value is set to seven days.

- **Recover request** *(finite duration)*:
  *Enabled on **escrow**:* Must be a valid finite duration. The default value is set to seven days.

Owner-related permissions

These permissions apply to the owners of a certificate (team or owner). An owner can always request the following actions, but this permission allows them to perform the action without validation.

- **Revoke** *(boolean)*:
  Grant self revoke permission. The default value is set to false.
- **Revoke (pop)** *(boolean)*:
  Grant self revoke permission with owner being determined by Proof of Possession. This is used by horizon-cli. The default value is set to false.
- **Update** *(boolean)*:
  Grant self update permission. The default value is set to false.
- **Update (pop)** *(boolean)*:
  Grant self update permission with owner being determined by Proof of Possession. This is used by horizon-cli. The default value is set to false.

Constraints

- **Allowed email domains** *(string input)*:
  Enter a valid regular expression that the inputted emails should match. This includes RFC822NAME and UPN SANs as well as the contact email

  This matches the domain of the email, not including anything before `@`.

- **Allowed DNS domains** *(string input)*:
  Enter a valid regular expression that the inputted domain should match.

CSR Data Mapping

1. Click on  to add a mapping.

2. Select a field and enter a value.

You can delete  the CSR Data Mapping.

Certificate Template

This section details how to define a custom structure for the fields `subject DN`, `SAN` & `extensions` of the requested certificate in order to match the configuration on the PKI side.



Defining a template will use the CSR to fill the available field. A CSR with unexpected fields will be rejected. Using a template also disables CSR Data

Mapping.

Subject DN composition

You can add more elements by clicking .

- **Element*** *(select)*:
  Select an attribute from the elements list.
- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the element.
- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the Subject DN template.



When a template is defined, at least one mandatory Common Name must be added to the DN Elements.

SAN composition

You can add more elements by clicking .

- **Element*** *(select)*:
  Select an attribute from the element list.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Minimum** *(int)*:
  The minimum number of value that this SAN must have.
- **Maximum** *(int)*:
  The maximum number of value that this SAN must have.

- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the SAN template.

Extensions

You can add more elements by clicking  .

- **Element*** *(select)*:
  Select an attribute from the elements list.
- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the element.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the Extensions template.



When adding a SAN, a DN element or an Extension and making it mandatory, make sure to either give it a default value or a computation rule or make it editable, otherwise the template will be unusable.

Certificate Metadata

This section details how to define a custom structure for the labels, ownership policy and technical metadata, allowing certificates to hold rich information.

Labels

You can add more labels by clicking  .

- **Name** *(select)*:

Select a preexisting label.

- **Mandatory** *(boolean)*:
  Should the label be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the label should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the label should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the label.
- **Label value restriction**
  - **Whitelist** *(string input multiple)*:
    The label value will have to be in the whitelist. Open the popup, enter the label value and press "enter" to add this value to the accepted value list. An empty whitelist means no restriction.
  - **Suggestions** *(string input multiple)*:
    Add suggestions that will be displayed to the user. The user will be able to choose one of these values or enter its own. Open the popup, enter your suggestions and press enter to add this value to the suggestions. An empty suggestions list means no restriction.
  - **Regex** *(regex)*:
    The label value will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this label to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can delete  or reorder (drag and drop)  the label template.

Ownership policy

- **Owner**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's owner is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when approving a request.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the owner to the value of the evaluated computation rule. This value will override any other value including the user input.
- **Contact email**

- **Mandatory** *(boolean)*:
  Specify if the certificate's contact email is mandatory when submitting a request.
- **Editable by requester** *(boolean)*:
  Specify if the certificate's contact email can be overridden by the requester when submitting a request.
- **Editable by approver** *(boolean)*:
  Specify if the certificate's contact email can be overridden by the requester when approving a request.
- **Default contact email** *(string input)*:
  Set a default contact email. This value must comply with the contact email restriction.
- **Contact email restriction**
  - **Whitelist** *(string input multiple)*:
    The contact email will have to be in the whitelist. Open the popup, enter the email and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
  - **Regex** *(regex)*:
    The contact email will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of the contact email to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

- **Team**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's team is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when approving a request.
  - **Default team** *(string input)*:
    Set a default team. This value must comply with the team restriction.
  - **Team restriction**
    - **Whitelist** *(string input multiple)*:
      The team will have to be in the whitelist. Enter the team and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The team will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:

Set the value of the team to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

Metadata policy *(overridable metadata)*



These metadata are technical metadata. They are used by Horizon or Third party connectors, updating them should be done with utmost care.



Metadata edition is not allowed on enroll.



Metadata edition is not available via the User Interface. It must be changed with API, using horizon-cli.

You can allow the override of technical metadata by clicking .

- **Metadata*** *(select)*:
  Select a metadata.
- **Editable by requester** *(boolean)*:
  Tells whether the metadata is editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the metadata is editable by the approver. The default value is set to false.

You can delete  a metadata policy. This will not delete the metadata but will make it non editable.

Third-party policy

- **Synchronize data from discovery** *(boolean)*:
  Enable overriding of third-party data during discovery. When enabled, the following logic applies:
  - If a discovery attempts to set third-party data on a connector that is not referenced in the profile (by a trigger), the data will not be added.
  - If the connector is referenced in the profile, the discovery will override the existing data.

Notifications

This section details how to configure notifications on certificate and request lifecycle events.

Certificate lifecycle notifications

Notifications are sent when one of the following event is triggered by a certificate:

Enrollment	Revocation	Expire	Update	Migrate	Renew

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to

associate it with an event.

Request lifecycle notifications

Notifications are sent when one of the following event is triggered by an Enroll/Revocation/Update/Migrate/Renew request:

Submit	Cancel	Revoke	Approve	Pending

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.



Submit request events are not triggered when the user has the permission to perform the action directly.

5. Click on the save button.

You can edit  , duplicate  or delete  the ACME Profile.



You won't be able to delete an ACME Profile if it is referenced somewhere else.

ACME EAB Policies

This section details how to configure External Account Binding policies.

Introduction

An EAB policy is a named, reusable set of constraints attached to EAB accounts. It allows users to define constraints once and apply them to every EAB account attached to the policy, instead of configuring them on each EAB account individually.

The constraints of an EAB policy always apply in combination with the constraints of the attached EAB accounts: when both levels define a constraint, requests must satisfy both of them. Constraints left empty impose no restriction at their level.

How to create an EAB policy

1. Log in to Horizon Administration Interface.

2. Access ACME from the drawer or card: **Protocols** › **ACME**, then open **EAB Policies** from the drawer or card.

3. Click on  .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful EAB policy name. It must be unique for each EAB policy. Horizon uses the name to identify the policy, and EAB accounts reference their policy by name. The name can no longer be edited after creation.

Additional Policy Constraints



The following constraints apply in addition to those of the attached EAB accounts and of the selected ACME profile. Leave a constraint empty to impose no restriction at this level.

- **Allowed Profiles** *(multiselect)*:
  Restricts which ACME profiles the associated EAB accounts may use. If no profile is selected, this policy imposes no profile restriction; each EAB account's own list still applies.
- **Order Identifier Constraint** *(regex)*:
  Every identifier in an order must match this regular expression, in addition to any expression set on the associated EAB account.
- **Email Constraint** *(regex)*:
  Every contact email address (provided at ACME account creation) must match this regular expression, in addition to any expression set on the associated EAB account.
- **Validation Methods** *(multiselect)*:
  Limits the validation methods allowed by this policy, in addition to those of the associated EAB account and the selected ACME profile. Leave empty for no additional restriction.

5. Click on the save button.

You can edit , duplicate  or delete  an EAB policy.



Any change made to a policy immediately applies to all EAB accounts attached to it.



You won't be able to delete an EAB policy if it is referenced by at least one EAB account.

ACME EAB Accounts

This section details how to configure External Account Bindings (EAB).

Introduction

An External Account Binding (EAB) is a pair of credentials made of a **MAC Key ID** (public identifier) and a **MAC Key** (shared secret), generated by Horizon and handed over to an ACME client to be used when registering an ACME account, as described in RFC 8555, section 7.3.4. Binding an ACME account to an EAB allows Horizon to:

- Control which clients can create ACME accounts, when the **Require External Account Binding** option is enabled on an ACME profile;
- Enforce additional constraints on the requests carried by the ACME accounts bound to the EAB.

Each EAB account is attached to an EAB policy, which adds a shared layer of constraints on top of the EAB account's own constraints.



An EAB account can only be used by ACME clients if its status is `Valid` and its expiration date, if any, is not passed. An expired EAB account behaves like a non-valid one: it blocks the enrollment of new ACME accounts and the use of linked ACME accounts.

Prerequisites

EAB Policy

How to create an EAB account

1. Log in to Horizon Administration Interface.

2. Access ACME from the drawer or card: **Protocols** › **ACME**, then open **EAB Accounts** from the drawer or card.

3. Click on  .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful EAB account name. It must be unique for each EAB account. The name can no longer be edited after creation.
- **Description** *(string input)*:
  Enter a description of the EAB account.
- **EAB Policy*** *(string select)*:
  Select an EAB policy previously created. Every constraint defined by the selected policy applies in addition to the EAB account's own constraints. If no EAB policy matches your needs, click on **Create an EAB policy** next to the field to create a new one on the fly: the newly created policy is automatically selected.
- **MAC Key Algorithm*** *(select)*:
  Select the algorithm used to generate the MAC Key: `HS256`, `HS384` or `HS512`. The default value is set to HS256.
- **Validity Duration** *(finite duration)*:
  Specify the validity duration of the EAB account, starting from its creation. Once the duration is elapsed, the EAB account is considered expired and can no longer be used. Leave empty to create the EAB account without expiration date. The validity duration can only be set at

creation; use the renew action to extend or remove it afterward.

Additional Account Constraints



The following constraints apply in addition to those of the attached EAB policy and of the selected ACME profile: when both levels define constraints, requests must satisfy both of them. Leave a constraint empty to impose no restriction at this level.

- **Allowed Profiles** *(multiselect)*:
  Restricts the ACME profiles allowed for this EAB account. If the linked EAB policy also defines a list, only profiles present in both lists will be accepted. Leave empty for no additional restriction; the linked EAB policy's list still applies.
- **Order Identifier Constraint** *(regex)*:
  Every identifier in an order must match this regular expression, in addition to any expression set on the EAB policy.
- **Email Constraint** *(regex)*:
  Every contact email address (provided at ACME account creation) must match this regular expression, in addition to any expression set on the EAB policy.
- **Validation Methods** *(multiselect)*:
  Limits the validation methods allowed for this EAB account, in addition to those of the EAB policy and the selected ACME profile. Leave empty for no additional restriction.

5. Click on the save button.

The MAC Key ID and the MAC Key are displayed once, right after the save, in a dedicated window: copy them and hand them over to the ACME client owner. Once the window is closed, the credentials cannot be displayed again.

From the EAB Accounts list, you can:

- edit an EAB account ,
- duplicate it ,
- delete it ,
- change its status ,
- renew its MAC Key ,
- and display the ACME accounts linked to it 



You won't be able to delete an EAB account if it is linked to at least one ACME account whose status is **Valid**, **Deactivated** or **Suspended**.

How to search EAB accounts

1. Log in to Horizon Administration Interface.

2. Access ACME from the drawer or card: **Protocols › ACME**, then open **EAB Accounts** from the drawer or card.

The EAB Accounts list can be searched using the search bar, in intermediate or expert mode.

Intermediate search

The intermediate search allows filtering the list by name and status.

Expert search

The expert mode allows building HEABQL (Horizon External Account Binding Query Language) queries, by combining elements, conditions and operators. The query structure is the following:

- `<element> <condition> <"value">` (`<operator> [<element> <condition> <"value">]`)

Table 2. Table element

Element	Description	Available conditions
`name`	EAB account name	`equals`, `not equals`, `contains`, `not contains`, `in`, `not in`
`status`	EAB account status	`equals`, `not equals`, `contains`, `not contains`, `in`, `not in`
`mackey.algorithm`	MAC key algorithm	`equals`, `not equals`, `contains`, `not contains`, `in`, `not in`
`expiration.date`	EAB account expiration date	`equals`, `before`, `after`, `not before`, `not after`, `exists`, `not exists`
`created.at`	EAB account creation date	`equals`, `before`, `after`, `not before`, `not after`
`eab.policy`	EAB policy name	`equals`, `not equals`, `contains`, `not contains`, `in`, `not in`
`validation.methods`	EAB validation methods	`exists`, `not exists`, `contains`, `not contains`, `in`, `not in`

Table 3. Table operator

Operator	Description
`or`	The EAB account matches at least one of the combined criteria
`and`	The EAB account matches all the combined criteria

MAC Key management

The EAB account credentials are materialized by:

- The **MAC Key ID**, which is the public identifier of the EAB account, provided by the ACME client when registering an ACME account;
- The **MAC Key**, which is the shared secret used by the ACME client to sign the registration request.

The MAC Key ID and the MAC Key are only displayed once, right after the EAB account creation and after each MAC Key renewal, in a dedicated window where they can be copied. Once this window is closed, the credentials can no longer be displayed from the Horizon Administration Interface: if the MAC Key is lost, renew the EAB account to generate a new one.



When duplicating an EAB account, a new MAC Key is generated: the copy does not share the credentials of the original EAB account.



The MAC Key must be kept secret: any party holding it can bind an ACME account to the EAB account and therefore benefit from its authorizations. If the MAC Key is suspected to be disclosed, renew it or compromise the EAB account.

Both values are only known by Horizon and the ACME client. At ACME account registration, the client sends the MAC Key ID and signs the registration request with the MAC Key; the MAC Key itself never appears in any ACME protocol exchange.

How to renew an EAB account

1. Click on  from the EAB account form or the EAB Accounts list.

2. Fill in the optional fields:

- **MAC Key Algorithm** *(select)*:
  Select a new algorithm to generate the MAC Key: `HS256`, `HS384` or `HS512`. Leave empty to keep generating the key with the current algorithm.
- **New validity duration** *(finite duration)*:
  Specify a new validity duration, starting from the renewal. Leave empty to renew the EAB account without any expiry.

3. Click on the confirm button.

Renewing an EAB account generates a new MAC Key while keeping the MAC Key ID unchanged. ACME accounts already bound to the EAB account are not impacted: only registrations performed with the previous key are rejected. The new MAC Key is displayed once, right after the renewal, in the same dedicated window as at creation: copy it and hand it over to the ACME clients that will enroll accounts using this EAB account. The number of key regenerations is displayed in the EAB account form.



ANSSI recommends a validity period of no more than 3 years for this kind of

secret.

EAB account statuses

The status of an EAB account can be changed by clicking on **Change status** from the EAB account form or the EAB Accounts list.

The following statuses are available:

Status	Description
Valid	Can enroll new ACME accounts, and the linked ACME accounts operate normally.
Disabled	Temporarily disables the account for any reason. Blocks the enrollment of new accounts and the use of linked accounts.
Suspended	Temporarily disables the account in case of a suspected compromise. Blocks the enrollment of new accounts and the use of linked accounts.
Deactivated	Deactivating this account deactivates all linked ACME accounts. Only compromising it will remain possible afterwards.
Compromised	Compromising an EAB forces the compromise of all linked ACME accounts and triggers the revocation of the certificates they enrolled. This status is irreversible.

Compromising an EAB account

When setting the status to Compromised, the following fields are requested:

- **Revoke certificates issued after** *(date input)*:
  Certificates issued after this date and time will be revoked. Leave empty to revoke all certificates linked to this EAB account.
- **Revocation reason** *(select)*:
  The revocation reason applied to every certificate revoked through this compromise, on this EAB account and on all the ACME accounts linked to it.

ACME Accounts

This section details how to manage the ACME accounts registered on Horizon.

Introduction

An ACME account is created by an ACME client when it registers on an ACME profile, as described in RFC 8555, section 7.3. An ACME account is identified by:

- its **Key ID**, a public identifier: the SHA-256 thumbprint of the account key, displayed in the Horizon Administration Interface;
- its **account key**, the private key of the key pair held by the client, used to sign its subsequent requests.



In the ACME protocol, the client identifies its account in the `kid` field of the JWS header of each request: this `kid` is the account URL returned by Horizon at registration. It is distinct from the **Key ID** displayed in the Horizon Administration Interface, which is the SHA-256 thumbprint of the account key.



The account key must be kept secret by the ACME client: any party holding it can act as the account, reusing its valid authorizations to enroll certificates without re-validating challenges. If the account key is suspected to be disclosed, compromise the ACME account to revoke the certificates issued to it.

ACME accounts cannot be created from the Horizon Administration Interface: this page allows administrators to audit the accounts registered on the ACME profiles and to manage their status.

An ACME account can be bound to an EAB account at registration, as described in EAB accounts. Changing the status of an EAB account can impact the ACME accounts bound to it, and compromising an EAB account also compromises the accounts bound to it.

Account details

1. Log in to Horizon Administration Interface.

2. Access ACME from the drawer or card: **Protocols** › **ACME**, then open **ACME Accounts** from the drawer or card.

Click on the Key ID or the view icon to display the account details.

The account details page displays:

- **Key ID** *(string)*: the account public identifier;
- **Account key (JWK)** *(string)*: the public JWK of the account key pair held by the client;
- **Contact** *(string)*: the contact email address(es) provided at registration;
- **Terms of service accepted** *(boolean)*: whether the client explicitly agreed to the terms of service;
- **EAB** *(string)*: the EAB account the account is bound to, if any;
- **Created at** *(date)*: the account creation date;
- **Initial IP** *(string)*: the IP address the account was registered from;
- **Status** *(chip)*: the account status, that can be changed with the **Change status** action;
- **Compromised at** / **Compromise reason** *(date / select)*: the compromise information, when relevant;
- **Orders** *(list)*: the orders placed by the account, with their identifier, the requested identifiers,

the validation type, the enrolled certificate (when available) and the order status (`Pending`, `Ready`, `Processing`, `Valid`, `Invalid`). The certificate reference navigates to the certificate page.

You can delete an ACME account  from the ACME Accounts list.



You won't be able to delete an ACME account if:

- One of its orders is not in a final status (`Valid` or `Invalid`);
- One of the certificates it enrolled is still valid.

Deleting an ACME account also deletes its orders.

ACME account statuses

The status of an ACME account can be changed by clicking on **Change status** from the account details or the ACME Accounts list. The following statuses are available:

Status	Description
Valid	Can request certificates and manage its orders and authorizations.
Deactivated	Temporarily disables this account for any reason. Blocks certificate requests and access to its orders and authorizations. Can be set back to Valid.
Suspended	Temporarily disables the account in case of a suspected compromise. Blocks certificate requests and access to its orders and authorizations. Can be set back to Valid.
Revoked	Revocation is final and will block this account. Once revoked, it can only be compromised.
Compromised	Certificates issued to this account will be revoked. This status is irreversible.
Compromise pending	Transitional status automatically set while a compromise is being processed and the certificates are being revoked.



ACME clients only see the statuses defined by RFC 8555: `Suspended` accounts are reported as `deactivated` and `Revoked` / `Compromised` accounts as `revoked`.

Compromising an ACME account

When setting the status to Compromised, the following fields are requested:

- **Revoke certificates issued after** *(date input)*:
  Certificates issued after this date and time will be revoked. Leave empty to revoke all

certificates issued to this account.

- **Revocation reason** *(select)*:
  The revocation reason applied to every certificate revoked through this compromise.



Compromising an ACME account is irreversible and triggers the revocation of the certificates issued to this account.

How to search ACME accounts

The ACME Accounts list can be searched using the search bar, in intermediate or expert mode.

Intermediate search

The intermediate search allows filtering the list by Key ID, contact or EAB account name.

Expert search

The expert mode allows building HAQL (Horizon ACME Query Language) queries, by combining elements, conditions and operators. The query structure is the following:

- `<element> <condition> <"value">` (`<operator> [<element> <condition> <"value">]`)

Table 4. Table element

Element	Description	Available conditions
`id`	Account ID	`equals`, `not equals`
`contact`	Account contact	`equals`, `not equals`, `contains`, `not contains`, `in`, `not in`, `exists`, `not exists`
`eab.name`	Name of the EAB account the ACME account is bound to	`equals`, `not equals`, `contains`, `not contains`, `in`, `not in`, `exists`, `not exists`
`status`	Account status	`equals`, `not equals`, `in`, `not in`
`created.at`	Account creation date	`equals`, `not equals`, `before`, `after`, `not before`, `not after`

Table 5. Table operator

Operator	Description
`or`	The account matches at least one of the combined criteria
`and`	The account matches all the combined criteria

ACME client usages

This section details how to use the most common Linux ACME clients.

Linux ACME clients

This section details how to use the **acme.sh**, **certbot** and **lego** ACME clients.

Overview

Certbot is able to run on any recent UNIX-like operating system equipped with Python 2.7 or 3.4+, while acme.sh can also run on any recent Linux distribution running either bash, dash or sh. lego is a self-contained binary written in Go, also available on Windows.

All three fully support the latest ACMEv2 protocol including its main latest feature: wildcard certificates (*.example.com).

All three clients support different modes for obtaining a certificate and in some cases automatically installing it.

The following tables lists the different modes for each clients:

Modes	certbot	acme.sh	lego	Notes
apache	Y	Y	N	Obtains and automatically installs a certificate using the running Apache server. (For acme.sh, this mode will only obtain a certificate without installing it)
nginx	Y	Y	N	Obtains and automatically installs a certificate using the running NGINX server. (For acme.sh, this mode will only obtain a certificate without installing it)
webroot	Y	Y	Y	Obtains a certificate by writing to the webroot directory of an already running web server
standalone	Y	Y	Y	Uses a "standalone" web server managed by Certbot, acme.sh or lego. This mode is useful on system with no web servers or if using the running web server is not desired
DNS	Y	Y	Y	This mode automates obtaining a certificate by modifying a DNS record to prove the control over a domain
`tls-alpn`	N	Y	Y	Uses a TLS server to validate the control over a domain

Requesting a certificate

certbot and acme.sh must be started using administrative privileges (`sudo`), except for acme.sh when using the webroot or DNS modes. lego only requires them to listen on a privileged port (below 1024) or to write to a protected webroot folder.

Each client requires only a few parameters to request a certificate.

acme.sh parameters:

Parameter	Description
`--issue`	Obtain or renew a certificate, but does not install it
`--accountemail [VALUE]`	The contact email address
`-w [VALUE]`	Path of the server's webroot folder
`-d [VALUE]`	The domain(s) to enroll.

certbot parameters:

Parameter	Description
`certonly`	Obtain or renew a certificate, but does not install it
`--webroot`	Place files in a server's webroot folder for authentication
`-w [VALUE]`	Path of the server's webroot folder
`-d [VALUE]`	The domain(s) to enroll.

Requesting a certificate for Apache using certbot:

```
(sudo) certbot run --apache --no-eff-email --agree-tos --server <Horizon ACME
endpoint, example: https://horizon.evertrust.fr/acme/profile1/directory> -m <contact
email address, example: kma@evertrust.fr> --domain <DNS name, example:
apache.evertrust.fr>
```

Where:

- `--apache`: Enables the Apache mode
- `--no-eff-email`: Does not share your email address with EFF
- `--agree-tos`: Explicitly agrees to the terms of service
- `--server`: Horizon ACME profile endpoint
- `-m`: Contact email address
- `--domain`: Requested DNS name (can be specified several times)

Requesting a certificate for nginx using certbot:

```
(sudo) certbot run --nginx --no-eff-email --agree-tos --server <Horizon ACME endpoint,
example: https://horizon.evertrust.fr/acme/profile1/directory> -m <contact email
address, example: kma@evertrust.fr> --domain <DNS name, example: nginx.evertrust.fr>
```

Where:

- `--nginx`: Enables the nginx mode
- `--no-eff-email`: Does not share your email address with EFF
- `--agree-tos`: Explicitly agrees to the terms of service
- `--server`: Horizon ACME profile endpoint
- `-m`: Contact email address
- `--domain`: Requested DNS name (can be specified several times)

Requesting a certificate for nginx using acme.sh:

```
(sudo) acme.sh --issue --nginx --server <Horizon ACME endpoint, example:
https://horizon.evertrust.fr/acme/profile1/directory> --accountemail <contact email
address, example: kma@evertrust.fr> -d <DNS name, example: nginx.evertrust.fr>
```

Where:

- `--issue`: Specifies that this is a certificate request
- `--nginx`: Enables the nginx mode
- `--server`: Horizon ACME profile endpoint
- `--accountemail`: Contact email address
- `-d`: Requested DNS name (can be specified several times)

Requesting a certificate in standalone mode using certbot:

```
(sudo) certbot certonly --standalone --no-eff-email --agree-tos --server <Horizon ACME
endpoint, example: https://horizon.evertrust.fr/acme/profile1/directory> -m <contact
email address, example: kma@evertrust.fr> --domain <DNS name, example:
apache.evertrust.fr>
```

Where:

- `--standalone`: Enables the standalone mode, i.e. certbot will start a local web server to serve the response
- `--no-eff-email`: Does not share your email address with EFF
- `--agree-tos`: Explicitly agrees to the terms of service
- `--server`: Horizon ACME profile endpoint
- `-m`: Contact email address
- `--domain`: Requested DNS name (can be specified several times)

Requesting a certificate in standalone mode using acme.sh:

```
(sudo) acme.sh --issue --standalone --server <Horizon ACME endpoint, example:
https://horizon.evertrust.fr/acme/profile1/directory> --accountemail <contact email
address, example: kma@evertrust.fr> -d <DNS name, example: apache.evertrust.fr>
```

Where:

- `--issue`: Specifies that this is a certificate request
- `--standalone`: Enables the standalone mode, i.e. acme.sh will start a local web server to serve the response
- `--server`: Horizon ACME profile endpoint
- `--accountemail`: Contact email address
- `-d`: Requested DNS name (can be specified several times)

Requesting a certificate in TLS-ALPN mode using lego:

```
lego run --server <Horizon ACME endpoint, example:
https://horizon.evertrust.fr/acme/profile1/directory> --accept-tos --email <contact
email address, example: kma@evertrust.fr> --domains <DNS name, example:
nginx.evertrust.fr> --tls --tls.address <port, example :4443>
```

Where:

- `run`: Requests or renews a certificate
- `--server`: Horizon ACME profile endpoint
- `--accept-tos`: Accepts the CA terms of service
- `--email`: Contact email address
- `--domains`: Requested DNS name (can be specified several times)
- `--tls`: Enables the TLS-ALPN-01 validation mode
- `--tls.address`: Address and port the challenge server listens on (default: `:443`): must match the TLS-ALPN-01 validation port of the ACME profile



If the Horizon endpoint is served with a certificate not trusted by the system running lego, add the `--tls-skip-verify` option to skip the TLS verification of the ACME server.

Requesting a certificate with an External Account Binding (EAB)

When the **Require External Account Binding (EAB)** option is enabled on the ACME profile, the ACME client must provide the credentials of an EAB account (MAC Key ID and MAC Key) when registering its ACME account. These credentials are only displayed once, at EAB account creation and at MAC Key renewal (see MAC Key management): copy them when they are displayed.

Requesting a certificate with EAB using certbot:

```
(sudo) certbot certonly --webroot -w <path of the webroot folder> --no-eff-email
--agree-tos --server <Horizon ACME endpoint, example:
https://horizon.evertrust.fr/acme/profile1/directory> -m <contact email address,
example: kma@evertrust.fr> -d <DNS name, example: apache.evertrust.fr> --eab-kid=<MAC
Key ID> --eab-hmac-key=<MAC Key>
```

Where:

- `--webroot`: Enables the webroot mode, i.e. certbot will place the challenge response in the server's webroot folder
- `-w`: Path of the server's webroot folder
- `--no-eff-email`: Does not share your email address with EFF
- `--agree-tos`: Explicitly agrees to the terms of service
- `--server`: Horizon ACME profile endpoint
- `-m`: Contact email address
- `-d`: Requested DNS name (can be specified several times)
- `--eab-kid`: MAC Key ID of the EAB account
- `--eab-hmac-key`: MAC Key of the EAB account



The `--eab-kid` and `--eab-hmac-key` options must be passed with the `=` form: a base64url value starting with a `-` would otherwise be interpreted by certbot as another option.

Requesting a certificate with EAB using acme.sh:

```
(sudo) acme.sh --issue -w <path of the webroot folder> --server <Horizon ACME
endpoint, example: https://horizon.evertrust.fr/acme/profile1/directory>
--accountemail <contact email address, example: kma@evertrust.fr> -d <DNS name,
example: apache.evertrust.fr> --eab-kid <MAC Key ID> --eab-hmac-key <MAC Key>
```

Where:

- `--issue`: Specifies that this is a certificate request
- `-w`: Path of the server's webroot folder
- `--server`: Horizon ACME profile endpoint
- `--accountemail`: Contact email address
- `-d`: Requested DNS name (can be specified several times)
- `--eab-kid`: MAC Key ID of the EAB account
- `--eab-hmac-key`: MAC Key of the EAB account

Requesting a certificate with EAB using lego:

```
lego run --server <Horizon ACME endpoint, example:
https://horizon.evertrust.fr/acme/profile1/directory> --accept-tos --email <contact
email address, example: kma@evertrust.fr> --domains <DNS name, example:
nginx.evertrust.fr> --tls --tls.address <port, example :4443> --eab --eab.kid <MAC Key
ID> --eab.hmac <MAC Key>
```

Where:

- `run`: Requests or renews a certificate
- `--server`: Horizon ACME profile endpoint
- `--accept-tos`: Accepts the CA terms of service
- `--email`: Contact email address
- `--domains`: Requested DNS name (can be specified several times)
- `--tls`: Enables the TLS-ALPN-01 validation mode
- `--tls.address`: Address and port the challenge server listens on (default: `:443`): must match the TLS-ALPN-01 validation port of the ACME profile
- `--eab`: Use External Account Binding for account registration
- `--eab.kid`: MAC Key ID of the EAB account
- `--eab.hmac`: MAC Key of the EAB account

Requesting a certificate for an IP address

An ACME client can request an IP identifier instead of, or alongside, a DNS name: the identifier type is defined by the ACME client when placing its order. Optionally, the **IP identifier constraint** of the ACME profile can restrict the IP addresses accepted (see ACME profile).

Requesting a certificate for an IP address using acme.sh:

```
(sudo) acme.sh --issue -w <path of the webroot folder> --server <Horizon ACME
endpoint, example: https://horizon.evertrust.fr/acme/profile1/directory>
--accountemail <contact email address, example: kma@evertrust.fr> -d <IP address,
example: 192.0.2.1>
```

Where:

- `--issue`: Specifies that this is a certificate request
- `-w`: Path of the webroot folder
- `--server`: Horizon ACME profile endpoint
- `--accountemail`: Contact email address
- `-d`: Requested IP address

Requesting a certificate for an IP address using certbot:

```
(sudo) certbot certonly --webroot -w <path of the webroot folder> --no-eff-email
--agree-tos --server <Horizon ACME endpoint, example:
https://horizon.evertrust.fr/acme/profile1/directory> -m <contact email address,
example: kma@evertrust.fr> --domain <DNS name, example: apache.evertrust.fr> --ip
-address <IP address, example: 192.0.2.1>
```

Where:

- `--webroot`: Enables the webroot mode
- `-w`: Path of the webroot folder
- `--no-eff-email`: Does not share your email address with EFF
- `--agree-tos`: Explicitly agrees to the terms of service
- `--server`: Horizon ACME profile endpoint
- `-m`: Contact email address
- `--domain`: Requested DNS name
- `--ip-address`: Requested IP address, added to the certificate as a SAN alongside the DNS name

Requesting a certificate for an IP address using lego:

```
lego run --server <Horizon ACME endpoint, example:
https://horizon.evertrust.fr/acme/profile1/directory> --accept-tos --email <contact
email address, example: kma@evertrust.fr> --domains <IP address, example: 192.0.2.1>
--tls --tls.address <port, example :4443>
```

Where:

- `run`: Requests or renews a certificate
- `--server`: Horizon ACME profile endpoint
- `--accept-tos`: Accepts the CA terms of service
- `--email`: Contact email address
- `--domains`: Requested IP address
- `--tls`: Enables the TLS-ALPN-01 validation mode
- `--tls.address`: Address and port the challenge server listens on (default: `:443`): must match the TLS-ALPN-01 validation port of the ACME profile

Revoking a certificate

Revoking a certificate using certbot:

```
(sudo) certbot revoke --cert-path <path of the certificate to revoke> --server
<Horizon ACME endpoint, example: https://horizon.evertrust.fr/acme/profile1/directory>
```

Where:

- `--cert-path`: Specifies the path of the certificate to revoke
- `--server`: Horizon ACME profile endpoint

Revoking a certificate using acme.sh:

```
(sudo) acme.sh --server <Horizon ACME endpoint, example:
https://horizon.evertrust.fr/acme/profile1/directory> --revoke -d <DNS name, example:
apache.evertrust.fr>
```

Where:

- `--server`: Horizon ACME profile endpoint
- `-d`: DNS name of the certificate to revoke

Revoking a certificate using lego:

```
lego certificates revoke --server <Horizon ACME endpoint, example:
https://horizon.evertrust.fr/acme/profile1/directory> --email <contact email address,
example: kma@evertrust.fr> --cert.name <DNS name, example: apache.evertrust.fr>
```

Where:

- `certificates revoke`: Revokes the certificate identified by its name
- `--server`: Horizon ACME profile endpoint
- `--email`: Contact email address used to register the account
- `--cert.name`: Name of the certificate to revoke (the first domain name by default)

2.10.2. ACME External

ACME External Introduction

This section details how to configure the ACME protocol to be managed by Horizon but enrolled on an external ACME endpoint.

The certificate are not enrolled on Horizon but managed thanks to automatic import by third parties such as the Horizon Client.

External ACME enrollment allows to configure:

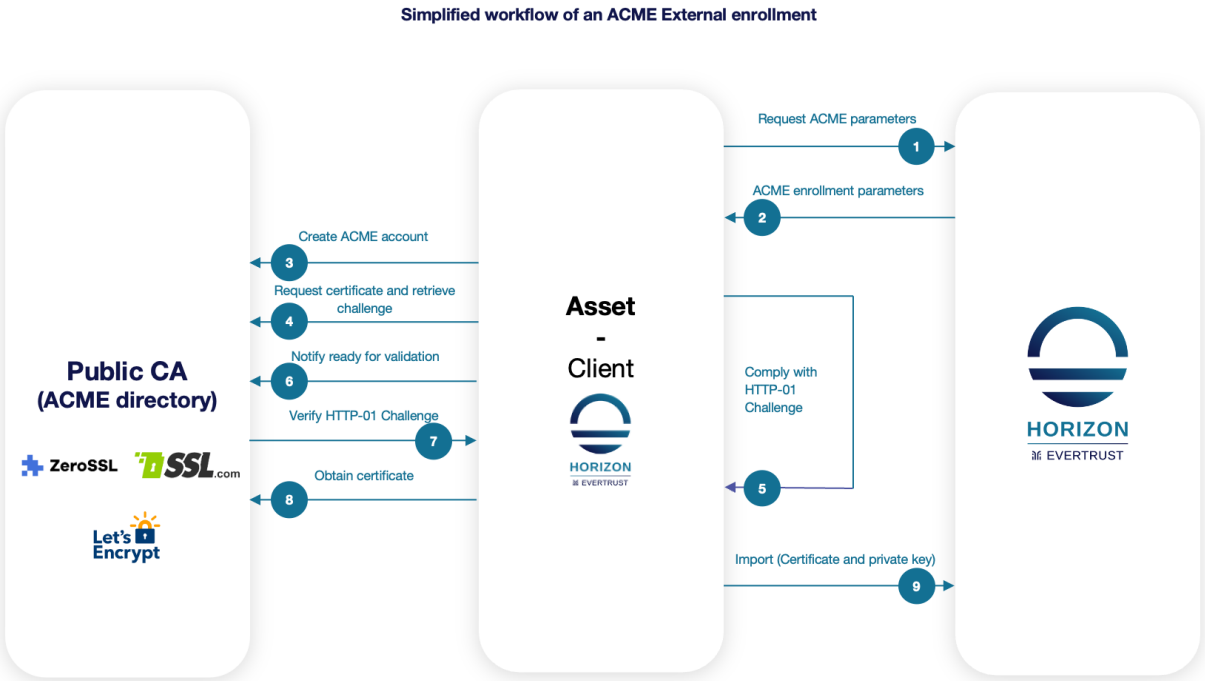
- Enrollment (will be performed by the third party);
- Renewal (will be performed by the third party, depending on the Horizon defined renewal period);
- Revocation (will be performed by Horizon).



ACME validation modes will be detailed later on. As of today, only `http-01` validation is supported.

The protocol paradigm can be described as follows: '**if the asset can prove it has authority on the DNS names (called identifiers in ACME) it is requesting for, the certificate should be automatically enrolled / renewed**', which is basically equivalent to a **Domain Validation**.

The following schema is a simplified workflow of an ACME External enrollment:



The protocol is based on the notion of challenge and offers three validation modes to actually verify challenges and prove that the asset owns authority on the requested DNS name(s), i.e. ACME identifiers:



As of today, only `http-01` validation is supported.

- `http-01`: For each requested identifier, the ACME repository will validate the challenge by connecting back in **HTTP** on the configured `http-01` *validation port* (TCP/80 by default) and retrieve the response to the challenge;
- `tls-alpn-01`: For each requested identifier, the ACME repository will validate the challenge by connecting back in **HTTPS** on the configured `tls-alpn-01` *validation port* (TCP/443 by default) and extract the response to the challenge from an **ALPN** extension in the asset / client **HTTPS** response; (not yet supported)
- `dns-01`: For each requested identifier, the ACME repository will validate the challenge through a DNS request and look for a specific TXT entry containing the response corresponding to the challenge for the considered identifier. (not yet supported)

Therefore, validation modes have the following constraints:

- `http-01 and tls-alpn-01`:
  - The ACME Repository must be able to access the asset on the validation port;

    - The validation port must be available and opened on the asset;
- `dns-01`: the ACME client must be configured with DNS credentials owning the permission to create TXT records on the requested domain(s).

Supported third parties

The following third party ACME directories have been tested with horizon-cli and Horizon:



Most third party vendors only support `keyCompromise` as revocation reason. Revocation with another reason will be rejected or ignored.

- Let's encrypt
- ZeroSSL
- SSL.com

ACME External Profile

This section details how to configure an ACME External Profile.

Prerequisites

PKI Connector

How to configure ACME External Profile

1. Log in to Horizon Administration Interface.

2. Access ACME External Profile from the drawer or card: **Protocols › ACME External**.

3. Click on  .

4. Fill in the mandatory fields.

ACME Profile Specific Configuration

General

- **Name*** *(string input)*:
  Enter a meaningful profile name. It must be unique for each profile. Horizon uses the name to identify the profile. As the name will be part of a URL, it is advisable to use only lower case letters and dashes.
- **Enable*** *(boolean)*:
  Indicates whether the profile is enabled or not. The default value is set to true.
- **PKI Connector** *(select)*:
  Select a PKI connector previously created. Only ACME connectors can be selected for this profile as only ACME revocation is supported.

Validations

- **Validation Methods*** *(select)*:
  Select the authorized ACME validation method(s) on the considered profile (`HTTP-01`).
- **ACME URL endpoint*** *(string input)*:
  Enter the ACME repository endpoint. It should end in `/acme/directory`.
- **Authorized CAs** *(select multiple)*:
  Select the authorized CAs for enrollment. Certificates not emitted on these CAs will not be able to be imported.
- **Require External Account Binding** *(boolean)*:
  Enable the requirement for the Horizon Client to ask for external account binding.

Max Certificate per Holder Policy

- **Maximum** *(int)*:
  When specified, define the maximum number of active certificates for a given holder.
- **Behavior** *(select)*:
  What behavior to have when the maximum number is reached:
  - **revoke** the previous certificates.
  - **reject** the current request.



In order to allow renewal in `reject` behavior, one more certificate is allowed when the certificate being renewed is in its renewal period.

- **Revocation reason** *(select)*:
  When the revoke behavior is selected, the revocation reason to revoke the certificate with.

Crypto policy

- **Default Key Type** *(select)*:
  Key Type that will be used by horizon-cli in certificate enrollment.
- **Authorized Key Types** *(multiselect)*:
  Key Types that can be used for enrollment. An empty value means no restrictions.
  - **Private key escrowing** *(boolean)*:
    Tells whether the private key should be escrowed by Horizon. This is true for ACME External profiles as the private key is required for revocation.

Common configuration for profiles

Languages

You can add more languages by clicking  .

- **Language*** *(select)*:
  Select a language. Supported languages are:

- **en**: English
- **fr**: French

- **Display Name** *(string input)*:
  Enter a display name. This will be the localized name of this profile.
- **Description** *(string input)*:
  Enter a description. This will be displayed on the list view of the profiles.

You can delete the localization.

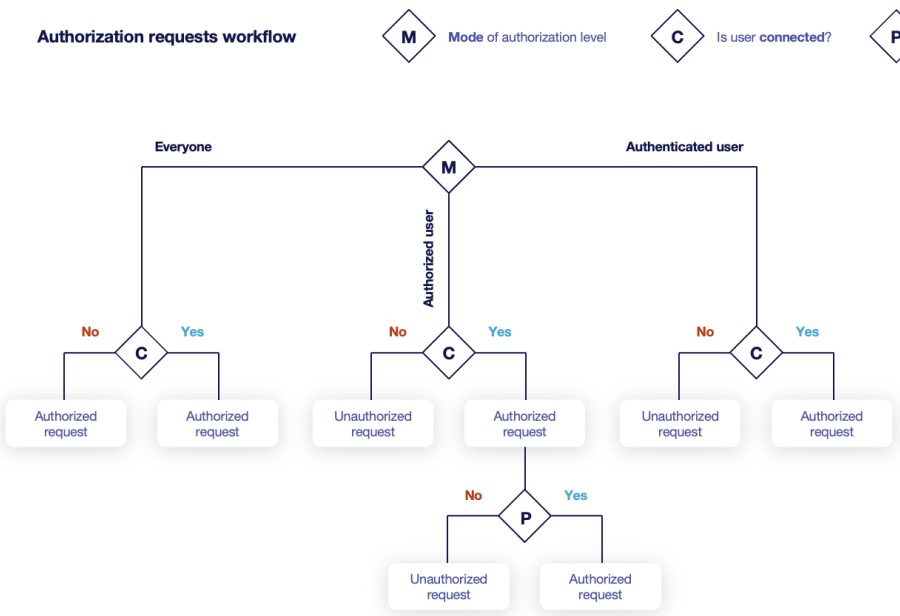
Grading Policies

You can select grading policies that will grade your certificate for a quick overview of its quality. For more information about the inner working of the grading policies in Horizon, please refer to the grading rules page.

Workflows builder

Configure custom rights for actions on this profile.

1. Select an authorization level for each workflow.



- **Everyone**:
  No authentication is required.
- **Authenticated**:
  User has to be authenticated.
- **Authorized**:
  User has to be authenticated and have an explicit authorizations.

2. Select an access level for identity providers.

You can remove the access level for an identity provider by clicking on 'x'.

Requests time to live

Configure the time your requests have before expiring.



After expiration, requests are stored for an additional 30 days. This can be changed using configuration files.

- **Enrollment request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Renewal request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Revocation request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Update request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Migration request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Recover request** *(finite duration)*:
  *Enabled on **escrow**:* Must be a valid finite duration. The default value is set to seven days.

Owner-related permissions

These permissions apply to the owners of a certificate (team or owner). An owner can always request the following actions, but this permission allows them to perform the action without validation.

- **Revoke** *(boolean)*:
  Grant self revoke permission. The default value is set to false.
- **Revoke (pop)** *(boolean)*:
  Grant self revoke permission with owner being determined by Proof of Possession. This is used by horizon-cli. The default value is set to false.
- **Update** *(boolean)*:
  Grant self update permission. The default value is set to false.
- **Update (pop)** *(boolean)*:
  Grant self update permission with owner being determined by Proof of Possession. This is used by horizon-cli. The default value is set to false.

Constraints

- **Allowed email domains** *(string input)*:
  Enter a valid regular expression that the inputted emails should match. This includes RFC822NAME and UPN SANs as well as the contact email



This matches the domain of the email, not including anything before `@`.

- **Allowed DNS domains** *(string input)*:
  Enter a valid regular expression that the inputted domain should match.

CSR Data Mapping

1. Click on  to add a mapping.

2. Select a field and enter a value.

You can delete  the CSR Data Mapping.

Certificate Metadata

This section details how to define a custom structure for the labels, ownership policy and technical metadata, allowing certificates to hold rich information.

Labels

You can add more labels by clicking  .

- **Name** *(select)*:
  Select a preexisting label.
- **Mandatory** *(boolean)*:
  Should the label be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the label should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the label should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the label.
- **Label value restriction**
  - **Whitelist** *(string input multiple)*:
    The label value will have to be in the whitelist. Open the popup, enter the label value and press "enter" to add this value to the accepted value list. An empty whitelist means no restriction.
  - **Suggestions** *(string input multiple)*:
    Add suggestions that will be displayed to the user. The user will be able to choose one of these values or enter its own. Open the popup, enter your suggestions and press enter to add this value to the suggestions. An empty suggestions list means no restriction.
  - **Regex** *(regex)*:
    The label value will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.

- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this label to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can delete  or reorder (drag and drop)  the label template.

Ownership policy

- **Owner**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's owner is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when approving a request.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the owner to the value of the evaluated computation rule. This value will override any other value including the user input.
- **Contact email**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's contact email is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when approving a request.
  - **Default contact email** *(string input)*:
    Set a default contact email. This value must comply with the contact email restriction.
  - **Contact email restriction**
    - **Whitelist** *(string input multiple)*:
      The contact email will have to be in the whitelist. Open the popup, enter the email and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The contact email will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the contact email to the value of the evaluated computation rule. This value

will override any other value including the user input and the default value.

- **Team**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's team is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when approving a request.
  - **Default team** *(string input)*:
    Set a default team. This value must comply with the team restriction.
  - **Team restriction**
    - **Whitelist** *(string input multiple)*:
      The team will have to be in the whitelist. Enter the team and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The team will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the team to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

Metadata policy *(overridable metadata)*



These metadata are technical metadata. They are used by Horizon or Third party connectors, updating them should be done with utmost care.



Metadata edition is not allowed on enroll.



Metadata edition is not available via the User Interface. It must be changed with API, using horizon-cli.

You can allow the override of technical metadata by clicking .

- **Metadata*** *(select)*:
  Select a metadata.
- **Editable by requester** *(boolean)*:
  Tells whether the metadata is editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the metadata is editable by the approver. The default value is set to false.

You can delete a metadata policy. This will not delete the metadata but will make it non editable.

Third-party policy

- **Synchronize data from discovery** *(boolean)*:
  Enable overriding of third-party data during discovery. When enabled, the following logic applies:
  - If a discovery attempts to set third-party data on a connector that is not referenced in the profile (by a trigger), the data will not be added.
  - If the connector is referenced in the profile, the discovery will override the existing data.

Notifications

This section details how to configure notifications on certificate and request lifecycle events.

Certificate lifecycle notifications

Notifications are sent when one of the following event is triggered by a certificate:

Enrollment	Revocation	Expire	Update	Migrate	Renew

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.

Request lifecycle notifications

Notifications are sent when one of the following event is triggered by an Enroll/Revocation/Update/Migrate/Renew request:

Submit	Cancel	Revoke	Approve	Pending

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.



Submit request events are not triggered when the user has the permission to perform the action directly.

5. Click on the save button.

You can edit , duplicate or delete the ACME Profile.



You won't be able to delete an ACME External Profile if it is referenced somewhere else.

2.10.3. CRMP

CRMP Introduction

This integration involves the following components:

- OpenTrust CMS
- EverTrust Horizon
- Cards to be enrolled

The integration is very simple, Horizon acting as an OpenTrust PKI for OpenTrust CMS. We will first see how to configure the Horizon CRMP Profile, which will define how to enroll your certificates, and then a step by step guide for a quick integration.

CRMP Profile

This section details how to configure the CRMP Profile

Prerequisites

PKI Connector

How to configure CRMP Profile

1. Log in to Horizon Administration Interface.

2. Access CRMP Profile from the drawer or card: **Protocol** › **CRMP**.

3. Click on  .

4. Fill in the mandatory fields.

CRMP profile specific configuration

General

- **Name*** *(string input)*:
  Enter a meaningful profile name. It must be unique for each profile. Horizon use the name to identify the profile.
- **Enable** *(boolean)*:
  Tells whether the profile is enabled for enrollment or not. The default value is set to true.
- **PKI Connector*** *(string select)*:
  Select a PKI connector previously created.
- **Data field identifier** *(select)*:
  *Enabled on **Escrow**:* When **recovering** a certificate, select on which Horizon field the field named `userprincipalname` on the CMS will be mapped. This will be used to identify a user, so this

data should be a unique identifier on the CMS side, in a field named `userprincipalname`, and mapped to the corresponding Horizon field in application configuration on the CMS.

Crypto policy

- **Default Key Type*** *(select)*:
  Key Type that will be used by the CMS in certificate enrollment.
- **Centralized enrollment** *(boolean)*:
  Enable centralized enrollment. In CRMP, only one enrollment mode can be enabled.
  - **Private key escrowing** *(boolean)*:
    Enable key escrow. Only available in centralized enrollment mode.
  - **PKCS#12 Password generation Mode*** *(select)*:
    *For certificate **recovery**:* Select a mode for PKCS#12 password generation:
    - **manual**: prompt the user to choose its password.
    - **random**: have the password generated on Horizon side.
  - **Password policy** *(select)*:
    Select a previously created password policy. It will be enforced on PKCS#12 password for recovery and CMS centralized enrollments.
  - **Store encryption type*** *(select)*:
    Select an encryption algorithm from the list. The PKCS#12 will use this algorithm. For CRMP it is enforced on DES Average because of CMS support.
  - **Transient key lifetime** *(finite duration)*:
    Gives the retention period for non-escrowed keys. During this period, triggers using the key can be retried.
- **Decentralized enrollment** *(boolean)*:
  Enable decentralized enrollment. In CRMP, only one enrollment mode can be enabled.

Max Certificate per Holder Policy

- **Maximum** *(int)*:
  When specified, define the maximum number of active certificates for a given holder.
- **Behavior** *(select)*:
  What behavior to have when the maximum number is reached:
  - **revoke** the previous certificates.
  - **reject** the current request.



In order to allow renewal in `reject` behavior, one more certificate is allowed when the certificate being renewed is in its renewal period.

- **Revocation reason** *(select)*:
  When the revoke behavior is selected, the revocation reason to revoke the certificate with.

Common configuration for profiles

Languages

You can add more languages by clicking  .

- **Language*** *(select)*:
  Select a language. Supported languages are:
  - **en**: English
  - **fr**: French
- **Display Name** *(string input)*:
  Enter a display name. This will be the localized name of this profile.
- **Description** *(string input)*:
  Enter a description. This will be displayed on the list view of the profiles.

You can delete  the localization.

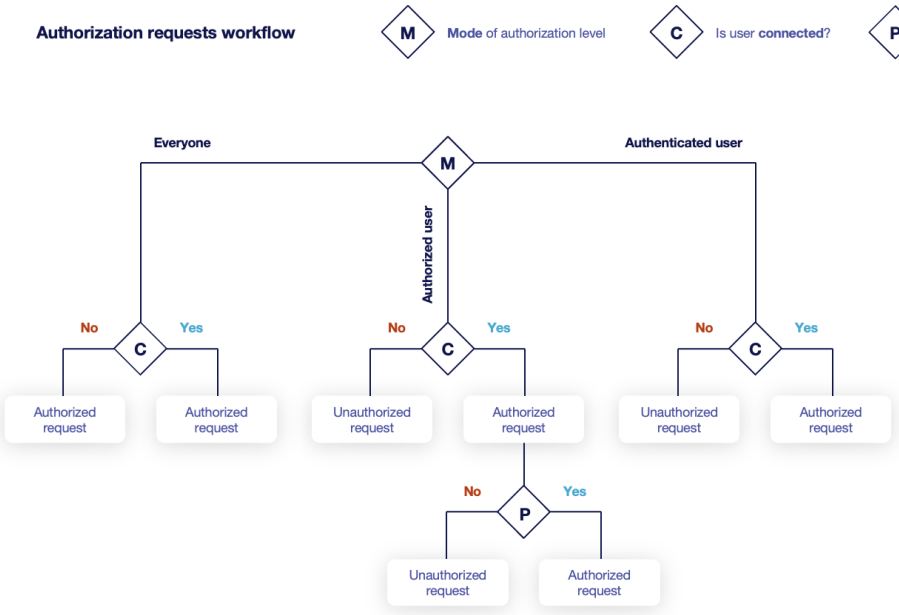
Grading Policies

You can select grading policies that will grade your certificate for a quick overview of its quality. For more information about the inner working of the grading policies in Horizon, please refer to the grading rules page.

Workflows builder

Configure custom rights for actions on this profile.

1. Select an authorization level for each workflow.



USER A : has workflow permission | USER B : has no workflow permission

Everyone		
USER A	not connected + no permissions	approve
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authenticated		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authorized		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	deny

- **Everyone**:
  No authentication is required.
- **Authenticated**:
  User has to be authenticated.
- **Authorized**:
  User has to be authenticated and have an explicit authorizations.

2. Select an access level for identity providers.

You can remove the access level for an identity provider by clicking on 'x'.

Requests time to live

Configure the time your requests have before expiring.



After expiration, requests are stored for an additional 30 days. This can be changed using configuration files.

- **Enrollment request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Renewal request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Revocation request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Update request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Migration request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Recover request** *(finite duration)*:
  *Enabled on **escrow**:* Must be a valid finite duration. The default value is set to seven days.

Owner-related permissions

These permissions apply to the owners of a certificate (team or owner). An owner can always request the following actions, but this permission allows them to perform the action without validation.

- **Revoke** *(boolean)*:
  Grant self revoke permission. The default value is set to false.
- **Recover** *(boolean)*:
  Grant self recover permission. The default value is set to false.
- **Update** *(boolean)*:
  Grant self update permission. The default value is set to false.

Constraints

- **Allowed email domains** *(string input)*:
  Enter a valid regular expression that the inputted emails should match. This includes RFC822NAME and UPN SANs as well as the contact email

  This matches the domain of the email, not including anything before `@`.

- **Allowed DNS domains** *(string input)*:
  Enter a valid regular expression that the inputted domain should match.

Certificate Template

This section details how to define a custom structure for the fields `subject DN`, `SAN` & `extensions` of the requested certificate in order to match the configuration on the PKI side.



In a CRMP profile, defining a template is mandatory.

Subject DN composition

You can add more elements by clicking .

- **Element*** *(select)*:
  Select an attribute from the elements list.
- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the element.
- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the Subject DN template.



When a template is defined, at least one mandatory Common Name must be added to the DN Elements.

SAN composition

You can add more elements by clicking  .

- **Element*** *(select)*:
  Select an attribute from the element list.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Minimum** *(int)*:
  The minimum number of value that this SAN must have.
- **Maximum** *(int)*:
  The maximum number of value that this SAN must have.
- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the SAN template.

Extensions

You can add more elements by clicking  .

- **Element*** *(select)*:
  Select an attribute from the elements list.
- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the element.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the

Extensions template.



When adding a SAN, a DN element or an Extension and making it mandatory, make sure to either give it a default value or a computation rule or make it editable, otherwise the template will be unusable.

Certificate Metadata

This section details how to define a custom structure for the labels, ownership policy and technical metadata, allowing certificates to hold rich information.

Labels

You can add more labels by clicking  .

- **Name** *(select)*:
  Select a preexisting label.
- **Mandatory** *(boolean)*:
  Should the label be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the label should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the label should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the label.
- **Label value restriction**
  - **Whitelist** *(string input multiple)*:
    The label value will have to be in the whitelist. Open the popup, enter the label value and press "enter" to add this value to the accepted value list. An empty whitelist means no restriction.
  - **Suggestions** *(string input multiple)*:
    Add suggestions that will be displayed to the user. The user will be able to choose one of these values or enter its own. Open the popup, enter your suggestions and press enter to add this value to the suggestions. An empty suggestions list means no restriction.
  - **Regex** *(regex)*:
    The label value will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this label to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can delete  or reorder (drag and drop)  the label template.

Ownership policy

- **Owner**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's owner is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when approving a request.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the owner to the value of the evaluated computation rule. This value will override any other value including the user input.
- **Contact email**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's contact email is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when approving a request.
  - **Default contact email** *(string input)*:
    Set a default contact email. This value must comply with the contact email restriction.
  - **Contact email restriction**
    - **Whitelist** *(string input multiple)*:
      The contact email will have to be in the whitelist. Open the popup, enter the email and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The contact email will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the contact email to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.
- **Team**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's team is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when submitting a

request.

- **Editable by approver** *(boolean)*:
  Specify if the certificate's team can be overridden by the requester when approving a request.
- **Default team** *(string input)*:
  Set a default team. This value must comply with the team restriction.
- **Team restriction**
  - **Whitelist** *(string input multiple)*:
    The team will have to be in the whitelist. Enter the team and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
  - **Regex** *(regex)*:
    The team will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of the team to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

Metadata policy *(overridable metadata)*



These metadata are technical metadata. They are used by Horizon or Third party connectors, updating them should be done with utmost care.



Metadata edition is not allowed on enroll.



Metadata edition is not available via the User Interface. It must be changed with API, using horizon-cli.

You can allow the override of technical metadata by clicking .

- **Metadata*** *(select)*:
  Select a metadata.
- **Editable by requester** *(boolean)*:
  Tells whether the metadata is editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the metadata is editable by the approver. The default value is set to false.

You can delete  a metadata policy. This will not delete the metadata but will make it non editable.

Third-party policy

- **Synchronize data from discovery** *(boolean)*:
  Enable overriding of third-party data during discovery. When enabled, the following logic

applies:

- If a discovery attempts to set third-party data on a connector that is not referenced in the profile (by a trigger), the data will not be added.
- If the connector is referenced in the profile, the discovery will override the existing data.

Notifications/Triggers

This section details how to configure notifications and triggers to perform actions on certificate and request lifecycle events.

Certificate lifecycle notifications

Notifications are sent when one of the following event is triggered by a certificate:

Enrollment	Revocation	Expire	Update	Migrate	Renew

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.

Request lifecycle notifications

Notifications are sent when one of the following event is triggered by an Enroll/Revocation/Update/Migrate/Renew request:

Submit	Cancel	Revoke	Approve	Pending

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.



Submit request events are not triggered when the user has the permission to perform the action directly.

Triggers

Horizon support the use of third-party triggers in the form of callbacks on specific events happening on the profile, giving a way to synchronize the third party repositories and Horizon.

- **Enrollment** *(select)*:
  Select the preexisting third party or MDM trigger(s) to call whenever a certificate is enrolled on this profile.
- **Renewal** *(select)*:
  Select the preexisting third party or MDM trigger(s) to call whenever a certificate is renewed on this profile.
- **Revocation** *(select)*:
  Select the preexisting third party or MDM trigger(s) to call whenever a certificate gets revoked on this profile.
- **Expire** *(select)*:

Select the preexisting third party or MDM trigger(s) to call whenever a certificate expires on this profile.

The available triggers are the following:

AKV Triggers	AWS Triggers	F5 Triggers	F5 AS3 Triggers
Netscaler Triggers	GCM Triggers	[admin-guide:third-parties-fortinet-triggers:::_fortinet_triggers]	[admin-guide:third-parties-paloalto-triggers:::_paloalto_triggers]
[admin-guide:third-parties-ldap-triggers:::_ldap_triggers]	Intune PKCS Triggers		

5. Click on the save button.

You can edit , duplicate or delete the CRMP Profile.



You won't be able to delete a CRMP Profile if a certificate is enrolled on it.

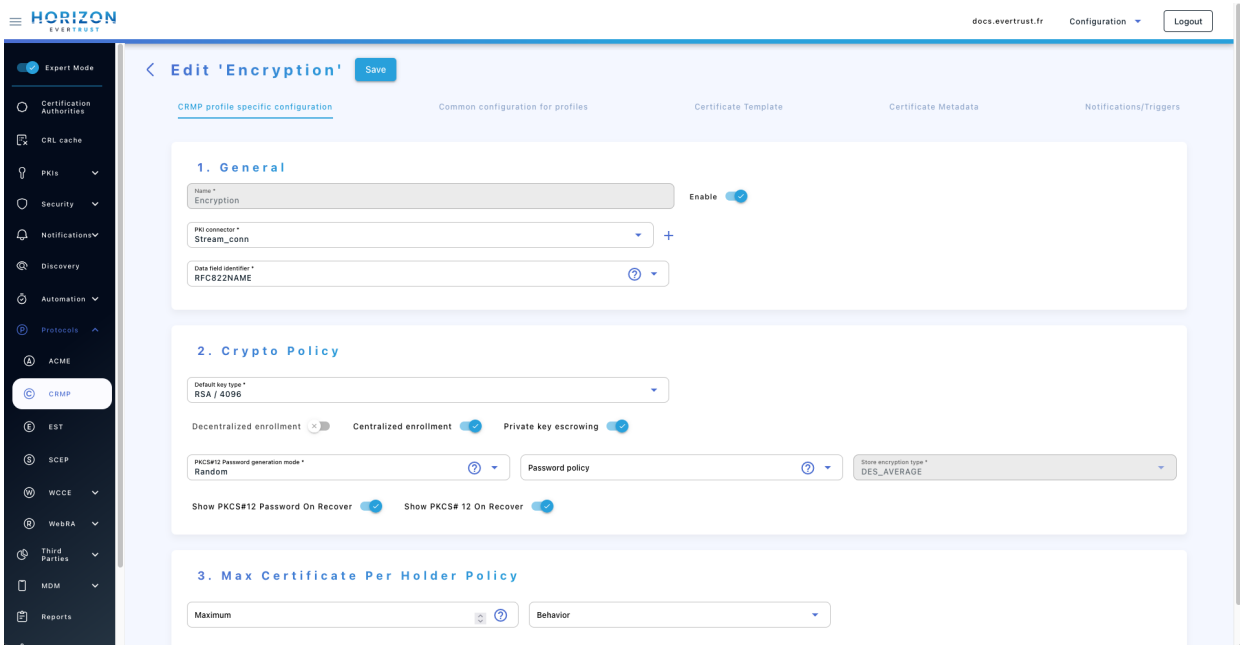
Enroll your first card with OpenTrust CMS

A step by step guide for a perfect integration between Horizon and OpenTrust CMS

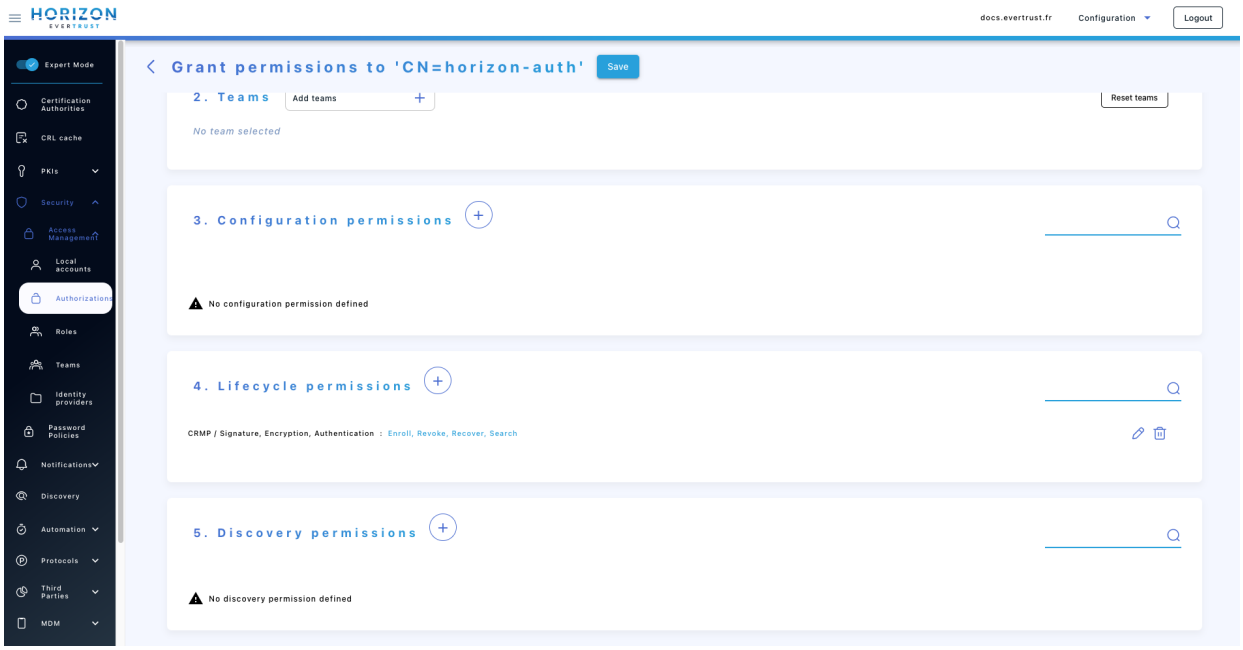
I. Configure Horizon

1. Create your profiles.

   In **Configuration › Protocols › CRMP** you will have the possibility to setup your profiles.
   Let's create three profiles, that will later result in 3 certificates for each user: an authentication certificate, a signing certificate and an encryption certificate.
   The first two will be decentralized profiles, and the encryption one will be centralized with escrow, so that we can always decrypt the user communications later. All configuration options are available in the profile section.



2. Create an account.
   OpenTrust CMS will need access to Horizon in order to manage your cards certificates.
   In order to do so, a certificate needs to be enrolled on a CA trusted by Horizon for client authentication.
   This certificate should be able to **enroll**, **revoke**, **recover** and **search** on the CRMP profiles you want it to manage.
   My certificate will here have for DN: `CN=horizon-auth`, and I will give it the appropriate rights.



II. Configure your CMS applications

1. Connect your applications.
   For each of the profiles on Horizon, a **CRMP** application must be created (If **CRMP** is not available, it must be installed on your CMS: refer to OpenTrust CMS documentation).
   It first needs to be able to connect.
   The server url must be set to `https://<horizon-url>/crmp`.
   The SSL client identity must then be set to the certificate created in step I.2.

Connection Settings	
Server URL	https://<horizon-url>/crmp
SSL Client Identity	CN=crmp-auth Modify
Connection to Application	Connect

2. Map your applications.
   The information setup on Horizon will be displayed, and the fields can be mapped.
   The enrolled certificate on Horizon will be the result of the values mapped in the Horizon Fields on the left.
   It should be noted that some Horizon fields are indexed, but the CMS does not display numbers. They are ordered in the same order as on Horizon, with mandatory fields first and then optional fields.

Certificate Management Profile Settings	
Profile	Encryption
PKI Version	2.004 (API 1, r2.004)
Type	Centralized Escrow Key Size: 4096
subject.cn.*	Datasource:cn X
san.rfc822name.*	Datasource:userprincipalname X
Email	Datasource:userprincipalname X



Escrow: Due to a technical limitation in the CMS, for certificates that are escrowed, a field with technical name `userprincipalname` must be mapped to the selected `Data Field Identifier` in the CRMP Profile. Otherwise, the user will not be able to recover its certificates. The field `userprincipalname` must then be able to uniquely identify each user.

2.10.4. EST

EST Introduction

This section refers to the EST protocol, as described by RFC 7030.

EST Profile

This section details how to configure the EST Profile

Prerequisites

PKI Connector

How to configure EST Profile

1. Log in to Horizon Administration Interface.

2. Access EST Profile from the drawer or card: **Protocol** › **EST**.

3. Click on  .

4. Fill in the mandatory fields.

EST Specific Configuration

General

- **Name*** *(string input)*:
  Enter a meaningful profile name. It must be unique for each profile. Horizon use the name to identify the profile.
- **Enable** *(boolean)*:
  Tells whether the profile is enabled or not. The default value is set to true.
- **PKI Connector** *(string select)*:
  Select a PKI connector previously created.
- **Terms of Service** *(string select)*:
  Select a previously created [admin-guide:system-terms_of_service:::_terms_of_service] entry. When set, the Terms of Service are displayed when issuing a challenge request and must be explicitly accepted by the requester before a challenge password is returned. Leave empty to keep the workflow unchanged.

Authorization and validation

- **Authorization mode** *(select)*:
  Select from the list.
- **Authorized**:

- **Enable whitelist** *(boolean)*:
  Tells whether whitelist is enabled or not. The default value is set to false.
- **CA*** *(select)*:
  Select a Certificate Authority previously created.

- **X509**:
  - **Enrollment CAs** *(select)*:
    Available only if mode at x509. Select a Certificate Authority previously created.
  - **Enable whitelist** *(boolean)*:
    Tells whether whitelist is enabled or not. The default value is set to false.
  - **CA*** *(select)*:
    Select a Certificate Authority previously created.
- **Challenge**:
  - **Password policy** *(select)*:
    Select a password policy previously created. It is used for the challenge generation.
  - **Enable whitelist** *(boolean)*:
    Tells whether whitelist is enabled or not. The default value is set to false.
  - **CA*** *(select)*:
    Select a Certificate Authority previously created.
- **Auto Validation**:

This enables auto validation.

- **CA*** *(select)*:
  Select a Certificate Authority previously created.

Max Certificate per Holder Policy

- **Maximum** *(int)*:
  When specified, define the maximum number of active certificates for a given holder.
- **Behavior** *(select)*:
  What behavior to have when the maximum number is reached:
  - **revoke** the previous certificates.
  - **reject** the current request.



In order to allow renewal in `reject` behavior, one more certificate is allowed when the certificate being renewed is in its renewal period.

- **Revocation reason** *(select)*:
  When the revoke behavior is selected, the revocation reason to revoke the certificate with.

Renewal management

- **Renewal period** *(finite duration)*:
  Must be a valid finite duration.

- **Renewal CAs** *(select)*:
  Select a Certificate Authority previously created.

Crypto Policy

- **Default Key Type** *(select)*:
  Select the default type of key to generate when using centralized enrollment mode.
- **Authorized Key Types** *(multiselect)*:
  Key Types that can be used for enrollment. An empty value means no restrictions.
- **Centralized enrollment** *(boolean)*:
  Tells whether the profile should be used with a centralized enrollment, i.e providing a PKCS#12. The default value is set to false.
  - **Private key escrowing** *(boolean)*:
    Tells whether the private key should be escrowed by Horizon. The default value is set to false.
    - **Show PKCS#12 Password On Recover** *(boolean)*:
      Tells whether the PKCS#12 password should be displayed on recover. The default value is set to false.
    - **Show PKCS#12 On Recover** *(boolean)*:
      Tells whether the PKCS#12 should be displayed on recover. The default value is set to false.
  - **PKCS#12 Password Mode*** *(select)*:
    Select how to generate PKCS#12 password:
    - **manual**: prompt the user to choose its password. This is the default behavior.
    - **random**: have the password generated on Horizon side.
  - **Password policy** *(select)*:
    Select a previously created password policy. It will be enforced on PKCS#12 password for recovery and centralized enrollments.
  - **Store encryption type*** *(select)*:
    Select an encryption algorithm from the list. The PKCS#12 will use this algorithm. The default value is set to DES_AVERAGE.
  - **Transient key lifetime** *(finite duration)*:
    Gives the retention period for non-escrowed keys. During this period, triggers using the key can be retried.
- **Decentralized enrollment** *(boolean)*:
  Tells whether the profile should be used with a decentralized enrollment mode, i.e CSR (PKCS#10) signing by the PKI. The default value is set to true.

Common configuration for profiles

Languages

You can add more languages by clicking .

- **Language*** *(select)*:
  Select a language. Supported languages are:
  - **en**: English
  - **fr**: French
- **Display Name** *(string input)*:
  Enter a display name. This will be the localized name of this profile.
- **Description** *(string input)*:
  Enter a description. This will be displayed on the list view of the profiles.

You can delete  the localization.

Grading Policies

You can select grading policies that will grade your certificate for a quick overview of its quality. For more information about the inner working of the grading policies in Horizon, please refer to the grading rules page.

Requests time to live

Configure the time your requests have before expiring.



After expiration, requests are stored for an additional 30 days. This can be changed using configuration files.

- **Enrollment request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Renewal request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Revocation request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Update request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Migration request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Recover request** *(finite duration)*:
  *Enabled on **escrow**:* Must be a valid finite duration. The default value is set to seven days.

Constraints

- **Allowed email domains** *(string input)*:
  Enter a valid regular expression that the inputted emails should match. This includes RFC822NAME and UPN SANs as well as the contact email



This matches the domain of the email, not including anything before `@`.

- **Allowed DNS domains** *(string input)*:
  Enter a valid regular expression that the inputted domain should match.

CSR Data Mapping

1. Click on  to add a mapping.

2. Select a field and enter a value.

You can delete  the CSR Data Mapping.

Workflow

Auto-validation

Configure auto validation rules to avoid needing permissions configuration.



A request permission must be available in order for the request to be created and then auto-validated. See workflows to modify request permissions.

1. To enable auto-validation, switch the profile mode to `auto-validation`

2. Add rules that will be evaluated on each request. For more details, see the validation rules reference.

3. Add the threshold. This is the number of rules that must pass in order for the request to be validated.

Data source flow

Configure which data sources to execute and in which order.

1. Select a data source to execute first, and fill its inputs with a computation rule.

2. Add other data sources if needed. Each datasource input can use outputs from previously executed data sources.

3. All data sources output are available in computation rules throughout the certificate template and metadata.

Datasource options

Stop on success

When this option is enabled, if the datasource data is successfully fetched, the flow will stop immediately.

This allows to search through a list of datasource, when the data can be in any of multiple datasource.

Mandatory

When this option is enabled, if the datasource return no results, the flow will stop immediately with an error.

This allows to block enrollment flows if a required record is not present, event if is not saved into the final certificate.



A datasource is considered as finding no results if:

- LDAP Datasource: no record match the filter
- DNS Datasource: no record match the query
- REST Datasource: the result code is in the `notFoundHttpCodes`

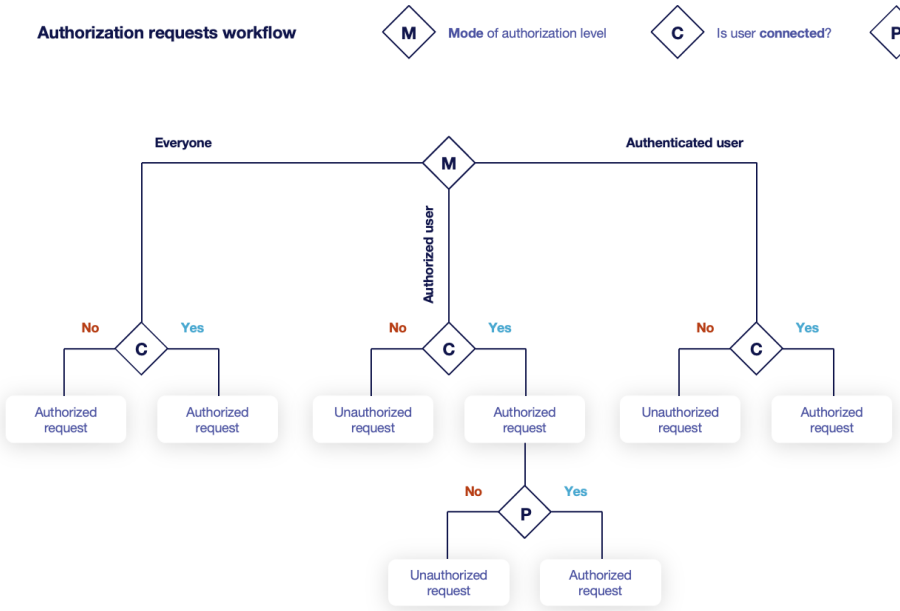


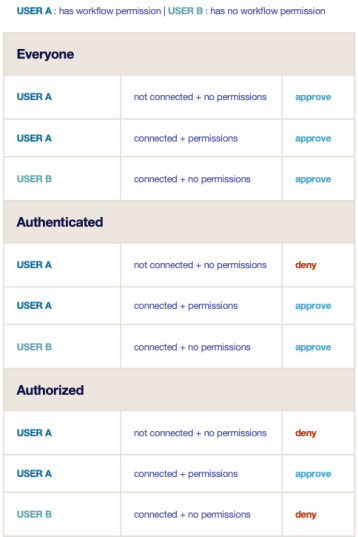
If a specific field from a record is considered as mandatory, including it in a mandatory label will also fail the enrollment flow if it is not found

Workflows builder

Configure custom rights for actions on this profile.

1. Select an authorization level for each workflow.





USER A : has workflow permission | USER B : has no workflow permission

Everyone		
USER A	not connected + no permissions	approve
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authenticated		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authorized		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	deny

- **Everyone**:
  No authentication is required.
- **Authenticated**:
  User has to be authenticated.
- **Authorized**:

User has to be authenticated and have an explicit authorizations.

2. Select an access level for identity providers.

You can remove the access level for an identity provider by clicking on 'x'.

Owner-related permissions

These permissions apply to the owners of a certificate (team or owner). An owner can always request the following actions, but this permission allows them to perform the action without validation.

- **Revoke** *(boolean)*:
  Grant self revoke permission. The default value is set to false.
- **Revoke (pop)** *(boolean)*:
  Grant self revoke permission with owner being determined by Proof of Possession. This is used by horizon-cli. The default value is set to false.
- **Recover** *(boolean)*:
  Grant self recover permission. The default value is set to false.
- **Update** *(boolean)*:
  Grant self update permission. The default value is set to false.
- **Update (pop)** *(boolean)*:
  Grant self update permission with owner being determined by Proof of Possession. This is used by horizon-cli. The default value is set to false.

Certificate Template

This section details how to define a custom structure for the fields `subject DN`, `SAN` *&* `extensions` *of the requested certificate in order to match the configuration on the PKI side.*



Defining a template will use the CSR to fill the available field. A CSR with unexpected fields will be rejected. Using a template also disables CSR Data Mapping.

Subject DN composition

You can add more elements by clicking .

- **Element*** *(select)*:
  Select an attribute from the elements list.
- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.

- **Default value** *(string input)*:
  Set a default value to the element.
- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button or reorder (drag and drop) the Subject DN template.



When a template is defined, at least one mandatory Common Name must be added to the DN Elements.

SAN composition

You can add more elements by clicking .

- **Element*** *(select)*:
  Select an attribute from the element list.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Minimum** *(int)*:
  The minimum number of value that this SAN must have.
- **Maximum** *(int)*:
  The maximum number of value that this SAN must have.
- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button or reorder (drag and drop) the SAN template.

Extensions

You can add more elements by clicking .

- **Element*** *(select)*:
  Select an attribute from the elements list.

- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the element.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the Extensions template.



When adding a SAN, a DN element or an Extension and making it mandatory, make sure to either give it a default value or a computation rule or make it editable, otherwise the template will be unusable.

Certificate Metadata

This section details how to define a custom structure for the labels, ownership policy and technical metadata, allowing certificates to hold rich information.

Labels

You can add more labels by clicking .

- **Name** *(select)*:
  Select a preexisting label.
- **Mandatory** *(boolean)*:
  Should the label be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the label should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the label should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the label.
- **Label value restriction**
  - **Whitelist** *(string input multiple)*:
    The label value will have to be in the whitelist. Open the popup, enter the label value and press "enter" to add this value to the accepted value list. An empty whitelist means no restriction.

- **Suggestions** *(string input multiple)*:
  Add suggestions that will be displayed to the user. The user will be able to choose one of these values or enter its own. Open the popup, enter your suggestions and press enter to add this value to the suggestions. An empty suggestions list means no restriction.
- **Regex** *(regex)*:
  The label value will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.

- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this label to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can delete  or reorder (drag and drop)  the label template.

Ownership policy

- **Owner**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's owner is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when approving a request.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the owner to the value of the evaluated computation rule. This value will override any other value including the user input.
- **Contact email**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's contact email is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when approving a request.
  - **Default contact email** *(string input)*:
    Set a default contact email. This value must comply with the contact email restriction.
  - **Contact email restriction**
    - **Whitelist** *(string input multiple)*:
      The contact email will have to be in the whitelist. Open the popup, enter the email and press "enter" to add this value to the accepted whitelist. An empty whitelist means no

restriction.

- **Regex** *(regex)*:
  The contact email will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.

- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of the contact email to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

- **Team**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's team is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when approving a request.
  - **Default team** *(string input)*:
    Set a default team. This value must comply with the team restriction.
  - **Team restriction**
    - **Whitelist** *(string input multiple)*:
      The team will have to be in the whitelist. Enter the team and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The team will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the team to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

Metadata policy *(overridable metadata)*



These metadata are technical metadata. They are used by Horizon or Third party connectors, updating them should be done with utmost care.



Metadata edition is not allowed on enroll.



Metadata edition is not available via the User Interface. It must be changed with API, using horizon-cli.

You can allow the override of technical metadata by clicking  .

- **Metadata*** *(select)*:
  Select a metadata.
- **Editable by requester** *(boolean)*:
  Tells whether the metadata is editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the metadata is editable by the approver. The default value is set to false.

You can delete a metadata policy. This will not delete the metadata but will make it non editable.

Third-party policy

- **Synchronize data from discovery** *(boolean)*:
  Enable overriding of third-party data during discovery. When enabled, the following logic applies:
  - If a discovery attempts to set third-party data on a connector that is not referenced in the profile (by a trigger), the data will not be added.
  - If the connector is referenced in the profile, the discovery will override the existing data.

Notifications

This section details how to configure notifications on certificate and request lifecycle events.

Certificate lifecycle notifications

Notifications are sent when one of the following event is triggered by a certificate:

Enrollment	Revocation	Expire	Update	Migrate	Renew

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.

Request lifecycle notifications

Notifications are sent when one of the following event is triggered by an Enroll/Revocation/Update/Migrate/Renew request:

Submit	Cancel	Revoke	Approve	Pending

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.



Submit request events are not triggered when the user has the permission to perform the action directly.

5. Click on the save button.

You can edit , duplicate or delete the EST Profile.



You won't be able to delete a EST Profile if this one is referenced somewhere else.

2.10.5. SCEP

SCEP Introduction

This section refers to the SCEP protocol, as described by RFC 8894.

SCEP Authorities

This section details how to configure SCEP Authorities.

The draft-nourse-scep-23 as well as RFC 8894 define how SCEP communications are secured. This involves using a SCEP Authority, which is a certificate and its associated private key, used to sign and encrypt communications between SCEP server and client.

Two setups are possible:

- the **CA mode** in which the SCEP Authority is a self-signed certificate. In that mode the SCEP server returns the self-signed certificate as `application/x-x509-ca-cert` when the client uses the `GetCaCert` call.
- the **RA mode** in which the SCEP Authority is a certificate signed by the CA that will issue certificates using the considered SCEP profile. In that mode, the SCEP server returns the SCEP Authority certificate and its issuing CA chain as `application/x-x509-ca-ra-cert` when the client uses the `GetCaCert` call.

Therefore, it is important in each SCEP or MDM Profile to align the SCEP mode with the characteristics of the SCEP Authority configured in the current section.

Prerequisites

- PKCS#12 containing the SCEP Authority certificate and private key. See above for explanation about the SCEP contents.

How to configure a SCEP Authority

SCEP Authorities are configured as credentials.

SCEP Profile

This section details how to configure the SCEP Profile

Prerequisites

PKI Connector	SCEP Authority

How to configure SCEP Profile

1. Log in to Horizon Administration Interface.

2. Access SCEP Profile from the drawer or card: **Protocol › SCEP**.

3. Click on  .

4. Fill in the mandatory fields.

SCEP Profile Specific Configuration

General

- **Name*** *(string input)*:
  Enter a meaningful profile name. It must be unique for each profile. Horizon use the name to identify the profile.
- **Enable** *(boolean)*:
  Tells whether the profile is enabled or not. The default value is set to true.
- **PKI Connector*** *(string select)*:
  Select a PKI connector previously created.
- **Authorize POST enrollment** *(boolean)*:
  Enable scep enrollment routes with HTTP POST method. The defaults value is set to false.
- **Authorization mode*** *(select)*:
  Select `Challenge` mode to allow enrollment using a pre validated request containing a challenge, `NDES` mode to use challenge validation but allow automatic request creation by a user with enroll permissions, `Authorized` to allow enrollment by a challenge containing credentials of a user with enroll permissions. In this mode, you can generate the credentials in the appropriate format by clicking on the shield icon. You will be asked to enter a username and password, then hit the 'Generate' button to display and/or copy the payload in the clipboard. Select `Auto Validation` to use auto validation.
- **Enable DN Whitelist*** *(boolean)*:
  Tells whether the DN whitelist is enabled or not. The default value is set to false.
- **Terms of Service** *(string select)*:
  Select a previously created [admin-guide:system-terms_of_service:::_terms_of_service] entry. When set, the Terms of Service are displayed when issuing a challenge request and must be explicitly accepted by the requester before a challenge password is returned. Leave empty to keep the workflow unchanged.

SCEP protocol parameters

- **Mode*** *(select)*:
  Choose from the two modes RA or CA. The default value is set to RA.
- **SCEP Authority*** *(select)*:
  Select a previously created SCEP Authority.

- **CAPS*** *(select)*:
  Select a caps from the list. The default value is set to SHA.
- **Encryption algorithm*** *(select)*:
  Select an encryption algorithm from the list.
  - **Password policy** *(select)*:
    Select a previously created password policy. It is used for the challenge generation.

Max Certificate per Holder Policy

- **Maximum** *(int)*:
  When specified, define the maximum number of active certificates for a given holder.
- **Behavior** *(select)*:
  What behavior to have when the maximum number is reached:
  - **revoke** the previous certificates.
  - **reject** the current request.



In order to allow renewal in `reject` behavior, one more certificate is allowed when the certificate being renewed is in its renewal period.

- **Revocation reason** *(select)*:
  When the revoke behavior is selected, the revocation reason to revoke the certificate with.

Renewal management

- **Renewal period** *(finite duration)*:
  Must be a valid finite duration.

Crypto Policy

- **Default Key Type** *(select)*:
  Key Type that will be used by horizon-cli in certificate enrollment.
- **Authorized Key Types** *(multiselect)*:
  Key Types that can be used for enrollment. An empty value means no restrictions.

Common configuration for profiles

Languages

You can add more languages by clicking .

- **Language*** *(select)*:
  Select a language. Supported languages are:
  - **en**: English
  - **fr**: French
- **Display Name** *(string input)*:

Enter a display name. This will be the localized name of this profile.

- **Description** *(string input)*:
  Enter a description. This will be displayed on the list view of the profiles.

You can delete  the localization.

Grading Policies

You can select grading policies that will grade your certificate for a quick overview of its quality. For more information about the inner working of the grading policies in Horizon, please refer to the grading rules page.

Requests time to live

Configure the time your requests have before expiring.



After expiration, requests are stored for an additional 30 days. This can be changed using configuration files.

- **Enrollment request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Renewal request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Revocation request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Update request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Migration request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Recover request** *(finite duration)*:
  *Enabled on **escrow**:* Must be a valid finite duration. The default value is set to seven days.

Constraints

- **Allowed email domains** *(string input)*:
  Enter a valid regular expression that the inputted emails should match. This includes RFC822NAME and UPN SANs as well as the contact email

  This matches the domain of the email, not including anything before `@`.

- **Allowed DNS domains** *(string input)*:
  Enter a valid regular expression that the inputted domain should match.

CSR Data Mapping

1. Click on  to add a mapping.

2. Select a field and enter a value.

You can delete  the CSR Data Mapping.

Workflow

Auto-validation

Configure auto validation rules to avoid needing permissions configuration.



A request permission must be available in order for the request to be created and then auto-validated. See workflows to modify request permissions.

1. To enable auto-validation, switch the profile mode to `auto-validation`

2. Add rules that will be evaluated on each request. For more details, see the validation rules reference.

3. Add the threshold. This is the number of rules that must pass in order for the request to be validated.

Data source flow

Configure which data sources to execute and in which order.

1. Select a data source to execute first, and fill its inputs with a computation rule.

2. Add other data sources if needed. Each datasource input can use outputs from previously executed data sources.

3. All data sources output are available in computation rules throughout the certificate template and metadata.

Datasource options

Stop on success

When this option is enabled, if the datasource data is successfully fetched, the flow will stop immediately.

This allows to search through a list of datasource, when the data can be in any of multiple datasource.

Mandatory

When this option is enabled, if the datasource return no results, the flow will stop immediately with an error.

This allows to block enrollment flows if a required record is not present, event if is not saved into the final certificate.



A datasource is considered as finding no results if:

- LDAP Datasource: no record match the filter
- DNS Datasource: no record match the query
- REST Datasource: the result code is in the `notFoundHttpCodes`

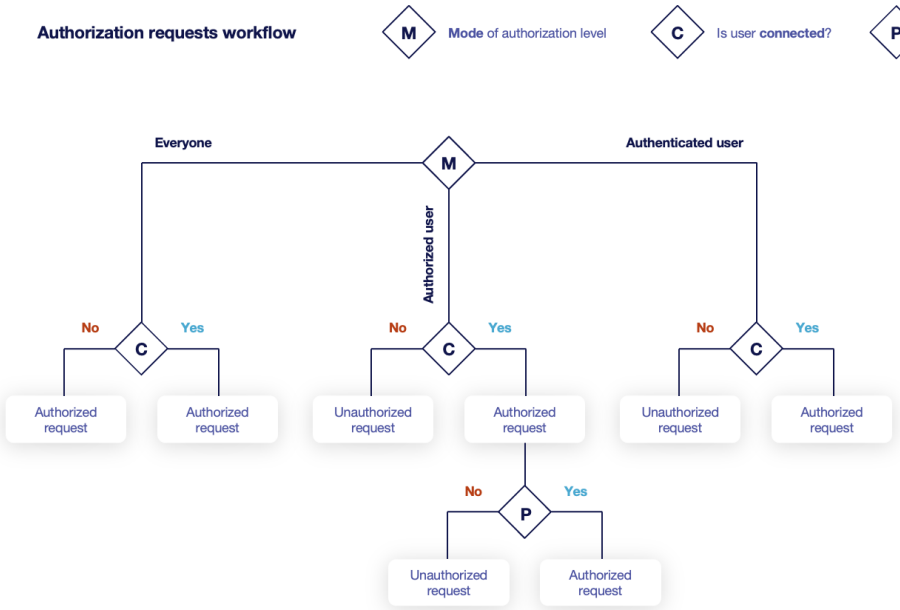


If a specific field from a record is considered as mandatory, including it in a mandatory label will also fail the enrollment flow if it is not found

Workflows builder

Configure custom rights for actions on this profile.

1. Select an authorization level for each workflow.



USER A : has workflow permission | USER B : has no workflow permission

Everyone		
USER A	not connected + no permissions	approve
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authenticated		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authorized		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	deny

- **Everyone**:
  No authentication is required.
- **Authenticated**:
  User has to be authenticated.
- **Authorized**:
  User has to be authenticated and have an explicit authorizations.

2. Select an access level for identity providers.

You can remove the access level for an identity provider by clicking on 'x'.

Owner-related permissions

These permissions apply to the owners of a certificate (team or owner). An owner can always request the following actions, but this permission allows them to perform the action without validation.

- **Revoke** *(boolean)*:
  Grant self revoke permission. The default value is set to false.
- **Revoke (pop)** *(boolean)*:
  Grant self revoke permission with owner being determined by Proof of Possession. This is used by horizon-cli. The default value is set to false.
- **Update** *(boolean)*:
  Grant self update permission. The default value is set to false.
- **Update (pop)** *(boolean)*:
  Grant self update permission with owner being determined by Proof of Possession. This is used by horizon-cli. The default value is set to false.

Certificate Template

This section details how to define a custom structure for the fields `subject DN`, `SAN` *&* `extensions` *of the requested certificate in order to match the configuration on the PKI side.*



Defining a template will use the CSR to fill the available field. A CSR with unexpected fields will be rejected. Using a template also disables CSR Data Mapping.

Subject DN composition

You can add more elements by clicking .

- **Element*** *(select)*:
  Select an attribute from the elements list.
- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the element.
- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the Subject DN template.



When a template is defined, at least one mandatory Common Name must be added to the DN Elements.

SAN composition

You can add more elements by clicking .

- **Element*** *(select)*:
  Select an attribute from the element list.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Minimum** *(int)*:
  The minimum number of value that this SAN must have.
- **Maximum** *(int)*:
  The maximum number of value that this SAN must have.
- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the SAN template.

Extensions

You can add more elements by clicking .

- **Element*** *(select)*:
  Select an attribute from the elements list.
- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:

Set a default value to the element.

- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button or reorder (drag and drop) the Extensions template.



When adding a SAN, a DN element or an Extension and making it mandatory, make sure to either give it a default value or a computation rule or make it editable, otherwise the template will be unusable.

Certificate Metadata

This section details how to define a custom structure for the labels, ownership policy and technical metadata, allowing certificates to hold rich information.

Labels

You can add more labels by clicking .

- **Name** *(select)*:
  Select a preexisting label.
- **Mandatory** *(boolean)*:
  Should the label be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the label should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the label should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the label.
- **Label value restriction**
  - **Whitelist** *(string input multiple)*:
    The label value will have to be in the whitelist. Open the popup, enter the label value and press "enter" to add this value to the accepted value list. An empty whitelist means no restriction.
  - **Suggestions** *(string input multiple)*:
    Add suggestions that will be displayed to the user. The user will be able to choose one of these values or enter its own. Open the popup, enter your suggestions and press enter to add this value to the suggestions. An empty suggestions list means no restriction.
  - **Regex** *(regex)*:
    The label value will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.

- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this label to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can delete  or reorder (drag and drop)  the label template.

Ownership policy

- **Owner**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's owner is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when approving a request.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the owner to the value of the evaluated computation rule. This value will override any other value including the user input.
- **Contact email**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's contact email is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when approving a request.
  - **Default contact email** *(string input)*:
    Set a default contact email. This value must comply with the contact email restriction.
  - **Contact email restriction**
    - **Whitelist** *(string input multiple)*:
      The contact email will have to be in the whitelist. Open the popup, enter the email and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The contact email will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the contact email to the value of the evaluated computation rule. This value

will override any other value including the user input and the default value.

- **Team**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's team is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when approving a request.
  - **Default team** *(string input)*:
    Set a default team. This value must comply with the team restriction.
  - **Team restriction**
    - **Whitelist** *(string input multiple)*:
      The team will have to be in the whitelist. Enter the team and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The team will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the team to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

Metadata policy *(overridable metadata)*



These metadata are technical metadata. They are used by Horizon or Third party connectors, updating them should be done with utmost care.



Metadata edition is not allowed on enroll.



Metadata edition is not available via the User Interface. It must be changed with API, using horizon-cli.

You can allow the override of technical metadata by clicking .

- **Metadata*** *(select)*:
  Select a metadata.
- **Editable by requester** *(boolean)*:
  Tells whether the metadata is editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the metadata is editable by the approver. The default value is set to false.

You can delete a metadata policy. This will not delete the metadata but will make it non editable.

Third-party policy

- **Synchronize data from discovery** *(boolean)*:
  Enable overriding of third-party data during discovery. When enabled, the following logic applies:
  - If a discovery attempts to set third-party data on a connector that is not referenced in the profile (by a trigger), the data will not be added.
  - If the connector is referenced in the profile, the discovery will override the existing data.

Notifications

This section details how to configure notifications on certificate and request lifecycle events.

Certificate lifecycle notifications

Notifications are sent when one of the following event is triggered by a certificate:

Enrollment	Revocation	Expire	Update	Migrate	Renew

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.

Request lifecycle notifications

Notifications are sent when one of the following event is triggered by an Enroll/Revocation/Update/Migrate/Renew request:

Submit	Cancel	Revoke	Approve	Pending

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.



Submit request events are not triggered when the user has the permission to perform the action directly.

5. Click on the save button.

You can edit , duplicate or delete the SCEP Profile.



You won't be able to delete a SCEP Profile if this one is referenced somewhere else.

2.10.6. WCCE

WCCE Introduction

This section details how to configure and consume the Windows Client Certificate Enrollment (WCCE) protocol.

Managing certificate lifecycle through the WCCE protocol involves up to three components:

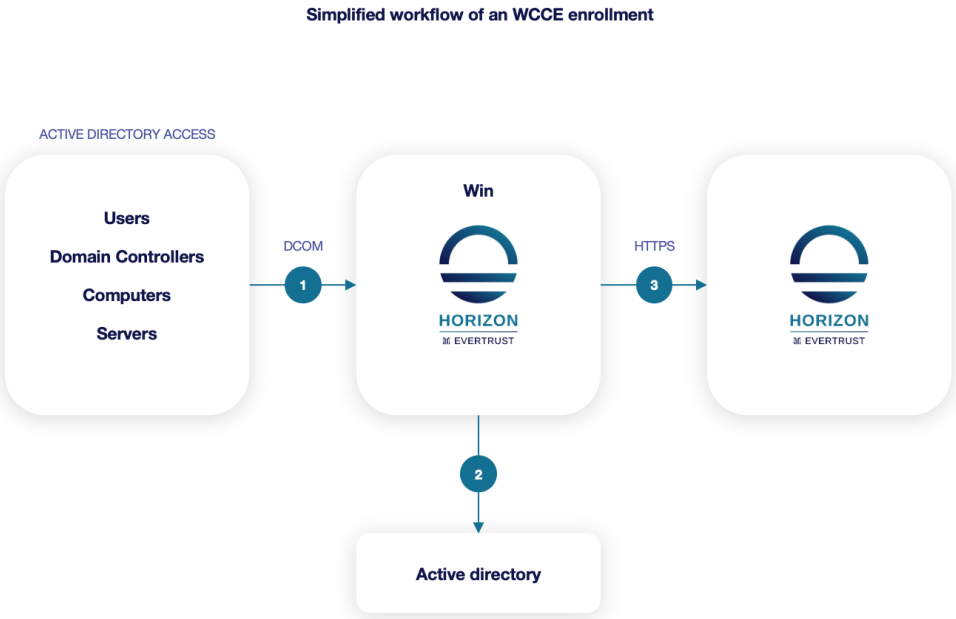
- Active Directory asset (domain controller, server, workstation, user) as WCCE Client;
- WinHorizon as the Active Directory enrollment service;
- Horizon as the WCCE proxy;



WCCE enrollment modes will be detailed later on.

The protocol paradigm can be described as follows: '**every Windows Active Directory member (machines, users) can use DCOM interfaces to interact with a CA to request certificate enrollment**'.

The following schema is a simplified workflow of an WCCE enrollment:



The protocol is based on the notion of Active Directory membership and configuration. Active Directory clients (such as machines and users) having rights on **Microsoft Certificate Templates** can use Active Directory **enrollment service** through DCOM interface to request certificate enrollment.

Horizon supports different WCCE enrollment modes:

- **Entity**: Certificate's elements are built using Active Directory content;
- **Enrollment On Behalf of Others (EOBO)**: Certificate signing request (CSR) is signed by

one/many Certificate Enrollment Agent(s);

- **Trust request**: Certificate signature request (CSR) content is fully trust and certificate will be created using its content.



For **Enrollment On Behalf of Others (EOBO)** enrollment mode, it is possible to configure a whitelist of **Authorized CAs** trusted as issuers of enrollment agent certificates.

Windows official resources

EverTrust WCCE implementation is based on official WCCE documentation provided by Microsoft:

- MS-WCCE: Windows Client Certificate Enrollment Protocol

Prerequisites

- WinHorizon should be installed using WinHorizon installation guide;
- WinHorizon and Active Directory should be configured using WinHorizon administration guide.

WCCE Forest

The first step is to register WCCE Forest on which you want to use WCCE protocol through Horizon.

Uses

MSAD Connector

How to configure WCCE Forest

1. Log in to Horizon Administration Interface.

2. Access WCCE Forest from the drawer or card: **Protocol** › **WCCE** › **Forest**.

3. Click on  .

4. Fill the mandatory fields.

- **Forest Name*** *(string input)*: Enter the Active Directory forest name.

5. Click on the save button.

You can duplicate  or delete  the WCCE Forest.



You won't be able to delete an WCCE Forest if it is referenced somewhere else.

WCCE Profile

*The second step details how to create and configure a WCCE Horizon profile. This profile is an **internal** Horizon profile.*

Uses

[admin-guide:protocols-wcce-wcce_template:::_wcce_template_mapping]	WCCE Scheduled Task

Prerequisites

PKI Connector

How to configure a WCCE Profile

1. Log in to Horizon Administration Interface.

2. Access WCCE Profile from the drawer or card: **Protocol** › **WCCE** › **Profiles**.

3. Click on ⊕ .

4. Fill the mandatory fields.

WCCE Profile Specific Configuration

General

- **Name*** *(string input)*:
  Enter a meaningful profile name. It must be unique for each profile. Horizon uses the name to identify the profile. As the name will be part of an URL, it is advisable to use only lower case letters and dashes.
- **Enable*** *(boolean)*:
  Indicates whether the profile is enabled or not. The default value is set to true.
- **PKI Connector** *(string select)*:
  Select a PKI connector previously created.
- **Exchange certificate*** *(select)*:

  *Enabled on **escrow**:* Select a preexisting Exchange Certificate or create one with the ⊕ .

Crypto Policy

- **Authorized Key Types** *(multiselect)*:
  Key Types that can be used for enrollment. An empty value means no restrictions.
- **Private key escrowing** *(boolean)*:
  Tells whether the private key should be escrowed by Horizon. The default value is set to false.



This can only be enabled using an Evertrust Stream PKI connector.

- **Show PKCS#12 Password On Recover** *(boolean)*:
  Tells whether the PKCS#12 password should be displayed on recover. The default value is set to false.
- **Show PKCS#12 On Recover** *(boolean)*:
  Tells whether the PKCS#12 should be displayed on recover. The default value is set to false.
- **PKCS#12 Password Mode*** *(select)*:
  Select how to generate PKCS#12 password:
  - **manual**: prompt the user to choose its password. This is the default behavior.
  - **random**: have the password generated on Horizon side.
- **Password policy** *(select)*:
  Select a previously created password policy. It will be enforced on PKCS#12 password for recovery and centralized enrollments.
- **Store encryption type*** *(select)*:
  Select an encryption algorithm from the list. The PKCS#12 will use this algorithm. The default value is set to DES_AVERAGE.

Max Certificate per Holder Policy

- **Maximum** *(int)*:
  When specified, define the maximum number of active certificates for a given holder.
- **Behavior** *(select)*:
  What behavior to have when the maximum number is reached:
  - **revoke** the previous certificates.
  - **reject** the current request.



In order to allow renewal in `reject` behavior, one more certificate is allowed when the certificate being renewed is in its renewal period.

- **Revocation reason** *(select)*:
  When the revoke behavior is selected, the revocation reason to revoke the certificate with.

Common configuration for profiles

Languages

You can add more languages by clicking .

- **Language*** *(select)*:
  Select a language. Supported languages are:
  - **en**: English
  - **fr**: French

- **Display Name** *(string input)*:
  Enter a display name. This will be the localized name of this profile.
- **Description** *(string input)*:
  Enter a description. This will be displayed on the list view of the profiles.

You can delete the localization.

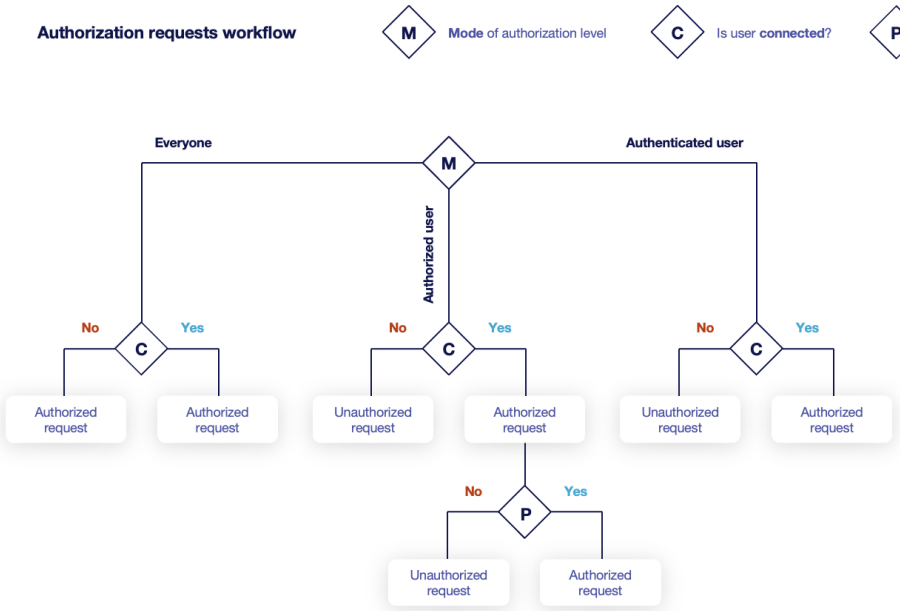
Grading Policies

You can select grading policies that will grade your certificate for a quick overview of its quality. For more information about the inner working of the grading policies in Horizon, please refer to the grading rules page.

Workflows builder

Configure custom rights for actions on this profile.

1. Select an authorization level for each workflow.



USER A : has workflow permission | USER B : has no workflow permission

Everyone		
USER A	not connected + no permissions	approve
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authenticated		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authorized		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	deny

- **Everyone**:
  No authentication is required.
- **Authenticated**:
  User has to be authenticated.
- **Authorized**:
  User has to be authenticated and have an explicit authorizations.

2. Select an access level for identity providers.

You can remove the access level for an identity provider by clicking on 'x'.

Requests time to live

Configure the time your requests have before expiring.



After expiration, requests are stored for an additional 30 days. This can be changed using configuration files.

- **Enrollment request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Renewal request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Revocation request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Update request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Migration request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Recover request** *(finite duration)*:
  *Enabled on **escrow**:* Must be a valid finite duration. The default value is set to seven days.

Owner-related permissions

These permissions apply to the owners of a certificate (team or owner). An owner can always request the following actions, but this permission allows them to perform the action without validation.

- **Revoke** *(boolean)*:
  Grant self revoke permission. The default value is set to false.
- **Recover** *(boolean)*:
  Grant self recover permission. The default value is set to false.
- **Update** *(boolean)*:
  Grant self update permission. The default value is set to false.

Constraints

- **Allowed email domains** *(string input)*:
  Enter a valid regular expression that the inputted emails should match. This includes RFC822NAME and UPN SANs as well as the contact email

  This matches the domain of the email, not including anything before `@`.

- **Allowed DNS domains** *(string input)*:
  Enter a valid regular expression that the inputted domain should match.

CSR Data Mapping

1. Click on  to add a mapping.

2. Select a field and enter a value.

You can delete  the CSR Data Mapping.

Certificate Template

This section details how to define a custom structure for the fields `subject DN`, `SAN` & `extensions` of the requested certificate in order to match the configuration on the PKI side.



Defining a template will use the CSR to fill the available field. A CSR with unexpected fields will be rejected. Using a template also disables CSR Data Mapping.

Subject DN composition

You can add more elements by clicking .

- **Element*** *(select)*:
  Select an attribute from the elements list.
- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the element.
- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the Subject DN template.



When a template is defined, at least one mandatory Common Name must be added to the DN Elements.

SAN composition

You can add more elements by clicking  .

- **Element*** *(select)*:
  Select an attribute from the element list.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Minimum** *(int)*:
  The minimum number of value that this SAN must have.
- **Maximum** *(int)*:
  The maximum number of value that this SAN must have.
- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the SAN template.

Extensions

You can add more elements by clicking  .

- **Element*** *(select)*:
  Select an attribute from the elements list.
- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the element.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the Extensions template.



When adding a SAN, a DN element or an Extension and making it mandatory, make sure to either give it a default value or a computation rule or make it editable, otherwise the template will be unusable.

Certificate Metadata

This section details how to define a custom structure for the labels, ownership policy and technical metadata, allowing certificates to hold rich information.

Labels

You can add more labels by clicking  .

- **Name** *(select)*:
  Select a preexisting label.
- **Mandatory** *(boolean)*:
  Should the label be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the label should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the label should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the label.
- **Label value restriction**
  - **Whitelist** *(string input multiple)*:
    The label value will have to be in the whitelist. Open the popup, enter the label value and press "enter" to add this value to the accepted value list. An empty whitelist means no restriction.
  - **Suggestions** *(string input multiple)*:
    Add suggestions that will be displayed to the user. The user will be able to choose one of these values or enter its own. Open the popup, enter your suggestions and press enter to add this value to the suggestions. An empty suggestions list means no restriction.
  - **Regex** *(regex)*:
    The label value will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this label to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can delete  or reorder (drag and drop)  the label template.

Ownership policy

- **Owner**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's owner is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when approving a request.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the owner to the value of the evaluated computation rule. This value will override any other value including the user input.
- **Contact email**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's contact email is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when approving a request.
  - **Default contact email** *(string input)*:
    Set a default contact email. This value must comply with the contact email restriction.
  - **Contact email restriction**
    - **Whitelist** *(string input multiple)*:
      The contact email will have to be in the whitelist. Open the popup, enter the email and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The contact email will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the contact email to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.
- **Team**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's team is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when submitting a

request.

- **Editable by approver** *(boolean)*:
  Specify if the certificate's team can be overridden by the requester when approving a request.
- **Default team** *(string input)*:
  Set a default team. This value must comply with the team restriction.
- **Team restriction**
  - **Whitelist** *(string input multiple)*:
    The team will have to be in the whitelist. Enter the team and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
  - **Regex** *(regex)*:
    The team will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of the team to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

Metadata policy *(overridable metadata)*



These metadata are technical metadata. They are used by Horizon or Third party connectors, updating them should be done with utmost care.



Metadata edition is not allowed on enroll.



Metadata edition is not available via the User Interface. It must be changed with API, using horizon-cli.

You can allow the override of technical metadata by clicking .

- **Metadata*** *(select)*:
  Select a metadata.
- **Editable by requester** *(boolean)*:
  Tells whether the metadata is editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the metadata is editable by the approver. The default value is set to false.

You can delete  a metadata policy. This will not delete the metadata but will make it non editable.

Third-party policy

- **Synchronize data from discovery** *(boolean)*:
  Enable overriding of third-party data during discovery. When enabled, the following logic

applies:

- If a discovery attempts to set third-party data on a connector that is not referenced in the profile (by a trigger), the data will not be added.
- If the connector is referenced in the profile, the discovery will override the existing data.

Notifications/Triggers

This section details how to configure notifications and triggers to perform actions on certificate and request lifecycle events.

Certificate lifecycle notifications

Notifications are sent when one of the following event is triggered by a certificate:

Enrollment	Revocation	Expire	Update	Migrate	Renew

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.

Request lifecycle notifications

Notifications are sent when one of the following event is triggered by an Enroll/Revocation/Update/Migrate/Renew request:

Submit	Cancel	Revoke	Approve	Pending

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.



Submit request events are not triggered when the user has the permission to perform the action directly.

Triggers

Horizon support the use of third-party triggers in the form of callbacks on specific events happening on the profile, giving a way to synchronize the third party repositories and Horizon.

- **Enrollment** *(select)*:
  Select the preexisting third party or MDM trigger(s) to call whenever a certificate is enrolled on this profile.
- **Renewal** *(select)*:
  Select the preexisting third party or MDM trigger(s) to call whenever a certificate is renewed on this profile.
- **Revocation** *(select)*:
  Select the preexisting third party or MDM trigger(s) to call whenever a certificate gets revoked on this profile.
- **Expire** *(select)*:

Select the preexisting third party or MDM trigger(s) to call whenever a certificate expires on this profile.

The available triggers are the following:

AKV Triggers	AWS Triggers	F5 Triggers	F5 AS3 Triggers
Netscaler Triggers	GCM Triggers	[admin-guide:third-parties-fortinet-triggers:::_fortinet_triggers]	[admin-guide:third-parties-paloalto-triggers:::_paloalto_triggers]
[admin-guide:third-parties-ldap-triggers:::_ldap_triggers]	Intune PKCS Triggers		

5. Click on the save button.

You can edit  , duplicate  or delete  the WCCE Profile .



You won't be able to delete a WCCE Profile if this one is referenced somewhere else.

WCCE Template Mapping

The third and last step is to configure mapping between Microsoft Certificate Template configured on Active Directory and Horizon WCCE profile. A mapping is created using a specific enrollment mode. As a result of this mapping, every Microsoft Certificate Template can issue certificate from different PKI (using PKI connector of WCCE profile associated to Microsoft Certificate Template).

Prerequisites

[admin-guide:protocols-wcce-wcce_forest:::_wcce_forest]	[admin-guide:protocols-wcce-wcce_profile:::_wcce_profile]

How to configure WCCE Template Mapping

1. Log in to Horizon Administration Interface.

2. Access WCCE Forest from the drawer or card: **Protocol** › **WCCE** › **Forest**.

3. Identify the section corresponds to the forest for which you want to add mapping. Click on + button.

4. Fill the mandatory fields.

- **Microsoft Template Name*** *(string input)*:
  Enter the Microsoft Certificate Template name created on Active Directory side.

- **Microsoft Template version** *(select)*:
  Select a version for the Microsoft template.
- **Enrollment mode** *(select)*:
  Specify the enrollment mode of this mapping.
- **EOBO CAs** *(select)*:
  Specify the CA(s) to use for EOBO enrolment.
- **Profile*** *(select)*:
  Select a previously created WCCE profile.

5. Click on the save button.

You can edit  or delete the WCCE Template mapping.

WCCE Test enrollment

This section details how to use the Microsoft Management Console (MMC) to manually retrieve a certificate through WCCE using different enrollment modes. If you want to enroll **machine certificate** you need to perform the following actions using Administrator Account.

1. Launch `mmc.exe`

2. Click on **File** > **Add/Remove or Remove Snap-ins**

3. On the left panel, click on **Certificates** then **Add**



If you don't have administrative privileges, the **User certificate store** will be automatically chosen. If your account has administrative privileges, it will be prompted a window to choose Microsoft Certificate Store to use. If you want to enroll **User** certificate please chose **My user account**. If you want to enroll **Machine** certificate (computer or IIS for example) please chose **Computer account**.

4. Navigate to **Personal** › **Certificates**

5. Right click on Windows and chose **All tasks** › **Request certificate**

6. Click on **Next**

7. On the next step, let default enrollment policy configuration, then click on **Next**

The next step lists all Microsoft Certificate Templates on which you have enrollment rights. The Microsoft Certificate template selection and last parts of this testing procedure are specific to the enrollment mode you want to perform.

Please refer to the proper section below.

Requesting a certificate using 'Entity' enrollment mode

8. Select the Microsoft Certificate Template configured on Horizon side as a part of a **Template Mapping** using **Entity** enrollment mode. Click on **Next**

9. Click on **Enroll** to request Enrollment.

10. Enrollment is requested to WinHorizon. Few seconds later, if enrollment is successful it will be displayed **STATUS: Succeeded**. Click on **Finish**.

11. Your certificate is displayed and available.

Requesting a certificate using 'Enrollment On Behalf of Others' enrollment mode

8. Identify the Microsoft Certificate Template configured on Horizon side as a part of a **Template Mapping** using **Enrollment On Behalf of Others (EOBO)** enrollment mode. Click on **Details** then **Properties**.

9. Navigate to **Extensions** tab and select **Enrollment Agent Certificate** (to be used to sign Certificate Request). Click on **OK**.

10. Click on **Enroll** to request Enrollment.

11. Enrollment is requested to WinHorizon. Few seconds later, if enrollment is successful it will be displayed **STATUS: Succeeded**. Click on **Finish**.

12. Your certificate is displayed and available.

Requesting a certificate using 'Trust request' enrollment mode

8. Identify the Microsoft Certificate Template configured on Horizon side as a part of a **Template Mapping** using **Trust request** enrollment mode. Click on **Details** then **Properties**.

9. Navigate to **Subject** tab to build your Certificate request manually. Click on **OK**.

10. Click on **Enroll** to request Enrollment.

11. Enrollment is requested to WinHorizon. Few seconds later, if enrollment is successful it will be displayed **STATUS: Succeeded**. Click on **Finish**.

12. Your certificate is displayed and available.

WCCE MSAD Connector

This section details how to to configure the Microsoft Active Directory Connectors.

Uses

WCCE Scheduled Task

Prerequisites

[admin-guide:protocols-wcce-wcce_forest:::_wcce_forest]

How to configure an MSAD Connector

1. Log in to Horizon Administration Interface.

2. Access MSAD Connectors from the drawer or card: **Protocol** › **WCCE** › **MSAD Connectors**.

3. Click on  .

4. Fill in the mandatory fields.

General

- **Name*** *(select)*:
  Select the Active Directory Forrest you want to use to set up the connector.
- **Hostname*** *(string input)*:
  DNS name or IP of the Active Directory domain.
- **Port** *(string input)*:
  Port to connect to the Active Directory. The default value is set to 636.
- **Proxy** *(string select)*:
  Select a proxy to connect to the Active Directory, if needed.
- **LDAP Credentials*** *(select)*:
  Select `Login` credentials containing the DN and password of the Active Directory account. Must have right privileges to browse and list objects.
- **Timeout*** *(finite duration)*:
  The time before Horizon stop trying to connect to Active Directory. Must be a valid finite duration.
- **Max stored certificate per holder** *(int)*:
  When specified, define the maximum number of active certificates for a given Holder.

Assets identification

- **Base DN*** *(string input)*:
  It can be the root of your domain or a restriction.
- **LDAP Filter** *(string input)*:
  This filter must respect LDAP filter syntax.

Actor management

These configuration elements mainly define the number of authorized interactions with the remote service on a defined period. For example, one needs to ensure that the remote service will not be requested more than 5 times per 3 seconds. *Throttle parallelism* defines the number of times and *Throttle duration* the period of time. Therefore, on the above example, throttle parallelism would be

set to 5 and throttle duration would be set to 3 seconds.

- **Throttle duration*** *(finite duration)*:
  The default value is set to 3 seconds. Must be a valid finite duration.
- **Throttle parallelism*** *(int)*:
  The default value is set to 3.

5. Click on the save button.

You can update  or delete  the MSAD Connector.



You won't be able to delete a MSAD Connector if this one is referenced somewhere else.

WCCE Scheduled Tasks

This section details how to schedule tasks that will run periodically on your WCCE profiles. You will be able to use MSAD Connector to browse Active Directory and retrieve changes (basically computer removal) to trigger certificate revocation. This mechanism works using comparison between Active Directory content (using MSAD connector) and Horizon certificate list based on a specific WCCE profile. If Horizon has a certificate for a holder that does not exist on Active Directory side a revocation will be triggered automatically.

Prerequisites

[admin-guide:protocols-wcce-wcce_forest:::_wcce_forest]	[admin-guide:protocols-wcce-wcce_profile:::wcce_profile]	[admin-guide:protocols-wcce-wcce_msad_connector:::_wcce_msad_connector]

How to configure WCCE Scheduled Tasks

1. Log in to Horizon Administration Interface.

2. Access WCCE scheduled tasks from the drawer or card: **Protocol › WCCE › Scheduled Tasks**.

3. Click on .

4. Fill the mandatory fields.

- **WCCE Profile*** *(select)*:
  Select the target WCCE profile.
- **Target Connector*** *(select)*:
  Select the MSAD connector to use as **golden source** of active Active Directory objects.
- **Cron scheduling in Quartz format** *(cron expression)*:
  Enter a Cron scheduling expression (in Quartz format). Default value is every 5 hours.
- **Revoke** *(boolean)*:

If true, will revoke all certificate that do not exist on the AD side.

- **Dry run** *(boolean)*:
  If enabled, revocation actions will not be performed. Instead, a message will be logged, explaining what would have been done.

5. Click on the save button.

You can run  , update  or delete  the Schedules Tasks.

2.10.7. WebRA

WebRA Introduction

WebRA is a powerful protocol designed by EverTrust. It allows a validation process with edition of all certificate fields, to perform enrollments with user friendly web interfaces on Horizon Registration Authority portal.

WebRA Profile

This section details how to configure the WebRA Profile.

Required By

WebRA Scheduled Task

Prerequisites

PKI Connector

How to configure WebRA Profile

1. Log in to Horizon Administration Interface.

2. Access WebRA Profiles from the drawer or card: **Protocols** › **WebRA** › **Profiles**.

3. Click on  .

4. Fill in the mandatory fields.

Profile specific configuration

General

- **Name*** *(string input)*:
  Enter a meaningful profile name, this setting will be the profile identifier. It must be unique for each profile.

- **Enable** *(boolean)*:
  Should the profile be enabled. The default value is set to true.
- **PKI Connector** *(string select)*:
  Select a previously created PKI connector.
- **Authorization Mode*** *(select)*:
  This concern enrollment requests only: Select `Authorized` to use the configured permissions, `Auto Validation` to use auto validation, `Auto Validation → Authorized` to use auto validation then fallback on the configured permissions, or `Challenge` to authorize enrollments with a one-time challenge (see Challenge mode).
- **Password policy*** *(select)*:
  Available only if the authorization mode is `Challenge`. Select a password policy previously created. It is used for the challenge generation.
- **Terms of Service** *(string select)*:
  Select a previously created [admin-guide:system-terms_of_service:::_terms_of_service] entry. When set, the Terms of Service are displayed during the WebRA enrollment workflow and must be explicitly accepted by the requester before the request can be submitted. Leave empty to keep the workflow unchanged.

Crypto Policy

- **Default Key Type** *(select)*:
  Select the default type of key to generate when using centralized enrollment mode.
- **Authorized Key Types** *(multiselect)*:
  Key Types that can be used for enrollment. An empty value means no restrictions.
- **Centralized enrollment** *(boolean)*:
  Tells whether the profile should be used with a centralized enrollment, i.e providing a PKCS#12. The default value is set to false.
  - **Private key escrowing** *(boolean)*:
    Tells whether the private key should be escrowed by Horizon. The default value is set to false.
    - **Show PKCS#12 Password On Recover** *(boolean)*:
      Tells whether the PKCS#12 password should be displayed on recover. The default value is set to false.
    - **Show PKCS#12 On Recover** *(boolean)*:
      Tells whether the PKCS#12 should be displayed on recover. The default value is set to false.
  - **Show PKCS#12 Password On Enroll** *(boolean)*:
    Tells whether the PKCS#12 password should be displayed on enroll. The default value is set to false.
  - **Show PKCS#12 On Enroll** *(boolean)*:
    Tells whether the PKCS#12 should be displayed on enroll. The default value is set to false.
  - **PKCS#12 Password Mode*** *(select)*:
    Select how to generate PKCS#12 password:

        - **manual**: prompt the user to choose its password. This is the default behavior.
        - **random**: have the password generated on Horizon side.
    - **Password policy** *(select)*:
      Select a previously created password policy. It will be enforced on PKCS#12 password for recovery and centralized enrollments.
    - **Store encryption type*** *(select)*:
      Select an encryption algorithm from the list. The PKCS#12 will use this algorithm. The default value is set to DES_AVERAGE.
    - **Transient key lifetime** *(finite duration)*:
      Gives the retention period for non-escrowed keys. During this period, triggers using the key can be retried.
- **Decentralized enrollment** *(boolean)*:
  Tells whether the profile should be used with a decentralized enrollment mode, i.e CSR (PKCS#10) signing by the PKI. The default value is set to true.

Max Certificate per Holder Policy

- **Maximum** *(int)*:
  When specified, define the maximum number of active certificates for a given holder.
- **Behavior** *(select)*:
  What behavior to have when the maximum number is reached:
    - **revoke** the previous certificates.
    - **reject** the current request.



In order to allow renewal in `reject` behavior, one more certificate is allowed when the certificate being renewed is in its renewal period.

- **Revocation reason** *(select)*:
  When the revoke behavior is selected, the revocation reason to revoke the certificate with.

Renewal

- **Renewal period** *(finite duration)*:
  Period before expiration during which a certificate is eligible for renewal. Must be a valid finite duration. Required to enable automatic renewal.

- **Automatic renewal** *(boolean)*:
  When enabled, certificates issued on this profile can be renewed automatically by a background process once they enter their renewal period.
    - **Default value** *(boolean)*:
      Default value of the `autoRenew` flag for certificates issued through this profile. The flag is set at enrollment, migration and import time. Existing certificates are not changed.
    - **Editable** *(boolean)*:
      Tells whether the `autoRenew` flag can be modified on individual requests or certificates.

When set to false, the **Default value** is enforced. The default value is set to false.



Renewing certificates centrally requires the profile to support centralized enrollment with PKCS#12 parameters. The Crypto Policy section should be configured accordingly so that the renewal background process can produce a usable key pair.



If the renewal period is greater than or equal to the certificate lifetime issued by the underlying PKI, every certificate will be considered eligible for renewal and will be renewed in a loop. Use the `reject` Max certificates per holder policy to guard against runaway renewals.



- Disabling **Automatic renewal** on a profile that previously had it enabled unsets the `autoRenew` flag from all certificates issued through this profile. A `CONFIGURATION` event is emitted to record the number of certificates impacted.
- Enabling **Automatic renewal** and setting a default value to enable will enable `autoRenew` on all certificates issued through this profile. To enable auto renew on new certificates only, first enable auto renew on the profile with `default: false`, and then update the profile again to change the default value.

Common configuration for profiles

Languages

You can add more languages by clicking  .

- **Language*** *(select)*:
  Select a language. Supported languages are:
  - **en**: English
  - **fr**: French
- **Display Name** *(string input)*:
  Enter a display name. This will be the localized name of this profile.
- **Description** *(string input)*:
  Enter a description. This will be displayed on the list view of the profiles.

You can delete  the localization.

Grading Policies

You can select grading policies that will grade your certificate for a quick overview of its quality. For more information about the inner working of the grading policies in Horizon, please refer to the grading rules page.

Requests time to live

Configure the time your requests have before expiring.



After expiration, requests are stored for an additional 30 days. This can be changed using configuration files.

- **Enrollment request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Renewal request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Revocation request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Update request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Migration request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Recover request** *(finite duration)*:
  *Enabled on **escrow**:* Must be a valid finite duration. The default value is set to seven days.

Constraints

- **Allowed email domains** *(string input)*:
  Enter a valid regular expression that the inputted emails should match. This includes RFC822NAME and UPN SANs as well as the contact email

  This matches the domain of the email, not including anything before `@`.

- **Allowed DNS domains** *(string input)*:
  Enter a valid regular expression that the inputted domain should match.



On WebRA profiles, only the constraints set on the profile apply. The constraints defined in the Horizon configuration are not enforced on WebRA.

CSR Data Mapping

1. Click on  to add a mapping.

2. Select a field and enter a value.

You can delete  the CSR Data Mapping.

Workflow

Auto-validation

Configure auto validation rules to avoid needing permissions configuration.



A request permission must be available in order for the request to be created and then auto-validated. See workflows to modify request permissions.

1. To enable auto-validation, switch the profile mode to `auto-validation`

2. Add rules that will be evaluated on each request. For more details, see the validation rules reference.

3. Add the threshold. This is the number of rules that must pass in order for the request to be validated.

Data source flow

Configure which data sources to execute and in which order.

1. Select a data source to execute first, and fill its inputs with a computation rule.

2. Add other data sources if needed. Each datasource input can use outputs from previously executed data sources.

3. All data sources output are available in computation rules throughout the certificate template and metadata.

Datasource options

Stop on success

When this option is enabled, if the datasource data is successfully fetched, the flow will stop immediately.

This allows to search through a list of datasource, when the data can be in any of multiple datasource.

Mandatory

When this option is enabled, if the datasource return no results, the flow will stop immediately with an error.

This allows to block enrollment flows if a required record is not present, event if is not saved into the final certificate.



A datasource is considered as finding no results if:

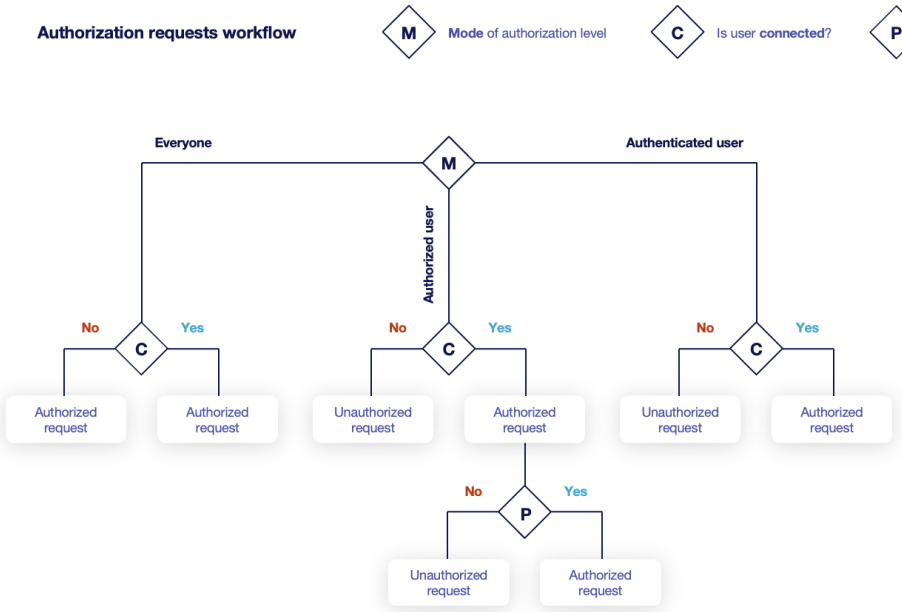
- LDAP Datasource: no record match the filter
- DNS Datasource: no record match the query
- REST Datasource: the result code is in the `notFoundHttpCodes`

If a specific field from a record is considered as mandatory, including it in a mandatory label will also fail the enrollment flow if it is not found

Workflows builder

Configure custom rights for actions on this profile.

1. Select an authorization level for each workflow.



USER A : has workflow permission | USER B : has no workflow permission

Everyone		
USER A	not connected + no permissions	approve
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authenticated		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authorized		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	deny

- **Everyone**:
  No authentication is required.
- **Authenticated**:
  User has to be authenticated.
- **Authorized**:
  User has to be authenticated and have an explicit authorizations.

2. Select an access level for identity providers.

You can remove the access level for an identity provider by clicking on 'x'.

Owner-related permissions

These permissions apply to the owners of a certificate (team or owner). An owner can always request the following actions, but this permission allows them to perform the action without validation.

- **Revoke** *(boolean)*:
  Grant self revoke permission. The default value is set to false.
- **Revoke (pop)** *(boolean)*:
  Grant self revoke permission with owner being determined by Proof of Possession. This is used

by horizon-cli. The default value is set to false.

- **Recover** *(boolean)*:
  Grant self recover permission. The default value is set to false.
- **Update** *(boolean)*:
  Grant self update permission. The default value is set to false.
- **Update (pop)** *(boolean)*:
  Grant self update permission with owner being determined by Proof of Possession. This is used by horizon-cli. The default value is set to false.
- **Renew** *(boolean)*:
  Grant self renew permission. The default value is set to false.
- **Renew (pop)** *(boolean)*:
  Grant self renew permission with owner being determined by Proof of Possession. This is used by horizon-cli. The default value is set to false.

Certificate Template

This section details how to define a custom structure for the fields `subject DN`, `SAN` *&* `extensions` *of the requested certificate in order to match the configuration on the PKI side.*



In a WebRA profile, defining a template is mandatory, unless the authorization mode is `Challenge`.

Subject DN composition

You can add more elements by clicking .

- **Element*** *(select)*:
  Select an attribute from the elements list.
- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the element.
- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the

Subject DN template.



When a template is defined, at least one mandatory Common Name must be added to the DN Elements.

SAN composition

You can add more elements by clicking .

- **Element*** *(select)*:
  Select an attribute from the element list.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Minimum** *(int)*:
  The minimum number of value that this SAN must have.
- **Maximum** *(int)*:
  The maximum number of value that this SAN must have.
- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the SAN template.

Extensions

You can add more elements by clicking .

- **Element*** *(select)*:
  Select an attribute from the elements list.
- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the element.

- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button or reorder (drag and drop) the Extensions template.



When adding a SAN, a DN element or an Extension and making it mandatory, make sure to either give it a default value or a computation rule or make it editable, otherwise the template will be unusable.

Certificate Metadata

This section details how to define a custom structure for the labels, ownership policy and technical metadata, allowing certificates to hold rich information.

Labels

You can add more labels by clicking .

- **Name** *(select)*:
  Select a preexisting label.
- **Mandatory** *(boolean)*:
  Should the label be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the label should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the label should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the label.
- **Label value restriction**
  - **Whitelist** *(string input multiple)*:
    The label value will have to be in the whitelist. Open the popup, enter the label value and press "enter" to add this value to the accepted value list. An empty whitelist means no restriction.
  - **Suggestions** *(string input multiple)*:
    Add suggestions that will be displayed to the user. The user will be able to choose one of these values or enter its own. Open the popup, enter your suggestions and press enter to add this value to the suggestions. An empty suggestions list means no restriction.
  - **Regex** *(regex)*:
    The label value will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this label to the value of the evaluated computation rule. This value will

override any other value including the user input and the default value.

You can delete  or reorder (drag and drop)  the label template.

Ownership policy

- **Owner**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's owner is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when approving a request.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the owner to the value of the evaluated computation rule. This value will override any other value including the user input.
- **Contact email**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's contact email is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when approving a request.
  - **Default contact email** *(string input)*:
    Set a default contact email. This value must comply with the contact email restriction.
  - **Contact email restriction**
    - **Whitelist** *(string input multiple)*:
      The contact email will have to be in the whitelist. Open the popup, enter the email and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The contact email will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the contact email to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.
- **Team**

- **Mandatory** *(boolean)*:
  Specify if the certificate's team is mandatory when submitting a request.
- **Editable by requester** *(boolean)*:
  Specify if the certificate's team can be overridden by the requester when submitting a request.
- **Editable by approver** *(boolean)*:
  Specify if the certificate's team can be overridden by the requester when approving a request.
- **Default team** *(string input)*:
  Set a default team. This value must comply with the team restriction.
- **Team restriction**
  - **Whitelist** *(string input multiple)*:
    The team will have to be in the whitelist. Enter the team and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
  - **Regex** *(regex)*:
    The team will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of the team to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

Metadata policy *(overridable metadata)*



These metadata are technical metadata. They are used by Horizon or Third party connectors, updating them should be done with utmost care.



Metadata edition is not allowed on enroll.



Metadata edition is not available via the User Interface. It must be changed with API, using horizon-cli.

You can allow the override of technical metadata by clicking .

- **Metadata*** *(select)*:
  Select a metadata.
- **Editable by requester** *(boolean)*:
  Tells whether the metadata is editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the metadata is editable by the approver. The default value is set to false.

You can delete  a metadata policy. This will not delete the metadata but will make it non editable.

Third-party policy

- **Synchronize data from discovery** *(boolean)*:
  Enable overriding of third-party data during discovery. When enabled, the following logic applies:
  - If a discovery attempts to set third-party data on a connector that is not referenced in the profile (by a trigger), the data will not be added.
  - If the connector is referenced in the profile, the discovery will override the existing data.

Notifications/Triggers

This section details how to configure notifications and triggers to perform actions on certificate and request lifecycle events.

Certificate lifecycle notifications

Notifications are sent when one of the following event is triggered by a certificate:

Enrollment	Revocation	Expire	Update	Migrate	Renew

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.

Request lifecycle notifications

Notifications are sent when one of the following event is triggered by an Enroll/Revocation/Update/Migrate/Renew request:

Submit	Cancel	Revoke	Approve	Pending

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.



Submit request events are not triggered when the user has the permission to perform the action directly.

Triggers

Horizon support the use of third-party triggers in the form of callbacks on specific events happening on the profile, giving a way to synchronize the third party repositories and Horizon.

- **Enrollment** *(select)*:
  Select the preexisting third party or MDM trigger(s) to call whenever a certificate is enrolled on this profile.
- **Renewal** *(select)*:
  Select the preexisting third party or MDM trigger(s) to call whenever a certificate is renewed on this profile.

- **Revocation** *(select)*:
  Select the preexisting third party or MDM trigger(s) to call whenever a certificate gets revoked on this profile.
- **Expire** *(select)*:
  Select the preexisting third party or MDM trigger(s) to call whenever a certificate expires on this profile.

The available triggers are the following:

AKV Triggers	AWS Triggers	F5 Triggers	F5 AS3 Triggers
Netscaler Triggers	GCM Triggers	[admin-guide:third-parties-fortinet-triggers:::_fortinet_triggers]	[admin-guide:third-parties-paloalto-triggers:::_paloalto_triggers]
[admin-guide:third-parties-ldap-triggers:::_ldap_triggers]	Intune PKCS Triggers		

5. Click on the save button.

You can edit , duplicate  or delete  the WebRA Profile.



You won't be able to delete a WebRA Profile if it is referenced somewhere else.

Challenge mode

When a WebRA profile is in `Challenge` authorization mode, an enrollment is authorized only if the request carries a valid challenge issued for this profile, as with the SCEP and EST challenge modes. A challenge can be used only once, and is rejected once expired.

Challenges are requested either from the Registration Authority Interface (see Requesting a WebRA challenge) or through the WebRA API, for instance by Horizon CLI.

In this mode, the certificate template may be left empty: the client then supplies the certificate identity when consuming the challenge. Identity fields defined in the template are enforced and cannot be overridden by the client.



When the certificate template is empty, the profile Constraints are the only restriction on the identity a client can request with a challenge. Set **Allowed DNS domains** and **Allowed email domains** accordingly.

WebRA Scheduled Tasks

This section details how to schedule tasks that will run periodically with your WebRA profiles.

Prerequisites

[admin-guide:third-parties-aws-connector:::_aws_connector]	[admin-guide:third-parties-akv-connector:::_azure_akv_connector]	[admin-guide:third-parties-f5-connector:::_f5_connector]	[admin-guide:third-parties-gcm-connector:::_gcm_connector]	[admin-guide:protocols-webra-webra_profile:::_webra_profile]

How to configure WebRA Scheduled Tasks

1. Log in to Horizon Administration Interface.

2. Access the "Scheduled tasks" from the drawer or card: **Protocols** › **WebRA** › **Scheduled Tasks**.

3. Click on  .

4. Fill in the mandatory fields.

- **WebRA Profile*** *(select)*:
  Select a previously created WebRA profile.
- **Target Connector*** *(select)*:
  Select a previously created third party connector.
- **Cron scheduling** *(cron expression)*:
  Enter a Cron scheduling expression (in Quartz format). The default expression is built to run the task every 5 hours.
- **Revoke** *(boolean)*:
  If enabled, will revoke all certificate whose container was deleted from the third party repository. The default value is set to false.
- **Dry run** *(boolean)*:
  If enabled, revocation actions will not be performed. Instead, a message will be logged, explaining what would have been done.

5. Click on the save button.

You can run  or edit  or delete  the Schedules Tasks.

2.10.8. Auto Validation

Auto-validation can be enabled on the following protocols:

- WebRA
- EST
- SCEP

When the auto-validation mode is enabled, Validation Rules are evaluated to allow or deny an enrollment request.

Multiple rules can be defined on each profile, and the minimum number of passing rules can be defined.

Validation Rules

A validation rule is a condition that can be true or false. Inputs are taken from dictionary entries and can be manipulated using Computation Rules and validation functions and operators.

For example, to allow all requests coming from a subnet and having all DNS SANs that resolves on the Horizon server, the following rule can be used:

```
{{http.request.ip}} in 154.12.45.0/24 and [[admin-guide:protocols-autovalidation:::csr.san.dnsname]] resolvesDNS
```

Here, two expressions are combined using the `and` operator:

- `{{http.request.ip}} in 154.12.45.0/24`: a computation rule `{{http.request.ip}}` fetches the value of the IP from the incoming request, and checks that this IP is `in` the `154.12.45.0/24` subnet, using the In operator.
- `[[admin-guide:protocols-autovalidation:::csr.san.dnsname]] resolvesDNS` : a computation rule `[[admin-guide:protocols-autovalidation:::csr.san.dnsname]]` fetches the values of the DNS SANs from the csr in the incoming request, and checks that these SANs `[.orange`#resolvesDNS#].

Examples

DNS Validation

To validate that all DNS requested in a WebRA enroll request resolve on the DNS Server with IP 192.10.132.2, the following rule can be used:

```
[[admin-guide:protocols-autovalidation:::webra.enroll.san.dnsname]] resolvesDNS(192.10.132.2:53)
```

Here all the dns sans from the request are fetched and are submitted to the dns server.

Email validation

To validate that the Email SAN requested in an EST enroll request are associated to the requester's Common Name, using a datasource that fetches the emails on an external LDAP server, the following rule can be used:

```
{{csr.san.rfc822name.1}} = {{ds.1.1.mail}}
```

Here the condition checks if the first Email SAN from the CSR is equal to the mail fetched from the LDAP datasource (supposing the LDAP datasource is the first in the flow).

Quick Reference

The table below lists the possible operators for a validation rule:

Operator Name	Syntax
And	`expression` and `expression`
Or	`expression` or `expression`
Equals	`expression` equals `expression`
In	`expression` in `expression`
Exists	`expression` exists
Contains	`expression` contains `expression`
Matches	`expression` matches `expression`
Within	`expression` within `expression`
Is Empty	`expression` is empty
Starts With	`expression` starts with `expression`
Ends With	`expression` ends with `expression`
Resolves DNS	`expression` resolvesDNS

And

```
left:<expression> and right:<expression>
```

This outputs the logical and operation on the result evaluated from `left` and `right`

```
"left" = "left" and "right"="right" => true
Upper("left") = "left" and "right"="right" => false
```

Or

```
left:<expression> or right:<expression>
```

This outputs the logical or operation on the result evaluated from `left` and `right`

```
"left" = "left" or "right"="right" => true
Upper("left") = "left" or "right"="right" => true
```

Equals

```
left:<expression> equals right:<expression>
left:<expression> = right:<expression>
```

This tests the equality operation on the result evaluated from `left` and `right`

```
"left" = "left"  => true
Upper("left") equals "left" => false
```

Not Equals

```
left:<expression> not equals right:<expression>
left:<expression> != right:<expression>
```

```
"left" != "left"  => false
Upper("left") not equals "left" => true
```

In

Element inclusion

```
elem:<single expression> in array:<multi expression>
```

This tests if `elem` is contained in `array`

```
"left" in [ "left" ]  => true
Upper("left") in ["left"] => false
```

Multiple Element inclusion

```
any of elems:<multi expression> in array:<multi expression>
all of elems:<multi expression> in array:<multi expression>
```

This tests if `all` or `any` element in `elems` is contained in `array`

```
any of ["left", "right"] in [ "left" ]  => true
all of ["left", "right"] in [ "left" ]  => false
```

IP in CIDR

```
ip:<single expression> in subnet:<subnet>
```

This tests if `ip` is contained in `subnet` (cidr notation)

```
"128.12.13.14" in 128.12.15.0/24  => false
```

```
"2001:0db8:85a3:0000:0000:0000:0000:0001" in 2001:db8:85a3::8a2e:370:7334/64 => true
```

Multiple IPs in CIDR

```
any of ips:<multi expression> in subnet:<subnet>
all of ips:<multi expression> in subnet:<subnet>
```

This tests if `all` or `any` IP in `ips` is contained in `subnet` (cidr notation)

```
any of ["128.12.13.14", "128.12.15.32" ] in 128.12.15.0/24  => true
all of [ "2001:0db8:85a3:0000:0000:0000:0000:0001",
"2002:0db8:85a3:0000:0000:0000:0000:0001"] in 2001:db8:85a3::8a2e:370:7334/64 => false
```

Element not included

```
elem:<single expression> not in array:<multi expression>
any of elems:<multi expression> not in array:<multi expression>
all of elems:<multi expression> not in array:<multi expression>
```

```
"left" not in ["right"]  => true
any of ["left", "right"] not in [ "left" ] => true
all of ["left", "right"] not in [ "left" ] => false
"128.12.13.14" not in 128.12.15.0/24  => true
```

Exists

```
elem:<expression> exists
```

This tests if `elem` exists. For single values, this will be true if the dictionary key is defined and for multi values if the array is not empty.

```
"" exists  => true
{{will.not.exist}} exists => false
[[will.not.exist]] exists => false
```

Not exists

```
elem:<expression> not exists
```

```
"" not exists  => false
```

```
{{will.not.exist}} not exists => true
[[will.not.exist]] not exists => true
```

Is Empty

```
elem:<expression> is empty
```

This tests if `elem` is empty. For single values, this will be true if the dictionary key is not defined or if the value is empty, and for multi values if the array is empty or if all values are empty.

```
"" is empty  => true
{{will.not.exist}} is empty => true
[[will.not.exist]] is empty => true
[ "", ""] is empty => true
```

Is Not empty

```
elem:<expression> is not empty
```

```
"" is not empty  => false
{{will.not.exist}} is not empty => false
[[will.not.exist]] is not empty => false
```

Contains

Single element contained

```
containing:<expression> contains elem:<single expression>
```

This tests if `containing` contains the `elem` value. If `containing` is a string, the presence of the substring elem is checked, if it is an array the presence of the element in the array is checked.

```
"google.com" contains "google"  => true
["google.com", "google.fr"] contains "google.fr"  => true
"google.com" contains {{does.not.exist}}  => true
```

Multiple elements contained

```
containing:<multi expression> contains all of elems:<multi expression>
containing:<multi expression> contains any of elems:<multi expression>
```

This tests if `containing` contains `all` or `any` of the elements of the `elems` array. An empty `elems` array will always be contained.

```
["google.com", "google.fr"] contains all of ["test.com"]  => false
["google.com"] contains all of [[does.not.exist]]  => true
["google.com", "google.fr"] contains any of ["google.com", "a"]  => true
```

Single element not contained

```
containing:<expression> not contains elem:<single expression>
```

```
"google.com" not contains ""  => false
```

Multiple elements not contained

```
containing:<multi expression> not contains all of elems:<multi expression>
containing:<multi expression> not contains any of elems:<multi expression>
```

```
["test", "abc" ] not contains all of ["abc", "d"] => true
["test", "abcf", "d"] not contains any of ["f", "e"] => true
```

Matches

Single element match

```
elem:<single expression> matches regex:<single expression>
elem:<single expression> ~ regex:<single expression>
```

This tests if `elem` matches the `regex`. If `regex` is None, this will output false.

```
"left" matches "\d+"  => false
Upper("left") ~ "[A-Z]+" => true
```

Multiple element match

```
any of elems:<multi expression> matches regex:<single expression>
all of elems:<multi expression> matches regex:<single expression>
```

This tests if `any` or `all` element in `elems` matches the `regex`. If `regex` is None, this will output false.

```
any of ["left", "42"] matches "\d+"  => true
all of Upper(["left", "42"]) matches "[A-Z]+" => false
```

Not matching

```
elem:<single expression> not matches regex:<single expression>
any of elems:<multi expression> not matches regex:<single expression>
all of elems:<multi expression> not matches regex:<single expression>
```

```
"left" not matches "\d+" => true
any of ["left", "aaaaa"] not matches "a+"  => true
all of ["left", "aaaaa"] not matches "a+"  => false
```

Within

Element matching

```
elem:<single expression> within array:<multi expression>
```

This tests if `elem` matches a regex in `array`.

```
"left" within [ "\d+", "[a-z]+" ]  => true
Upper("left") within [ "\d+", "[a-z]+" ] => false
```

Multiple Element matching

```
any of elems:<multi expression> within array:<multi expression>
all of elems:<multi expression> within array:<multi expression>
```

This tests if `all` or `any` element in `elems` matches a regex in `array`

```
any of ["left", "aaaaa"] within [ "\d+", "a+" ]  => true
all of ["left", "aaaaa"] within [ "\d+", "a+" ]  => false
```

Element not matching

```
elem:<single expression> not within array:<multi expression>
any of elems:<multi expression> not within array:<multi expression>
all of elems:<multi expression> not within array:<multi expression>
```

```
"left" not within [ "\d+", "[a-z]+" ]  => false
any of ["left", "aaaaa"] not within [ "\d+", "a+" ]  => true
all of ["left", "aaaaa"] not within [ "\d+", "a+" ]  => false
```

Starts With

Element matching

```
elem:<single expression> starts with start:<single expression>
```

This tests if `elem` starts with `start` value. An empty `elem` will send return false.

```
"left" starts with "le"  => true
Upper("left") starts with "le" => false
```

Multiple Element matching

```
any of elems:<multi expression> starts with start:<single expression>
all of elems:<multi expression> starts with start:<single expression>
```

This tests if `all` or `any` element in `elems` starts with `start`

```
any of ["left", "aaaaa"] starts with "aaa" => true
all of ["left", "aaaaa"] starts with "aaa" => false
```

Element not matching

```
elem:<single expression> starts not with start:<single expression>
any of elems:<multi expression> starts not with start:<single expression>
all of elems:<multi expression> starts not with start:<single expression>
```

```
"left" starts not with "le"  => false
any of ["left", "aaaaa"] starts not with "a"  => true
all of ["left", "aaaaa"] starts not with "a"  => false
```

Ends With

Element matching

```
elem:<single expression> ends with start:<single expression>
```

This tests if `elem` ends with `start` value. An empty `elem` will send return false.

```
"left" ends with "ft"  => true
Upper("left") ends with "ft" => false
```

Multiple Element matching

```
any of elems:<multi expression> ends with start:<single expression>
all of elems:<multi expression> ends with start:<single expression>
```

This tests if `all` or `any` element in `elems` ends with `start`

```
any of ["left", "aaaaa"] ends with "aaa" => true
all of ["left", "aaaaa"] ends with "aaa" => false
```

Element not matching

```
elem:<single expression> ends not with start:<single expression>
any of elems:<multi expression> ends not with start:<single expression>
all of elems:<multi expression> ends not with start:<single expression>
```

```
"left" ends not with "ft"  => false
any of ["left", "aaaaa"] ends not with "a"  => true
all of ["left", "aaaaa"] ends not with "a"  => false
```

Resolves DNS

```
host:<expression> resolvesDNS
host:<expression> resolvesDNS(101.12.13.14:53)
```

This tests if `host` resolves on the DNS server. An optional DNS server in the `ip:port` format can be used. If `host` is an array, DNS must resolve for each value. An empty array returns `false`.

```
"google.com" resolvesDNS  => true
["google.com", "google.fr"] resolvesDNS  => true
["google.com", "not.resolving"] resolvesDNS  => false
```

Not Resolves DNS

```
host:<expression> not resolvesDNS
```

```
host:<expression> not resolvesDNS(101.12.13.14:53)
```

```
"google.com" not resolvesDNS  => false
["google.com", "google.fr"] not resolvesDNS  => false
```

2.10.9. URL Parameters

On the ACME, EST and SCEP protocols, EVERTRUST has designed a way to add certificate metadata such as labels, ownership and technical metadata.

This syntax works by editing the Horizon profile name to provide these metadata.

The possible items are the following:

- team
- owner
- mail
- label.<label name>
- metadata.<technical metadata type>

These items value can be given by adding `:` and then the value.

These can be added to the profile name following a `~`, as follows:

`<profile name>~<metadata 1>,<metadata 2>`

For example, to add:

- the label `my-label` with value `test-label`
- the owner with value `my-owner`

to the following EST endpoint for profile `est-profile`: `https://horizon.evertrust/.well-known/est/est-profile/cacerts`

The new endpoint is: `https://horizon.evertrust/.well-known/est/est-profile~my-label:test-label,owner:my-owner/cacerts`



Base64 encoding for the metadata values is also allowed. For the above example, this would make the new name `est-profile~bXktbGFiZWw6dGVzdC1sYWJlbCxvd25lcjpteS1vd25lcgo=`



URLs are transmitted in plaintext when using TLS versions prior to 1.3, exposing this information to potential interception

2.11. Datasources

2.11.1. Datasource Introduction

Datasources are external assets that contain data useful for certificate enrollment or enrollment validation.

Datasources are fetched on enrollment request **submission** and are used to fill the request dictionary. This dictionary is then available to use in computation rules and validation rules.

To define which datasources to fetch, and with which parameters, a **datasource flow** is available on all certificate profiles. A Datasource Flow is a collection of datasources to fetch, with inputs from the request.



Keys fetched from datasources are prefixed with `ds.<i>`, `i` being the index of the datasource in the flow (starting from 1).

2.11.2. DNS Datasource

This section details how to configure a DNS datasource.

How to configure DNS datasource

1. Log in to Horizon Administration Interface.

2. Access Datasources from the drawer or card: **Datasources**.

3. Click on  .

4. Select **DNS Type**

5. Fill in the mandatory fields.

Datasource specific configuration

General

- **Name*** *(string input)*:
  Enter a meaningful datasource name, this setting will be the datasource identifier. It must be unique for each datasource.
- **Description** *(string input)*:
  Enter a description to describe this datasource usage.

DNS Parameters

- **Hostname and port***: *(select & string input)*
  Choose the hostname and port of your DNS server.

- **Record types** *(select string)*:
  Choose the record types to fetch. If none are selected, all record types are fetched.
- **Lookup*** *(string input)*:
  Lookup to query. It is a template string and can contain keys for parametrization.
- **Timeout** *(finite duration)*:
  Set by default at 10 seconds. Must be a valid finite duration.

2.11.3. LDAP Datasource

This section details how to configure a LDAP datasource.

How to configure LDAP datasource

1. Log in to Horizon Administration Interface.

2. Access Datasources from the drawer or card: **Datasources**.

3. Click on  .

4. Select **LDAP Type**

5. Fill in the mandatory fields.

Datasource specific configuration

General

- **Name*** *(string input)*:
  Enter a meaningful datasource name, this setting will be the datasource identifier. It must be unique for each datasource.
- **Description** *(string input)*:
  Enter a description to describe this datasource usage.

LDAP Parameters

- **Hostname*** *(string input)*:
  Enter the URL pointing to LDAP.
- **Port** *(int)*:
  Enter the port where to reach the running LDAP instance (default values are 389 for LDAP and 636 for LDAPS).
- **LDAP Credentials*** *(select)*:
  Select `Login` credentials containing the technical user created for Horizon login DN and password.
- **Base DN*** *(string input)*:
  Enter the Base DN where Horizon should publish the certificate. It is a template string and can contain keys for parametrization.

- **Filter** *(string input)*:
  Enter the custom filter. It is a template string and can contain keys for parametrization.
- **Proxy** *(string select)*:
  The HTTP/HTTPS proxy used to reach LDAP, if any.
- **Timeout** *(finite duration)*:
  Set by default at 10 seconds. Must be a valid finite duration.
- **Follow referrals** *(boolean)*:
  Allow publication to follow LDAP referrals.
- **Secure** *(boolean)*:
  Only use secure LDAP connection
- **Disable hostname validation** *(boolean)*:
  Allow non validated hostname during LDAP connection
- **Limit*** *(integer)*:
  Maximum number of results for the LDAP query

2.11.4. REST Datasource

This section details how to configure a REST datasource.

How to configure REST datasource

1. Log in to Horizon Administration Interface.

2. Access Datasources from the drawer or card: **Datasources**.

3. Click on  .

4. Select **REST Type**

5. Fill in the mandatory fields.

Datasource specific configuration

General

- **Name*** *(string input)*:
  Enter a meaningful datasource name, this setting will be the datasource identifier. It must be unique for each datasource.
- **Description** *(string input)*:
  Enter a description to describe this datasource usage.

REST Parameters

- **HTTP Method and URL***: *(select & string input)*
  Choose the HTTP method and the destination URL for your notification. The URL is a template string and can contain keys for parametrization.

- **Proxy**: *(select)*
  Define a proxy for this REST API call.
- **Timeout*** *(finite duration)*:
  Connection timeout when executing the REST API call. Must be a valid finite duration.
- **Accepted response HTTP code*** *(multiselect | input)*:
  Response codes meaning the REST call was a success. If another one is received, a failure will be logged.
- **Authentication type and credentials*** *(select & select)*:
  Choose the authentication type and the credentials to perform the authentication. Custom authentication allows the credentials values to be accessible in headers.
- **Headers** *(input string & input string)*:
  Choose the header name and value. Header values are template strings and can contain keys for parametrization.
- **Body*** *(string input)*:
  Enter the REST body. It is a template string and can contain keys for parametrization.

2.12. Third parties

2.12.1. AWS

AWS Introduction

This section refers to the AWS Certificate Manager (ACM) integration with Horizon, used to enroll certificates held in ACM.

This integration involves at least two infrastructure components:

- AWS Certificate Manager
- EverTrust Horizon

AWS Connector

Here is the section to manage the AWS Connector.

Required By

[admin-guide:third-parties-aws-triggers:::_aws_trigger]

Prerequisites

On Horizon side, you might need to set up a Proxy , used to reach AWS, if necessary.

On AWS side, you need to create a user using the AWS IAM module, and following AWS guide. You should create an access key for that user, and give him appropriate permissions. The created user should hold the following permissions:

- `AWSResourceGroupsReadOnlyAccess`
- `ResourceGroupsandTagEditorReadOnlyAccess`
- `AWSCertificateManagerFullAccess`

After performing these steps, you will get the following information, required later:

- the AWS Region
- the User Access Key ID
- the User Access Key Secret

On top of that, you need to define a Resource Group, using AWS Resource Groups and Tags Editor, with the following characteristics:

- Group Type: Tag based
- Resource Type: `AWS::CertificateManager::Certificate`
- Tag key and value (e.g. key=manage and value=HRZ)

After performing this steps, you will get the following information, required later:

- The Resource Group name
- the Tag name
- the Tag value

How to configure AWS Connector

1. Log in to Horizon Administration Interface.

2. Access AWS Connectors from the drawer or card: **Third Parties › AWS › Connectors**.

3. Click on  .

4. Fill the mandatory fields.

Connection

- **Name*** *(string input)*:
  Enter a meaningful connector name. It must be unique for each connector. Horizon uses the name to identify the connector.
- **Region*** *(string input)*:
  Enter a valid AWS region. Here's the region list from AWS.
- **AWS Access Key Credentials** *(select)*:
  Select `Login` credentials containing the User Access Key ID and secret used by Horizon to connect to AWS.
- **Proxy** *(string select)*:
  The HTTP/HTTPS proxy to use to reach AWS, if any.

- **Timeout*** *(finite duration)*:
  The timeout for Horizon-initiated connections to AWS. Must be a valid finite duration.

Assets identification

- **Resource group name** *(string input)*:
  Name of the resource group pointing to the tag name and value.
- **Role ARN** *(string input)*:
  Name of the AWS role Horizon will impersonate in ACM.
- **Tag key** *(string input)*:
  Name of the tag used to identify certificates managed by Horizon in ACM.
- **Tag value** *(string input)*:
  Value of the tag used to identify certificates managed by Horizon in ACM.

Actors and renewal management

- **Throttle duration*** *(finite duration)*:
  Set by default at 3 seconds. Must be a valid finite duration.

5. Click on the save button.

You can update  or delete  the AWS Connector.



You won't be able to delete an AWS Connector if it is referenced somewhere else.

Synchronize your third party

Your third-party certificates can be synchronized with Horizon using scheduled tasks.

Scheduled tasks are a functionality of WebRA that allows to synchronize automatic renewal or revocation events with a third party periodically with what occurs on a WebRA profile. To be more specific, it will periodically check whether the certificate has entered the "renewal period" that was defined in the connector's configuration, and renew it automatically if necessary.

1. Refer to the third party connector documentation to create a third party connector.

2. Ensure you have an existing [admin-guide:protocols-webra-webra_profile:::_webra_profile]: renewal will be automated on the selected profile.

3. Follow the documentation of the [admin-guide:protocols-webra-webra_schedule_tasks:::_webra_scheduled_tasks] section to properly configure a scheduled task.

Connector actions

The following actions are available once the connector is configured

-  **Retry failed triggers**: This will run every retryable trigger linked to this connector. A

trigger is retryable if it failed and all required parameters for it (like the private key) are still available.



This action could trigger a lot of retries.

AWS Trigger

Here is the section to manage the Triggers that will be used by Profiles to push or delete certificates to/from AWS ACM.

Prerequisites

[admin-guide:third-parties-aws-connector:::_aws_connector]

How to configure AWS Trigger

1. Log in to Horizon Administration Interface.

2. Access AWS Triggers from the drawer or card: **Third Parties** › **AWS** › **Triggers**.

3. Click on  .

4. Fill the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful trigger name. It must be unique for each trigger. Horizon uses the name to identify the trigger.
- **AWS Connector*** *(select)*:
  Select an AWS connector previously created.
- **Retries in case of error** *(int)*:
  Number of times to retry to push the change on the AWS repository in case of error. Must be an integer between 1 and 15.

5. Click on the save button.

You can update  or delete  the AWS Trigger.



You won't be able to delete an AWS Trigger if it is referenced somewhere else.

Synchronization using triggers

Triggers are a functionality of WebRA, EST, Intune PKCS, WCCE and CRMP profiles that allows to push lifecycle events into a third party whenever they occur on a profile.

1. Refer to the trigger documentation to create a trigger.

2. Create or modify the profile you wish to use the triggers on.

3. Go to the **Triggers** tab, then on **Certificate lifecycle triggers**

4. Chose which lifecycle events you wish to use triggers upon (enrollment, revocation, expiration)

5. Select one or more existing triggers from the menu (if several are selected, they will all be called whenever the selected event occurs)

6. Click on the Save button.

From now on, whenever a selected lifecycle event will occur on the configured profile, the trigger will be called and the and the certificate will be pushed into or removed from the third party container.

2.12.2. AKV

AKV Introduction

This section refers to the Azure Key Vault (AKV) integration with Horizon, used to enroll certificates held in AKV.

This integration involves at least three infrastructure components:

- Azure Key Vault
- Azure Active Directory
- EverTrust Horizon

Azure AD is used to authenticate Horizon, which should be a registered application.

Azure AKV Connector

Here is the section to manage the Azure AKV Connector.

Required By

Azure AKV Trigger

Prerequisites

On Horizon side, you might need to set up a Proxy used to reach Azure, if necessary.

On Azure AD side, it is required to set up an application by following Microsoft's guide.



Horizon supports only client secret authentication

After performing these steps, you will get the following information, required later:

- the Tenant ID
- the Application ID

- the Application Authentication Key

Finally, you should give all Certificate Permissions to the Application you created for Horizon inside the target Azure Key Vault "Access policies" menu entry, using the "Add Access Policy" link.

How to configure AKV Connector

1. Log in to Horizon Administration Interface.

2. Access AKV Connectors from the drawer or card: **Third Parties › AKV › Connectors**.

3. Click on  .

4. Fill the mandatory fields.

Connection

- **Name*** *(string input)*:
  Enter a meaningful Connector Name.
- **Azure Tenant*** *(string input)*:
  Enter the Tenant, which is the domain name after the @ sign in your account.
- **App Registration Credentials*** *(select)*:
  Select `Login` credentials containing your app registration ID and secret key.
- **Proxy** *(string select)*:
  The HTTP/HTTPS proxy used to reach Azure AD and AKV, if necessary.
- **Timeout** *(finite duration)*:
  Set on the connections used to reach Azure AD and AKV. Configured by default at 10 seconds. Must be a valid finite duration.
- **Vault fully qualified domain name*** *(string input)*:
  Fully qualified domain name used to reach the Azure Key Vault to be managed by Horizon.

Assets identification and management

- **Prefix** *(string input)*: Used to filter the certificates managed by Horizon in the specified Azure Key Vault. Defaults to "HRZ-"
- **With chain** *(boolean)*:
  If enabled, the CA chain is imported in the Key Vault certificate along with the leaf certificate and its private key. Disabled by default.

Actors and renewal management

These configuration elements mainly define the number of authorized interactions with the remote service on a defined period. For example, one needs to ensure that the remote service will not be contacted more than 5 times per 3 seconds. *Throttle parallelism* defines the number of times and *Throttle duration* the period of time. Therefore, on the above example, throttle parallelism would be set to 5 and throttle duration would be set to 3 seconds.

- **Throttle duration*** *(finite duration)*:
  Set by default at 3 seconds. Must be a valid finite duration.
- **Throttle parallelism*** *(int)*:
  Set by default at 3.

5. Click on the save button.

You can update  or delete  the AKV Connector.



You will not be able to delete an AKV Connector if it is referenced in any other configuration element.

Synchronize your third party

Your third-party certificates can be synchronized with Horizon using scheduled tasks.

Scheduled tasks are a functionality of WebRA that allows to synchronize automatic renewal or revocation events with a third party periodically with what occurs on a WebRA profile. To be more specific, it will periodically check whether the certificate has entered the "renewal period" that was defined in the connector's configuration, and renew it automatically if necessary.

1. Refer to the third party connector documentation to create a third party connector.

2. Ensure you have an existing [admin-guide:protocols-webra-webra_profile:::_webra_profile]: renewal will be automated on the selected profile.

3. Follow the documentation of the [admin-guide:protocols-webra-webra_schedule_tasks:::_webra_scheduled_tasks] section to properly configure a scheduled task.

Connector actions

The following actions are available once the connector is configured

-  **Retry failed triggers**: This will run every retryable trigger linked to this connector. A trigger is retryable if it failed and all required parameters for it (like the private key) are still available.



This action could trigger a lot of retries.

AKV Trigger

This section details how to configure the Triggers that will be used by Profiles to push or delete certificates to/from AKV.

Prerequisites

[admin-guide:third-parties-akv-connector:::_azure_akv_connector]

How to configure AKV Trigger

1. Log in to Horizon Administration Interface.

2. Access AKV Triggers from the drawer or card: **Third Parties › AKV › Triggers**.

3. Click on  .

4. Fill the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful trigger name. It must be unique for each trigger. Horizon uses the name to identify the trigger.
- **Azure Key Vault Connector*** *(select)*:
  Select an AKV connector previously created.
- **Retries in case of error** *(int)*:
  Number of times to retry to push the change on the AKV repository in case of error. Must be an integer between 1 and 15.

5. Click on the save button.

You can update  or delete  the AKV Trigger.

Synchronization using triggers

Triggers are a functionality of WebRA, EST, Intune PKCS, WCCE and CRMP profiles that allows to push lifecycle events into a third party whenever they occur on a profile.

1. Refer to the trigger documentation to create a trigger.

2. Create or modify the profile you wish to use the triggers on.

3. Go to the **Triggers** tab, then on **Certificate lifecycle triggers**

4. Chose which lifecycle events you wish to use triggers upon (enrollment, revocation, expiration)

5. Select one or more existing triggers from the menu (if several are selected, they will all be called whenever the selected event occurs)

6. Click on the Save button.

From now on, whenever a selected lifecycle event will occur on the configured profile, the trigger will be called and the and the certificate will be pushed into or removed from the third party container.

2.12.3. F5

F5 Introduction

This section refers to the F5 BigIP integration with Horizon, used to enroll certificates used by F5 BigIP.

This integration involves at least two infrastructure components:

- F5 BigIP
- EverTrust Horizon

Horizon connects to the F5 BigIP using the iControl REST administration API in order to manage the lifecycle of certificates associated to Client SSL Profiles within the BigIP.

F5 Connector

This section details how to configure the F5 Connector.

Required By

[admin-guide:third-parties-f5-triggers:::_f5_trigger]

Prerequisites

On the F5 BigIP side, you need to create a technical user for Horizon, and give it `resource-admin` access on the partitions you wish to manage. This is required because only `resource-admin` have the right to upload certificates on an F5 BigIP.

After performing these steps, you will get the following information, required later:

- the technical user login/username
- the technical user password

How to configure F5 Connector

1. Log in to Horizon Administration Interface.

2. Access F5 Connectors from the drawer or card: **Third Parties › F5 › Connectors**.

3. Click on  .

4. Fill the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful connector name. It must be unique for each connector. Horizon uses the name to identify the connector.
- **F5 BigIP hostname*** *(string input)*:

Enter the F5 BigIP hostname (DNS or IP address).

- **F5 BigIP credentials*** *(select)*:
  Select `Login` credentials containing the username and password created for Horizon in the F5 BigIP. Must have administrator rights.
- **F5 Login Provider** *(string input)*:
  Login provider to use in TACACS authentication mode. `tmos` for example
- **Proxy** *(string select)*:
  The HTTP/HTTPS proxy to use.
- **Timeout** *(finite duration)*:
  Set by default at 10 seconds. Must be a valid finite duration.
- **Max stored certificates per holder** *(int)*:
  When specified, define the maximum number of certificates stored in the third party for a given holder.
- **TLS Insecure** *(boolean)*:
  If enabled, TLS validation will ignore expired, invalid or untrusted certificates.



This is not recommended for production usage

Assets identification

- **Partition** *(string input)*:
  F5 BigIP partition to manage. `Common` by default.
- **SSL parent** *(string input)*:
  Name of the parent Client SSL Profile. `clientssl` by default. A short name such as `clientssl` refers to a profile located in the managed **Partition**. To use a profile located in the `Common` partition, such as the standard F5 profile, use the `/Common/<name>` syntax, for example `/Common/clientssl`.
- **Prefix** *(string input)*:
  Used to filter the certificates managed by Horizon in the specified F5 Client. `hrz-` by default.
- **Cipher group** *(string input)*:
  Name of the Cipher group. `None` by default. As for the **SSL parent**, a short name refers to the managed **Partition**, and a cipher group located in the `Common` partition is referenced with the `/Common/<name>` syntax. When no cipher group is set, the Client SSL Profiles created by Horizon inherit the ciphers and the cipher group of their **SSL parent**. This allows using a parent profile with TLS 1.3 enabled without setting a cipher group on the connector.
- **Version** *(string input)*:
  Major version of the F5 BigIp instance. For a F5 instance in `15.1.10` use `15.0.0` for example. `13.0.0` by default.
- **Override profile configuration** *(boolean)*:
  If enabled, the SSL Parent and the Cipher Group will be overridden when updating a client profile. `true` by default.
- **Persist configuration** *(boolean)*:
  If enabled, Horizon saves the F5 BigIP configuration to `bigip.conf` after each successful

deployment, so that pushed certificates are not lost if the appliance restarts before its scheduled configuration save. If the save fails, the deployment is reported as failed. `false` by default.



Saving the configuration requires an administrator-level F5 BigIP account.



The CA chain objects pushed to the F5 BigIP are named after the **Prefix**, followed by the leaf certificate name when the F5 object name length allows it, and end with `<chain hash>-CHAIN.crt`. CA chains pushed by earlier versions under the hash-only name are left in place, and the client SSL profiles managed by Horizon are bound to the renamed chain on their next push.

Actors and renewal management

These configuration elements mainly define the number of authorized interactions with the remote service on a defined period. For example, one needs to ensure that the remote service will not be contacted more than 5 times per 3 seconds. *Throttle parallelism* defines the number of times and *Throttle duration* the period of time. Therefore, on the above example, throttle parallelism would be set to 5 and throttle duration would be set to 3 seconds.

- **Throttle duration*** *(finite duration)*:
  Set by default at 3 seconds. Must be a valid finite duration.
- **Throttle parallelism*** *(int)*:
  Set by default at 3.

5. Click on the save button.

You can update  or delete  the F5 Connector.



You will not be able to delete an F5 Connector if it is referenced in any other configuration element.

Synchronize your third party

Your third-party certificates can be synchronized with Horizon using scheduled tasks.

Scheduled tasks are a functionality of WebRA that allows to synchronize automatic renewal or revocation events with a third party periodically with what occurs on a WebRA profile. To be more specific, it will periodically check whether the certificate has entered the "renewal period" that was defined in the connector's configuration, and renew it automatically if necessary.

1. Refer to the third party connector documentation to create a third party connector.

2. Ensure you have an existing [admin-guide:protocols-webra-webra_profile:::_webra_profile]: renewal will be automated on the selected profile.

3. Follow the documentation of the [admin-guide:protocols-webra-webra_schedule_tasks:::_webra_scheduled_tasks] section to properly configure a scheduled task.

F5 Trigger

This section details how to configure the Triggers that will be used by Profiles to push or delete certificates to/from F5 BigIP.

Prerequisites

[admin-guide:third-parties-f5-connector:::_f5_connector]

How to configure F5 Trigger

1. Log in to Horizon Administration Interface.

2. Access F5 Triggers from the drawer or card: **Third Parties** › **F5** › **Triggers**.

3. Click on  .

4. Fill the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful trigger name. It must be unique for each trigger. Horizon uses the name to identify the trigger.
- **F5 Connector*** *(select)*:
  Select a F5 connector previously created.
- **Retries in case of error** *(int)*:
  Number of times to retry to push the change on the F5 BigIP repository in case of error. Must be an integer between 1 and 15.

5. Click on the save button.

You can update  or delete  the F5 Trigger.

Synchronization using triggers

Triggers are a functionality of WebRA, EST, Intune PKCS, WCCE and CRMP profiles that allows to push lifecycle events into a third party whenever they occur on a profile.

1. Refer to the trigger documentation to create a trigger.

2. Create or modify the profile you wish to use the triggers on.

3. Go to the **Triggers** tab, then on **Certificate lifecycle triggers**

4. Chose which lifecycle events you wish to use triggers upon (enrollment, revocation, expiration)

5. Select one or more existing triggers from the menu (if several are selected, they will all be called whenever the selected event occurs)

6. Click on the Save button.

From now on, whenever a selected lifecycle event will occur on the configured profile, the trigger will be called and the and the certificate will be pushed into or removed from the third party container.

2.12.4. F5 AS3

F5 Introduction

This section refers to the F5 BigIP integration with Horizon, used to enroll certificates used by F5 BigIP.

This integration involves at least two infrastructure components:

- F5 BigIP
- F5 AS3 enabled
- EverTrust Horizon

Horizon connects to the F5 BigIP using the AS3 declarative document API in order to manage the lifecycle of certificates within the BigIP.

Limitations

Horizon can only manage the lifecycle of certificate already on the F5 AS3. It cannot push new certificate to it.

Horizon can **renew** certificates that need to be renewed on the AS3 and **replace** the previous certificate.

Horizon can **revoke** certificates that are removed from the AS3 and are managed in Horizon.

Horizon cannot remove certificates from the AS3 after a revocation on Horizon.

You will need to import your F5 AS3 certificates into Horizon, it is recommended to use **horizon-cli** to do so.

F5 AS3 Connector

This section details how to configure the F5 AS3 Connector.

Required By

[admin-guide:third-parties-f5as3-triggers:::_f5as3_trigger]
WEBRA Scheduled task

Prerequisites

On the F5 AS3 side, you need to create a technical user for Horizon and give it full administrator rights. This is required because AS3 is a declarative way of managing the configuration, you have

either the permission to manage it or not.

After performing these steps, you will get the following information required later:

- the technical user login/username
- the technical user password

How to configure F5 Connector

1. Log in to Horizon Administration Interface.

2. Access F5 AS3 Connectors from the drawer or card: **Third Parties › F5 AS3 › Connectors**.

3. Click on  .

4. Fill the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful connector name. It must be unique for each connector. Horizon uses the name to identify the connector.
- **F5 hostname*** *(string input)*:
  Enter the F5 hostname (DNS or IP address).
- **F5 credentials*** *(select)*:
  Select `Login` credentials containing the username and password created for Horizon in the F5 BigIP. Must have administrator rights.
- **F5 Login Provider** *(string input)*:
  Login provider to use in TACACS authentication mode. `tmos` for example
- **Proxy** *(string select)*:
  The HTTP/HTTPS proxy to use.
- **Timeout** *(finite duration)*:
  Set by default at 10 seconds. Must be a valid finite duration.
- **TLS Insecure** *(boolean)*:
  If enabled, TLS validation will ignore expired, invalid or untrusted certificates.
- **Bundle chain** *(boolean)*:
  If enabled, The trust chain of the certificate will also be pushed.



This is not recommended for production usage

Actors and renewal management

These configuration elements mainly define the number of authorized interactions with the remote service on a defined period. For example, one needs to ensure that the remote service will not be contacted more than 5 times per 3 seconds. *Throttle parallelism* defines the number of times and

Throttle duration the period of time. Therefore, on the above example, throttle parallelism would be set to 5 and throttle duration would be set to 3 seconds.

- **Throttle duration*** *(finite duration)*:
  Set by default at 3 seconds. Must be a valid finite duration.
- **Throttle parallelism*** *(int)*:
  Set by default at 3.

5. Click on the save button.

You can update or delete the F5 Connector.



You will not be able to delete an F5 AS3 Connector if it is referenced in any other configuration element.

Synchronize your third party

Your third-party certificates can be synchronized with Horizon using scheduled tasks.

Scheduled tasks are a functionality of WebRA that allows to synchronize automatic renewal or revocation events with a third party periodically with what occurs on a WebRA profile. To be more specific, it will periodically check whether the certificate has entered the "renewal period" that was defined in the connector's configuration, and renew it automatically if necessary.

1. Refer to the third party connector documentation to create a third party connector.

2. Ensure you have an existing [admin-guide:protocols-webra-webra_profile:::_webra_profile]: renewal will be automated on the selected profile.

3. Follow the documentation of the [admin-guide:protocols-webra-webra_schedule_tasks:::_webra_scheduled_tasks] section to properly configure a scheduled task.

Connector actions

The following actions are available once the connector is configured

- **Retry failed triggers**: This will run every retryable trigger linked to this connector. A trigger is retryable if it failed and all required parameters for it (like the private key) are still available.



This action could trigger a lot of retries.

F5 AS3 Trigger

This section details how to configure the Triggers that will be used by Profiles to push or delete certificates to/from F5 BigIP.

Prerequisites

F5 Connector

How to configure F5 Trigger

1. Log in to Horizon Administration Interface.

2. Access F5 AS3 Triggers from the drawer or card: **Third Parties** › **F5 AS3** › **Triggers**.

3. Click on  .

4. Fill the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful trigger name. It must be unique for each trigger. Horizon uses the name to identify the trigger.
- **F5 AS3 Connector*** *(select)*:
  Select a F5 AS3 connector previously created.
- **Retries** *(int)*:
  Number of times to retry to push the change on the F5 BigIP repository in case of error. Must be an integer between 1 and 15.
- **On execution error** *(notification trigger)*: In case of an error happening during the trigger execution, notification defined here will be sent.

5. Click on the save button.

You can update  or delete  the F5 Trigger.

Synchronization using triggers

Triggers are a functionality of WebRA, EST, Intune PKCS, WCCE and CRMP profiles that allows to push lifecycle events into a third party whenever they occur on a profile.

1. Refer to the trigger documentation to create a trigger.

2. Create or modify the profile you wish to use the triggers on.

3. Go to the **Triggers** tab, then on **Certificate lifecycle triggers**

4. Chose which lifecycle events you wish to use triggers upon (enrollment, revocation, expiration)

5. Select one or more existing triggers from the menu (if several are selected, they will all be called whenever the selected event occurs)

6. Click on the Save button.

From now on, whenever a selected lifecycle event will occur on the configured profile, the trigger will be called and the and the certificate will be pushed into or removed from the third party

container.

2.12.5. Fortinet

Fortinet Introduction

This section refers to the Fortinet integration with Horizon, used to deploy certificates on Fortinet appliances.

This integration involves at least two infrastructure components:

- A Fortinet FortiGate firewall or a FortiManager appliance
- EverTrust Horizon

Two connectors are available:

- The FortiGate connector connects to a single FortiGate firewall using the FortiOS REST API, and deploys certificates in its local certificate store, either in the global scope or in a given virtual domain (VDOM).
- The FortiManager connector connects to a FortiManager appliance using its JSON-RPC API, and deploys certificates either on the FortiManager unit itself or on a FortiGate device it manages.

Using triggers, Horizon deploys the certificate and its private key on enrollment and renewal, and removes it from the appliance on revocation.

For each holder, only the latest certificate is kept on the appliance: deploying a new certificate removes the previously deployed one. A certificate is only removed if it is still found under the name Horizon gave it; if it was renamed or moved on the appliance, the removal is skipped.



Only FortiOS 7.0 and later is supported.

FortiGate Connector

This section details how to configure the FortiGate Connector.

Required By

FortiGate Trigger

Prerequisites

On the FortiGate side, you need to create a REST API administrator for Horizon, with an administrator profile allowed to read and write the certificate configuration. Make sure the Horizon IP address is part of the trusted hosts of this administrator.

After performing these steps, you will get the following information, required later:

- the REST API key generated for this administrator

- optionally, a client certificate if the FortiGate requires mutual TLS authentication

How to configure FortiGate Connector

1. Log in to Horizon Administration Interface.

2. Access FortiGate Connectors from the drawer or card: **Third Parties** › **FortiGate** › **Connectors**.

3. Click on  .

4. Fill the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful connector name. It must be unique for each connector. Horizon uses the name to identify the connector.
- **Hostname*** *(string input)*:
  Enter the FortiGate hostname or URL.
- **Credentials*** *(select)*:
  Select `API Token` credentials containing the FortiGate REST API key.
- **Certificate credentials** *(select)*:
  Select `Certificate` credentials to authenticate with mutual TLS, in addition to the API key.
- **VDOM** *(string input)*:
  Virtual domain to deploy certificates to. When empty, certificates are deployed in the global scope.
- **Prefix*** *(string input)*:
  Prefix of the certificate names on the FortiGate, used to identify the certificates managed by Horizon.
- **Proxy** *(select)*:
  The HTTP/HTTPS proxy to use.
- **Timeout*** *(finite duration)*:
  Maximum time Horizon waits for a response from the FortiGate.
- **TLS Insecure** *(boolean)*:
  If enabled, TLS validation will ignore expired, invalid or untrusted certificates.



This is not recommended for production usage

Actors and renewal management

These configuration elements mainly define the number of authorized interactions with the remote service on a defined period. For example, one needs to ensure that the remote service will not be contacted more than 5 times per 3 seconds. *Throttle parallelism* defines the number of times and *Throttle duration* the period of time. Therefore, on the above example, throttle parallelism would be set to 5 and throttle duration would be set to 3 seconds.

- **Throttle duration*** *(finite duration)*:
  Must be a valid finite duration.
- **Throttle parallelism*** *(int)*:
  Number of deployments processed in parallel.

5. Click on the save button.

You can update  or delete  the FortiGate Connector.



You will not be able to delete a FortiGate Connector if it is referenced in any other configuration element.

FortiManager Connector

This section details how to configure the FortiManager Connector.

Required By

FortiManager Trigger

Prerequisites

On the FortiManager side, you need to create an administrator account for Horizon with JSON-RPC API access in read-write mode, allowed to manage certificates on the FortiManager and, when deploying to managed devices, on the targeted ADOM.

After performing these steps, you will get the following information, required later:

- the administrator login/username
- the administrator password

How to configure FortiManager Connector

1. Log in to Horizon Administration Interface.

2. Access FortiManager Connectors from the drawer or card: **Third Parties** › **FortiManager** › **Connectors**.

3. Click on  .

4. Fill the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful connector name. It must be unique for each connector. Horizon uses the name to identify the connector.

- **Hostname*** *(string input)*:
  Enter the FortiManager hostname or URL.
- **Credentials*** *(select)*:
  Select `Login` credentials containing the username and password created for Horizon in the FortiManager.
- **Prefix*** *(string input)*:
  Prefix of the certificate names on the FortiManager, used to identify the certificates managed by Horizon.
- **Proxy** *(select)*:
  The HTTP/HTTPS proxy to use.
- **Timeout*** *(finite duration)*:
  Maximum time Horizon waits for a response from the FortiManager.
- **TLS Insecure** *(boolean)*:
  If enabled, TLS validation will ignore expired, invalid or untrusted certificates.



This is not recommended for production usage

Target

- **Target*** *(select)*:
  Select where certificates are deployed:
  - `Unit`: the certificate store of the FortiManager unit itself.
  - `Device`: a FortiGate device managed by the FortiManager. The following fields are then required:
    - **ADOM*** *(string input)*:
      Administrative domain the device belongs to.
    - **Device*** *(string input)*:
      Name of the managed device.
    - **VDOM*** *(string input)*:
      Virtual domain on the managed device.
    - **Synchronize devices** *(boolean)*:
      If enabled, Horizon installs the configuration on the device after importing the certificate. Otherwise, the certificate is only imported in the FortiManager device database. Disabled by default.

Actors and renewal management

These configuration elements mainly define the number of authorized interactions with the remote service on a defined period. For example, one needs to ensure that the remote service will not be contacted more than 5 times per 3 seconds. *Throttle parallelism* defines the number of times and *Throttle duration* the period of time. Therefore, on the above example, throttle parallelism would be set to 5 and throttle duration would be set to 3 seconds.

- **Throttle duration*** *(finite duration)*:

Must be a valid finite duration.

- **Throttle parallelism*** *(int)*:
  Number of deployments processed in parallel.

Deployment jobs retry

Deployments to this third party are run as asynchronous jobs. When a job fails, Horizon retries it using an exponential backoff strategy.

- **Attempts*** *(int)*:
  Maximum number of retry attempts before the job is considered failed.
- **Minimum backoff*** *(finite duration)*:
  Delay to wait before the first retry.
- **Maximum backoff*** *(finite duration)*:
  Maximum delay between two retries, capping the exponential backoff.
- **Random factor*** *(decimal)*:
  Random jitter added to each delay to avoid simultaneous retries (e.g. `0.1` adds up to 10%).

5. Click on the save button.

You can update  or delete  the FortiManager Connector.



You will not be able to delete a FortiManager Connector if it is referenced in any other configuration element.

Fortinet Triggers

This section details how to configure the Triggers that will be used by Profiles to push or delete certificates to/from Fortinet appliances.

FortiGate Trigger

Prerequisites

[admin-guide:third-parties-fortinet-fortigate_connector:::_fortigate_connector]

How to configure FortiGate Trigger

1. Log in to Horizon Administration Interface.

2. Access FortiGate Triggers from the drawer or card: **Third Parties** › **FortiGate** › **Triggers**.

3. Click on  .

4. Fill the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful trigger name. It must be unique for each trigger. Horizon uses the name to identify the trigger.
- **FortiGate Connector*** *(select)*:
  Select a FortiGate connector previously created.
- **Retries in case of error** *(int)*:
  Number of times to retry to push the change on the FortiGate in case of error. Must be an integer between 1 and 15.

5. Click on the save button.

You can update  or delete  the FortiGate Trigger.

FortiManager Trigger

Prerequisites

[admin-guide:third-parties-fortinet-fortimanager_connector:::_fortimanager_connector]

How to configure FortiManager Trigger

1. Log in to Horizon Administration Interface.

2. Access FortiManager Triggers from the drawer or card: **Third Parties** › **FortiManager** › **Triggers**.

3. Click on  .

4. Fill the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful trigger name. It must be unique for each trigger. Horizon uses the name to identify the trigger.
- **FortiManager Connector*** *(select)*:
  Select a FortiManager connector previously created.
- **Retries in case of error** *(int)*:
  Number of times to retry to push the change on the FortiManager in case of error. Must be an integer between 1 and 15.

5. Click on the save button.

You can update  or delete  the FortiManager Trigger.

Synchronization using triggers

Triggers are a functionality of WebRA, EST, Intune PKCS, WCCE and CRMP profiles that allows to

push lifecycle events into a third party whenever they occur on a profile.

1. Refer to the trigger documentation to create a trigger.

2. Create or modify the profile you wish to use the triggers on.

3. Go to the **Triggers** tab, then on **Certificate lifecycle triggers**

4. Chose which lifecycle events you wish to use triggers upon (enrollment, revocation, expiration)

5. Select one or more existing triggers from the menu (if several are selected, they will all be called whenever the selected event occurs)

6. Click on the Save button.

From now on, whenever a selected lifecycle event will occur on the configured profile, the trigger will be called and the and the certificate will be pushed into or removed from the third party container.

2.12.6. Netscaler

Netscaler Introduction

This section refers to the netscaler integration with Horizon, used to enroll certificates used by Netscaler.

This integration involves at least two infrastructure components:

- Netscaler
- EverTrust Horizon

Horizon connects to the Netscaler instance using the nitro REST administration API in order to manage the lifecycle of certificates associated to SSL Cert Key objects within Netscaler.

Netscaler Connector

This section details how to configure the Netscaler Connector.

Required By

[admin-guide:third-parties-netscaler-triggers:::_netscaler_trigger]

Prerequisites

On the Netscaler side, you need to create a technical user for Horizon.

This account should be allowed to manage the API Management interface, and should have the following command policy:

- Action: `ALLOW`

- Command Spec: `(^\S+\s+system\s+file\s\S+\s.*)|(^\S+\s+ssl\s+certKey.*)`

This allows Horizon to upload and delete certificate files, as well as managing the SSL Cert Key objects.

After performing these steps, you will get the following information, required later:

- the technical user login/username
- the technical user password

How to configure Netscaler Connector

1. Log in to Horizon Administration Interface.

2. Access Netscaler Connectors from the drawer or card: **Third Parties › Netscaler › Connectors**.

3. Click on  .

4. Fill the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful connector name. It must be unique for each connector. Horizon uses the name to identify the connector.
- **Netscaler hostname*** *(string input)*:
  Enter the Netscaler hostname (DNS or IP address).
- **Netscaler credentials*** *(select)*:
  Select `Login` credentials containing the username and password created for Horizon in the Netscaler.
- **Certificate store path*** *(string input)*:
  Path where certificates are stored on the Netscaler.
- **Max stored certificates per holder** *(int)*:
  When specified, define the maximum number of certificates stored in the third party for a given holder.
- **Prefix** *(string input)*:
  Used to filter the certificates managed by Horizon in the Netscaler. `hrz-` by default.
- **Proxy** *(string select)*:
  The HTTP/HTTPS proxy to use.
- **Timeout*** *(finite duration)*:
  Set by default at 10 seconds. Must be a valid finite duration.
- **TLS Insecure** *(boolean)*:
  If enabled, TLS validation will ignore expired, invalid or untrusted certificates.



This is not recommended for production usage

Actors and renewal management

These configuration elements mainly define the number of authorized interactions with the remote service on a defined period. For example, one needs to ensure that the remote service will not be contacted more than 5 times per 3 seconds. *Throttle parallelism* defines the number of times and *Throttle duration* the period of time. Therefore, on the above example, throttle parallelism would be set to 5 and throttle duration would be set to 3 seconds.

- **Throttle duration*** *(finite duration)*:
  Set by default at 3 seconds. Must be a valid finite duration.
- **Throttle parallelism*** *(int)*:
  Set by default at 3.

5. Click on the save button.

You can update  or delete  the Netscaler Connector.



You will not be able to delete an Netscaler Connector if it is referenced in any other configuration element.

Synchronize your third party

Your third-party certificates can be synchronized with Horizon using scheduled tasks.

Scheduled tasks are a functionality of WebRA that allows to synchronize automatic renewal or revocation events with a third party periodically with what occurs on a WebRA profile. To be more specific, it will periodically check whether the certificate has entered the "renewal period" that was defined in the connector's configuration, and renew it automatically if necessary.

1. Refer to the third party connector documentation to create a third party connector.

2. Ensure you have an existing [admin-guide:protocols-webra-webra_profile:::_webra_profile]: renewal will be automated on the selected profile.

3. Follow the documentation of the [admin-guide:protocols-webra-webra_schedule_tasks:::_webra_scheduled_tasks] section to properly configure a scheduled task.

Netscaler Trigger

This section details how to configure the Triggers that will be used by Profiles to push or delete certificates to/from Netscaler.

Prerequisites

[admin-guide:third-parties-netscaler-connector:::_netscaler_connector]

How to configure Netscaler Trigger

1. Log in to Horizon Administration Interface.

2. Access Netscaler Triggers from the drawer or card: **Third Parties** › **Netscaler** › **Triggers**.

3. Click on  .

4. Fill the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful trigger name. It must be unique for each trigger. Horizon uses the name to identify the trigger.
- **Netscaler Connector*** *(select)*:
  Select a Netscaler connector previously created.
- **Retries in case of error** *(int)*:
  Number of times to retry to push the change on the Netscaler BigIP repository in case of error. Must be an integer between 1 and 15.

5. Click on the save button.

You can update  or delete  the Netscaler Trigger.

Synchronization using triggers

Triggers are a functionality of WebRA, EST, Intune PKCS, WCCE and CRMP profiles that allows to push lifecycle events into a third party whenever they occur on a profile.

1. Refer to the trigger documentation to create a trigger.

2. Create or modify the profile you wish to use the triggers on.

3. Go to the **Triggers** tab, then on **Certificate lifecycle triggers**

4. Chose which lifecycle events you wish to use triggers upon (enrollment, revocation, expiration)

5. Select one or more existing triggers from the menu (if several are selected, they will all be called whenever the selected event occurs)

6. Click on the Save button.

From now on, whenever a selected lifecycle event will occur on the configured profile, the trigger will be called and the and the certificate will be pushed into or removed from the third party container.

2.12.7. Palo Alto

Palo Alto Introduction

This section refers to the Palo Alto Networks integration with Horizon, used to deploy certificates on PAN-OS firewalls.

This integration involves at least two infrastructure components:

- A Palo Alto PAN-OS firewall or a Panorama appliance
- EverTrust Horizon

Two connectors are available, both relying on the PAN-OS XML API:

- The Panorama connector connects to a Panorama appliance and deploys certificates in a template, template stack or virtual system (VSYS). It can optionally commit the configuration to the managed firewalls.
- The PAN-OS Firewall connector connects to a standalone firewall and deploys certificates on it, optionally in a given virtual system (VSYS).

Using triggers, Horizon deploys the certificate and its private key on enrollment and renewal, and removes it from the target on revocation or expiration. A certificate is only removed if it is still found under the name Horizon gave it; if it was renamed or moved, the removal is skipped.



Only PAN-OS 10.2 and later is supported.

Panorama Connector

This section details how to configure the Palo Alto Panorama Connector.

Required By

Panorama Trigger

Prerequisites

On the Panorama side, you need to create an administrator account for Horizon, with an admin role allowing XML API access to import certificates, edit the configuration and, if devices are synchronized, commit and push the configuration.

After performing these steps, you will get the following information, required later:

- the administrator login/username
- the administrator password

How to configure Panorama Connector

1. Log in to Horizon Administration Interface.

2. Access Panorama Connectors from the drawer or card: **Third Parties › Panorama › Connectors**.

3. Click on ⊕ .

4. Fill the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful connector name. It must be unique for each connector. Horizon uses the name to identify the connector.
- **Hostname*** *(string input)*:
  Enter the Panorama hostname or URL.
- **Credentials*** *(select)*:
  Select `Login` credentials containing the username and password created for Horizon in Panorama.
- **Proxy** *(select)*:
  The HTTP/HTTPS proxy to use.
- **Timeout*** *(finite duration)*:
  Maximum time Horizon waits for a response from Panorama.
- **TLS Insecure** *(boolean)*:
  If enabled, TLS validation will ignore expired, invalid or untrusted certificates.



This is not recommended for production usage

Assets identification

- **Template** *(string input)*:
  Name of the Panorama template to push certificates to.
- **Template stack** *(string input)*:
  Name of the Panorama template stack to push changes to.
- **VSYS** *(string input)*:
  Name of the virtual system to target within the template. Requires **Template** to be set.
- **Prefix*** *(string input)*:
  Prefix of the certificate names on Panorama, used to identify the certificates managed by Horizon.
- **Synchronize devices** *(boolean)*:
  If enabled, Horizon commits the configuration to the devices managed by Panorama after pushing the certificate. Otherwise, the certificate stays in the Panorama candidate configuration until an operator commits it.

Actors and renewal management

These configuration elements mainly define the number of authorized interactions with the remote service on a defined period. For example, one needs to ensure that the remote service will not be contacted more than 5 times per 3 seconds. *Throttle parallelism* defines the number of times and

Throttle duration the period of time. Therefore, on the above example, throttle parallelism would be set to 5 and throttle duration would be set to 3 seconds.

- **Throttle duration*** *(finite duration)*:
  Must be a valid finite duration.
- **Throttle parallelism*** *(int)*:
  Number of deployments processed in parallel.

Deployment jobs retry

Deployments to this third party are run as asynchronous jobs. When a job fails, Horizon retries it using an exponential backoff strategy.

- **Attempts*** *(int)*:
  Maximum number of retry attempts before the job is considered failed.
- **Minimum backoff*** *(finite duration)*:
  Delay to wait before the first retry.
- **Maximum backoff*** *(finite duration)*:
  Maximum delay between two retries, capping the exponential backoff.
- **Random factor*** *(decimal)*:
  Random jitter added to each delay to avoid simultaneous retries (e.g. `0.1` adds up to 10%).

5. Click on the save button.

You can update  or delete  the Panorama Connector.



You will not be able to delete a Panorama Connector if it is referenced in any other configuration element.

PAN-OS Firewall Connector

This section details how to configure the Palo Alto PAN-OS Firewall Connector, for standalone firewalls not managed by Panorama.

Required By

PAN-OS Firewall Trigger

Prerequisites

On the firewall side, you need to create an administrator account for Horizon, with an admin role allowing XML API access to import certificates and edit the configuration.

After performing these steps, you will get the following information, required later:

- the administrator login/username
- the administrator password

How to configure PAN-OS Firewall Connector

1. Log in to Horizon Administration Interface.

2. Access PAN-OS Firewall Connectors from the drawer or card: **Third Parties › PAN-OS Firewall › Connectors**.

3. Click on  .

4. Fill the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful connector name. It must be unique for each connector. Horizon uses the name to identify the connector.
- **Hostname*** *(string input)*:
  Enter the firewall hostname or URL.
- **Credentials*** *(select)*:
  Select `Login` credentials containing the username and password created for Horizon in the firewall.
- **Proxy** *(select)*:
  The HTTP/HTTPS proxy to use.
- **Timeout*** *(finite duration)*:
  Maximum time Horizon waits for a response from the firewall.
- **TLS Insecure** *(boolean)*:
  If enabled, TLS validation will ignore expired, invalid or untrusted certificates.



This is not recommended for production usage

Assets identification

- **VSYS** *(string input)*:
  Name of the virtual system to target, for multi-VSYS firewalls.
- **Prefix*** *(string input)*:
  Prefix of the certificate names on the firewall, used to identify the certificates managed by Horizon.

Actors and renewal management

These configuration elements mainly define the number of authorized interactions with the remote service on a defined period. For example, one needs to ensure that the remote service will not be contacted more than 5 times per 3 seconds. *Throttle parallelism* defines the number of times and *Throttle duration* the period of time. Therefore, on the above example, throttle parallelism would be set to 5 and throttle duration would be set to 3 seconds.

- **Throttle duration*** *(finite duration)*:
  Must be a valid finite duration.
- **Throttle parallelism*** *(int)*:
  Number of deployments processed in parallel.

Deployment jobs retry

Deployments to this third party are run as asynchronous jobs. When a job fails, Horizon retries it using an exponential backoff strategy.

- **Attempts*** *(int)*:
  Maximum number of retry attempts before the job is considered failed.
- **Minimum backoff*** *(finite duration)*:
  Delay to wait before the first retry.
- **Maximum backoff*** *(finite duration)*:
  Maximum delay between two retries, capping the exponential backoff.
- **Random factor*** *(decimal)*:
  Random jitter added to each delay to avoid simultaneous retries (e.g. `0.1` adds up to 10%).

5. Click on the save button.

You can update  or delete  the PAN-OS Firewall Connector.



You will not be able to delete a PAN-OS Firewall Connector if it is referenced in any other configuration element.

Palo Alto Triggers

This section details how to configure the Triggers that will be used by Profiles to push or delete certificates to/from Palo Alto appliances.

Panorama Trigger

Prerequisites

[admin-guide:third-parties-paloalto-panorama_connector:::_panorama_connector]

How to configure Panorama Trigger

1. Log in to Horizon Administration Interface.

2. Access Panorama Triggers from the drawer or card: **Third Parties** › **Panorama** › **Triggers**.

3. Click on .

4. Fill the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful trigger name. It must be unique for each trigger. Horizon uses the name to identify the trigger.
- **Panorama Connector*** *(select)*:
  Select a Panorama connector previously created.
- **Retries in case of error** *(int)*:
  Number of times to retry to push the change on Panorama in case of error. Must be an integer between 1 and 15.

5. Click on the save button.

You can update  or delete  the Panorama Trigger.

PAN-OS Firewall Trigger

Prerequisites

[admin-guide:third-parties-paloalto-firewall_connector:::_panos_firewall_connector]

How to configure PAN-OS Firewall Trigger

1. Log in to Horizon Administration Interface.

2. Access PAN-OS Firewall Triggers from the drawer or card: **Third Parties** › **PAN-OS Firewall** › **Triggers**.

3. Click on  .

4. Fill the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful trigger name. It must be unique for each trigger. Horizon uses the name to identify the trigger.
- **PAN-OS Firewall Connector*** *(select)*:
  Select a PAN-OS Firewall connector previously created.
- **Retries in case of error** *(int)*:
  Number of times to retry to push the change on the firewall in case of error. Must be an integer between 1 and 15.

5. Click on the save button.

You can update  or delete  the PAN-OS Firewall Trigger.

Synchronization using triggers

Triggers are a functionality of WebRA, EST, Intune PKCS, WCCE and CRMP profiles that allows to

push lifecycle events into a third party whenever they occur on a profile.

1. Refer to the trigger documentation to create a trigger.

2. Create or modify the profile you wish to use the triggers on.

3. Go to the **Triggers** tab, then on **Certificate lifecycle triggers**

4. Chose which lifecycle events you wish to use triggers upon (enrollment, revocation, expiration)

5. Select one or more existing triggers from the menu (if several are selected, they will all be called whenever the selected event occurs)

6. Click on the Save button.

From now on, whenever a selected lifecycle event will occur on the configured profile, the trigger will be called and the and the certificate will be pushed into or removed from the third party container.

2.12.8. GCM

GCM Introduction

This section refers to the Google Certificate Manager integration with Horizon, used to enroll certificates used by Google Certificate Manager.

This integration involves at least two infrastructure components:

- Google Certificate Manager
- EverTrust Horizon

GCM Connector

This section details how to configure the Google Certificate Manager Connector.

Required By

[admin-guide:third-parties-gcm-triggers:::_gcm_trigger]

Prerequisites

On Horizon side, you might need to set up a Proxy , used to reach GCM, if necessary.

On Google Cloud side, you need to create a service account using the IAM, and grant that SA the appropriate permissions, as documented here. Typically, these can be granted through the Certificate Manager Owner role (`roles/certificatemanager.owner`), or through the individual following permissions:

- `certificatemanager.certs.create`

- `certificatemanager.certs.list`
- `certificatemanager.certs.get`
- `certificatemanager.certs.update`
- `certificatemanager.certs.delete`

After performing these steps, you will get the following information, required later:

- the GCP Project
- the GCP Location
- the JSON key file of the GCP Service Account

How to configure GCM Connector

1. Log in to Horizon Administration Interface.

2. Access GCM Connectors from the drawer or card: **Third Parties › GCM › Connectors**.

3. Click on  .

4. Fill the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful connector name. It must be unique for each connector. Horizon uses the name to identify the connector.
- **GCM Service Account Credentials*** *(select)*:
  Select `API Token` credentials whose JSON Token holds the full content of the service account JSON key file.
- **Proxy** *(string select)*:
  The HTTP/HTTPS proxy to use.
- **Timeout** *(finite duration)*:
  Set by default at 10 seconds. Must be a valid finite duration.

Assets identification

- **Project name*** *(string input)*:
  ID of the GCP project hosting the certificates.
- **Location*** *(string input)*:
  Location of the Certificate Manager resources, `global` or a region.
- **Label** *(string inputs)*:
  Used to filter the certificates managed by Horizon in GCM.
  - **Key** *(string input)*:
    The label key. *manage* by default.

- **Value** *(string input)*:
  The label value. *horizon* by default.

Actors and renewal management

This configuration element paces the calls made by the scheduled tasks described below, which process certificates in batches. Pushes fired by the profile triggers are not throttled. The GCM connector issues one call per throttle window, so *Throttle duration* alone sets that pace.

- **Throttle duration*** *(finite duration)*:
  Set by default at 3 seconds. Must be a valid finite duration.

5. Click on the save button.

You can update  or delete  the GCM Connector.



You will not be able to delete a GCM Connector if it is referenced in any other configuration element.

Synchronize your third party

Your third-party certificates can be synchronized with Horizon using scheduled tasks.

Scheduled tasks are a functionality of WebRA that allows to synchronize automatic renewal or revocation events with a third party periodically with what occurs on a WebRA profile. To be more specific, it will periodically check whether the certificate has entered the "renewal period" that was defined in the connector's configuration, and renew it automatically if necessary.

1. Refer to the third party connector documentation to create a third party connector.

2. Ensure you have an existing [admin-guide:protocols-webra-webra_profile:::_webra_profile]: renewal will be automated on the selected profile.

3. Follow the documentation of the [admin-guide:protocols-webra-webra_schedule_tasks:::_webra_scheduled_tasks] section to properly configure a scheduled task.

Connector actions

The following actions are available once the connector is configured

-  **Retry failed triggers**: This will run every retryable trigger linked to this connector. A trigger is retryable if it failed and all required parameters for it (like the private key) are still available.



This action could trigger a lot of retries.

GCM Trigger

This section details how to configure the Triggers that will be used by Profiles to push or delete certificates to/from Google Certificate Manager.

Prerequisites

[admin-guide:third-parties-gcm-connector:::_gcm_connector]

How to configure GCM Trigger

1. Log in to Horizon Administration Interface.

2. Access GCM Triggers from the drawer or card: **Third Parties** › **GCM** › **Triggers**.

3. Click on  .

4. Fill the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful trigger name. It must be unique for each trigger. Horizon uses the name to identify the trigger.
- **GCM Connector*** *(select)*:
  Select a GCM connector previously created.
- **Retries in case of error** *(int)*:
  Number of times to retry to push the change on the GCM repository in case of error. Must be an integer between 1 and 15.

5. Click on the save button.

You can update  or delete  the GCM Trigger.

Synchronization using triggers

Triggers are a functionality of WebRA, EST, Intune PKCS, WCCE and CRMP profiles that allows to push lifecycle events into a third party whenever they occur on a profile.

1. Refer to the trigger documentation to create a trigger.

2. Create or modify the profile you wish to use the triggers on.

3. Go to the **Triggers** tab, then on **Certificate lifecycle triggers**

4. Chose which lifecycle events you wish to use triggers upon (enrollment, revocation, expiration)

5. Select one or more existing triggers from the menu (if several are selected, they will all be called whenever the selected event occurs)

6. Click on the Save button.

From now on, whenever a selected lifecycle event will occur on the configured profile, the trigger will be called and the and the certificate will be pushed into or removed from the third party container.



Google Certificate Manager requires the certificate CN to be a valid DNS hostname. If you try to push a certificate with a CN that is not a valid DNS hostname, you may receive a validation error stating that the "domain name doesn't comply with RFC 1034 3.5 preferred name syntax (relaxed by RFC 1123 2.1)".

Therefore, we recommend validating the certificate CN using Horizon validation rules to ensure consistency between certificates in Horizon and on Google Certificate Manager.

2.12.9. LDAP

LDAP Introduction

This section details the LDAP integration with Horizon, used to publish and unpublish certificates on LDAP.

The integration will require to set up the following elements (on Horizon side):

- an LDAP Connector, which holds the configuration items required by Horizon to connect to LDAP
- an LDAP Trigger, which holds the configuration items specifying how Horizon should publish/unpublish certificates for the specified LDAP connector



Only SMIME Certificates can be published

LDAP Connector

This section details how to configure an LDAP Connector.

Required By

LDAP Trigger

Prerequisites

On the LDAP side, it is required to create a technical user with permissions to write in the LDAP sub DN, so that Horizon will be able to search by email, to publish and to unpublish certificates using that technical user. The following information will be required later:

- LDAP Hostname
- a login DN
- a password

- Base DN to publish SMIME certificates

How to configure LDAP Connector

1. Log in to Horizon Administration Interface.

2. Access LDAP Connector from the drawer or card: **Third Parties › LDAP › Connectors**.

3. Click on  .

4. Fill the mandatory fields.

Connection

- **Name*** *(string input)*:
  Enter a meaningful connector name. It must be unique for each connector. Horizon uses the name to identify the connector.
- **Hostname*** *(string input)*:
  Enter the URL pointing to LDAP.
- **LDAP Credentials*** *(select)*:
  Select `Login` credentials containing the technical user created for Horizon login DN and password.
- **Base DN*** *(string input)*:
  Enter the Base DN where Horizon should publish the certificate.
- **Max stored certificates per holder*** *(int)*:
  When specified, define a maximum number of certificates stored in the third party.
- **Port** *(int)*:
  Enter the port where to reach the running LDAP instance (default values are 389 for LDAP and 636 for LDAPS).
- **Proxy** *(string select)*:
  The HTTP/HTTPS proxy used to reach LDAP, if any.
- **Timeout** *(finite duration)*:
  Set by default at 10 seconds. Must be a valid finite duration.
- **TLS Insecure** *(boolean)*:
  If enabled, TLS validation will ignore expired, invalid or untrusted certificates.



This is not recommended for production usage

Assets identification and management

- **Filter** *(string input)*:
  Enter the custom filter. By default, LDAP Identities are filtered by `(objectclass=user)`. If you are using inetOrgPerson as type, you will have to manually set the following filter: `(objectclass=inetOrgPerson)`.

- **Target LDAP publication attribute** *(string input)*:
  When specified, the certificate will be published on the specified attribute. In most LDAP applications you will have to set the field to: `userCertificate;binary` but in MSAD the field is already well managed.
- **Target LDAP user identifier attribute** *(string select)*:
  The LDAP attribute that will be used to identify a user for publication. Possible values are `CN`, `MAIL`, `UID`.
- **Certificate user identifier attribute** *(string select)*:
  The Certificate attribute value that will be used to identity a user for publication, possible values are `UID SERIALNUMBER SURNAME GIVENNAME T UNSTRUCTUREDADDRESS UNSTRUCTUREDNAME E OU ORGANIZATIONIDENTIFIER PSEUDONYM UNIQUEIDENTIFIER STREET ST L O C DESCRIPTION DC RFC822NAME DNSNAME URI IPADDRESS OTHERNAME_UPN OTHERNAME_GUID`.
- **Follow referrals** *(boolean)*:
  Allow publication to follow LDAP referrals.
- **Create LDAP entry** *(boolean)*:
  If true, an LDAP entry will be created for this certificate if no entry matching the filter and the identifier attribute are detected. This entry will have its `objectClass` set to the filter value.

Actors and renewal management

These configuration elements mainly define the number of authorized interactions with the remote service on a defined period. For example, one needs to ensure that the remote service will not be contacted more than 5 times per 3 seconds. *Throttle parallelism* defines the number of times and *Throttle duration* the period of time. Therefore, on the above example, throttle parallelism would be set to 5 and throttle duration would be set to 3 seconds.

- **Throttle duration*** *(finite duration)*:
  Set by default to 3 seconds. Must be a valid finite duration.
- **Throttle parallelism*** *(int)*:
  Set by default to 3.

5. Click on the save button.

You can update  or delete  the LDAP Connector.



You won't be able to delete a LDAP Connector if it is referenced in any other configuration element.

Connector actions

The following actions are available once the connector is configured

-  **Retry failed triggers**: This will run every retryable trigger linked to this connector. A trigger is retryable if it failed and all required parameters for it (like the private key) are still available.



This action could trigger a lot of retries.

LDAP Triggers

Here is the section to manage the Triggers that will be used by profiles to publish or unpublish certificates into LDAP.

Prerequisites

[admin-guide:third-parties-ldap-connector:::_ldap_connector]

How to configure LDAP trigger

1. Log in to Horizon Administration Interface.

2. Access LDAP triggers from the drawer or card: **Third Parties** › **LDAP** › **Triggers**.

3. Click on  .

4. Fill the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful trigger name. It must be unique for each trigger. Horizon uses the name to identify the trigger.
- **LDAP Connector Certificate Publication*** *(select)*:
  Select an LDAP connector previously created.
- **Retries in case of error** *(int)*:
  Number of times to retry to push the change on the Intune PKCS repository in case of error. Must be an integer between 1 and 15.

5. Click on the save button.

You can run  or update  or delete  the trigger.

Synchronization using triggers

Triggers are a functionality of WebRA, EST, Intune PKCS, WCCE and CRMP profiles that allows to push lifecycle events into a third party whenever they occur on a profile.

1. Refer to the trigger documentation to create a trigger.

2. Create or modify the profile you wish to use the triggers on.

3. Go to the **Triggers** tab, then on **Certificate lifecycle triggers**

4. Chose which lifecycle events you wish to use triggers upon (enrollment, revocation, expiration)

5. Select one or more existing triggers from the menu (if several are selected, they will all be called

whenever the selected event occurs)

6. Click on the Save button.

From now on, whenever a selected lifecycle event will occur on the configured profile, the trigger will be called and the and the certificate will be pushed into or removed from the third party container.

2.13. MDM

2.13.1. Intune

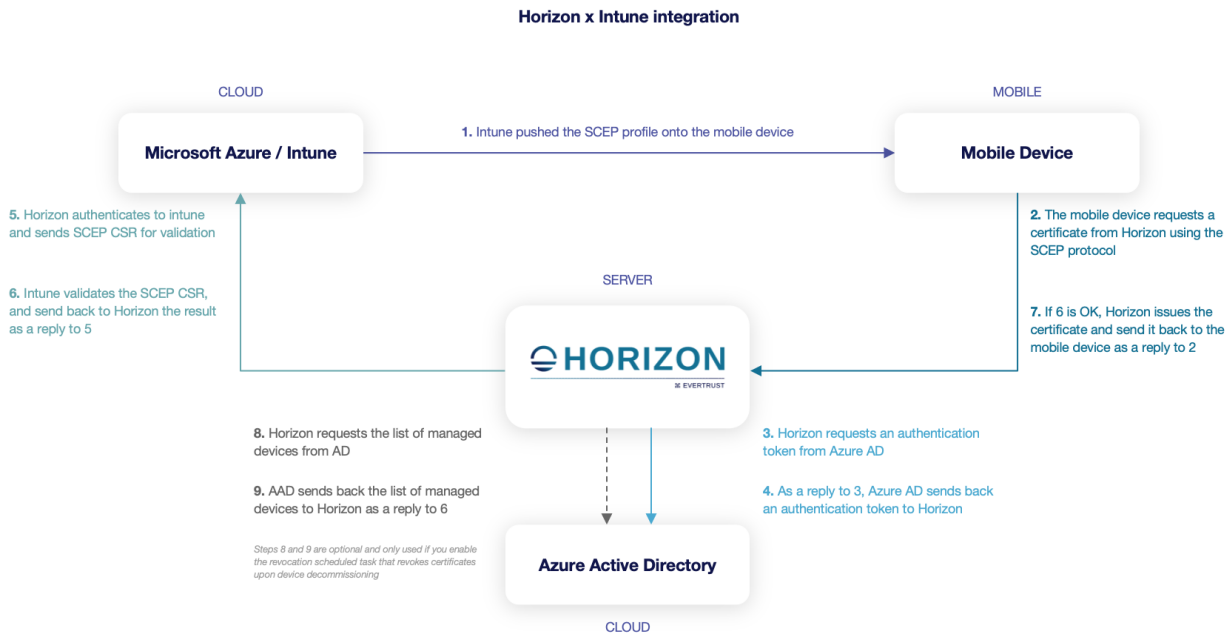
Intune Introduction

This section details the Microsoft Endpoint Manager - Intune SCEP integration with Horizon, used to enroll, renew and revoke certificates on Intune managed devices.

This integration involves at least three infrastructure components:

- Microsoft Endpoint Manager / Intune
- Azure Active Directory
- EverTrust Horizon

The enrolled devices interface with these components in order to retrieve their certificate.



The diagram displays these components as well as the various flows involved in an enrollment.

Microsoft describes the integration principles on their website: https://docs.microsoft.com/en-us/mem/intune/protect/certificate-authority-add-scep-overview

Finally, this integration will require to set up, on Horizon side, the following elements:

- an [admin-guide:third-parties-intune-intune_connector:::_intune_connector], which holds the configuration items required for Horizon to connect to Azure AD and Intune
- an [admin-guide:third-parties-intune-intune_profile:::_intune_profile], which holds the configuration items specifying how Horizon should issue certificates for the specified Intune Connector
- an Intune Scheduled Task, which holds configuration items defining the scheduled task in charge of performing revocation upon decommissioning devices from Azure AD. This is optional.

Intune Connector

This section details how to configure an Intune Connector.

Required By

[admin-guide:third-parties-intune-intune_profile:::_intune_profile]	[admin-guide:third-parties-intune-intune_connector:::_intune_connector]

Prerequisites

On Horizon side, you might need to set up a Proxy, used to reach Azure/Intune, if necessary. Note that the Horizon instance must also be reachable from the Azure AD endpoint, hence being reachable from the Internet.

On Azure AD side, it is required to set up an application by following Microsoft's guide. Please note that you must add the **Microsoft Graph** / **Application.Read.All** permission as well for the revocation feature to work properly. After performing these steps, you will get the following information, required later:

- the Tenant ID
- the Application ID
- the Application Authentication Key

How to configure Intune Connector

1. Log in to Horizon Administration Interface.

2. Access Intune Connector from the drawer or card: **MDM** › **Intune** › **Connectors**.

3. Click on  .

4. Fill the mandatory fields.

Connection

- **Name*** *(string input)*:
  Enter a meaningful connector name. It must be unique for each connector. Horizon uses the name to identify the connector.
- **Azure Tenant*** *(string input)*:
  Enter the Tenant ID.
- **App Registration Credentials***(select)*:
  Select `Login` credentials containing your app registration ID and secret key.
- **Proxy** *(string select)*:
  The HTTP/HTTPS proxy used to reach Azure AD and Intune.
- **Timeout** *(finite duration)*:
  Timeout set on the connection used to reach Azure AD and Intune. Configured by default at 10 seconds. Must be a valid finite duration.

Assets identification and management

- **OS query string** *(string input)*:
  Only used with the legacy revocation mode. Restricts by operating system the devices listed in Intune to detect the devices that are no longer managed. Leave blank to use the default setting if unsure.
- **Intune resource URL** *(string input)*:
  This allows to point at a specific Intune installation. Used only in Hybrid Intune setups, leave blank otherwise.
- **Legacy revocation mode** *(boolean)*:
  Deprecated, only kept for backward compatibility. When enabled, Horizon revokes the certificates of the devices that are no longer managed by Intune, instead of processing the revocation requests queued by Intune. Default value is set to false.

Actors management

These configuration elements mainly define the number of authorized interactions with the remote service on a defined period. For example, one needs to ensure that the remote service will not be contacted more than 5 times per 3 seconds. *Throttle parallelism* defines the number of times and *Throttle duration* the period of time. Therefore, on the above example, throttle parallelism would be set to 5 and throttle duration would be set to 3 seconds.

- **Throttle duration*** *(finite duration)*:
  Set by default to 3 seconds. Must be a valid finite duration.
- **Throttle parallelism*** *(int)*:
  Set by default to 3.

5. Click on the save button.

You can update  or delete  the Intune Connector.



You will not be able to delete an Intune Connector if it is referenced in any other configuration element.

Intune Profile

This section details how to configure an Intune Profile.

Required By

[admin-guide:third-parties-intune-intune_scheduled_task:::_intune_scheduled_tasks]

Prerequisites

[admin-guide:third-parties-intune-intune_connector:::_intune_connector]	PKI Connector	SCEP Authority

Setting up an SCEP Authority requires you to issue a certificate from the underlying PKI with the following characteristics:

- the issuing CA should be the same as the one that will issue certificates through the PKI Connector that will be linked to the Intune Profile
- the certificate key usages must include **Digital Signature** and **Key Encipherment**
- the certificate must be issued as PKCS#12 and then imported into Horizon

How to configure Intune Profile

1. Log in to Horizon Administration Interface.

2. Access Intune Profile from the drawer or card: **MDM › Intune › Profiles**.

3. Click on  .

4. Fill the mandatory fields.

Intune Profile Specific Configuration

General

- **Name*** *(string input)*:
  Enter a meaningful profile name. It must be unique for each profile. Horizon uses the name to identify the profile. As the name will be part of an URL, it is advised to use only lower case letters and dashes.
- **Enable*** *(boolean)*:
  Indicates whether the profile is enabled or not. Set to true by default.

- **Intune Connector*** *(select)*:
  Select an Intune Connector previously created.
- **PKI Connector** *(string select)*:
  Select a PKI connector previously created.

Assets identification

- **Device ID field name** *(string input)*:
  Only used with the legacy revocation mode. Subject DN field holding the Entra ID device ID, which must be set to `{{AAD_Device_ID}}` on Intune side, e.g. if you select "L", the configured Subject DN in the SCEP profile in Intune must then contain `L={{AAD_Device_ID}}`. Leave blank otherwise.
- **Device ID separator** *(string input)*:
  Only used with the legacy revocation mode. Separator used to retrieve the Device ID in the device ID field. Leave blank otherwise.

SCEP protocol parameters

- **Mode*** *(select)*:
  Choose from one of the two modes RA or CA. Usually this should be set to **RA**.
- **SCEP Authority** *(select)*:
  Select a SCEP Authority previously created. See Prerequisites for details.
- **CAPS** *(select)*:
  Select one or many SCEP Capabilities from the list. If unsure, leave the default.
- **Encryption algorithm** *(select)*:
  Select a SCEP Encryption Algorithm algorithms from the list. If unsure, leave the default.
- **Authorize POST enrollment** *(boolean)*:
  Enable SCEP enrollment with the HTTP POST method. When enabled, and if both SHA-256 and AES are selected in the CAPS, Horizon also advertises the `SCEPStandard` capability (RFC 8894). The default value is set to false.

Crypto Policy

- **Authorized Key Types** *(multiselect)*:
  Key Types that can be used for enrollment. An empty value means no restrictions.

Max Certificate per Holder Policy

- **Maximum** *(int)*:
  When specified, define the maximum number of active certificates for a given holder.
- **Behavior** *(select)*:
  What behavior to have when the maximum number is reached:
  - **revoke** the previous certificates.
  - **reject** the current request.



In order to allow renewal in `reject` behavior, one more certificate is allowed when the certificate being renewed is in its renewal period.

- **Revocation reason** *(select)*:
  When the revoke behavior is selected, the revocation reason to revoke the certificate with.

Common configuration for profiles

Languages

You can add more languages by clicking  .

- **Language*** *(select)*:
  Select a language. Supported languages are:
  - **en**: English
  - **fr**: French
- **Display Name** *(string input)*:
  Enter a display name. This will be the localized name of this profile.
- **Description** *(string input)*:
  Enter a description. This will be displayed on the list view of the profiles.

You can delete  the localization.

Grading Policies

You can select grading policies that will grade your certificate for a quick overview of its quality. For more information about the inner working of the grading policies in Horizon, please refer to the grading rules page.

Requests time to live

Configure the time your requests have before expiring.



After expiration, requests are stored for an additional 30 days. This can be changed using configuration files.

- **Enrollment request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Renewal request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Revocation request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Update request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.

- **Migration request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Recover request** *(finite duration)*:
  *Enabled on **escrow**:* Must be a valid finite duration. The default value is set to seven days.

Constraints

- **Allowed email domains** *(string input)*:
  Enter a valid regular expression that the inputted emails should match. This includes RFC822NAME and UPN SANs as well as the contact email

  This matches the domain of the email, not including anything before `@`.

- **Allowed DNS domains** *(string input)*:
  Enter a valid regular expression that the inputted domain should match.

CSR Data Mapping

1. Click on  to add a mapping.

2. Select a field and enter a value.

You can delete  the CSR Data Mapping.

Workflow

Data source flow

Configure which data sources to execute and in which order.

1. Select a data source to execute first, and fill its inputs with a computation rule.

2. Add other data sources if needed. Each datasource input can use outputs from previously executed data sources.

3. All data sources output are available in computation rules throughout the certificate template and metadata.

Datasource options

Stop on success

When this option is enabled, if the datasource data is successfully fetched, the flow will stop immediately.

This allows to search through a list of datasource, when the data can be in any of multiple datasource.

Mandatory

When this option is enabled, if the datasource return no results, the flow will stop immediately with an error.

This allows to block enrollment flows if a required record is not present, event if is not saved into the final certificate.



A datasource is considered as finding no results if:

- LDAP Datasource: no record match the filter
- DNS Datasource: no record match the query
- REST Datasource: the result code is in the `notFoundHttpCodes`

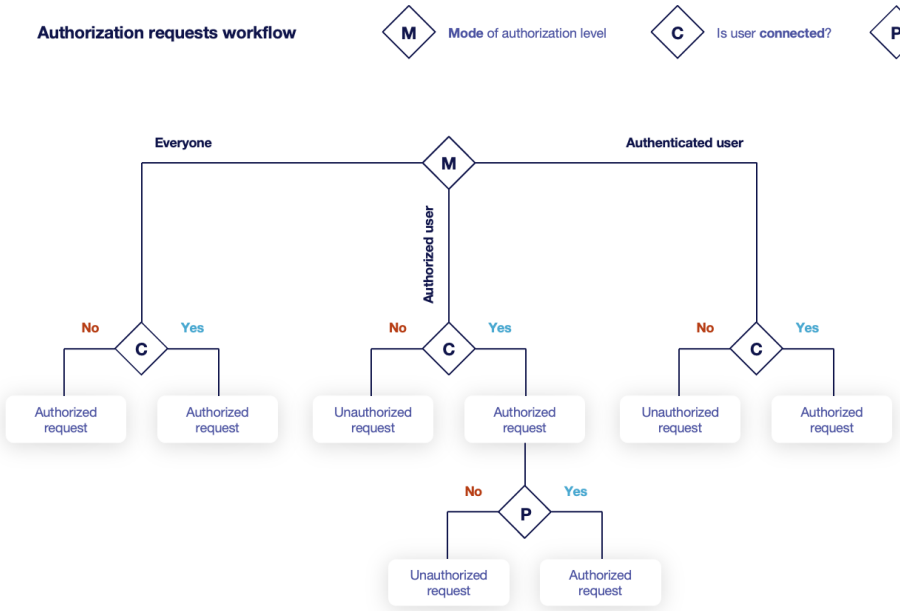


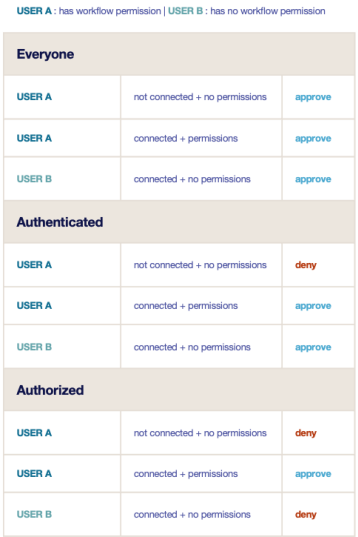
If a specific field from a record is considered as mandatory, including it in a mandatory label will also fail the enrollment flow if it is not found

Workflows builder

Configure custom rights for actions on this profile.

1. Select an authorization level for each workflow.





USER A : has workflow permission | USER B : has no workflow permission

Everyone		
USER A	not connected + no permissions	approve
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authenticated		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authorized		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	deny

- **Everyone**:
  No authentication is required.
- **Authenticated**:
  User has to be authenticated.
- **Authorized**:

User has to be authenticated and have an explicit authorizations.

2. Select an access level for identity providers.

You can remove the access level for an identity provider by clicking on 'x'.

Owner-related permissions

These permissions apply to the owners of a certificate (team or owner). An owner can always request the following actions, but this permission allows them to perform the action without validation.

- **Revoke** *(boolean)*:
  Grant self revoke permission. The default value is set to false.
- **Update** *(boolean)*:
  Grant self update permission. The default value is set to false.

Certificate Template

This section details how to define a custom structure for the fields `subject DN`, `SAN` *&* `extensions` *of the requested certificate in order to match the configuration on the PKI side.*

Subject DN composition

You can add more elements by clicking  .

- **Element*** *(select)*:
  Select an attribute from the elements list.
- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the element.
- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the Subject DN template.



When a template is defined, at least one mandatory Common Name must be added

to the DN Elements.

SAN composition

You can add more elements by clicking  .

- **Element*** *(select)*:
  Select an attribute from the element list.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Minimum** *(int)*:
  The minimum number of value that this SAN must have.
- **Maximum** *(int)*:
  The maximum number of value that this SAN must have.
- **Regex** *(regex)*:
  Enter a regular expression that the element should match.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button  or reorder (drag and drop)  the SAN template.

Extensions

You can add more elements by clicking  .

- **Element*** *(select)*:
  Select an attribute from the elements list.
- **Mandatory** *(boolean)*:
  Should the element be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the element should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the element should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the element.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this element to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can remove an element by clicking the delete button or reorder (drag and drop) the Extensions template.



When adding a SAN, a DN element or an Extension and making it mandatory, make sure to either give it a default value or a computation rule or make it editable, otherwise the template will be unusable.

Certificate Metadata

This section details how to define a custom structure for the labels, ownership policy and technical metadata, allowing certificates to hold rich information.

Labels

You can add more labels by clicking .

- **Name** *(select)*:
  Select a preexisting label.
- **Mandatory** *(boolean)*:
  Should the label be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the label should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the label should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the label.
- **Label value restriction**
  - **Whitelist** *(string input multiple)*:
    The label value will have to be in the whitelist. Open the popup, enter the label value and press "enter" to add this value to the accepted value list. An empty whitelist means no restriction.
  - **Suggestions** *(string input multiple)*:
    Add suggestions that will be displayed to the user. The user will be able to choose one of these values or enter its own. Open the popup, enter your suggestions and press enter to add this value to the suggestions. An empty suggestions list means no restriction.
  - **Regex** *(regex)*:
    The label value will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this label to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can delete or reorder (drag and drop) the label template.

Ownership policy

- **Owner**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's owner is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when approving a request.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the owner to the value of the evaluated computation rule. This value will override any other value including the user input.
- **Contact email**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's contact email is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when approving a request.
  - **Default contact email** *(string input)*:
    Set a default contact email. This value must comply with the contact email restriction.
  - **Contact email restriction**
    - **Whitelist** *(string input multiple)*:
      The contact email will have to be in the whitelist. Open the popup, enter the email and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The contact email will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the contact email to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.
- **Team**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's team is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when submitting a

request.

- **Editable by approver** *(boolean)*:
  Specify if the certificate's team can be overridden by the requester when approving a request.
- **Default team** *(string input)*:
  Set a default team. This value must comply with the team restriction.
- **Team restriction**
  - **Whitelist** *(string input multiple)*:
    The team will have to be in the whitelist. Enter the team and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
  - **Regex** *(regex)*:
    The team will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of the team to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

Metadata policy *(overridable metadata)*



These metadata are technical metadata. They are used by Horizon or Third party connectors, updating them should be done with utmost care.



Metadata edition is not allowed on enroll.



Metadata edition is not available via the User Interface. It must be changed with API, using horizon-cli.

You can allow the override of technical metadata by clicking .

- **Metadata*** *(select)*:
  Select a metadata.
- **Editable by requester** *(boolean)*:
  Tells whether the metadata is editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the metadata is editable by the approver. The default value is set to false.

You can delete  a metadata policy. This will not delete the metadata but will make it non editable.

Third-party policy

- **Synchronize data from discovery** *(boolean)*:
  Enable overriding of third-party data during discovery. When enabled, the following logic

applies:

- If a discovery attempts to set third-party data on a connector that is not referenced in the profile (by a trigger), the data will not be added.
- If the connector is referenced in the profile, the discovery will override the existing data.

Notifications

This section details how to configure notifications on certificate and request lifecycle events.

Certificate lifecycle notifications

Notifications are sent when one of the following event is triggered by a certificate:

Enrollment	Revocation	Expire	Update	Migrate	Renew

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.

Request lifecycle notifications

Notifications are sent when one of the following event is triggered by an Enroll/Revocation/Update/Migrate/Renew request:

Submit	Cancel	Revoke	Approve	Pending

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.



Submit request events are not triggered when the user has the permission to perform the action directly.

5. Click on the save button.

You can update  or delete  the Intune Profile once it has been created.



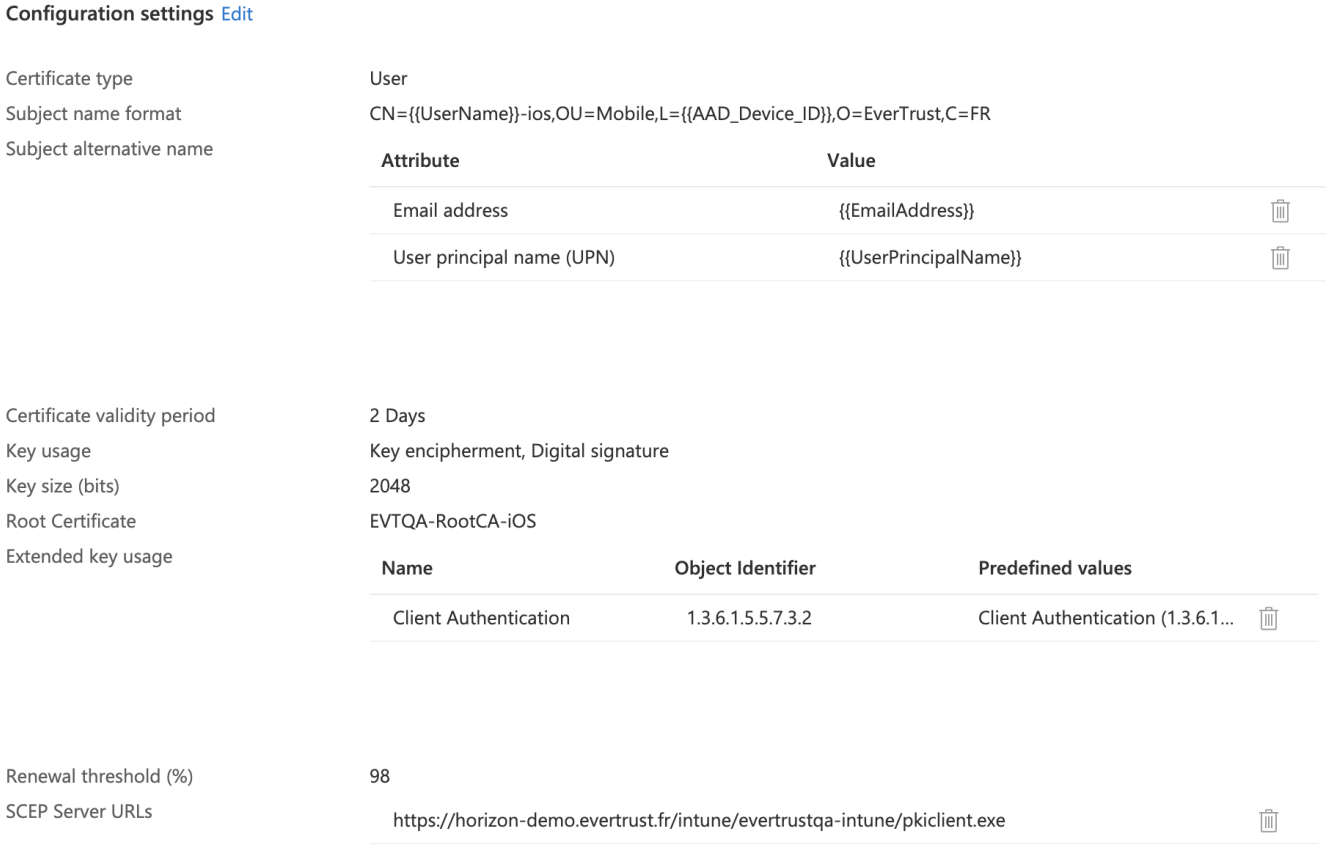
You won't be able to delete an Intune Profile if it is referenced somewhere else.

Last steps

Once the profile is created in Horizon, you need to setup a SCEP profile in Intune by following Microsoft documentation. You will need to match the parameters in the Intune SCEP profile with what has been set up in Horizon and in the underlying PKI. You need to pay special attention to:

- the certificate lifetime and renewal interval, which must match throughout the solution
- the Subject and Subject Alternative Name settings must match throughout the solution. In the end, the issued certificate must contain exactly what was configured in Intune for these fields, or the renewal will not work.

- the SCEP server URL, where you need to input the URL given in the Intune Profile that you created in Horizon





To enroll **Windows** machines or users using Intune, you need to remove the trailing `/pkiclient.exe` (including the slash) from the SCEP server URL, since the Windows SCEP client appends it to the configured URL

Intune Scheduled Tasks

This section details how to configure scheduled tasks which will run periodically on your Intune profiles, in order to manage automatic revocation upon device decommission.

Prerequisites

[admin-guide:third-parties-intune-intune_connector:::_intune_connector]	[admin-guide:third-parties-intune-intune_profile:::_intune_profile]

How to configure Intune Scheduled Tasks

1. Log in to Horizon Administration Interface.

2. Access Intune Scheduled Tasks from the drawer or card: **MDM › Intune › Scheduled Tasks**.

3. Click on  .

4. Fill the mandatory fields.

- **Intune Profile*** *(select)*:
  Select an Intune profile previously created.
- **Target Connector*** *(select)*:
  Select an Intune connector previously created.
- **Cron scheduling** *(cron expression)*:
  Set to every 5 hours by default.
- **Revoke** *(boolean)*:
  Set to true by default. If true, Horizon retrieves the revocation requests that Intune queues for decommissioned devices (device retired, wiped or deleted in Intune) and revokes the matching certificates.
- **Dry run** *(boolean)*:
  If enabled, revocation actions will not be performed. Instead, a message will be logged, explaining what would have been done.

5. Click on the save button.

You can run  , update  or delete  the Scheduled Tasks.

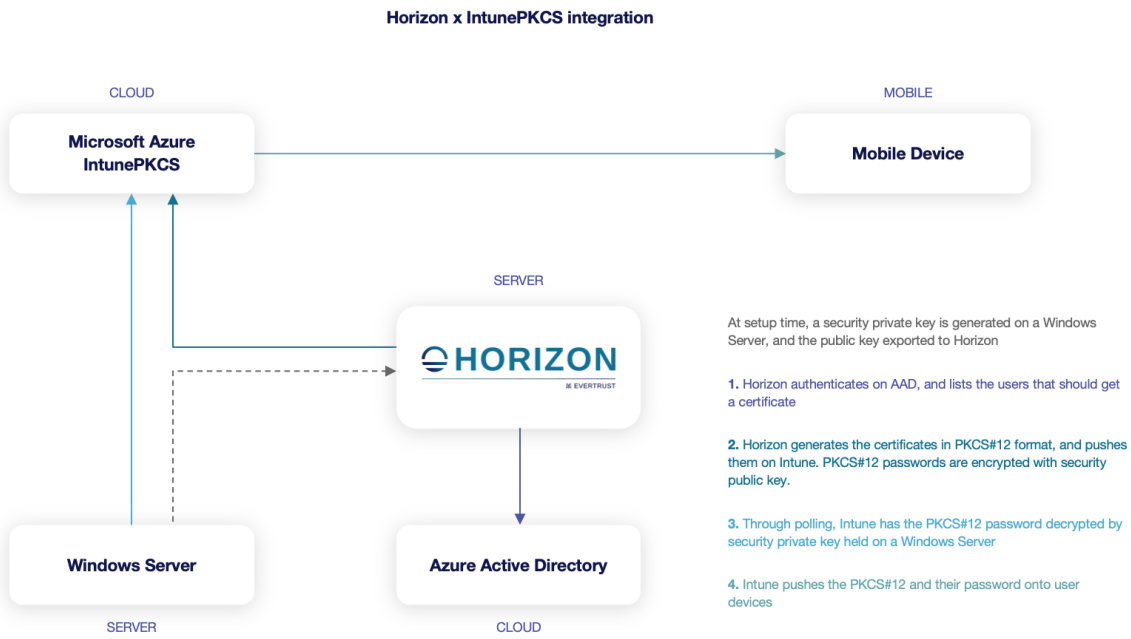
2.13.2. Intune PKCS

Intune PKCS Introduction

This integration involves at least three infrastructure components:

- Microsoft Endpoint Manager / Intune
- Azure Active Directory
- EverTrust Horizon

The enrolled devices interface with these components in order to retrieve their certificate.



The diagram displays these components as well as the various flows involved in an enrollment. The integration is further explained in the Microsoft Intune PKCS documentation.

Intune PKCS Connector

This section details how to configure the Intune PKCS Connector.

Required By

[admin-guide:third-parties-intune_pkcs-intune_pkcs_profile:::_intune_pkcs_profile]

How to configure Intune PKCS Connector

Configuring the Microsoft Certificate Connector

The first step of the Intune PKCS connector is to actually understand the workflow that it bears, explained in introduction. Working with IntunePKCS requires the Microsoft Certificate Connector MSI to be uploaded to any Windows machine connected to the Internet. This connector is available on the Microsoft Documentation.

1. Run the certificate connector MSI on the machine and click on "Configure now". Configure the connector to fit your infrastructure, just remember to only check the *PKCS imported* box whenever prompted. This step should end with a connection to Azure;

2. Retrieve the **Horizon Key Manager** from Horizon and upload it to the same machine where the **Microsoft Certificate Connector** was installed;

3. Open a command-line prompt as Administrator;

4. Generate an import key through the command-line tool:

```
$ PKCSImport.exe generate [KeyName]
```

Replace [KeyName] with what you want to name your key as. The next steps of the documentation will assume that the name is set to "PKCSImportKey".

5. Use the tool to export the generated key:

```
$ PKCSImport.exe export PKCSImportKey PKCSImportKey.pub
```

This will export the public key part of the PKCS Import Key to the *PKCSImportKey.pub* file as base64 format.

Configuring the IntunePKCS Connector in Horizon

This step assumes that the previous one has been thoroughly followed. The only extra pre-requisite for this step is to retrieve the Azure resource ID of the group that will be using the escrowed certificates. Note that the app registration for Horizon must have the "*DeviceManagementConfiguration.ReadWrite.All*" permission granted as tenant admin. Read more about that in the Microsoft documentation

1. Log in to Horizon Administration Interface.

2. Access Intune PKCS Connectors from the drawer or card: **MDM › Intune PKCS › Connectors**.

3. Click on  .

4. Fill the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful connector name. It must be unique for each connector. Horizon uses the name to identify the connector.
- **Azure Tenant*** *(string input)*:
  Value must be set to the Azure tenant.
- **App Registration Credentials***(select)*:
  Select `Login` credentials containing your app registration ID and secret key. The app registration must have the "*DeviceManagementConfiguration.ReadWrite.All*" permission granted as tenant admin.
- **Proxy** *(string select)*:
  The HTTP/HTTPS proxy to use.
- **Timeout** *(finite duration)*:
  Set by default at 10 seconds. Must be a valid finite duration.
- **Search Filter** *(string input)*:
  This value must be set to : groups/{Azure AD group object ID}/members This will apply the PKCS

Import policy to all the members of the referenced Azure AD group object ID (the one that was retrieved at the beginning of the step).

- **Max stored certificates per holder** *(int)*:
  When specified, define the maximum number of certificates stored in the third party for a given holder. As an example, when set to 2, Intune will store the current certificate as well as the previous one (whether expired or revoked), so that it can still be used to decrypt resources. When a third one is going to be enrolled, the older one will be flushed out of Intune.

Assets identification and management

- **Key Name** *(string input)*:
  Enter the key name that was specified in the **Horizon Key Manager** (*PKCSImportKey* in the example).
- **Key Type** *(select)*:
  Select one key type from the list. If the **Horizon Key Manager** was used, select **RSA-2048**.
- **Provider Name** *(string input)*:
  Enter provider name. If the **Horizon Key Manager** was used, leave it blank.
- **Public Key** *(string input)*:
  Paste the base64 exported public key generated at step 5 of the previous part.
- **Intended Purpose** *(select)*:
  Select one intended certificate usage from the list. As an example, if you want to use the escrowed certificates through this connector to encrypt email, select S/MIME.

Actors and renewal management

These configuration elements mainly define the number of authorized interactions with the remote service on a defined period. For example, one needs to ensure that the remote service will not be contacted more than 5 times per 3 seconds. *Throttle parallelism* defines the number of times and *Throttle duration* the period of time. Therefore, on the above example, throttle parallelism would be set to 5 and throttle duration would be set to 3 seconds.

- **Throttle duration*** *(finite duration)*:
  Set by default at 3 seconds. Must be a valid finite duration.
- **Throttle parallelism*** *(int)*:
  Set by default at 3.

5. Click on the save button.

You can update  or delete  the Intune PKCS Connector.



You won't be able to delete an Intune PKCS Connector if it is referenced in any other configuration element.

Connector actions

The following actions are available once the connector is configured

- **Retry failed triggers**: This will run every retryable trigger linked to this connector. A trigger is retryable if it failed and all required parameters for it (like the private key) are still available.



This action could trigger a lot of retries.

Intune PKCS Profile

This section details how to configure the Intune PKCS Profile

Required By

[admin-guide:third-parties-intune_pkcs-intune_pkcs_scheduled_tasks:::_intune_pkcs_scheduled_tasks]

Prerequisites

PKI Connector

How to configure Intune PKCS Profile

1. Log in to Horizon Administration Interface.

2. Access Intune PKCS Profiles from the drawer or card: **MDM** › **Intune PKCS** › **Profiles**.

3. Click on  .

4. Fill the mandatory fields.

Intune PKCS Profile Specific Configuration

General

- **Name*** *(string input)*:
  Enter a meaningful profile name. It must be unique for each profile. Horizon uses the name to identify the profile.
- **Enable*** *(boolean)*:
  Is the profile enabled or not. Set at true by default.
- **PKI Connector** *(string select)*:
  Select a PKI connector previously created. CAUTION: The selected PKI connector must support the msUPN SAN and, if used for S/MIME encryption, the RFC822NAME SAN (email).
- **Intune PKCS Connector*** *(select)*:
  Select an Intune PKCS Connector previously created.

Crypto Policy

- **Default Key Type** *(select)*:
  Select the default type of key to generate when using centralized enrollment mode.
- **Authorized Key Types** *(multiselect)*:
  Key Types that can be used for enrollment. An empty value means no restrictions.
- **Private key escrowing** *(boolean)*:
  Tells whether the private key should be escrowed by Horizon. The default value is set to false.
  - **Show PKCS#12 Password On Recover** *(boolean)*:
    Tells whether the PKCS#12 password should be displayed on recover. The default value is set to false.
  - **Show PKCS#12 On Recover** *(boolean)*:
    Tells whether the PKCS#12 should be displayed on recover. The default value is set to false.
- **PKCS#12 Password Mode*** *(select)*:
  Select how to generate PKCS#12 password:
  - **manual**: prompt the user to choose its password. This is the default behavior.
  - **random**: have the password generated on Horizon side.
- **Password policy** *(select)*:
  Select a previously created password policy. It will be enforced on PKCS#12 password for recovery and centralized enrollments.
- **Store encryption type*** *(select)*:
  Select an encryption algorithm from the list. The PKCS#12 will use this algorithm. The default value is set to DES_AVERAGE.

Max Certificate per Holder Policy

- **Maximum** *(int)*:
  When specified, define the maximum number of active certificates for a given holder.
- **Behavior** *(select)*:
  What behavior to have when the maximum number is reached:
  - **revoke** the previous certificates.
  - **reject** the current request.



In order to allow renewal in `reject` behavior, one more certificate is allowed when the certificate being renewed is in its renewal period.

- **Revocation reason** *(select)*:
  When the revoke behavior is selected, the revocation reason to revoke the certificate with.

Common configuration for profiles

Languages

You can add more languages by clicking ⊕.

- **Language*** *(select)*:
  Select a language. Supported languages are:
  - **en**: English
  - **fr**: French
- **Display Name** *(string input)*:
  Enter a display name. This will be the localized name of this profile.
- **Description** *(string input)*:
  Enter a description. This will be displayed on the list view of the profiles.

You can delete 🗑 the localization.

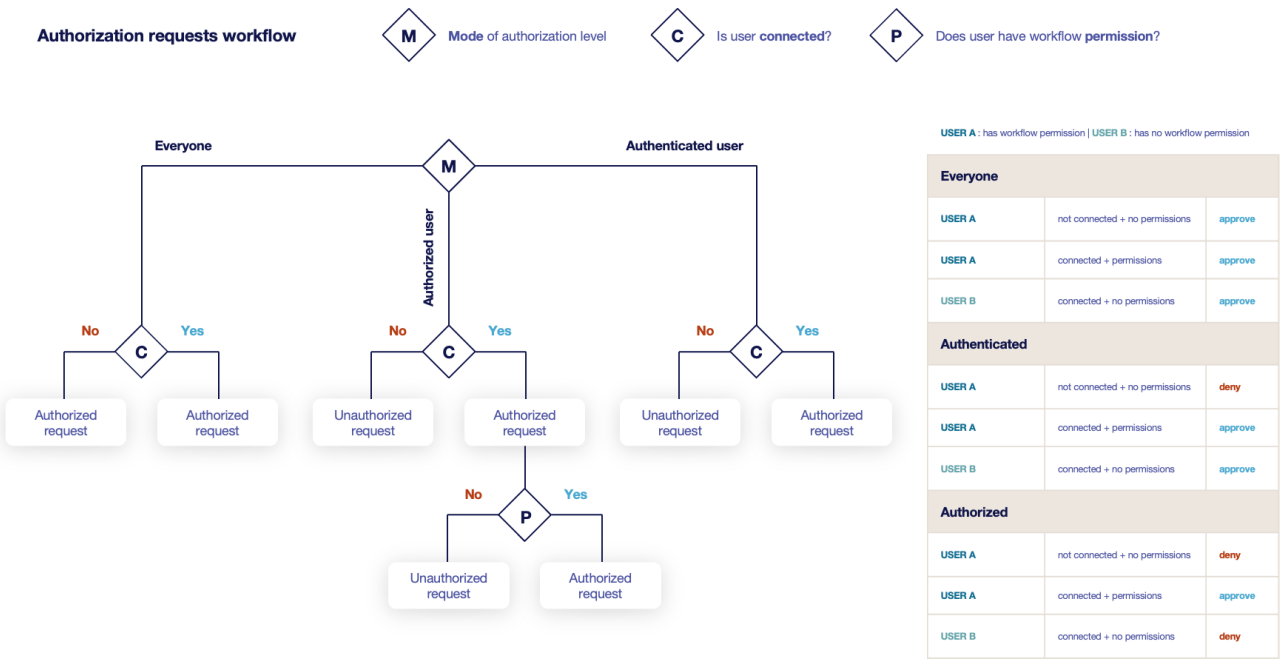
Grading Policies

You can select grading policies that will grade your certificate for a quick overview of its quality. For more information about the inner working of the grading policies in Horizon, please refer to the grading rules page.

Workflows builder

Configure custom rights for actions on this profile.

1. Select an authorization level for each workflow.



- **Everyone**:

No authentication is required.

- **Authenticated**:
  User has to be authenticated.
- **Authorized**:
  User has to be authenticated and have an explicit authorizations.

2. Select an access level for identity providers.

You can remove the access level for an identity provider by clicking on 'x'.

Requests time to live

Configure the time your requests have before expiring.



After expiration, requests are stored for an additional 30 days. This can be changed using configuration files.

- **Enrollment request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Renewal request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Revocation request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Update request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Migration request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Recover request** *(finite duration)*:
  *Enabled on **escrow**:* Must be a valid finite duration. The default value is set to seven days.

Owner-related permission

These permissions apply to the owners of a certificate (team or owner). An owner can always request the following actions, but this permission allows them to perform the action without validation.

- **Revoke** *(boolean)*:
  Grant self revoke permission. The default value is set to false.
- **Recover** *(boolean)*:
  Grant self recover permission. The default value is set to false.
- **Update** *(boolean)*:
  Grant self update permission. The default value is set to false.

Constraints

- **Allowed email domains** *(string input)*:

Enter a valid regular expression that the inputted emails should match. This includes RFC822NAME and UPN SANs as well as the contact email



This matches the domain of the email, not including anything before `@`.

- **Allowed DNS domains** *(string input)*:
  Enter a valid regular expression that the inputted domain should match.

CSR Data Mapping

1. Click on  to add a mapping.

2. Select a field and enter a value.

You can delete  the CSR Data Mapping.

Certificate Metadata

This section details how to define a custom structure for the labels, ownership policy and technical metadata, allowing certificates to hold rich information.

Labels

You can add more labels by clicking .

- **Name** *(select)*:
  Select a preexisting label.
- **Mandatory** *(boolean)*:
  Should the label be mandatory. The default value is set to false.
- **Editable by requester** *(boolean)*:
  Tells whether the label should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the label should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the label.
- **Label value restriction**
  - **Whitelist** *(string input multiple)*:
    The label value will have to be in the whitelist. Open the popup, enter the label value and press "enter" to add this value to the accepted value list. An empty whitelist means no restriction.
  - **Suggestions** *(string input multiple)*:
    Add suggestions that will be displayed to the user. The user will be able to choose one of these values or enter its own. Open the popup, enter your suggestions and press enter to add this value to the suggestions. An empty suggestions list means no restriction.

- **Regex** *(regex)*:
  The label value will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.

- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this label to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can delete  or reorder (drag and drop)  the label template.

Ownership policy

- **Owner**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's owner is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when approving a request.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the owner to the value of the evaluated computation rule. This value will override any other value including the user input.
- **Contact email**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's contact email is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's contact email can be overridden by the requester when approving a request.
  - **Default contact email** *(string input)*:
    Set a default contact email. This value must comply with the contact email restriction.
  - **Contact email restriction**
    - **Whitelist** *(string input multiple)*:
      The contact email will have to be in the whitelist. Open the popup, enter the email and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The contact email will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no

restrictions.

- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of the contact email to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

- **Team**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's team is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when approving a request.
  - **Default team** *(string input)*:
    Set a default team. This value must comply with the team restriction.
  - **Team restriction**
    - **Whitelist** *(string input multiple)*:
      The team will have to be in the whitelist. Enter the team and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The team will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the team to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

Metadata policy *(overridable metadata)*



These metadata are technical metadata. They are used by Horizon or Third party connectors, updating them should be done with utmost care.



Metadata edition is not allowed on enroll.



Metadata edition is not available via the User Interface. It must be changed with API, using horizon-cli.

You can allow the override of technical metadata by clicking .

- **Metadata*** *(select)*:
  Select a metadata.
- **Editable by requester** *(boolean)*:
  Tells whether the metadata is editable by the requester. The default value is set to false.

- **Editable by approver** *(boolean)*:
  Tells whether the metadata is editable by the approver. The default value is set to false.

You can delete  a metadata policy. This will not delete the metadata but will make it non editable.

Third-party policy

- **Synchronize data from discovery** *(boolean)*:
  Enable overriding of third-party data during discovery. When enabled, the following logic applies:
  - If a discovery attempts to set third-party data on a connector that is not referenced in the profile (by a trigger), the data will not be added.
  - If the connector is referenced in the profile, the discovery will override the existing data.

Notifications/Triggers

This section details how to configure notifications and triggers to perform actions on certificate and request lifecycle events.

Certificate lifecycle notifications

Notifications are sent when one of the following event is triggered by a certificate:

Enrollment	Revocation	Expire	Update	Migrate	Renew

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.

Request lifecycle notifications

Notifications are sent when one of the following event is triggered by an Enroll/Revocation/Update/Migrate/Renew request:

Submit	Cancel	Revoke	Approve	Pending

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.



Submit request events are not triggered when the user has the permission to perform the action directly.

Triggers

Horizon support the use of third-party triggers in the form of callbacks on specific events happening on the profile, giving a way to synchronize the third party repositories and Horizon.

- **Enrollment** *(select)*:

Select the preexisting third party or MDM trigger(s) to call whenever a certificate is enrolled on this profile.

- **Renewal** *(select)*:
  Select the preexisting third party or MDM trigger(s) to call whenever a certificate is renewed on this profile.
- **Revocation** *(select)*:
  Select the preexisting third party or MDM trigger(s) to call whenever a certificate gets revoked on this profile.
- **Expire** *(select)*:
  Select the preexisting third party or MDM trigger(s) to call whenever a certificate expires on this profile.

The available triggers are the following:

AKV Triggers	AWS Triggers	F5 Triggers	F5 AS3 Triggers
Netscaler Triggers	GCM Triggers	[admin-guide:third-parties-fortinet-triggers:::_fortinet_triggers]	[admin-guide:third-parties-paloalto-triggers:::_paloalto_triggers]
[admin-guide:third-parties-ldap-triggers:::_ldap_triggers]	Intune PKCS Triggers		

5. Click on the save button.

You can update  or delete  the Intune PKCS Profile.



You won't be able to delete an Intune PKCS Profile if it is referenced in any other configuration element.

Intune PKCS Scheduled Tasks

This section details how to schedule tasks that will run periodically on your Intune PKCS profiles.

Prerequisites

[admin-guide:third-parties-intune_pkcs-intune_pkcs_connector:::_intune_pkcs_connector]	[admin-guide:third-parties-intune_pkcs-intune_pkcs_profile:::_intune_pkcs_profile]

How to configure Intune PKCS Scheduled Tasks

1. Log in to Horizon Administration Interface.

2. Access Intune PKCS Scheduled Tasks from the drawer or card: **MDM › Intune PKCS › Scheduled**

Tasks.

3. Click on  .

4. Fill the mandatory fields.

- **Enable** *(boolean)*:
  Tells whether the Scheduled task should be enabled. Set by default at true.
- **Intune PKCS Profile*** *(select)*:
  Select an Intune PKCS profile previously created.
- **Target Connector*** *(select)*:
  Select an Intune PKCS connector previously created.
- **Cron scheduling** *(cron expression)*:
  By default set at every 5 hours.
- **Enroll?** *(boolean)*:
  If enabled, will enroll all certificate from the third party repository. Set to false by default.
- **Revoke?** *(boolean)*:
  If enabled, will revoke all certificate whose container was deleted from the third party repository. Set to false by default.
- **Renew?** *(boolean)*:
  If enabled, will renew all certificate who are about to expire. Set to false by default.
- **Dry run** *(boolean)*:
  If enabled, enroll, revocation and renewal actions will not be performed. Instead, a message will be logged, explaining what would have been done.

5. Click on the save button.

You can run  or update  or delete  the Schedules Tasks.

Intune PKCS Trigger

This section details how to configure the Triggers that will run automatically on your Intune PKCS connectors.

Prerequisites

[admin-guide:third-parties-intune_pkcs-intune_pkcs_connector:::_intune_pkcs_connector]

How to configure Intune PKCS Trigger

1. Log in to Horizon Administration Interface.

2. Access Intune PKCS Triggers from the drawer or card: **MDM** › **Intune PKCS** › **Triggers**.

3. Click on  .

4. Fill the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful trigger name. It must be unique for each trigger. Horizon uses the name to identify the trigger.
- **Intune PKCS Connector*** *(select)*:
  Select an Intune PKCS connector previously created.
- **Retries in case of error** *(int)*:
  Number of times to retry to push the change on the Intune PKCS repository in case of error. Must be an integer between 1 and 15.

5. Click on the save button.

You can update  or delete  the Intune PKCS Trigger.



You won't be able to delete an Intune PKCS Trigger if it is referenced in any other configuration element.

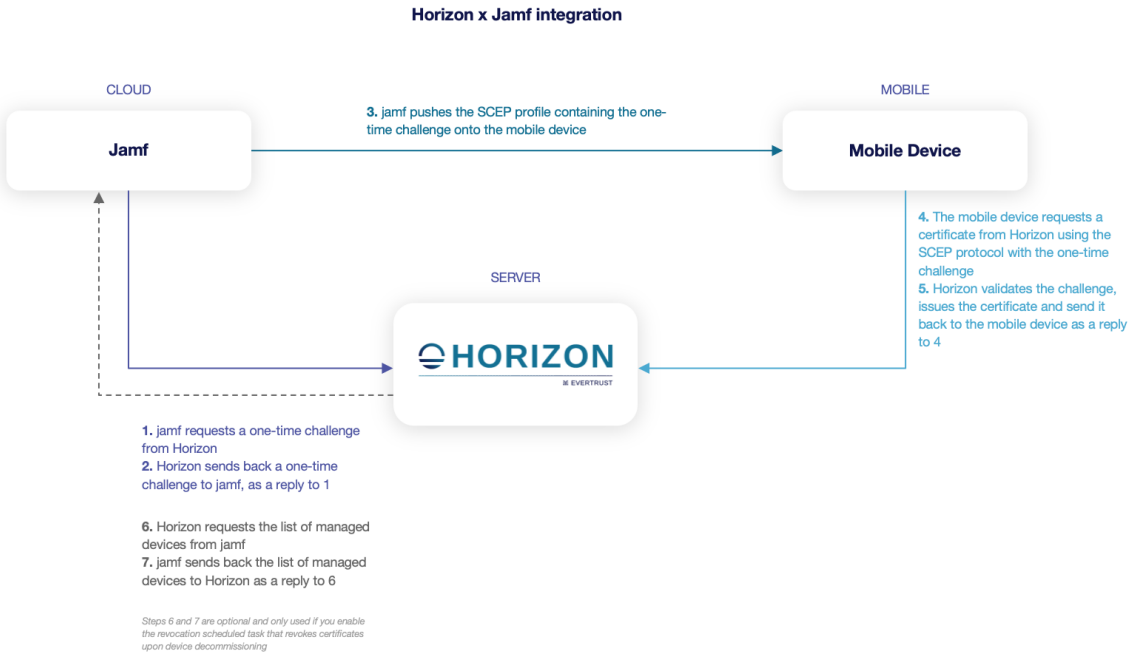
2.13.3. Jamf

Jamf Introduction

This section details the Jamf Pro integration with Horizon, used to enroll, renew and revoke certificates on Jamf Pro managed devices.

This integration involves the following components:

- Jamf Pro server or Cloud instance
- EverTrust Horizon
- Devices to be enrolled



The diagram displays these components as well as the various flows involved in an enrollment.

Finally, this integration will require to setup, on Horizon side, the following elements:

- a [admin-guide:third-parties-jamf-jamf_connector:::_jamf_connector], which holds the configuration items required for Horizon to connect to Jamf Pro
- a [admin-guide:third-parties-jamf-jamf_profile:::_jamf_profile], which holds the configuration items specifying how Horizon should issue certificates for the specified Jamf Connector
- a Jamf Schedule Task, which holds configuration items defining the scheduled task in charge of performing revocation upon decommissioning devices from Jamf Pro. This is optional.

Jamf Connector

This section details how to configure a Jamf Connector.

Required By

[admin-guide:third-parties-jamf-jamf_profile:::_jamf_profile]

Prerequisites

On Horizon side, you might need to set up a Proxy used to reach Jamf Pro, if necessary.

On Jamf Pro side, it is required to create a technical user for Horizon, and give it **Auditor** rights, so that Horizon will be able to list the devices managed by Jamf Pro and thus be able to trigger certificate revocation upon decommissioning. Please follow the steps from the Jamf Pro documentation. After performing these steps, you will be given the following information, required later:

- a login
- a password

How to configure Jamf Connector

1. Log in to Horizon Administration Interface.

2. Access Jamf Connector from the drawer or card: **MDM › Jamf › Connectors**.

3. Click on  .

4. Fill the mandatory fields.

Connection

- **Name*** *(string input)*:
  Enter a meaningful connector name. It must be unique for each connector. Horizon uses the name to identify the connector.
- **Jamf endpoint URL*** *(string input)*:
  Enter the URL pointing to the Jamf deployment or the Jamf Cloud instance.
- **Jamf user account credentials*** *(select)*:
  Select `Login` credentials containing the username and password created for Horizon in Jamf.
- **Proxy** *(string select)*:
  The HTTP/HTTPS proxy used to reach Jamf Pro, if any.
- **Timeout** *(finite duration)*:
  Set by default at 10 seconds. Must be a valid finite duration.

Actors management

These configuration elements mainly define the number of authorized interactions with the remote service on a defined period. For example, one needs to ensure that the remote service will not be contacted more than 5 times per 3 seconds. *Throttle parallelism* defines the number of times and *Throttle duration* the period of time. Therefore, on the above example, throttle parallelism would be set to 5 and throttle duration would be set to 3 seconds.

- **Throttle duration*** *(finite duration)*:
  Set by default to 3 seconds. Must be a valid finite duration.
- **Throttle parallelism*** *(int)*:
  Set by default to 3.

5. Click on the save button.

You can update  or delete  the Jamf Connector.



You won't be able to delete a Jamf Connector if it is referenced in any other configuration element.

Jamf Profile

This section details how to configure a Jamf Profile

Prerequisites

[admin-guide:third-parties-jamf-jamf_connector:::_jamf_connector]	PKI Connector	SCEP Authority

The SCEP Authority setup requires you to issue a certificate from the underlying PKI with the following characteristics:

- in RA mode (iOS and macOS devices):
  - the issuing CA should be the same as the one that will issue certificates through the PKI Connector that will be linked to the Jamf Profile
  - the certificate key usages must include **Digital Signature** and **Key Encipherment**
  - the certificate must be issued as PKCS#12 and then imported into Horizon
- in CA mode (older macOS versions that do not support the RA mode):
  - the certificate should be self-signed
  - the certificate key usages must include **Digital Signature** and **Key Encipherment**
  - the certificate must be issued as PKCS#12 and then imported into Horizon

How to configure Jamf Profile

1. Log in to Horizon Administration Interface.

2. Access Jamf Profiles from the drawer or card: **MDM** › **Jamf** › **Profiles**.

3. Click on  .

4. Fill the mandatory fields.

Jamf Profile Specific Configuration

General

- **Name*** *(string input)*:
  Enter a meaningful profile name. It must be unique for each profile. Horizon uses the name to identify the profile. As the name will be part of an URL, it is advised to use only lower case letters and dashes.
- **Enable** *(boolean)*:
  Is the profile enabled or not. Set at true by default.
- **Jamf Connector** *(select)*:

Select a Jamf connector previously created.

- **PKI connector*** *(string select)*:
  Select a PKI connector previously created.

Assets identification

- **DN field containing the device `UDID`*** *(select)*:
  Field used to retrieve the Device ID. The selected field must be set to `$UDID/$COMPUTERNAME` on Jamf side, e.g. if you select "L", the configured Subject DN in the SCEP profile in Jamf pro must then contain `L=$UDID` for iOS or `L=$COMPUTERNAME` for macOS devices. This allows to use the automated revocation upon device decommissioning feature.

SCEP protocol parameters

- **Mode*** *(select)*:
  Choose from the two modes RA or CA. Select the **RA** mode to enroll certificates on **iOS** and **macOS** devices. The **CA** mode is only kept for older macOS versions that do not support the RA mode.
- **SCEP Authority*** *(select)*:
  Select a SCEP Authority previously created. See Prerequisites for details.
- **CAPS** *(select)*:
  Select one or many SCEP Capabilities from the list. If unsure, leave the default.
- **Encryption algorithm*** *(select)*:
  Select a SCEP Encryption Algorithm algorithms from the list. If unsure, leave the default.
- **Authorize POST enrollment** *(boolean)*:
  Enable SCEP enrollment with the HTTP POST method. When enabled, and if both SHA-256 and AES are selected in the CAPS, Horizon also advertises the `SCEPStandard` capability (RFC 8894). The default value is set to false.
- **Password policy** *(select)*:
  Choose from the password policy you might have previously created. If unsure, leave the default.

Crypto Policy

- **Authorized Key Types** *(multiselect)*:
  Key Types that can be used for enrollment. An empty value means no restrictions.

Max Certificate per Holder Policy

- **Maximum** *(int)*:
  When specified, define the maximum number of active certificates for a given holder.
- **Behavior** *(select)*:
  What behavior to have when the maximum number is reached:
  - **revoke** the previous certificates.
  - **reject** the current request.



In order to allow renewal in `reject` behavior, one more certificate is allowed when the certificate being renewed is in its renewal period.

- **Revocation reason** *(select)*:
  When the revoke behavior is selected, the revocation reason to revoke the certificate with.

Common configuration for profiles

Languages

You can add more languages by clicking  .

- **Language*** *(select)*:
  Select a language. Supported languages are:
  - **en**: English
  - **fr**: French
- **Display Name** *(string input)*:
  Enter a display name. This will be the localized name of this profile.
- **Description** *(string input)*:
  Enter a description. This will be displayed on the list view of the profiles.

You can delete  the localization.

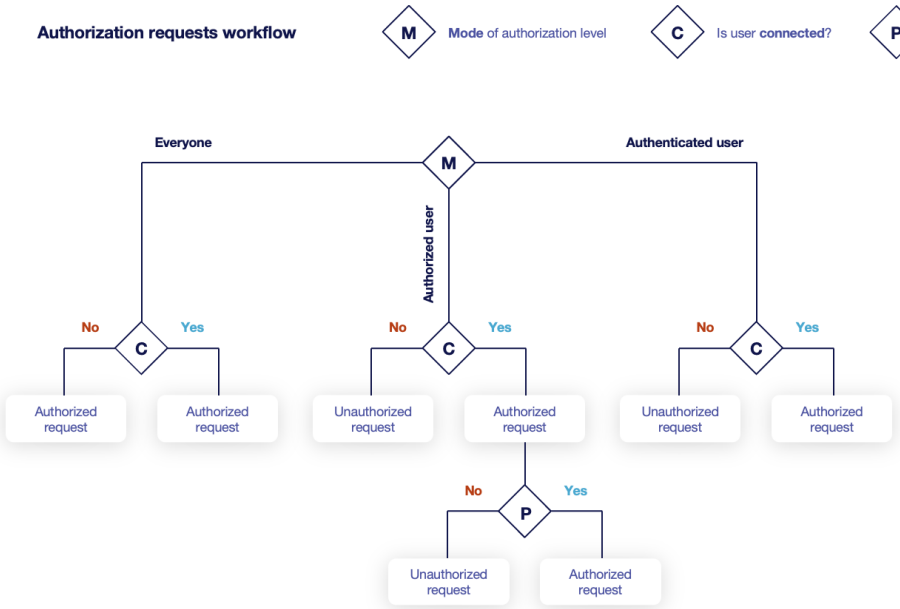
Grading Policies

You can select grading policies that will grade your certificate for a quick overview of its quality. For more information about the inner working of the grading policies in Horizon, please refer to the grading rules page.

Workflows builder

Configure custom rights for actions on this profile.

1. Select an authorization level for each workflow.



USER A : has workflow permission | USER B : has no workflow permission

Everyone		
USER A	not connected + no permissions	approve
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authenticated		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	approve
Authorized		
USER A	not connected + no permissions	deny
USER A	connected + permissions	approve
USER B	connected + no permissions	deny

- **Everyone**:
  No authentication is required.
- **Authenticated**:
  User has to be authenticated.
- **Authorized**:
  User has to be authenticated and have an explicit authorizations.

2. Select an access level for identity providers.

You can remove the access level for an identity provider by clicking on 'x'.

Requests time to live

Configure the time your requests have before expiring.



After expiration, requests are stored for an additional 30 days. This can be changed using configuration files.

- **Enrollment request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Renewal request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Revocation request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Update request*** *(finite duration)*:
  Must be a valid finite duration. The default value is set to seven days.
- **Migration request*** *(finite duration)*:

Must be a valid finite duration. The default value is set to seven days.

- **Recover request** *(finite duration)*:
  *Enabled on **escrow**:* Must be a valid finite duration. The default value is set to seven days.

Owner-related permission

These permissions apply to the owners of a certificate (team or owner). An owner can always request the following actions, but this permission allows them to perform the action without validation.

- **Revoke** *(boolean)*:
  Grant self revoke permission. The default value is set to false.
- **Update** *(boolean)*:
  Grant self update permission. The default value is set to false.

Constraints

- **Allowed email domains** *(string input)*:
  Enter a valid regular expression that the inputted emails should match. This includes RFC822NAME and UPN SANs as well as the contact email

  This matches the domain of the email, not including anything before `@`.

- **Allowed DNS domains** *(string input)*:
  Enter a valid regular expression that the inputted domain should match.

CSR Data Mapping

1. Click on  to add a mapping.

2. Select a field and enter a value.

You can delete  the CSR Data Mapping.

Certificate Metadata

This section details how to define a custom structure for the labels, ownership policy and technical metadata, allowing certificates to hold rich information.

Labels

You can add more labels by clicking .

- **Name** *(select)*:
  Select a preexisting label.
- **Mandatory** *(boolean)*:
  Should the label be mandatory. The default value is set to false.

- **Editable by requester** *(boolean)*:
  Tells whether the label should be editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the label should be editable by the approver. The default value is set to false.
- **Default value** *(string input)*:
  Set a default value to the label.
- **Label value restriction**
  - **Whitelist** *(string input multiple)*:
    The label value will have to be in the whitelist. Open the popup, enter the label value and press "enter" to add this value to the accepted value list. An empty whitelist means no restriction.
  - **Suggestions** *(string input multiple)*:
    Add suggestions that will be displayed to the user. The user will be able to choose one of these values or enter its own. Open the popup, enter your suggestions and press enter to add this value to the suggestions. An empty suggestions list means no restriction.
  - **Regex** *(regex)*:
    The label value will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of this label to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

You can delete  or reorder (drag and drop)  the label template.

Ownership policy

- **Owner**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's owner is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's owner can be overridden by the requester when approving a request.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the owner to the value of the evaluated computation rule. This value will override any other value including the user input.
- **Contact email**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's contact email is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:

Specify if the certificate's contact email can be overridden by the requester when submitting a request.

- **Editable by approver** *(boolean)*:
  Specify if the certificate's contact email can be overridden by the requester when approving a request.
- **Default contact email** *(string input)*:
  Set a default contact email. This value must comply with the contact email restriction.
- **Contact email restriction**
  - **Whitelist** *(string input multiple)*:
    The contact email will have to be in the whitelist. Open the popup, enter the email and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
  - **Regex** *(regex)*:
    The contact email will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
- **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
  Set the value of the contact email to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

- **Team**
  - **Mandatory** *(boolean)*:
    Specify if the certificate's team is mandatory when submitting a request.
  - **Editable by requester** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when submitting a request.
  - **Editable by approver** *(boolean)*:
    Specify if the certificate's team can be overridden by the requester when approving a request.
  - **Default team** *(string input)*:
    Set a default team. This value must comply with the team restriction.
  - **Team restriction**
    - **Whitelist** *(string input multiple)*:
      The team will have to be in the whitelist. Enter the team and press "enter" to add this value to the accepted whitelist. An empty whitelist means no restriction.
    - **Regex** *(regex)*:
      The team will have to match the regex. Open the popup, enter the regular expression and click on the submit button to set the regex. An empty regex means no restrictions.
  - **Computation rule** *([admin-guide:other-computation_rules:::_computation_rule] input)*:
    Set the value of the team to the value of the evaluated computation rule. This value will override any other value including the user input and the default value.

Metadata policy *(overridable metadata)*



These metadata are technical metadata. They are used by Horizon or Third party connectors, updating them should be done with utmost care.



Metadata edition is not allowed on enroll.



Metadata edition is not available via the User Interface. It must be changed with API, using horizon-cli.

You can allow the override of technical metadata by clicking .

- **Metadata*** *(select)*:
  Select a metadata.
- **Editable by requester** *(boolean)*:
  Tells whether the metadata is editable by the requester. The default value is set to false.
- **Editable by approver** *(boolean)*:
  Tells whether the metadata is editable by the approver. The default value is set to false.

You can delete  a metadata policy. This will not delete the metadata but will make it non editable.

Third-party policy

- **Synchronize data from discovery** *(boolean)*:
  Enable overriding of third-party data during discovery. When enabled, the following logic applies:
  - If a discovery attempts to set third-party data on a connector that is not referenced in the profile (by a trigger), the data will not be added.
  - If the connector is referenced in the profile, the discovery will override the existing data.

Notifications

This section details how to configure notifications on certificate and request lifecycle events.

Certificate lifecycle notifications

Notifications are sent when one of the following event is triggered by a certificate:

Enrollment	Revocation	Expire	Update	Migrate	Renew

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.

Request lifecycle notifications

Notifications are sent when one of the following event is triggered by an Enroll/Revocation/Update/Migrate/Renew request:

Submit	Cancel	Revoke	Approve	Pending

Select a preexisting email, [admin-guide:notifications-rest:::_REST] or groupware notification to associate it with an event.



Submit request events are not triggered when the user has the permission to perform the action directly.

5. Click on the save button.

You can update  or delete  the Jamf Profile .



You won't be able to delete a Jamf Profile if it is referenced somewhere else.

Last Steps

The integration between Jamf Pro and Horizon can be done in the following modes:

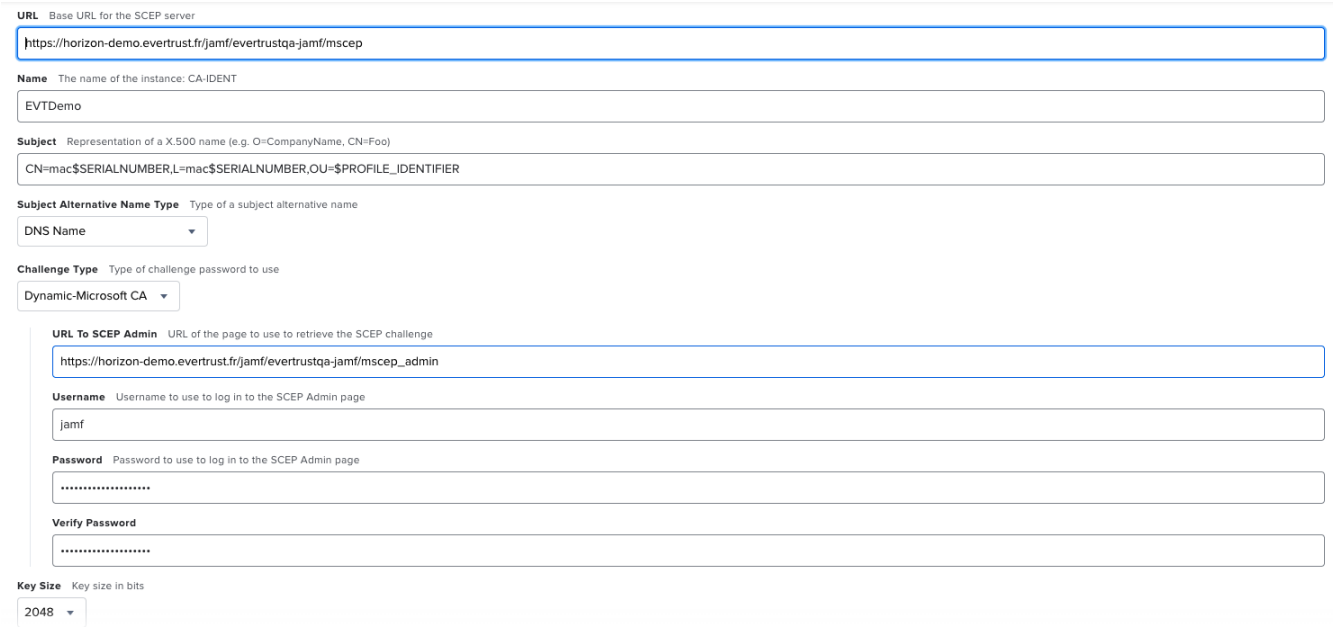
- Jamf Pro SCEP Proxy mode
- iOS SCEP Profile
- macOS SCEP Profile
- macOS SCEP Profile with Proxy

In all these modes, the Challenge type to use on Jamf Pro side is **Dynamic-Microsoft CA**, and you should point to the corresponding `mscep` and `mscep_admin` URI on Horizon side, that can be found in the Jamf Profile after it has been created.

Jamf Pro SCEP Proxy mode

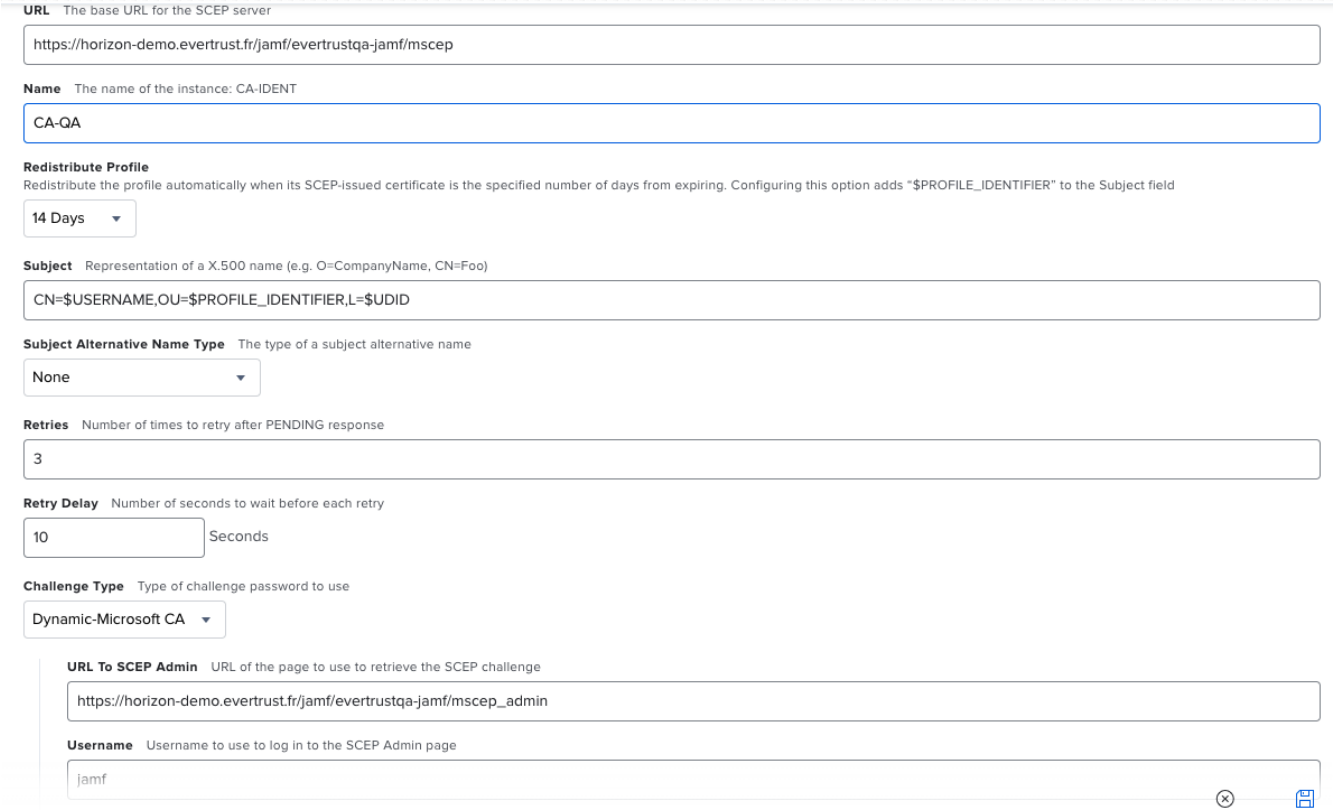
This mode requires to provide the SCEP Authority PKCS#12 to Jamf Pro, so that it can be uploaded in the appropriate profile.

Other than that, the configuration looks like the following on Jamf Pro side:



iOS/macOS SCEP Profile

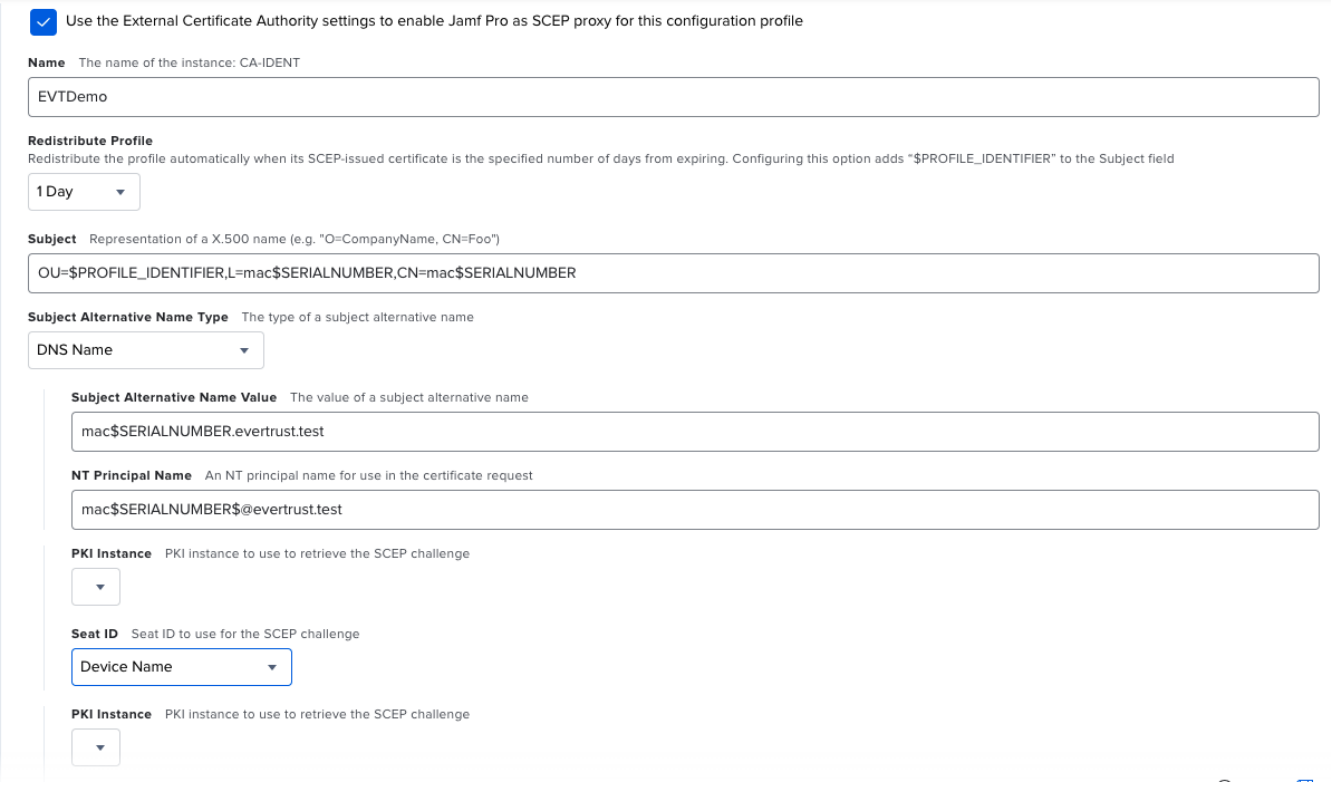
On Jamf Pro side, the profile configuration looks like the following:



macOS SCEP Profile with Proxy

This mode requires:

1. to set up the SCEP Proxy mode on Jamf Pro side
2. to configure a profile on Jamf Pro side, that looks like the following:



Jamf Scheduled Tasks

This section details how to schedule tasks that will run periodically on your jamf profiles, in order to manage automatic revocation upon device decommissioning.

Prerequisites

[admin-guide:third-parties-jamf-jamf_connector:::_jamf_connector]	[admin-guide:third-parties-jamf-jamf_profile:::_jamf_profile]

How to configure Jamf Scheduled Tasks

1. Log in to Horizon Administration Interface.

2. Access Jamf Scheduled Tasks from the drawer or card: **MDM › Jamf › Scheduled Tasks**.

3. Click on ⊕ .

4. Fill the mandatory fields.

- **Enable** *(boolean)*:
  Tells whether the Scheduled task should be enabled. Set by default at true.
- **Jamf Profile*** *(select)*:
  Select a Jamf profile previously created.
- **Target Connector*** *(select)*:
  Select an Jamf connector previously created.
- **Cron scheduling** *(cron expression)*:

Set to every 5 hours by default.

- **Revoke** *(boolean)*:
  Set to true by default. If true, Horizon will revoke any certificate associated to a device that is no longer present in Jamf Pro (for instance once the device has been deleted from Jamf Pro, and hence decommissioned).
- **Dry run** *(boolean)*:
  If enabled, revocation actions will not be performed. Instead, a message will be logged, explaining what would have been done.

5. Click on the save button.

You can run  or update  or delete  the Scheduled Tasks.

2.14. System configuration

2.14.1. Labels

This section details how to configure the labels. Labels are metadata used to store information provided by the en-users in Horizon database, associated to a given certificate, but not contained in the certificate.

You will be able to associate the labels created in this section with your profiles in order to enrich the certificates that will be issued from them.

How to configure a Label

1. Log in to Horizon Administration Interface.

2. Access Labels from the drawer or card: **System** › **Labels**.

3. Click on  .

4. Fill the following fields:

- **Name*** *(string input)*:
  Enter a meaningful Label name.

You can add more languages by clicking  .

- **Language*** *(select)*:
  Select a language. Supported languages are:
  - **en**: English
  - **fr**: French
- **Display Name** *(string input)*:
  Enter a display name. This will be the localized name of the label.

- **Description** *(string input)*:
  Enter a description. This will be displayed when making a request with this label and when adding it to a profile.

You can delete  the localization.

5. Click on the create button to save.

You can update  or delete  Labels.



You won't be able to delete a Label if it is referenced in any other configuration element.

2.14.2. HTTP Proxy

In this section you will be able to set up HTTP Proxies. HTTP Proxies may be used by Horizon to establish connection to various services.

How to configure an HTTP Proxy

1. Log in to Horizon Administration Interface.

2. Access HTTP Proxy from the drawer or card: **System › HTTP Proxies**.

3. Click on  .

4. Fill the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful HTTP Proxy name.
- **Host*** *(string input)*:
  The Hostname or IP Address of the HTTP/HTTPS proxy to use.
- **Port*** *(int)*:
  The Port of the HTTP/HTTPS proxy to use.
- **Credentials** *(select)*:
  Select `Password` credentials containing basic authentication credentials for the proxy.

5. Click on the create button to save.

You can update  or delete  the HTTP Proxy.



You won't be able to delete an HTTP Proxy if it is referenced in any other configuration element.

2.14.3. Grading Rules

The grading rules feature enhances the governance capabilities of Horizon, clearly displaying the quality of a certificate using different criteria. Currently, there is only one grading policy which is the Horizon grading policy designed by EverTrust experts using common reference documents.

The grading mechanism works as following:

1. Each rule is evaluated individually;
2. The score of each ruleset is calculated by adding the scores of each of its rules and dividing it by the max note for each ruleset, giving a score `s_i \in [-1,1]` for each ruleset;
3. The effective score for the grading policy is calculated through a weighted sum: `S = \sum_i w_i * s_i` with `w_i` being the weight of each ruleset;
4. The sum of the weights is calculated: `W = \sum_i w_i` ;
5. The score of the certificate for this grading policy is then calculated by dividing S by W: `cert\_score = \frac{S}{W} \in [-1,1]` , then the score is put back over 100 and the certificate grade is applied with the following scale:



Breakdown of the grading rules

ANSSI Cryptographic Content

The ANSSI Cryptographic Content Ruleset is created from the good practices advocated by the French ANSSI to ensure good cryptographic material when dealing with X509 certificates (based on the RGS). This ruleset has a maximum possible score of 70 and has a weight of 50 in the Horizon Grading Policy.

▼ *Details*

Rule	Score if satisfied	Score if not satisfied
Certificate Policy OID should be specified	10	0
Certificate should contain at least a CRLDP or an AIA OCSP URL	10	0
Certificate should contain the subject key identifier extension	10	0

Rule	Score if satisfied	Score if not satisfied
Certificate subject and issuer should differ and authority key identifier should be defined	10	0
Certificate issuer should contain the country element ('C')	5	0
Certificate issuer should contain the organization element ('O')	5	0
Certificate issuer should contain the organizational unit element ('OU') or organisational identifier ('organizationIdentifier')	5	0
Certificate subject should contain the country element ('C')	5	0
Certificate subject should contain the organization element ('O')	5	0
Certificate subject should contain the organizational unit element ('OU') or organisational identifier ('organizationIdentifier')	5	0
Total	70	0

CA/B Forum Ruleset

The CA/B Forum Ruleset contains good practices for certificates from the CA/B Forum recommendations.
This ruleset has a maximum possible score of 40 and has a weight of 60 in the Horizon Grading Policy.

▼ *Details*

Rule	Score if satisfied	Score if not satisfied
CP OID extension is not empty	10	0
Character '_' is forbidden in SAN DNS *(penalty rule)*	0	-10
SAN DNS field must not end with '.' *(penalty rule)*	0	-10
Certificate lifetime is less than 397 days	10	0
Certificate serial number is longer than 8 bytes	10	0
SAN DNS field is not empty	10	0
Total	40	-20

NIST and ANSSI ECDSA Cryptographic Ruleset (Weight 100, Maximum score 35)

The NIST and ANSSI ECDSA Cryptographic Ruleset contains good practices when dealing with elliptic curves cryptography for the certificate's private key.
This ruleset has a maximum possible score of 35 and has a weight of 100 in the Horizon Grading Policy.

▼ *Details*

Rule	Score if satisfied	Score if not satisfied
EC key algorithm should be P-256, P-384 or P-521	25	0
Signing hash algorithm should be SHA-256, SHA-384, SHA-512, SHA-3-256, SHA-3-384 or SHA-3-512	10	0
Certificate expiring after January 1st 2035 must be hybrid	0	-60
Hybrid certificate	5	0
Total	40	-60

Email Certificate Ruleset

The Email Certificate Ruleset contains good practices written by the EverTrust experts regarding the use of S/MIME certificates.
This ruleset has a maximum possible score of 20 and has a weight of 60 in the Horizon Grading Policy.

▼ *Details*

Rule	Score if satisfied	Score if not satisfied
Certificate with extended key usages 'emailProtection' should contain any of the following key usages: 'digitalSignature', 'nonRepudiation', 'keyEncipherment', 'dataEncipherment'	10	0
SAN Email (RFC822Name) field is not empty	10	0
Total	20	0

IETF PKIX Ruleset

The IETF PKIX Ruleset contains good practices from the IETF PKIX recommendations.
This ruleset has a maximum possible score of 30 and has a weight of 100 in the Horizon Grading Policy.

▼ *Details*

Rule	Score if satisfied	Score if not satisfied
An entity certificate should not contain a pathlen	10	0
Issuer must not be empty	5	0
Subject must not be empty	5	0
Subject key identifier extension should not be empty	5	0
Certificate subject and issuer should differ and authority key identifier should be defined	5	0
If defined, AIA OCSP URL should use HTTP *(penalty rule)*	0	-10

Rule	Score if satisfied	Score if not satisfied
If defined, CRLDP should use LDAP or HTTP *(penalty rule)*	0	-10
Non-CA certificate cannot be self-signed *(penalty rule)*	0	-30
The certificate is issued by an untrusted CA *(penalty rule)*	0	-15
Certificate KeyUsage cannot be empty *(penalty rule)*	0	-10
Total	30	-75

NIST PQC Cryptographic Ruleset

The NIST PQC Cryptographic Ruleset contains good practices when dealing with post-quantum cryptography for the certificate's private key.
This ruleset has a maximum possible score of 25 and has a weight of 100 in the Horizon Grading Policy.

▼ *Details*

Rule	Score if satisfied	Score if not satisfied
PQC certificate	25	0
Total	25	0

NIST and ANSSI RSA Cryptographic Ruleset

The NIST and ANSSI RSA Cryptographic Ruleset contains good practices when dealing with RSA cryptography for the certificate's private key.
This ruleset has a maximum possible score of 40 and has a weight of 100 in the Horizon Grading Policy.

▼ *Details*

Rule	Score if satisfied	Score if not satisfied
RSA key size should be greater or equals to 2048 bits	10	0
RSA key size should be greater or equals to 3072 bits	5	0
RSA key exponent should be greater than 2^16	10	0
Signing hash algorithm should be SHA-256, SHA-384, SHA-512, SHA-3-256, SHA-3-384 or SHA-3-512	10	0
RSA key size should not be less than 2048 bits *(penalty rule)*	0	-10
RSA key size must not be less than 1024 bits *(penalty rule)*	0	-10
Certificate expiring after January 1st 2030 should be hybrid	0	-15
Certificate expiring after January 1st 2035 must be hybrid	0	-50
Hybrid certificate	5	0

Rule	Score if satisfied	Score if not satisfied
Total	40	-85

TLS Certificate Ruleset

The TLS certificate ruleset contains good practices for certificates used to identify web servers.
This ruleset has a maximum possible score of 20 and has a weight of 60 in the Horizon Grading Policy.

▼ *Details*

Rule	Score if satisfied	Score if not satisfied
Certificate with extended key usages 'TLSServer' should contain key usage 'digitalSignature'	10	0
Certificate with extended key usage 'TLSServer' should not have a subject containing the following elements: 'givenname', 'surname' *(penalty rule)*	0	-5
SAN DNS field is not empty	10	0
Total	20	-5

Applying the grading policy

All certificates that are in Horizon can be graded using grading policies, whether they are discovered or fully managed by the product. If you want to add a grading policy to a profile, simply go to the profile settings then in the "Common configuration for profile" tab select the grading policies that will be used to grade certificates on this profile.
To remove a grading policy from a profile you just have to unselect it from the drop-down menu.

You can also grade discovered certificates: in the **Discovery** menu, click on the campaign that you want to apply the grading policies on and then select the grading policies that you want to apply from the drop-down menu.
Again, to remove a grading policy from a discovery campaign, just unselect it from the same drop-down menu.

Manually re-grading certificates

In case anything went wrong in the initial grading of certificates, or if you manually added a new grading policy to an existing profile and you want to manually re-evaluate a grading policy, follow these steps:

- **1.** Go to **System** › **Grading Rules**;
- **2.** Select the Grading Policy that you want to manually relaunch and click the .

All certificates concerned by this grading policy will now be re-graded.

2.14.4. Global configuration

These configurations handle various Horizon global parameters directly via the Web Interface.

Internal monitoring

This parameter configures the internal monitoring execution interval. Internal monitoring refers to an action that will check the expiration and usage status of credentials and license, and send the configured notifications if needed.

By default, this action will be executed every day at midnight UTC. The notifications will keep being sent each day for as long as an action is needed.

License configuration

The license configuration panel allows to configure [admin-guide:notifications-mail:::_email], [admin-guide:notifications-groupware:::_groupware] or [admin-guide:notifications-rest:::_REST] notifications to be sent. These can be configured on:

- license expiration: using a notification on the `License Expiration` event and the `Delay before notification sending` field in the notification configuration, notifications configured here will be sent by the internal monitoring action.
- license usage: using the `usage threshold`, if this threshold is exceeded, notifications configured here will be sent by the internal monitoring action.

Interface configuration

An image can be defined here. It will be added on the Web interface in the header and in the login menu.

Announcements

Announcements allow administrators to broadcast system-wide notices in the Web interface, typically to communicate upcoming maintenances or ongoing incidents.

Each announcement is defined by:

- a **level**, which controls the icon and color of the banner:
  - `info`: used for general communications.
  - `warning`: used for non-blocking issues or upcoming maintenances.
  - `danger`: used for major incidents.
- a **content**, displayed in a banner at the top of the Web interface. Clicking the banner opens a popup with the fully formatted content. The content cannot be empty.

Multiple announcements can be configured simultaneously and will all be displayed in the Web interface. Announcements are displayed to all users, including users who are not logged in.

2.14.5. Storage Backends

This section details how to declare and use storage backends for artifacts produced by Horizon (magic link reports and archives).

Horizon supports two storage backend types:

- **local**: MongoDB GridFS, stored in Horizon's primary database. A single `local` backend exists implicitly and does not require any UI configuration.
- **s3**: an Amazon S3 bucket or any S3-compatible object storage. Any number of `s3` backends can be declared.



The backend types that can actually be created are constrained by the optional `horizon.storage.whitelist` HOCON key:

- if the key is absent or set to an empty list, all backend types are authorized;
- if the key is set, only the listed types are available through the API. For example, `horizon.storage.whitelist = ["s3"]` prevents creating or using the implicit `local` backend.

Whitelisting applies to both static and dynamic backends. Non-whitelisted entries are filtered out of `list` and `get` operations.

How to configure a Storage Backend

1. Log in to Horizon Administration Interface.

2. Access Storages from the drawer or card: **Configuration › System › Storages**.

3. Click on  .

4. Fill in the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful name for the storage backend. It must be unique across all storage backends.
- **Type*** *(select)*:
  Only `s3` is creatable. The `local` backend is implicit and cannot be created from the UI.
- **Endpoint*** *(string input)*:
  URL of the S3 endpoint (e.g. `https://s3.eu-west-3.amazonaws.com` or the URL of a self-hosted S3-compatible service).
- **Region*** *(string input)*:
  S3 region the bucket belongs to.
- **Bucket*** *(string input)*:
  Name of the bucket where artifacts are written.
- Path style access (boolean):
 Use path-style

URLs (<code><a href="https://<endpoint>/<bucket>/<key>" class="bare">https://<endpoint>/<bucket>/<key&gt;</code>) instead of virtual-hosted style. Required for most S3-compatible services. The default value is set to false.

- **Credentials*** *(select)*:
  Select an `API Token` credentials entry holding the S3 access key and secret key. Credentials must declare `storage` as their target.
- **Proxy** *(string select)*:
  HTTP/HTTPS proxy used to reach the S3 endpoint, if any.

5. Click on the save button.

You can update  or delete  a Storage Backend.



A storage backend cannot be deleted while it is referenced in the system storage configuration (see below).

How to assign Storage Backends to artifacts

Once a storage backend is declared, it must be selected as the destination for each artifact type. This is done from the System storage configuration drawer (expert mode only):

- **Archive storage** *(string select)*:
  Storage backend used to write archives. If left empty, archives are written to the implicit `local` backend (only allowed if `local` is whitelisted).
- **Magic link report storage** *(string select)*:
  Storage backend used to write magic link reports. If left empty, reports are written to the implicit `local` backend (only allowed if `local` is whitelisted).



Changing the destination only affects newly produced artifacts. Existing artifacts remain on the backend they were originally written to. Refer to the archives page for the impact on archive deletion when using a non-`local` backend.

2.14.6. Terms of Service

This section details how to configure Terms of Service entries that can be referenced from WebRA, SCEP and EST profiles to require the requester's acceptance before enrollment.

A Terms of Service entry holds localized Markdown content. When a profile references it, the Terms of Service are displayed during the enrollment workflow (WebRA) or when issuing a challenge request (SCEP, EST) and must be explicitly accepted by the requester before the request can be submitted. When the profile does not reference any Terms of Service entry, the corresponding enrollment workflow remains unchanged.

How to configure Terms of Service

1. Log in to Horizon Administration Interface.

2. Access Terms of Service from the drawer or card: **Configuration › System › Terms of Service**.

3. Click on  .

4. Fill in the mandatory fields.

- **Name*** *(string input)*:
  Enter a meaningful name. It must be unique and will be used to reference the Terms of Service from profiles.
- **Description** *(string input)*:
  Enter a description. This is displayed in the list view to help administrators identify the entry.

Languages

You can add more languages by clicking  .

- **Language*** *(select)*:
  Select the language this entry applies to. Supported languages are:
  - **en**: English
  - **fr**: French
- **Content*** *(markdown)*:
  Markdown content to display to the requester. Paragraphs, headings, bullet lists, bold/italic text and HTTP(S) links are supported. Embedded HTML is blocked to prevent XSS.

You can delete  a language entry.

5. Click on the save button.

You can update  or delete  a Terms of Service entry.



A Terms of Service entry cannot be deleted while it is referenced by any profile.

2.15. Common configuration elements

2.15.1. Cron Expression

Cron expressions are composed of 6 required fields and one optional field separated by white spaces. The fields are respectively described as follows:

Field Name	Allowed Values	Allowed Special Character
Seconds	0-59	- * /
Minutes	0-59	- * /
Hours	0-23	- * /

Field Name	Allowed Values	Allowed Special Character
Day-of-month	1-31	- * ? / L W
Month	1-12 or JAN-DEC	- *
Day-of-Week	1-7 or SUN-SAT	- * ? / L #
Year (Optional)	empty, 1970-2199	- * /

Special characters

- * ("all values") - used to select all values within a field. For example, "*" in the minute field means *every minute*.
- **?** ("no specific value") - useful when you need to specify something in one of the two fields in which the character is allowed, but not the other. For example, if I want my trigger to fire on a particular day of the month (say, the 10th), but don't care what day of the week that happens to be, I would put "10" in the day-of-month field, and "?" in the day-of-week field. See the examples below for clarification.
- **-** - used to specify ranges. For example, "10-12" in the hour field means "the hours 10, 11 and 12".
- **,** - used to specify additional values. For example, "MON,WED,FRI" in the day-of-week field means "the days Monday, Wednesday, and Friday".
- / - used to specify increments. For example, "0/15" in the seconds field means "the seconds 0, 15, 30, and 45". And "5/15" in the seconds field means "the seconds 5, 20, 35, and 50". You can also specify '/' after the '*' character - in this case '*' is equivalent to having '0' before the '/'. '1/3' in the day-of-month field means "fire every 3 days starting on the first day of the month".
- **L** ("last") - has different meaning in each of the two fields it is allowed into. For example, the value "L" in the day-of-month field means "the last day of the month" - day 31 for January, day 28 for February on non-leap years. If used in the day-of-week field by itself, it simply means "7" or "SAT". But if used in the day-of-week field after another value, it means "the last xxx day of the month" - for example "6L" means "the last Friday of the month". You can also specify an offset from the last day of the month, such as "L-3" which would mean the third-to-last day of the calendar month. When using the 'L' option, it is important not to specify lists, or ranges of values, as you'll get confusing/unexpected results.
- **W** ("weekday") - used to specify the weekday (Monday-Friday) nearest the given day. As an example, if you were to specify "15W" as the value for the day-of-month field, the meaning is: "the nearest weekday to the 15th of the month". So if the 15th is a Saturday, the trigger will fire on Friday the 14th. If the 15th is a Sunday, the trigger will fire on Monday the 16th. If the 15th is a Tuesday, then it will fire on Tuesday the 15th. However if you specify "1W" as the value for day-of-month, and the 1st is a Saturday, the trigger will fire on Monday the 3rd, as it will not 'jump' over the boundary of a month's days. The 'W' character can only be specified when the day-of-month is a single day, not a range or list of days.
- # - used to specify "the nth" XXX day of the month. For example, the value of "6#3" in the day-of-week field means "the third Friday of the month" (day 6 = Friday and "#3" = the 3rd one in the month). Other examples: "2#1" = the first Monday of the month and "4#5" = the fifth Wednesday of the month. Note that if you specify "#5" and there is not 5 of the given day-of-week in the month, then no firing will occur that month.



The 'L' and 'W' characters can also be combined in the day-of-month field to yield '`LW`', which translates to **"last weekday of the month"**.

2.15.2. Finite Duration

The format of a *Finite Duration* is `"<length><unit>"`, where:

- White space is allowed between the parts.
- Length is a positive integer without the "+" sign.
- Valid possible units are described in the below table:

Unit	Short name	Long names
DAYS	d	day days
HOURS	h	hour hours
MINUTES	m	min mins minute minutes
SECONDS	s	sec secs second seconds
MILLISECONDS	ms	milli millis millisecond milliseconds

For example, 10 seconds will be written as "10 s", "10s", "10 sec" or "10 seconds".

2.15.3. Regex

Regex are a powerful tool to match patterns in strings.Horizon is developed in Java, so regex must follow the Java regex engine syntax.

In Horizon, most regex input are used for validation of a field input.As such, they validate one line, and a good practice is enforced:



Regex must start with `^` and end with `$`.

2.15.4. Dictionaries

Here is the list of available dictionary keys to use in computation rules, depending on the usage.

In notifications

Certificate dictionary

This dictionary is available for notifications on the following events:

- `on_enroll`
- `on_revoke`
- `on_update`

- `on_recover`
- `on_migrate`
- `on_expire`
- `on_renew`

Key	Description	Type	Available in Computation Rule
certificate.id	Horizon Id of the certificate	Single value	Yes
certificate.module	Module of the certificate	Single value	Yes
certificate.not_after	Expiration date of the certificate	Single value	Yes
certificate.not_before	Start date of the certificate	Single value	Yes
certificate.serial	Serial number of the certificate	Single value	Yes
certificate.thumbprint	Thumbprint of the certificate	Single value	Yes
certificate.public_key_thumbprint	Thumbprint of the public key of the certificate	Single value	Yes
certificate.revoked	true if the certificate is revoked, false otherwise	Single value	Yes
certificate.key_type	Key Type of the certificate	Single value	Yes
certificate.signing_algorithm	Signing Algorithm of the certificate	Single value	Yes
certificate.holder_id	Holder Id of the certificate	Single value	Yes
certificate.friendly_name	Friendly name of the certificate	Single value	Yes
certificate.pem	PEM Encoded certificate	Single value	Yes
certificate.profile	The profile of the certificate	Single value	Yes
certificate.revocation_date	The revocation date of the certificate	Single value	Yes

Key	Description	Type	Available in Computation Rule
certificate.revocation_reason	The revocation reason of the certificate	Single value	Yes
certificate.mail	The contact email of the certificate	Single value	Yes
certificate.owner	Principal owning the certificate	Single value	Yes
certificate.issuer	The issuer of the certificate	Single value	No
certificate.dn	The Distinguished Name of the certificate	Single value	No
certificate.sans	All the SANs of the certificate, in <type>: <value> comma separated format	Single value	No
certificate.extensions	All the extensions of the certificate, in <type>: <value> comma separated format	Single value	No
certificate.metadata	All the metadata of the certificate, in <type>: <value> comma separated format	Single value	No
certificate.labels	All the labels of the certificate, in <name>: <value> comma separated format	Single value	No
certificate.metadata.<metadata name>	The value of metadata `metadata name` defined in the certificate	Single value	Yes
certificate.subject	The values of the certificate subject	Subject dictionary	Yes
certificate.san	The values of the certificate sans	Sans dictionary	Yes
certificate.extension	The values of the certificate extensions	Extensions dictionary	Yes
certificate.label	The values of the certificate label	Labels dictionary	Yes
certificate.team	The values of the certificate team	Team dictionary	Yes

Request dictionary

This dictionary is available for notifications on the following events:

- `on_submit_enroll`
- `on_cancel_enroll`
- `on_approve_enroll`
- `on_deny_enroll`
- `on_pending_enroll`
- `on_submit_revoke`
- `on_cancel_revoke`
- `on_approve_revoke`
- `on_deny_revoke`
- `on_pending_revoke`
- `on_submit_update`
- `on_cancel_update`
- `on_approve_update`
- `on_deny_update`
- `on_pending_update`
- `on_submit_recover`
- `on_cancel_recover`
- `on_approve_recover`
- `on_deny_recover`
- `on_pending_recover`
- `on_submit_migrate`
- `on_cancel_migrate`
- `on_approve_migrate`
- `on_deny_migrate`
- `on_pending_migrate`
- `on_submit_renew`
- `on_cancel_renew`
- `on_approve_renew`
- `on_deny_renew`
- `on_pending_renew`

Key	Description	Type	Available in Computation Rule
request.id	Horizon Id of the request	Single value	Yes
request.workflow	Workflow of the request	Single value	Yes
request.module	Module of the request	Single value	Yes
request.status	Status of the request	Single value	Yes
request.profile	Profile of the request	Single value	Yes
request.requester	Requester of the request	Single value	Yes
request.approver	Approver of the request	Single value	Yes
request.requester_comment	Comment of the requester	Single value	Yes
request.approver_comment	Comment of the approver	Single value	Yes
request.registration_date	Registration date of the request	Single value	Yes
request.last_modification_date	Last modification date of the request	Single value	Yes
request.password	PKCS#12 password or challenge value of the request	Single value	Yes
request.mail	The contact email of the request	Single value	Yes
request.owner	Principal owning the request	Single value	Yes
request.my.url	Generates the link to access the request in the 'My Requests' drawer. Should be used after specifying the hostname without trailing slash: https://horizon.fr{{request.my.url}}	Single value	No

Key	Description	Type	Available in Computation Rule
request.manage.url	Generates the link to access the request in the 'Manage Requests' drawer. Should be used after specifying the hostname without trailing slash: https://horizon.fr{{request.my.url}}	Single value	No
request.dn	The Distinguished Name of the request	Single value	No
request.sans	All the SANs of the request, in <type>: <value> comma separated format	Single value	No
request.extensions	All the extensions of the request, in <type>: <value> comma separated format	Single value	No
request.metadata	All the metadata of the request, in <type>: <value> comma separated format	Single value	No
request.labels	All the labels of the request, in <name>: <value> comma separated format	Single value	No
request.subject	The values of the request subject	Subject dictionary	Yes
request.san	The values of the request sans	Sans dictionary	Yes
request.extension	The values of the request extensions	Extensions dictionary	Yes
request.label	The values of the request label	Labels dictionary	Yes
request.metadata.<metadata name>	The value of metadata `metadata name` defined in the request	Single value	Yes
request.certificate	The value of the certificate contained in the request	Certificate Dictionary	Yes

Key	Description	Type	Available in Computation Rule
request.team	The value of the team contained in the request	Team Dictionary	Yes

Previous Certificate dictionary

This dictionary is available for notifications on the following events:

- `on_renew`

Key	Description	Type	Available in Computation Rule
previous.certificate	The value of the certificate that is being renewed	Certificate dictionary	Yes

Credentials dictionary

This dictionary is available for notifications on the `on_credentials_expiration` event.

Key	Description	Type	Available in Computation Rule
credentials.name	Name of the credentials	Single value	Yes
credentials.description	Description of the credentials	Single value	Yes
credentials.type	Type of the credentials	Single value	Yes
credentials.expiration_date	Expiration date of the credentials	Single value	Yes

Profile dictionary

Key	Description	Type	Available in Computation Rule
profile.name	Technical name of the profile	Single value	Yes
profile.module	Module of the profile	Single value	Yes
profile.displaynames	Display names of the profile in <lang>:<value> comma separated format	Single value	No

Key	Description	Type	Available in Computation Rule
profile.descriptions	Descriptions of the profile in <lang>: <value> comma separated format	Single value	No
profile.<name>.display name.<lang>	Display name of the profile in <lang> (two letter identifier) language	Single value	No
profile.<name>.description.<lang>	Description of the profile in <lang> (two letter identifier) language	Single value	No

License dictionary

This dictionary is available for notifications on the `on_license_expiration` and `on_license_usage` event.

Key	Description	Type	Available in Computation Rule
license.expiration_date	Expiration date of the license	Single value	Yes
license.used	Number of holders on the license (only available on `on_license_usage` event)	Single value	Yes
license.percent_used	Percent of the license used (only available on `on_license_usage` event)	Single value	Yes

Failed trigger dictionary

This dictionary is available for notifications on the `on_trigger_error` event.

Key	Description	Type
trigger.name	Name of the trigger	Single value
trigger.event	Event on which the trigger was run	Single value
trigger.lastExecutionDate	Last execution date of the trigger	Single value
trigger.status	Status of the trigger	Single value

Key	Description	Type
trigger.retryable	true if the trigger can be retried, false otherwise	Single value
trigger.type	Type of the trigger	Single value
trigger.retries	Number of remaining retries	Single value
trigger.nextExecutionDate	Date at which the trigger will be rerun	Single value
trigger.nextDelay	Delay between the current and next iteration	Single value
trigger.detail	Details about the failure	Single value

In profile

The following dictionaries are available in a certificate template in profile configuration, for auto validation and datasource flow configuration.

General

The dictionary keys listed here are available in all protocols.



All indexes start at 1.

Principal

This dictionary regroups the information of the user making the request, the 'principal'.

Key	Description	Type
principal.identifier	The identifier of the user	Single value
principal.team	The teams of the user	Multi valued
principal.team.<index>	The team at index `index`	Single value
principal.name	The name of the user	Single value
principal.mail	The email of the user	Single value
principal.provider.name	The name of the identity provider of the principal	Single value
principal.certificate.subject	The values of the principal certificate subject	Subject dictionary
principal.certificate.san	The values of the principal certificate sans	Sans dictionary
principal.certificate.extension	The values of the principal certificate extensions	Extensions dictionary

CSR

This dictionary regroups the information of the csr used for enrollment. It can be sent via a client (horizon-cli, estclient, sscep) or via web interfaces with WebRA protocol.



This only concerns decentralized enrollment.

Key	Description	Type
csr.subject	The values of the csr subject	Subject dictionary
csr.san	The values of the csr sans	Sans dictionary
csr.extension	The values of the csr extensions	Extensions dictionary

HTTP Request

This dictionary regroups the information of the http request that initiated the enrollment.

Key	Description	Type
http.request.ip	The IP from which the request originated	Single value
http.request.method	The HTTP method used by the request	Single value
http.request.path	The path requested	Single value
http.request.host	The host requested	Single value
http.request.header.<header name>	Value of the <header name> header	Multi value

WebRA

Enrollment request

Certificate fields can be filled by the user on Horizon interface. This information is available through the following dictionary.

Key	Description	Type
webra.enroll.subject	The values of the subject defined in the challenge request	Subject dictionary
webra.enroll.san	The values of the sans defined in the challenge request	Sans dictionary
webra.enroll.extension	The values of the extensions defined in the challenge request	Extensions dictionary
webra.enroll.label.<label name>	The value of label `label name` defined in the challenge request	Single value

Key	Description	Type
webra.enroll.metadata.<metadata name>	The value of metadata `metadata name` defined in the challenge request	Single value
webra.enroll.mail	The value of the contact email defined in the challenge request	Single value
webra.enroll.owner	The value of the owner defined in the challenge request	Single value
webra.enroll.team	The value of the team defined in the challenge request	Single value

EST

Enrollment request

In case of a prevalidated enroll, certificate fields can be filled by the user on Horizon interface. This information is available through the following dictionary.

Key	Description	Type
est.enroll.subject	The values of the subject defined in the challenge request	Subject dictionary
est.enroll.san	The values of the sans defined in the challenge request	Sans dictionary
est.enroll.extension	The values of the extensions defined in the challenge request	Extensions dictionary
est.enroll.label.<label name>	The value of label `label name` defined in the challenge request	Single value
est.enroll.metadata.<metadata name>	The value of metadata `metadata name` defined in the challenge request	Single value
est.enroll.mail	The value of the contact email defined in the challenge request	Single value
est.enroll.owner	The value of the owner defined in the challenge request	Single value
est.enroll.team	The value of the team defined in the challenge request	Single value

Url passed parameters

Horizon allows the use of url parameters to pass certificate metadata info. These are notably used by the horizon-cli client. See the dedicated page for more information.

Key	Description	Type
url.enroll.label.<label name>	The value of label `label name` passed in the url	Single value
url.enroll.metadata.<metadata name>	The value of metadata `metadata name` passed in the url	Single value
url.enroll.mail	The value of the contact email passed in the url	Single value
url.enroll.owner	The value of the owner passed in the url	Single value
url.enroll.team	The value of the team passed in the url	Single value

SCEP

Enrollment request

In case of a prevalidated enroll, certificate fields can be filled by the user on Horizon interface. This information is available through the following dictionary.

Key	Description	Type
scep.enroll.subject	The values of the subject defined in the challenge request	Subject dictionary
scep.enroll.san	The values of the sans defined in the challenge request	Sans dictionary
scep.enroll.extension	The values of the extensions defined in the challenge request	Extensions dictionary
scep.enroll.label.<label name>	The value of label `label name` defined in the challenge request	Single value
scep.enroll.metadata.<metadata name>	The value of metadata `metadata name` defined in the challenge request	Single value
scep.enroll.mail	The value of the contact email defined in the challenge request	Single value
scep.enroll.owner	The value of the owner defined in the challenge request	Single value
scep.enroll.team	The value of the team defined in the challenge request	Single value

Url passed parameters

Horizon allows the use of url parameters to pass certificate metadata info. These are notably used by the horizon-cli client. See the dedicated page for more information.

Key	Description	Type
url.enroll.label.<label name>	The value of label `label name` passed in the url	Single value
url.enroll.metadata.<metadata name>	The value of metadata `metadata name` passed in the url	Single value
url.enroll.mail	The value of the contact email passed in the url	Single value
url.enroll.owner	The value of the owner passed in the url	Single value
url.enroll.team	The value of the team passed in the url	Single value

ACME

Order

This dictionary regroups the information of the acme order used for enrollment.

Key	Description	Type
acme.order.initialip	The initial IP of the acme order	Single value
acme.order.label.<label name>	The value of label `label name`	Single value
acme.order.metadata.<metadata name>	The value of metadata `metadata name`	Single value
acme.order.mail	The value of the contact email of the acme order	Single value
acme.order.owner	The value of the owner of the acme order	Single value
acme.order.team	The value of the team of the acme order	Single value

Account

This dictionary regroups the information of the acme account used for enrollment.

Key	Description	Type
acme.account.initialip	The initial IP of the acme account	Single value
acme.account.contact.<index>	The value of contact email address of the account at index `index`	Single value

CRMP

Enrollment request

Certificate fields can be filled by the user on CMS interface. This information is available through the following dictionary.

Key	Description	Type
crmp.enroll.subject	The values of the subject defined in the challenge request	Subject dictionary
crmp.enroll.san	The values of the sans defined in the challenge request	Sans dictionary
crmp.enroll.extension	The values of the extensions defined in the challenge request	Extensions dictionary
crmp.enroll.label.<label name>	The value of label `label name` defined in the challenge request	Single value
crmp.enroll.metadata.<metadata name>	The value of metadata `metadata name` defined in the challenge request	Single value
crmp.enroll.mail	The value of the contact email defined in the challenge request	Single value
crmp.enroll.owner	The value of the owner defined in the challenge request	Single value
crmp.enroll.team	The value of the team defined in the challenge request	Single value

WCCE

Caller identity

The information of the caller identity in a WCCE enroll.

Key	Description	Type
calleridentity.dn	The dn of the caller identity	Single value
calleridentity.subject	The dn of the caller identity, split in addressable form	Subject dictionary
calleridentity.cn	The cn of the caller identity	Single value
calleridentity.msguid	The guid of the caller identity	Single value
calleridentity.msupn	The upn of the caller identity	Single value
calleridentity.c	The country of the caller identity	Single value

Key	Description	Type
calleridentity.company	The company of the caller identity	Single value
calleridentity.department	The department of the caller identity	Single value
calleridentity.description	The description of the caller identity	Single value
calleridentity.displayname	The display name of the caller identity	Single value
calleridentity.dnshostname	The dns host name of the caller identity	Single value
calleridentity.employeeid	The employee id of the caller identity	Single value
calleridentity.employeenumber	The employee number of the caller identity	Single value
calleridentity.mail	The email of the caller identity	Single value
calleridentity.o	The organization of the caller identity	Single value
calleridentity.ou	The OU of the caller identity	Single value
calleridentity.samaccountname	The sam account name of the caller identity	Single value
calleridentity.serialnumber	The serial number of the caller identity	Single value
calleridentity.sn	The sn of the caller identity	Single value
calleridentity.title	The title of the caller identity	Single value
calleridentity.uid	The uid of the caller identity	Single value
calleridentity.sid	The sid of the caller identity	Single value

Raw certificate dictionary

This dictionary is available when evaluating template strings against a raw X.509 certificate that is not yet known to Horizon, such as the X509 identity mapping applied at client certificate authentication time. It exposes a subset of the certificate dictionary.

Key	Description	Type
certificate.not_after	Expiration date of the certificate	Single value
certificate.not_before	Start date of the certificate	Single value
certificate.serial	Serial number of the certificate	Single value

Key	Description	Type
certificate.thumbprint	Thumbprint of the certificate	Single value
certificate.public_key_thumbprint	Thumbprint of the public key of the certificate	Single value
certificate.key_type	Key Type of the certificate	Single value
certificate.signing_algorithm	Signing Algorithm of the certificate	Single value
certificate.holder_id	Holder Id of the certificate	Single value
certificate.friendly_name	Friendly name of the certificate	Single value
certificate.pem	PEM Encoded certificate	Single value
certificate.issuer	The issuer of the certificate	Single value
certificate.dn	The Distinguished Name of the certificate	Single value
certificate.sans	All the SANs of the certificate, in <type>: <value> comma separated format	Single value
certificate.extensions	All the extensions of the certificate, in <type>: <value> comma separated format	Single value
certificate.subject	The values of the certificate subject	Subject dictionary
certificate.san	The values of the certificate sans	Sans dictionary
certificate.extension	The values of the certificate extensions	Extensions dictionary

Sub dictionaries

These dictionary cannot be used alone but can be completed with one of the other ones. For example, a valid key is:

```
principal.certificate.subject.cn.1
```

Subject dictionary

Key	Description	Type
subject.<dn field type>	All values of subject field of type `dn field type`	Multi valued
subject.<dn field type>.<index>	Value of subject field of type `dn field type` at index `index`	Single value



The valid dn field types are: cn, uid, serialnumber, surname, givenname, unstructuredaddress, unstructuredname, e, ou, organizationidentifier, uniqueidentifier, street, st, l, o, c, description, dc.

Sans dictionary

Key	Description	Type
san.<san field type>	All values of SAN fields of type `san field type`	Multi valued
san.<san field type>.<index>	Value of subject field of type `san field type` at index `index`	Single value



The valid SAN field types are: `rfc822name`, `dnsname`, `uri`, `ipaddress`, `othername_upn`, `othername_guid`, `registered_id`.

Extensions dictionary

Key	Description	Type
extension.<extension type>	Value of extension of type `extension type`	Single value



The valid extension types are: `ms_sid`, `ms_template`, `ms_template_v2`.

Labels dictionary

Key	Description	Type	Available in Computation Rule
label.<name>	Value of the <name> label	Single value	Yes
label.<name>.displaynames	Display names of the label in <lang>: <value> comma separated format	Single value	No
label.<name>.descriptions	Descriptions of the label in <lang>: <value> comma separated format	Single value	No
label.<name>.displayname.<lang>	Display name of the label in <lang> (two letter identifier) language	Single value	No

Key	Description	Type	Available in Computation Rule
label.<name>.description.<lang>	Description of the label in <lang> (two letter identifier) language	Single value	No

Team dictionary

Key	Description	Type	Available in Computation Rule
team	Value of the team	Single value	Yes
team.displaynames	Display names of the team in <lang>: <value> comma separated format	Single value	No
team.descriptions	Descriptions of the team in <lang>: <value> comma separated format	Single value	No
team.displayname.<lang>	Display name of the team in <lang> (two letter identifier) language	Single value	No
team.<name>.description.<lang>	Description of the team in <lang> (two letter identifier) language	Single value	No

2.15.5. Computation rule

Computation Rules are expressions that describe operations to apply to dictionary keys. These keys can come from diverse data sources such as a certification request or a user entry. The available operations and their usage are detailed in this part.

Example

Let’s start by an example:

My CSR contains a DNSNAME subject alternate name with the following value:

```
host.evertrust.fr
```

I want my final certificate to have 2 SANs, this value and its short name: "host".

In order to do that, in **Profile** › **Certificate Template** › **Subject Alternate Names** , I add a

DNSNAME SAN with the following computation rule:

```
[{{csr.san.dnsname.1}}, Extract({{csr.san.dnsname.1}}, "(.*?)\.", 1)]
```

This will output, in my final certificate, two SANs with values:

```
host.evertrust.fr, host
```

To explain this result, the value "host.evertrust.fr" was retrieved by choosing the first DNSNAME SAN of the CSR: `{{csr.san.dnsname.1}}`. The function `Extract` extracted the first catching group from the regex `(.*?)\.`, resulting in the "host" value.

The computation rule language has a lot more possible operations, allowing complex use cases to become reality.

Dictionary keys

Dictionary keys are a way to name the information from the available sources. For instance, for a webra enroll, the available sources are the given csr, the webra enroll form data and the principal information if it is authenticated. The full list of available dictionary keys is available on the dictionary page.

Enrollment

A key can reference a single element or a list of elements. It is separated in three main parts: the source of data (csr, webra enroll data form), the section of the data, and an optional number

For example, the following is a valid key with these 3 parts:

`{{csr.subject.cn.1}}`

The csr is the data source, the subject.cn the requested information and the 1 is the index. It allows to retrieve the first, common name from the subject, from the CSR.

Without an index, the key is still valid, but it will output all the corresponding values. For example

`[[csr.subject.ou]]`

This retrieves all the ou from the subject, from the CSR.



When a key is expected to output a single value it should be written as a single dictionary key, and one outputting a list of values as a multi dictionary key, otherwise it will be none.

Basic expressions

Basic string expressions
The following expressions are evaluated as a string or None.

Expression Name	Syntax	Allowed Values	Description	Example
Single dictionary key	{{<key>}}	key: a-zA-A-._	This retrieves a key value from the dictionary, none if it does not exist	{{csr.subject.cn.1}}
Number	<number>	number: -\d+	This will output the given number	-4
Literal	"<literal>"	literal: any string	This will output the given literal	"iAmAString"
Null	NULL	NULL	This will output None	NULL
Now	NOW	NOW	This will output the current instant	NOW

Basic list expressions

The following expressions are evaluated as a list of string or None.

Expression Name	Syntax	Allowed Values	Description	Example
Multi dictionary key	[[<key>]]	key: a-zA-A-._	This retrieves all values that start with key from the dictionary	[[admin-guide:other-computation_rules:::csr.subject.cn]]
Array	[<simpleExpression>, ... <simpleExpression>]	simpleExpression: any expression that will be evaluated to a single element	This will output a multi expression composed of all inserted simple expressions	["iAmAString", {{csr.san.dnsname.1}}]

Quick reference



Function names are not case sensitive but keys are

Function Name	Syntax
Upper	Upper(`expression`:<expression>)
Lower	Lower(`expression`:<expression>)
Trim	Trim(`expression`: <expression>)
Substr	Substr(`expression`: <expression>, `start`: <number>)
Substr	Substr(`expression`: <expression>, `start`: <number>, `end`: <number>)
Concat	Concat(`expression`: <expression>, ... <expression>)

Function Name	Syntax
Extract	Extract(expression: <expression>, regex: <literal>)
Extract	Extract(expression: <expression>, regex: <literal>, group: <number>)
Replace	Replace(expression: <expression>, regex: <literal>, replacement: <expression>)
OrElse	OrElse(expression: <expression>, ... <expression>)
Match	Match(expression: <simpleExpression>, regex: <literal>)
DateTimeFormat	DateTimeFormat(expression: <simpleExpression>, format: <literal>)
Get	Get(expression: <multiExpression>, index: <number>)
First	First(expression: <multiExpression>)
Last	Last(expression: <multiExpression>)
Filter	Filter(expression: <multiExpression>, regex: <literal>)
Slice	Slice(expression: <multiExpression>, start: <number>)
Slice	Slice(expression: <multiExpression>, start: <number>, end: <number>)
Join	Join(expression: <multiExpression>, separator: <literal>)
Split	Split(expression: <singleExpression>, separator: <literal>)
Sort	Sort(expression: <multiExpression>)
ShortenDNS	ShortenDNS(expression: <singleExpression>)
DomainDNS	DomainDNS(expression: <singleExpression>)
EmailUser	EmailUser(expression: <singleExpression>)
EmailDomain	EmailDomain(expression: <singleExpression>)
SamAccountNameUser	SamAccountNameUser(expression: <singleExpression>)
SamAccountNameDomain	SamAccountNameDomain(expression: <singleExpression>)

Any expression functions

Upper

```
Upper(expression:<expression>)
```

This outputs the result evaluated from `expression` with only upper case characters and None if no value was evaluated

```
Upper("string") => "STRING"
Upper(["string1", "string2"]) => ["STRING1", "STRING2"]
```

Lower

```
Lower(expression:<expression>)
```

This outputs the result evaluated from `expression` with only lower case characters and None if no value was evaluated

```
Lower("STRING") => "string"
Lower(["STRING1", "STRING2"]) => ["string1", "string2"]
```

Trim

```
Trim(expression:<expression>)
```

This outputs the trimmed result evaluated from `expression` and None if no value was evaluated

```
Trim(" STRING") => "STRING"
Trim(["string1 ", "  string2  "]) => ["string1", "string2"]
```

Substr

```
Substr(expression: <expression>, start: <number>)
```

This outputs the substring from index `start` to the end of the string evaluated from `expression` and None if no value was evaluated or the result of substring is empty. `start` can be negative and it will be computed from end of string.

```
Substr("STRING", 2) => "TRING"
Substr(["string", "longerString", "s"], -2) => ["ng", "ng", "s"]
```

```
Substr("tooShort", 15) => None
```

Substr

```
Substr(expression: <expression>, start: <number>, end: <number>)
```

This outputs the substring from index `start` to `end` of the string evaluated from `expression` and None if no value was evaluated or the result of substring is empty. `start` and `end` can be negative and it will be computed from end of string.

```
Substr("STRING", 2, 4)  => "TRI"
Substr(["string", "longerString", "s"], 2, -2) => ["tri", "ongerStri"]
Substr("tooShort", -2, 4) => None
```

Concat

```
Concat(expression: <expression>, ...<expression>)
```

This outputs the concatenation of evaluated expressions: if they are all simple expression, a string concatenation will take place, otherwise an array with all the values will be evaluated. If the final result is empty, None will be returned.

```
Concat("start", " middle ", "end") => "start middle end"
Concat(["string1", "string2", "string3"], "string4") => ["string1", "string2",
"string3", "string4"]
```

Extract

```
Extract(expression: <expression>, regex: <literal>)
```

This extracts from the evaluated `expression` string(s) the part that matches the `regex`

```
Extract("abcd@domain.com", ".*@") => "abcd@"
Extract(["string1", "string2", "string3"], "\d") => ["1", "2", "3"]
```

Extract

```
Extract(expression: <expression>, regex: <literal>, group: <number>)
```

This extracts from the evaluated `expression` string(s) the group at index `group` that matches the `regex`

```
Extract("abcd@domain.com", "(.*)@", 1) => "abcd"
Extract(["string1", "string2", "string3"], "(.*)\d", 1) => ["string", "string",
"string"]
```

Replace

```
Replace(expression: <expression>, regex: <literal>, replacement: <expression>)
```

This replaces parts of the evaluated `expression` string(s) that matches the `regex` with the evaluated `replacement`. If `replacement` is None, values will be replaced by an empty string.

```
Replace("abcdATdomain.com", "AT", "@") => "abcd@domain.com"
Replace(["string1", "string2", "string3"], "\d", CONCAT("This", " was ", " a number"))
=> ["stringThis was a number", "stringThis was a number", "stringThis was a number"]
```

OrElse

```
OrElse(expression: <expression>, ...<expression>)
```

This outputs the first non None result of the given expressions, or None if they are all None

```
OrElse({{not.a.value}}, "abcd@domain.com") => "abcd@domain.com"
OrElse([[no.values]], "value") => ["value"]
OrElse([[no.values]], {{not.a.value}}) => None
```

String functions



The following functions output a string or None.

Match

```
Match(expression: <simpleExpression>, regex: <literal>)
```

This outputs the expression if it matches the regex, otherwise None

```
Match("abcd", "[a-z]+") => "abcd"
Match("abcd", "\d+") => None
```

DateTimeFormat

```
DateTimeFormat(expression: <simpleExpression>, format: <literal>)
```

This outputs the expression formatted as format. If expression is not a date, no formatting takes place. Available formats are:

- Custom format in Java DateFormatter syntax
- MILLIS
- BASIC_ISO_DATE
- ISO_LOCAL_DATE
- ISO_OFFSET_DATE
- ISO_DATE
- ISO_LOCAL_TIME
- ISO_OFFSET_TIME
- ISO_TIME
- ISO_LOCAL_DATE_TIME
- ISO_ZONED_DATE_TIME
- ISO_DATE_TIME
- ISO_ORDINAL_DATE
- ISO_WEEK_DATE
- ISO_INSTANT
- RFC_1123_DATE_TIME

```
DateTimeFormat(NOW, "MILLIS") => "1709290260764"
DateTimeFormat(NOW, "hh:mm:ss") => "10:54:57"
```

Get

```
Get(expression: <multiExpression>, index: <number>)
```

This outputs the string at `index` index in the `expression` list, and None if the index does not exist. The index can be negative to get from the end of the list.

```
Get(["string1", "string2", "string3", "string4"], -2) => "string3"
Get(["string1", "string2"], 3) => None
```

First

```
First(expression: <multiExpression>)
```

This outputs the first string of the `expression` list, and None if it does not exist. The index can be negative to get from the end of the list.

```
First(["string1", "string2", "string3", "string4"]) => "string1"
First([[no.values]]) => None
```

Last

```
Last(expression: <multiExpression>)
```

This outputs the last string of the `expression` list, and None if it does not exist. The index can be negative to get from the end of the list.

```
Last(["string1", "string2", "string3", "string4"]) => "string4"
Last([[no.values]]) => None
```

Join

```
Join(`expression`: <multiExpression>, `separator`: <literal>)
```

This outputs the values of the `expression` joined with the `separator` string.

```
Join(["string1", "string2"], ".") => "string1.string2"
```

List of string functions



The following functions output a list of string or None.

Filter

```
Filter(expression: <multiExpression>, regex: <literal>)
```

This outputs a list of string from `expression` that matches the `regex`, None if none matches

```
Filter(["string1", "string2", "match"], "[a-z]+") => ["match"]
Filter(["string1", "string2"], "[a-z]+") => None
```

Slice

```
Slice(expression: <multiExpression>, start: <number>)
```

This outputs the slice of the `expression` list between `start` index and its end, or None if the slice is invalid. The index can be negative to get from the end of the list.

```
Slice(["string1", "string2", "string3", "string4"], -2) => ["string3", "string4"]
Slice(["string1", "string2"], 3) => None
```

Slice

```
Slice(expression: <multiExpression>, start: <number>, end: <number>)
```

This outputs the slice of the `expression` list between `start` and `end` index, or None if the slice is invalid. The index can be negative to get from the end of the list.

```
Slice(["string1", "string2", "string3", "string4"], 1, 3) => ["string1", "string2",
"string3"]
Slice(["string1", "string2"], 3) => None
```

Sort

```
Sort(`expression`: <multiExpression>)
```

This outputs the values of the `expression` sorted in alphabetical order.

```
Sort(["b", "a"]) => ["a","b"]
```

Split

```
Split(`expression`: <singleExpression>, `separator`: <literal>)
```

This outputs the values of the `expression` split with the `separator` string.

```
Split("string1andstring2", "and") => ["string1", "string2"]
Split("string1.string2", ".") => ["string1", "string2"]
```

ShortenDNS

```
ShortenDNS(`expression`: <singleExpression>)
```

This retrieves the first element of the DNS FQDN from `expression`.

```
ShortenDNS("subdomain.domain.com") => "subdomain"
```

DomainDNS

```
DomainDNS(`expression`: <singleExpression>)
```

This retrieves the domain FQDN of the DNS FQDN from `expression`.

```
DomainDNS("subdomain.domain.com") => "domain.com"
```

EmailUser

```
EmailUser(`expression`: <singleExpression>)
```

This retrieves the part before the @ from `expression`.

```
EmailUser("user@domain.com") => "user"
```

EmailDomain

```
EmailDomain(`expression`: <singleExpression>)
```

This retrieves the part after the @ from `expression`.

```
EmailDomain("user@domain.com") => "domain.com"
```

SamAccountNameUser

```
SamAccountNameUser(`expression`: <singleExpression>)
```

This retrieves the user part (before the \) from a SamAccountName in `expression`.

```
SamAccountNameUser("DOMAIN\User") => "User"
```

SamAccountNameDomain

```
SamAccountNameDomain(`expression`: <singleExpression>)
```

This retrieves the domain part (before the \) from a SamAccountName in `expression`.

```
SamAccountNameDomain("DOMAIN\User") => "DOMAIN"
```

2.15.6. Template Strings

Template Strings are augmented strings. They can be used as normal text but can also be augmented:

Using dictionary values

Using the following format, a dictionary key will be interpreted to its value when sending the notification:

```
{{<dictionary key>}}
```

Example:

```
I am enrolling on {{ca.name}}
```

Depending on the notification event, values will be added to context to be interpreted.



If the value is not available in the context, the dictionary value will not be replaced

Using computation rules

Using the following format, a computation rule will be interpreted to its value when sending the notification:

```
{{<computation rule>}}
```

Example:

```
I am enrolling on {{ Lower({{ca.name}}) }}
```

Depending on the notification event, values will be added to context to be interpreted in the computation rule.



If the computation rule result is None, an empty string will be displayed. If it is an array, it will be in a comma separated string

2.16. Reports

A report is a CSV file sent in a scheduled email. The CSV content is managed by:

- *HCQL query (certificates), HRQL query (requests)*
- *CSV fields shown*

Prerequisites

You may need [admin-guide:security-teams:::_teams].

How to configure Reports

1. Log in to Horizon Administration Interface.

2. Access Reports from the drawer or card: **Reports**.

3. Click on  .

4. Fill in the mandatory fields.

Details

- **Enable** *(boolean)*:
  Tells whether the reporting task should be enabled. Set by default at true.
- **Name*** *(string input)*:
  Enter a meaningful report name. It must be unique.
- **Cron scheduling expression in Quartz format*** *(cron expression)*:
  Enter a Cron scheduling expression (in Quartz format). The default expression is built to run the task every hour.

Recipients

Click on  to add a recipient.

You can either target:

- A static (recipient): you will need to set a valid email address.
- A team contact: you will need to select one of the enabled teams.

- A team manager: you will need to select one of the enabled teams.

Email

Common fields

- **From*** *(string input)*:
  Enter the email address that will appear in the "From" field of the email.
- **Subject*** *(string input)*:
  Enter the subject of the email.
- **Body** *(string input)*:
  Enter the body of the email.
- **Is HTML** *(boolean)*: (boolean):
  Sets whether the email body contains HTML code (true) or plain text (false). The default value is set to false.

ReportType

The report type determines how the CSV report will be sent. Depending on the user's choice, a set of parameters may appear.

If set to **link_email**, a temporary link will be made available in the email notification dictionary. In this case, the user must set a retention period to specify how long the CSV can be downloaded for.



- The provided email format is a relative URL, and the expected instance hostname should be prefixed to it.
- The user should add the temporary link to the email body.

- **Retention Period*** *(finite duration)*:
  Specify the period during which the report can be downloaded after generation. Using long periods is not recommended.

If set to `attachment_email`, then the report CSV will be attached to the email. In this case, the user can specify whether or not to compress the file.

- **CSV file name** *(string input)*:
  Enter the name that will be given to the attached csv file.
- **Compress file** *(boolean)*: (boolean):
  Sets whether the CSV must be compressed using gzip and adds the `.csv.gz` extension to the file.

HQL

- **HQL Type*** *(select)*:
  Either chose Certificate or Request. It will define the HQL Query type to set and the enabled CSV fields.
- **Query** *(string input or select)*:

HCQL (Certificate) or HRQL (Request). You can select one of your saved queries.

CSV

You can select which fields will appear on the CSV file.

5. Click on the save button.

You can run  , edit  or delete  the report .

2.17. Archives

Horizon has the capability to archive (extract) and purge (remove) the following elements:

- *Events: events older than a finite duration. For example, it is possible to enforce a retention of 3 months (90 days) of events in the database*
- *Certificates: expired certificates according to a filter expressed using HCQL. For example, all expired certificates on a certificate profile*

Archiving and purging can be performed for various reasons, the main being:

- *GDPR compliance: certificates and events may contain data that falls under the GDPR regulation*
- *performances: the volume of expired certificates and events may have impact on the performance of the database, specifically in the certificate dashboard and the event search*

Data are archived through parquet file. Parquet is a perfect choice for archiving data for the following reasons:

- *It is a standard format and offer broad compatibility with many data reading tools*
- *It offers strong compression capability*

As of now, archives can be stored:

- *In the MongoDB database using gridFS (default) _ In a S3 bucket (require advance configuration, please refer to the following documentation section)*

How to configure Archives

1. Log in to Horizon Administration Interface.

2. Access Archives from the drawer or card: **Archives**.

3. Click on  .

4. Fill in the mandatory fields.

General

- **Name*** *(string input)*:
  Enter a meaningful archive name. It must be unique.
- **Filename*** *(string input)*:
  Enter the file name. It must be unique on your storage.



By default archive storage is the mongo database for on premise instances. Horizon also supports S3, see the Advanced configuration guide to configure it.

- **Type*** *(select)*:
  Select the archive type:
  - certificate
  - event

Certificate

- **Archive Keys*** *(boolean)*:
  If enabled, escrowed private key will be added to the archive (encrypted).
- **Filter** *(string input)*:
  The HCQL filter to apply to the archive.

Event

- **Before*** *(date)*:
  Date before which to archive events. By default, only events older than 3 months are eligible for archiving.

5. Click on the save button.

Archive actions

Once an archive has been created, several actions are available:

-  Retry in case of failure
-  Download the archive parquet file
-  Cancel the archive. This will restore archived certificates as well as delete the archive file
-  Delete the archive. This is only available after a security period has passed. This period is 7 days by default but can be overridden



Archive deletion does NOT delete archives if using another storage backend than `gridfs`.

2.18. Page Moved

This page was moved to the Installation Guide

2.19. Logging

2.20. Event Codes

All events displayed in this document work in a similar manner. In case of failure, the event will display the reason of said failure. This behavior is also valid for warning-status events.

ACME

- **ACME-ACCOUNT-DELETE**
  This event is triggered when an account is deleted via the management API.
- **ACME-ACCOUNT-KEY-CHANGE**
  This event is triggered when an account key is updated.
- **ACME-ACCOUNT-REGISTER**
  This event is triggered when an account is unsuccessfully registered. Mainly due to errors in registration parameters (mail, name, ...)
- **ACME-ACCOUNT-STATUS-UPDATE**
  This event is triggered when an account status is updated via the management API.
- **ACME-ACCOUNT-UPDATE**
  This event is triggered when an account is unsuccessfully updated. Mainly due to errors in updated parameters (mail, name, ...)
- **ACME-AUTHORIZATION-DEACTIVATE**
  This event is triggered when an authorization is unsuccessfully deactivated.
- **ACME-CHALLENGE-REQUEST-VERIFY**
  This event is triggered when trying to use the challenge feature as an authentication method. It issues a warning if this is not applicable to this authentication case.
- **ACME-CHALLENGE-VERIFY**
  This event is triggered when a challenge is used as an authentication method. It issues a warning if the challenge is invalid or if the user doesn't correspond to the challenge.
- **ACME-EAB-ADD**
  This event is triggered when an External Account Binding is created via the management API.
- **ACME-EAB-DELETE**
  This event is triggered when an External Account Binding is deleted via the management API.
- **ACME-EAB-POLICY-ADD**
  This event is triggered when an External Account Binding policy is created via the management API.
- **ACME-EAB-POLICY-DELETE**
  This event is triggered when an External Account Binding policy is deleted via the management

API.

- **ACME-EAB-POLICY-UPDATE**
  This event is triggered when an External Account Binding policy is updated via the management API.
- **ACME-EAB-RENEW**
  This event is triggered when an External Account Binding MAC key is renewed via the management API.
- **ACME-EAB-STATUS-UPDATE**
  This event is triggered when an External Account Binding status is updated via the management API.
- **ACME-EAB-UPDATE**
  This event is triggered when an External Account Binding is updated via the management API.
- **ACME-ORDER-CERTIFICATE**
  This event is triggered when a user tries to access a certificate. It presents a failure in case the user doesn't have the necessary rights and permissions.
- **ACME-ORDER-FINALIZE**
  This event traces the status of the certificate's status. It presents a failure if the certificate is pending or if is not valid.
- **ACME-ORDER-GET**
  This event is triggered when a user tries to get orders via POST-AS-GET request for instance. It issues a failure if the user doesn't have the necessary rights and permissions for requesting that.
- **ACME-ORDER-NEW**
  This event is triggered when a user tries to order a certificate. It issues a failure if the user doesn't have the necessary rights and permissions for requesting this type of new certificate.
- **ACME-ORDER-UPDATE**
  This event is triggered when a user tries to order an update on certificate. It issues a failure if the user doesn't have the necessary rights and permissions to update this type of certificates.
- **ACME-REVOKE**
  This event is triggered when Horizon tries revoking a certificate using the ACME protocol. A warning can occur if the certificate is already revoked. A failure can occur if the certificate cannot be found based on the provided thumbprint.

ANALYTICS

- **ANALYTICS-CERTIFICATES-FLUSH**
  This event occurs when the certificate analytics database is manually flushed.
- **ANALYTICS-DISCOVERY-EVENTS-FLUSH**
  This event occurs when the discovery event analytics database is manually flushed.
- **ANALYTICS-EVENTS-FLUSH**
  This event occurs when the event analytics database is manually flushed.

ARCHIVE

- **ARCHIVE-CANCEL**
  This event is triggered when an error occurs while launching a restoration process.
- **ARCHIVE-DOWNLOAD**
  This event is triggered when an error occurs while downloading an archive.
- **ARCHIVE-END**
  This event is triggered when an archive process ends.
- **ARCHIVE-RUN**
  This event is triggered when an error occurs while launching an archiving process.
- **ARCHIVE-START**
  This event is triggered when an archive process starts.

ASYNC-ENROLLMENT

- **ASYNC-ENROLLMENT-POLL**
  This event is triggered when the background process polls an in-progress enrollment request.

AUTO-RENEW

- **AUTO-RENEW**
  This event is triggered on the auto renew process.

BOOTSTRAP

Bootstrap events relate to the initial setup of the Horizon platform.

- **BOOTSTRAP-ADMINISTRATOR-ACCOUNT**
  This event is triggered when installing Horizon, it corresponds to the creation of the administrator local identity on Horizon.
- **BOOTSTRAP-ADMINISTRATOR-PRINCIPAL**
  This event is triggered when installing Horizon, it corresponds to the creation of a link between the administrator account and its rights.
- **BOOTSTRAP-GRADING-POLICY**
  This event is triggered when installing Horizon, it corresponds to the creation of the "Horizon Grading Policy" which itself contains different grading rulesets.
- **BOOTSTRAP-GRADING-RULESET**
  This event is triggered when installing Horizon, it corresponds to the creation of different grading rulesets. For more information about those grading rule sets, click here.
- **BOOTSTRAP-LOCAL-IDENTITY-PROVIDER**
  This event is triggered when installing Horizon, it corresponds to the creation of a provider of type Local so that the administrator can connect after startup.
- **BOOTSTRAP-PASSWORD-POLICY**
  This event is triggered when installing Horizon, it corresponds to the creation of the Horizon-

Default password policy.

- **BOOTSTRAP-SYSTEM-CONFIGURATION**
  This event is triggered when installing Horizon, it corresponds to the creation of internal configuration elements such as the CRON internal monitor.

CA

- **CA-CERT-SYNC**
  This event is triggered when a Certification Authority is revoked and certificates managed in Horizon are subsequently revoked. The synchronization revokes all the underlying certificates.
- **CA-CRL-SYNC**
  This event is triggered when Horizon tries fetching a CRL from a specified CRLDP and synchronizes the revocation status in the database.
- **CA-CRL-UPDATE**
  This event is triggered when Horizon tries fetching a CRL from a specified CRLDP and updates the cached entries.

CONF

CONF events are triggered when users interact with configuration elements. This includes protocol profiles, notification triggers, Certification Authorities…

- **CONF-ADD**
  This event is triggered when a user tries to add a configuration element.
- **CONF-DELETE**
  This event is triggered when a user tries to delete a configuration element.
- **CONF-EXPORT**
  This event is triggered when a user tries to export configuration.
- **CONF-IMPORT**
  This event is triggered when a user tries to import configuration.
- **CONF-TEST**
  This event is triggered when a notification test happens.
- **CONF-UPDATE**
  This event occurs when a user tries to modify a configuration element.

CRMP

- **CRMP-AUTHENTICATION**
  This event occurs when a user tries to authenticate. It fails if the authentication is invalid.
- **CRMP-BAD-REQUEST**
  This event occurs when a wrong request is issued. For instance if an unavailable action is requested.
- **CRMP-ENROLL**
  This event occurs when an enrollment request happens. It fails if the CRMP enrollment is

unsuccessful.

- **CRMP-LIST**
  This event occurs when a user tries to access the profiles list. Fails if he doesn't have the required rights and authorisations.
- **CRMP-PROFILE-PROPERTIES**
  This event occurs when a user tries to access a profile. Fails if he doesn't have the required rights and authorisations or if the profile doesn't exist.
- **CRMP-RECOVER**
  This event occurs when a user tries to recover a CRMP certificate. It fails if it is not technically possible or if the user doesn't have the necessary rights and permissions.
- **CRMP-RETRIEVE**
  This event occurs when a user tries to retrieve certificates. It issues a warning if the research field is empty.
- **CRMP-REVOKE**
  This event occurs when a user tries to revoke a certificate. It fails or issues a warning respectively if the user doesn't have the necessary rights and permissions or if the certificate is expired.

DATASOURCE

- **DATASOURCE-IGNORED**
  This event occurs when a datasource is not executed because its inputs where not filled. This could indicate a misconfiguration of the datasource flow.

DCV

- **DCV-POLICY-END**
  This event is triggered when a DCV policy execution is finished with a summary on what happened
- **DCV-POLICY-ERROR**
  This event is triggered when a DCV policy fails to run (provider missing, connector error, etc.).
- **DCV-POLICY-RUN**
  This event is triggered when a DCV policy starts processing domains.

DISCOVERY

- **DISCOVERY-CAMPAIGN-FLUSH**
  This event is triggered when running a Discovery campaign.

EST

- **EST-CACERTS**
  This event is triggered when an error occurs during the call to the CACert endpoint when using the EST protocol.

- **EST-REVOKE-ON-RENEW**
  This event is triggered when enforcing max certificate per holder on the EST protocol.

  Deprecated since version 2.4.0

- **EST-SIMPLE-ENROLL**
  This event is triggered when enrolling a certificate through the EST protocol.
- **EST-SIMPLE-REENROLL**
  This event is triggered when re-enrolling a certificate through the EST protocol.

EVENT COMPLIANCE

- **INVALID-SEAL-PENDING-EVENT**
  This event occurs when a pending event has an invalid seal (indicating data corruption in the pending events collection).
- **UNSEALED-PENDING-EVENT**
  This event occurs when a pending event has no seal (indicating data corruption in the pending events collection).

GRADING

- **GRADING-END**
  This event is triggered at the end of the grading process of a certificate.
- **GRADING-ERROR**
  This event is triggered if an error occurs while grading a certificate.
- **GRADING-START**
  This event is triggered at the beginning of the grading process of a certificate.

INTERNAL MONITOR

- **INTERNAL-MONITOR-INIT**
  This event occurs when a bad initialization of the internal monitor happens. It is a failure case, happening for instance when it is not configured
- **INTERNAL-MONITOR-RUN**
  This event occurs when the internal monitor completes successfully.

LICENSE

- **LICENSE-ERROR**
  This event occurs when an error is related to the License. For example, when the license in use is expired.
- **LICENSE-LIMIT-REACHED**
  This event is triggered when a limit built into the license is the reached. For example, if only one discovery campaign is available, then reaching that threshold will trigger an error saying "Maximum number of discovery campaign(s) reached (x)" where x is the availability threshold.

LIFECYCLE

- **LIFECYCLE-CANCEL**
  This event is triggered when Horizon tries to cancel a certificate request.
- **LIFECYCLE-ENROLL**
  This event is triggered when a user tries to enroll an end-entity certificate. The event specifies the Distinguished Name of the enrolled certificate, its serial number as well as the Certificate Authority that enrolled said certificate in case of success. In case of failure, the reason of the failure is specified (e.g.: "Unauthorized DN element").
- **LIFECYCLE-ESCROW**
  This event is triggered when Horizon tries to escrow a key for an issued certificate.
- **LIFECYCLE-IMPORT**
  This event is triggered when trying to import a certificate in Horizon. Import here is the use of the import workflow.
- **LIFECYCLE-MAX-CERT-PER-HOLDER**
  This event is triggered when an error occurs trying to enforce the max certificates per holder parameter.
- **LIFECYCLE-MIGRATE**
  This event is triggered when trying to migrate certificates. This means taking under Horizon management a discovered certificate.
- **LIFECYCLE-RECOVER**
  This event is triggered when a user tries to recover a certificate.
- **LIFECYCLE-RENEW**
  This event is triggered when Horizon tries to renew a certificate.
- **LIFECYCLE-REVOKE**
  This event occurs when a user tries to revoke a certificate. Note that no event is triggered when a certificate expires.
- **LIFECYCLE-UPDATE**
  This event is triggered when a user tries updating the details related to a certificate. The Labels and the Ownership can be edited.

PKI CONNECTOR

- **ACTOR**
  This event is triggered when a PKI connector cannot be properly built between Horizon and the chosen PKI.

  Deprecated since version 2.4.0
- **PKI-CONNECTOR**
  This event is triggered when a PKI connector cannot be properly built between Horizon and the chosen PKI.

REPORT

- **REPORT-CSV-DELETE-ERROR**
  This event is triggered when someone is trying to delete a csv form a report but the deletion failed.
- **REPORT-CSV-DOWNLOAD-ERROR**
  This event is triggered when someone is trying to download a csv form a report but the download failed.
- **REPORT-CSV-FETCH-ERROR**
  This event is triggered when someone is trying to list all csv metadata linked to a report from the gridFS, but the list failed.
- **REPORT-CSV-LIST-ERROR**
  This event is triggered when someone is trying to list all reports metadata from the gridFS, but the list failed.

REQUEST

- **REQUEST-APPROVE**
  This event is triggered when approving a request.
- **REQUEST-CANCEL**
  This event is triggered when cancelling a request.
- **REQUEST-DENY**
  This event is triggered when a request is denied.
- **REQUEST-SUBMIT**
  This event is triggered when submitting a request.
- **REQUEST-TEMPLATE**
  This event is triggered when requesting a template. It can fail when trying to enroll a workflow without a module.
- **WEBRA-CHALLENGE-SUBMIT**
  This event is triggered when a WebRA challenge is submitted.

SCEP

- **SCEP-ENROLL**
  This event is triggered when enrolling a certificate via SCEP. Fails when missing mandatory certificate's elements or when missing rights and/or permissions to enroll the certificate.
- **SCEP-GET-CA-CERT**
  This event is triggered when requesting a CA certificate via SCEP. Fails when missing mandatory certificate's elements or when missing rights and/or permissions to enroll the certificate.
- **SCEP-GET-CERT-INITIAL**
  This event is triggered when requesting the initial certificate via SCEP. Fails when missing mandatory certificate's elements or when missing rights and/or permissions to enroll the certificate.

- **SCEP-GET-RA**
  This event is triggered when the Horizon API Gateway retrieves a SCEP Registration authority for validation. It fails if an unexpected error happens during the process.
- **SCEP-NDES-EMULATION**
  This event is triggered when requesting a certificate with the scep profile template using the NDES server. It fails if the two don't comply with one another.
- **SCEP-PKI-CLIENT**
  This event is triggered when using the pkiclient profile. It fails if the request is invalid, if the operation is not allow for the type of certificate the user wants to manage, or if the user doesn't have the necessary rights and permissions to execute the action.
- **SCEP-PKI-OPERATION**
  This event is triggered when operating through the PKI.
- **SCEP-RENEW**
  This event is triggered when renewing a certificate. It fails he is system fails to enroll the new certificate.
- **SCEP-REVOKE-ON-RENEW**
  This event is triggered when enforcing max certificate per holder on the SCEP protocol.



Deprecated since version 2.4.0

SCHEDULED TASK

- **SCHEDULED-TASK-COMPLETE**
  This event is triggered when a scheduled task end. It fails if the task fails.
- **SCHEDULED-TASK-RUN**
  This event is triggered when trying to pass a scheduled task to "running" status. Fails if this status is not achieved.

SECURITY

- **SEC-AUTHENTICATION**
  This event is triggered when a user tries to connect. The local or OpenID identifier is specified whether it is a failure or a success

AUTHORIZATION



These events relate to the Security>Access Management>Authorizations tab under configuration.

- **SEC-AUTHORIZATION-ADD**
  This event is triggered when a user tries to create an authorization object.
- **SEC-AUTHORIZATION-DELETE**
  This event is triggered when a user tries to delete an authorization object.
- **SEC-AUTHORIZATION-UPDATE**

This event is triggered when a user tries to modify elements inside an authorization object. The event specifies the modified fields.

CREDENTIALS



These events relate to the Security>Credentials tab under configuration.

- **SEC-CREDENTIALS-ADD**
  This event occurs when a user tries creating new credentials.
- **SEC-CREDENTIALS-DELETE**
  This event occurs when a user tries deleting credentials.
- **SEC-CREDENTIALS-UPDATE**
  This event occurs when a user tries updating credentials.

IDENTITY



These events relate to the Security>Access Management>Identity tab under configuration.

- **SEC-IDENTITY-PROVIDER-ADD**
  This event occurs when a user tries creating an identity provider profile.
- **SEC-IDENTITY-PROVIDER-DELETE**
  This event occurs when a user tries deleting an identity provider profile.
- **SEC-IDENTITY-PROVIDER-UPDATE**
  This event occurs when a user tries modifying an identity provider profile. The modified fields are specified in the event.

LOCAL IDENTITY



These events relate to the Security>Access Management>Local accounts tab under configuration.

- **SEC-LOCAL-IDENTITY-ADD**
  This event is triggered when a user tries creating a local account.
- **SEC-LOCAL-IDENTITY-DELETE**
  This event is triggered when a user tries to delete a local account.
- **SEC-LOCAL-IDENTITY-RESET**
  This event is triggered when executing the reset password workflow.
- **SEC-LOCAL-IDENTITY-UPDATE**
  This event is triggered when a user tries modifying a local account. The modified fields are specified. Updating the password falls in this event.

PASSWORD POLICY



These events relate to the Security>Password Policies tab under configuration.

- **SEC-PASSWORD-POLICY-ADD**
  This event is triggered when a user tries creating a new password policy.
- **SEC-PASSWORD-POLICY-DELETE**
  This event is triggered when a user tries deleting a password policy.
- **SEC-PASSWORD-POLICY-UPDATE**
  This event is triggered when a user tries modifying a password policy.

ROLE



These events relate to the Security>Access Management>Roles tab under configuration.

- **SEC-ROLE-ADD**
  This event is triggered when a user tries to create a new role.
- **SEC-ROLE-DELETE**
  This event is triggered when a user tries to delete a role.
- **SEC-ROLE-MEMBERS-ADD**
  This event is triggered when a user tries to add members to a role.
- **SEC-ROLE-MEMBERS-REMOVE**
  This event is triggered when a user tries to remove members from a role.
- **SEC-ROLE-UPDATE**
  This event is triggered when a user tries to modify a role. The modified fields are specified in the event.

SCIM PROFILE



These events relate to the Security>SCIM Profiles tab under configuration.

- **SEC-SCIM-PROFILE-ADD**
  This event is triggered when a user tries creating a new SCIM profile.
- **SEC-SCIM-PROFILE-DELETE**
  This event is triggered when a user tries deleting a SCIM profile.
- **SEC-SCIM-PROFILE-UPDATE**
  This event is triggered when a user tries modifying a SCIM profile.

SERVICEACCOUNT

- **SERVICE-ACCOUNT-ADD**
  This event is triggered when adding a new service account
- **SERVICE-ACCOUNT-DELETE**

This event is triggered when deleting a service account

- **SERVICE-ACCOUNT-GET**
  This event is triggered when getting a service account
- **SERVICE-ACCOUNT-UPDATE**
  This event is triggered when updating a service account

TEAM



These events relate to the Security>Teams tab under configuration.

- **SEC-TEAM-ADD**
  This event is triggered when a user tries creating a team.
- **SEC-TEAM-DELETE**
  This event is triggered when a user tries deleting a team.
- **SEC-TEAM-MEMBERS-ADD**
  This event is triggered when a user tries to add members to a team.
- **SEC-TEAM-MEMBERS-REMOVE**
  This event is triggered when a user tries to remove members from a team.
- **SEC-TEAM-SWITCH**
  This event is triggered when using the team switch feature (renaming team).
- **SEC-TEAM-UPDATE**
  This event is triggered when a user tries modifying a team element (that does not include adding/removing users).
- **TEAM-SWITCH**
  This event is triggered when using the team switch feature (renaming team).



Deprecated since version 2.4.0

TENANT

- **SEC-TENANT-ACCOUNT-RESET**
  This event occurs when a user tries to reset a tenant's admin account.
- **SEC-TENANT-ADD**
  This event occurs when a user tries to add a tenant.
- **SEC-TENANT-DELETE**
  This event occurs when a user tries to delete a tenant.
- **SEC-TENANT-RESTORE**
  This event occurs when a user tries to restore a tenant.
- **SEC-TENANT-UPDATE**
  This event occurs when a user tries to update a tenant.

SERVICE

- **SERVICE-START**
  This event is triggered when the Horizon service is started.
- **SERVICE-STOP**
  This event is triggered when the Horizon service is manually stopped.

SYNC

Synchronization events are triggered by scheduled task when synchronizing a third party connector state with Horizon

- **SYNC-ENROLL**
  This event is triggered when syncing with a third party triggers an enrollment.
- **SYNC-RENEW**
  This event is triggered when syncing with a third party triggers a renewal.
- **SYNC-REVOKE**
  This event is triggered when syncing with a third party triggers a revocation.

THIRD PARTY

- **THIRD-PARTY-CONNECTOR**
  This event is triggered as a warning when Horizon cannot build a connection with a third party.

TRIGGER

Trigger events relate to *Notifications* and can occur based on configurations made under *Third Parties* or under *Protocols*.

- **TRIGGER-DELETE**
  This event occurs when Horizon tries deleting a certificate from a third party.
- **TRIGGER-EMAIL**
  This event occurs when a Trigger that sends an email is activated. The event specifies to whom the email is addressed.
- **TRIGGER-NOTIFICATION**
  This event occurs when a Trigger that sends a notification is activated.
- **TRIGGER-PUSH**
  This event occurs when Horizon tries to push a certificate to a third party.
- **TRIGGER-REMOVE**
  This event occurs when Horizon orders a third party to remove a certificate.

WCCE

- **WCCE-ENROLL**
  This event is triggered when a client tries to enroll a certificate through Horizon using the

WCCE protocol.

Chapter 3. User guide

Description

The user guide describes how end-users should interact with Horizon.

Prerequisites

To use Horizon, you need the following prerequisites:

- an up and running Horizon instance, which you can access through your web browser;
- a correctly configured platform;

3.1. Managing requests on the WebRA

Each Request has the same lifecycle described by the following figure.

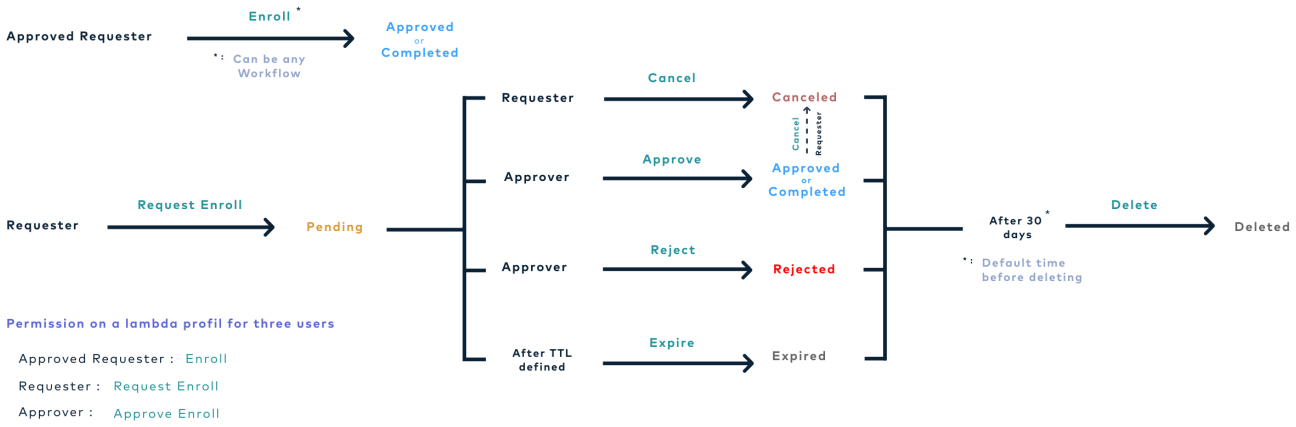


Figure 1. Request Workflow

Requester

A requester is someone who is granted the permission to request a certificate (enroll, renew, revoke, update, recover).

Approver

An approver is someone who is granted the permission to approve a request (enroll, renew, revoke, update, recover). An approver cannot approve its own request.

Owner

A request owner is someone who is designated as the benefactor for the request. It can view the request like the requester (in the `My requests` drawer), but unlike the requester, they can also access the certificate information (PKCS#12, challenge password).

The owner is computed according to the following rules:

- **enroll, update, migrate**: the owner is the one defined in the request template (ownership tab)
- **renew**: the owner of the request is the owner of the renewed certificate
- **recover**: the owner is the requester of the recover request
- **revoke**: no owner is associated with the request

Table 6. Owner vs Requester

User type	Can view the request	Can view the PKCS#12	Can view the challenge password
Requester	Yes	No	No
Owner	Yes	Yes	Yes



Any user with the `Enroll API/ Renew API` permission can access the PKCS#12 or the challenge password for the workflow regardless of ownership status

3.1.1. How to enroll a certificate using the WebRA

1. Log in to Horizon registration authority Interface

2. Access Request Certificate from the drawer:  **Request Certificate**

Profile tab

3. Fill in all the mandatory fields

- **Certificate profile***_(string select)_:
  The certificate profile will be used in order to build the next step of the enrollment.

If decentralized enrollment is enabled for the profile:

Either:

- **CSR***_(string)_:
  The CSR in PEM format
- **Import a CSR file***_(file)_:
  The CSR file

If centralized enrollment is enabled for the profile:

- **Key type***_(string select)_:
  The key type will be used for the private key generation

In case of the definition of a password policy:

- **Password***_(string)_:
  The password will be used for the PKCS#12 encryption



You must comply with the configured password policy.

4. Click on Next button.

Data tab

5. Fill in all the mandatory fields:

- **Subject***(string)*:
  Fill the subject fields of the certificate
- **Subject Alternatives Names***(string)*:
  Fill the Subject Alternative Names of the certificate
- **Extensions***(string)*:
  Fill the extensions of the certificate



In decentralized mode, CSR values will be used as default for the corresponding fields.



You must comply with the configured regular expression(s) that you can get with the ? icon.

6. Click on next button.

Labels tab

7. Fill in all the mandatory fields:

- **Labels***(string)*:
  The labels will be used for permission, email and certificate search.



You must comply with the configured regular expression(s) that you can get with the ? icon.

- **Requester comment** *(string)*:
  This comment appears:
  - to the approver when your request is in the pending status.
  - in the certificate info after the enrollment.

8. Click on next button.

Ownership tab

9. Fill in all the fields:

- **Owner** *(string input)*:
  Displayed if an owner policy is set. The owner of the certificate can search it, and request other actions on it (such as revoke, recover, ..).

- **Contact email address** *(string email format)*:
  Displayed if an email policy is set. An email can be sent each time the request status changes (see request lifecycle). This will also set the contact email of the certificate.
- **Team** *(string input)*:
  Displayed if a team policy is set. A team has the same rights as an owner on a certificate.

10. Click on next button.

Summary tab

If you own the enrolling permission

11. Click on enroll button
You can download the PKCS#12 after the enrollment if you are allowed to in the profile

If you own the request certificate permission

11. Click on request button
You have to wait until your request is approved, afterward you will be able to download the PKCS#12 if you are allowed to in the profile

3.1.2. How to request a certificate revocation

1. Log in to Horizon registration authority Interface

2. Access Either my certificates or Search certificates from the drawer: **My Certificates/Search Certificates**

3. Click on the Revoke icon 



The revoke icon appears only if you own the permission to revoke the certificate

Revocation Options tab

4. Fill in all the mandatory fields.

- **Revocation reason*** *(String select)*:
  The revocation reason that will appear on the CRL
- **Contact email address** *(string email format)*:
  Used if an email configuration is set. An email can be sent each time the request status changes (see request lifecycle)
- **Requester comment** *(String)*:
  This comment appears:
  - by the approver when your request is in the pending status
  - to the certificate info after the revocation

5. Click on Certificate tab

Certificate tab

6. Check the certificate's information

7. Click on Ownership tab

Ownership tab

8. Check the certificate's ownership information

If you have the revoke permission

9. Click on the revoke button
The certificate is now revoked.

9. Click on request button
You have to wait until your request is approved, afterward you will be able to see the certificate as revoked when you search for it

3.1.3. How to request a certificate update

1. Log in to Horizon Registration Authority Interface

2. Access request update from the drawer: **My certificates** or **Search certificates**

3. Click on request update button 

Labels tab

4. You can update in the labels section the labels

- **Label** *(string input)*:
  Enter a correct label

Ownership tab

5. You can update the ownership information

- **Owner** *(string input)*:
  Displayed if an owner policy is set. The owner of the certificate can search it, and request other actions on it (such as revoke, recover, ..).
- **Contact email address** *(string email format)*:
  Displayed if an email policy is set. An email can be sent each time the request status changes (see request lifecycle). This will also set the contact email of the certificate.
- **Team** *(string input)*:
  Displayed if a team policy is set. A team has the same rights as an owner on a certificate.

6. You can also check the details information

Certificate tab

7. You can also check the certificate information

8. Once you have made changes you can request the update by clicking on the update button

3.1.4. How to request a certificate duplication

A Duplication is a simplification of the enroll process. When choosing the duplication on a certificate, a new certificate enrollment request is created with the information from the previous certificate. Certificate data and metadata are still editable, as opposed to a renewal.

1. Log in to Horizon Registration Authority Interface

2. Access request duplication from the drawer: **My certificates** or **Search certificates**

3. Click on request duplication button 

Profile tab

4. Fill in all the mandatory fields

- **Key type*** *(string)*:
  The key type will be used for the private key generation

In case of the definition of a password policy:

- **Password***(string)*:
  The password will be used for the PKCS#12 encryption

5. Go to enroll (same as duplicate) and follow all the steps

3.1.5. How to request a certificate renewal

A certificate renewal will enroll a certificate strictly identical to the previous one. No edition of certificate data or metadata can take place.

1. Log in to Horizon Registration Authority Interface

2. Access request renew from the drawer: **My certificates** or **Search certificates**

3. Click on request renew button 

Renew options tab

4. Fill in all the fields

- **Key type*** *(string)*:
  *Enabled on **centralized** enrollment:* The key type will be used for the private key generation.
- **Password***(string)*:
  *Enabled on **centralized** enrollment with **manual** password policy:* The password will be used for the PKCS#12 encryption.

- **CSR*** *(string)*:
  *Enabled on **decentralized** enrollment:* The CSR, defining the public key of the enrolled certificate.
- **Comment** *(string)*:
  This comment appears:
  - to the approver when your request is in the pending status.
  - in the certificate info after the enrollment.

Certificate tab

5. You can also check the certificate information

Ownership tab

6. You can also check the certificate ownership information

7. Renew. You will obtain a strictly identical certificate to the one used for renewal, except for the key.

3.1.6. How to request a certificate recovery

1. Log in to Horizon Registration Authority Interface

2. Access request recover from the drawer: **My certificates** or **Search certificates**

3. Click on request recover button 

Recover Options tab

4. Fill in the information you want to add.

- **Contact_Email** *(string email format)*:
  Used if an email configuration is set. An email can be sent every time the request status change (see request lifecycle).
- **Recover comment** *(string input)*:
  This comment appears:
  - to the approver when your request is in the pending status.
  - in the certificate info after the enrollment.

Certificate tab

5. You can also check the certificate information

Ownership tab

6. You can also check the certificate ownership information

7. Once you have checked and added the information you wanted you can request the recover by

clicking on the recover button

8. You will be able to see and copy the password and download the certificate PKCS#12

3.2. Requesting a SCEP challenge

This section details how you can get a SCEP Challenge.

1. Log in to Horizon Registration Authority Interface

2. Access Request a SCEP Challenge from the drawer: **Request a SCEP Challenge**



You must have the permission to request a SCEP challenge on at least one SCEP profile.

Profile tab

1. Select the SCEP profile

2. Click on next button

Metadata tab

1. Fill in all the mandatory fields:

- Labels(string):
  The labels are used for permission, email and request search.
- Contact email address(string email format):
  Used if an email configuration is set. An email can be sent each time the request status changes (see request lifecycle).
- Requester comment(string):
  This comment appears:
  - to the approver when your request is in the pending status
  - in the certificate information after the enrollment

2. Click on next button

Summary

If you own the enrolling permission on the SCEP profile:

1. Click on the Retrieve challenge button

If you own the request permission on the SCEP profile:

1. Click on request button



You have to wait that your request is approved by an operator and its status is 'completed', in order to use your SCEP challenge

2. click on View Request 

You now have access to your SCEP challenge



In order to enroll using SCEP you will need at least a challenge and the SCEP endpoint:

- https://<horizon_url>/scep/<profile>/pkiclient.exe

In case you use the NDES emulation, the enrollment and challenge URLs will be respectively: - https://<horizon_url>/certsrv/<profile>/mscep - https://<horizon_url>/certsrv/<profile>/mscep_admin



You can cancel your request at any time, as long as the request status is pending, by clicking on 

3.3. Requesting an EST challenge

This section details how you can get an EST Challenge.

1. Log in to Horizon Registration Authority Interface

2. Access Request an EST Challenge from the drawer:  **Request an EST Challenge**



You must have the permission to request an EST challenge on at least one EST profile.

Profile tab

1. Select the EST profile.

2. Click on next button.

Metadata tab

1. Fill in all the mandatory fields:

- Labels(string):
  The labels are used for permission, email and request search.
- Contact email address(string email format):
  Used if an email notification is set. An email can be sent each time the request status changes

(see request lifecycle).

- Requester comment(string):
  This comment appears:
  - to the approver when your request is in the pending status
  - in the certificate information after the enrollment

2. Click on next button

Summary

If you own the enrolling permission on the EST profile:

1. Click on the Retrieve challenge button

If you own the "request" permission on the EST profile:

1. Click on request button



You have to wait that your request is approved by an operator and its status is 'completed', in order to use your EST challenge

2. click on View Request 

You now have access to your EST challenge



You can cancel your request at any time, as long as the request status is pending, by clicking on 

How to enroll using EST

This section details how to enroll using the Horizon Client (`horizon-cli`). It is also possible to use another EST client implementation, as long as it complies with RFC 7030.

Prerequisites

You need the `horizon-cli` tools

Enroll with Horizon Client

1. Set the horizon root endpoint

```
export ``ENDPOINT``=https://<horizon_url>
```



The `endpoint` can instead be set in `horizon-cli` configuration file

2. Enroll with `horizon-cli`

```
horizon-cli est --enroll <your_challenge> --profile <est_profile> --key
<link_to_the_privatekey> --cn <certificate_cn> --cert <name_of_the_output_certificate>
```



If the enrollment succeeds, the challenge is no longer usable, as it is a one-time password.

3.4. Requesting a WebRA challenge

This section details how you can get a WebRA Challenge, used to enroll a certificate on a WebRA profile in Challenge mode, for instance with Horizon CLI.

1. Log in to Horizon Registration Authority Interface

2. Access Request a WebRA Challenge from the drawer: **Request a WebRA Challenge**



You must have the permission to request a WebRA challenge on at least one WebRA profile in Challenge mode.

Profile tab

1. Select the WebRA profile. Only profiles in Challenge mode are listed.

2. Click on next button.

Metadata tab

1. Fill in all the mandatory fields:

- Labels(string):
  The labels are used for permission, email and request search.
- Requester comment(string):
  This comment appears:
  - to the approver when your request is in the pending status
  - in the certificate information after the enrollment

2. Click on next button

Ownership tab

1. Fill in the owner, contact email and team of the future certificate, according to the ownership policy of the profile.

2. Click on next button

Summary

If you own the enrolling permission on the WebRA profile:

1. Click on the Retrieve challenge button. The challenge is displayed and can be copied.

If you own the "request" permission on the WebRA profile:

1. Click on request button



You have to wait that your request is approved by an operator and its status is 'completed', in order to use your WebRA challenge

2. click on View Request 

You now have access to your WebRA challenge



The challenge can be used only once, and must be used before it expires.

3.5. Managing requests (operator)

An Operator is someone who owns the permission to approve or deny a request.

Manage Request

This section details how to manage a request (view, approve, deny).

1. Log in to Horizon Registration Authority Interface

2. Access Manages requests from the drawer: **Manage requests**

How to view a request

3. Click on view request button

4. Check all the information from the request

5. At the end you can either approve or deny the request

How to approve a request

3. Click on approve request button and approve the request



If the certificate has mandatory metadata you will need to fill it in before approving the request, otherwise you will get an error.

How to deny a request

3. Click on deny request button and deny the request

3.6. Searching requests and certificates

Search Request

Here is the section where you can search easily find all information regarding the request.

How to do a simple request search

1. Log in to Horizon Registration Authority Interface

2. Access request search from the drawer: **My request** or **Request dashboard**

3. Fill in the information you want to look at:

- **Search in request DNs** *(string input)*:
  Enter the Certificate DNs you are looking for in a request
- **Search in IDs** *(string input)*:
  Enter the IDs you are looking for in a request
- **Search in Requester** *(string input)*:
  Enter the Requester you are looking for in a request
- **Search in Protocols** *(string input)*:
  Select the Protocols you are looking for in a request
- **Include status** *(string select multiple)*:
  Select the status you are looking for in a request
- **Include workflow** *(string select)*:
  Select if the workflow you are looking in a request
- **Include expired requests** *(string select)*:
  Select if the request you are looking is expired

4. Click on the filter button

You can reset the search by clicking on reset button

Search Certificate

Here is the section where you can search easily find all information regarding the certificate.

How to do a simple certificate search

1. Log in to Horizon Registration Authority Interface

2. Access certificate search from the drawer: **My certificates** or **Search certificates** or **Certificates dashboard**

3. Fill in the information you want to look at:

- **Search in DNs** *(string input)*:
  Enter the DNs you are looking for in a certificate
- **Search in SANs** *(string input)*:
  Enter the SANs you are looking for in a certificate
- **Search in serials** *(string input)*:
  Enter the serials you are looking for in a certificate
- **Search in issuer DNs** *(string input)*:
  Enter the issuer DNs you are looking for in a certificate
- **Expiration date start** *(string input)*:
  Enter the expiration date start you are looking for in a certificate
- **Expiration end start** *(string input)*:
  Enter the expiration end start you are looking for in a certificate
- **Search in profiles** *(string input)*:
  Enter the profile you are looking for in a certificate
- **Search in modules** *(string select multiple)*:
  Select the module you are looking for in a certificate
- **Search in Discovery campaigns** *(string input)*:
  Enter the discovery campaigns you are looking for in a certificate
- **Include status** *(string select multiple)*:
  Select the status you are looking for in a certificate
- **Include signed** *(string select)*:
  Select if the certificate you are looking is Self-Signed or Not Self-Signed or all
- **Include discovery** *(string select)*:
  Select if the certificate you are looking is Discovered trusted or Discovered not trusted

4. Click on the search button

You can reset the search by clicking on reset button or try the expert mode by clicking on expert mode button

How to do an expert certificate search

1. Log in to Horizon Registration Authority Interface

2. Access certification search from the drawer: **My certificates** or **Search certificate** or **Certificate dashboard**

3. Enter your research line.

To do so you will need to click on the input field. A list appears and you will be able to choose between all selector, then condition, then field, and you can add an operator to refine the search.

- **Element*** *(string input)*:
  Enter the element you are looking for in a certificate
- **Condition*** *(string input)*:
  Enter the condition you are looking for in a certificate for this element
- **Field*** *(string input)*:
  Enter the name of the element
- **Operation*** *(string input)*:
  Choose an operator if you want to refine your search

Certificate Search Structure

- <element> <condition> <"name"> (<operator> [<element> <condition> <"name">])

Table 7. Table element

Name	Type	Description
`dn`	*string*	Distinguished name
`san`	*string*	Subject Alternative name
`serial`	*string*	Certificate serial number
`issuer`	*string*	Issuer distinguished name
`status`	*string*	Certificate status
`module`	*string*	Certificate module
`profile`	*string*	Certificate profile
`valid.until`	*date*	Certificate 'not after' date
`valid.from`	*date*	Certificate 'not before' date
`keytype`	*string*	Certificate key type
`signingalgorithm`	*string*	Certificate signing algorithm
`owner`	*string*	Certificate owner
`holderid`	*string*	Certificate holder ID
`metadata.contact_email`	*string*	Contact email
`metadata.pki_connector`	*string*	PKI Connector

Name	Type	Description
`label.`	*string*	Label
`discoveryinfo.campaign`	*string*	Discovery campaign
`discoverydata.ip`	*string*	Discovery IP
`discoverydata.hostnames`	*string*	Discovery Hostnames
`discoverydata.tls.port`	*int*	Discovery TLS port
`discoverydata.tls.version`	*string*	Discovery TLS version
`discoverydata.operatingsystems`	*string*	Discovery TLS version
`discoverydata.sources`	*string*	Discovery TLS version
`thirdparty.id`	*string*	Third-Party ID
`thirdparty.connector`	*string*	Third-Party connector
`thirdparty.fingerprint`	*string*	Third-Party fingerprint

Table 8. Table condition combination

Name	Description
contains	Field contains the specified value (case insensitive)
not contains	Criteria does not contain
equals	Criteria exactly equals to
not equals	Criteria exactly not equals
in	Criteria exactly in the following array
not in	Criteria exactly not in the following array
within	Criteria contain in the following array
not within	Criteria contain not in the following array

Table 9. Table operation combination

Name	Description
and	Evaluate a AND logical operation on two criteria or set of criteria
or	Evaluate a OR logical operation on two criteria or set of criteria

Name	Contains	Not contains	Equals	Not equals	In	Not in	Within	Not within
dn	✓	✓	✓	✓	✓	✓	✓	✓
san	✓	✓	✓	✓	✓	✓	✓	✓
serial	✓	✓	✓	✓	✓	✓	✓	✓
issuer	✓	✓	✓	✓	✓	✓	✓	✓
status	✓	✓	✓	✓	✓	✓	✓	✓
module	✓	✓	✓	✓	✓	✓	✓	✓
profile	✓	✓	✓	✓	✓	✓	✓	✓
valid.until	✗	✗	✓	✗	✗	✗	✗	✗
valid.from	✗	✗	✓	✗	✗	✗	✗	✗
keytype	✓	✓	✓	✓	✓	✓	✓	✓
signingalgorithm	✓	✓	✓	✓	✓	✓	✓	✓
owner	✓	✓	✓	✓	✓	✓	✓	✓
holderid	✓	✓	✓	✓	✓	✓	✓	✓
metadata.contact_email	✓	✓	✓	✓	✓	✓	✓	✓
metadata.pki_connector	✓	✓	✓	✓	✓	✓	✓	✓
label.	✓	✓	✓	✓	✓	✓	✓	✓
discoveryinfo.campaign	✓	✓	✓	✓	✓	✓	✓	✓
discoverydata.ip	✓	✓	✓	✓	✓	✓	✓	✓
discoverydata.hostnames	✓	✓	✓	✓	✓	✓	✓	✓
discoverydata.tls.port	✗	✗	✓	✓	✓	✓	✗	✗
discoverydata.tls.version	✓	✓	✓	✓	✓	✓	✓	✓
discoverydata.operatingsystems	✓	✓	✓	✓	✓	✓	✓	✓

Name	Contains	Not contains	Equals	Not equals	In	Not in	Within	Not within
discoverydata.sources	✓	✓	✓	✓	✓	✓	✓	✓
thirdparty.id	✓	✓	✓	✓	✓	✓	✓	✓
thirdparty.connector	✓	✓	✓	✓	✓	✓	✓	✓
thirdparty.fingerprint	✓	✓	✓	✓	✓	✓	✓	✓

Name (part 2)	Is	Before	After	Exists
dn	✗	✗	✗	✗
san	✗	✗	✗	✗
serial	✗	✗	✗	✗
issuer	✗	✗	✗	✗
status	✗	✗	✗	✗
module	✗	✗	✗	✗
profile	✗	✗	✗	✗
valid.until	✗	✓	✓	✗
valid.from	✗	✓	✓	✗
keytype	✗	✗	✗	✗
signingalgorithm	✗	✗	✗	✗
owner	✗	✗	✗	✗
holderid	✗	✗	✗	✗
metadata.contact_email	✗	✗	✗	✓
metadata.pki_connector	✗	✗	✗	✓
label.	✗	✗	✗	✓

`discoveryinfo.campaign`	⊗	⊗	⊗	⊗
`discoverydata.ip`	⊗	⊗	⊗	⊗
`discoverydata.hostnames`	⊗	⊗	⊗	⊗
`discoverydata.tls.port`	⊗	⊗	⊗	⊗
`discoverydata.tls.version`	⊗	⊗	⊗	⊗
`discoverydata.operatingsystems`	⊗	⊗	⊗	⊗
`discoverydata.sources`	⊗	⊗	⊗	⊗
`thirdparty.id`	⊗	⊗	⊗	⊗
`thirdparty.connector`	⊗	⊗	⊗	⊗
`thirdparty.fingerprint`	⊗	⊗	⊗	⊗

4. Click on the search button

3.7. Sharing custom dashboards

This section details how to share a custom dashboard with other users by exporting it to a JSON file and importing it.

Custom dashboards are saved in the profile of the user who created them. To share a dashboard, export it as a JSON file, then send the file to the users who will import it.

How to export a custom dashboard

1. Log in to Horizon Registration Authority Interface.

2. Open the custom dashboard to share from the drawer.

3. Click on the download button, then select **Export as JSON**.

A `horizon-dashboard-<name>.json` file is downloaded. It contains the dashboard charts, but not the filters currently applied.

Built-in dashboards can only be downloaded as PNG images.

How to import a custom dashboard

1. Log in to Horizon Registration Authority Interface.

2. Click on **Import dashboard** in the drawer, below **New dashboard**.

3. Select the JSON file previously exported.

4. Review the dashboard name and description, which are prefilled from the file. The name must not match one of your existing dashboards.

5. Click on the save button.

The imported dashboard is saved in your profile and can then be modified independently of the original one.

Dashboards can only be imported on the Horizon instance they were exported from. The file is rejected if it is not valid JSON, does not follow the dashboard format, exceeds 1 MB or contains an unsupported chart type.

Chapter 4. Knowledge base

4.1. Configure tunnels

For some Horizon features to work, such as pushing certificates to third-party connectors, it will need to have network access to the third-party. Sometimes, this is not possible due to network restrictions. In these cases, you can configure a tunnel service such as Chisel to allow Horizon to communicate with the third-party service through encrypted tunnels over HTTP.



Horizon will eventually support connecting to on-premises resources, so you won't need to set up a separate tunnel service. However, this feature is not yet available in Horizon.

Generate a server key

An SSH key will be required to establish a secure connection between the Chisel server and clients. To generate a key, install Chisel on your local machine and run the following command:

```
$ chisel server --keygen /tmp/chisel_key
```

Then, start the server with the `--key` argument pointing to the generated key:

```
$ chisel server --key /tmp/chisel_key
```

The server will output a fingerprint that you will need to copy. This fingerprint will be used to verify the connection from the Chisel client. The output will look something like this:

```
* server: Fingerprint: Nz5NRzt20kNugkeyHDcxEXQ3+D4Noy8lThsPzkiNJc8=
```

Set up the server

The Chisel server should be set up in the same network location as Horizon. This can be on the same server as Horizon or on a separate server that has access to Horizon. The Chisel server will listen for incoming connections from the Chisel client and forward requests to the third-party service.

RPM

Fetch the Chisel RPM from their releases page and install it on the server that will run the Chisel server:

```
$ yum install chisel_1.10.1_linux_amd64.rpm
```

Copy the generated key to the server, in a path such as `/var/chisel/chisel_key`. Make sure the key is readable by the user that will run the Chisel server.

When starting the server, add the `--key` argument with the path to the key:

```
$ chisel server --key /var/chisel/chisel_key --port 80 --reverse
```



You'll need to ensure that the server will run as a service, such as by using `systemd`. You'll also need to expose the Chisel server with a reverse proxy, such as Nginx, to make it accessible over the network.

Kubernetes

Create a Kubernetes Secret to store the Chisel key. You can do this by running the following command, replacing `<base64-encoded-chisel-key>` with the base64-encoded content of your `chisel_key` file:

```
$ kubectl create secret generic chisel-key --from-file=chisel_key=/tmp/chisel_key
```

You can deploy the Chisel server to a cluster by applying the following manifests:

chisel.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: chisel
  labels:
    app.kubernetes.io/name: chisel
spec:
  replicas: 1
  selector:
    matchLabels:
      app.kubernetes.io/name: chisel
  template:
    metadata:
      labels:
        app.kubernetes.io/name: chisel
    spec:
      containers:
        - name: chisel
          image: jpillora/chisel:1
          args: ["server", "--port", "80", "--reverse"]
          ports:
            - containerPort: 80
            volumeMounts:
                - name: chisel-key
                  mountPath: /var/chisel
      volumes:
        - name: chisel-key
          secret:
            secretName: chisel-key
---
apiVersion: v1
kind: Service
```

```yaml
metadata:
  name: chisel
spec:
  selector:
    app.kubernetes.io/name: chisel
  ports:
    - name: chisel
      protocol: TCP
      port: 80
      targetPort: 80
    - name: server
      protocol: TCP
      port: 9000
      targetPort: 9000
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: chisel
spec:
  ingressClassName: nginx
  rules:
    - host: tunnel.example.org
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: chisel
                port:
                  number: 80
```

```
$ kubectl apply -f chisel.yaml
```

Configure the client

Next to the third-party service, you will need to run the Chisel client. This can be done on the same server as the third-party service or on a separate server that has access to the third-party service.

In the example below, we will assume that the third-party service is running on `example-third-party.local` and that you want to expose it on port `9000` of the Chisel client.

RPM

Fetch the Chisel RPM from their releases page and install it on the server that will run the Chisel client:

```
$ yum install chisel_1.10.1_linux_amd64.rpm
```

When starting the client, add the `--fingerprint` argument with the value you copied from the server logs:

```
$ chisel client tunnel.example.org --fingerprint <server-fingerprint> R:9000:example-third-party.local:80
```



You'll need to ensure that the client will run as a service, such as by using `systemd`.

Kubernetes

You can deploy the Chisel client to a cluster by applying the following manifests. The fingerprint can be provided as a command-line argument:

chisel-client.yaml

```
apiVersion: apps/v1
```

```
kind: Deployment
metadata:
  name: chisel-client
  labels:
    app.kubernetes.io/name: chisel-client
spec:
  replicas: 1
  selector:
    matchLabels:
      app.kubernetes.io/name: chisel-client
  template:
    metadata:
      labels:
        app.kubernetes.io/name: chisel-client
    spec:
      containers:
        - name: chisel-client
          image: jpillora/chisel:1
          args: ["client", "tunnel.example.org", "--fingerprint", "<server-fingerprint>", "R:9000:example-third-
party.local:80"]
          volumeMounts:
            - name: chisel-key
              mountPath: /var/chisel
          ports:
            - containerPort: 9000
      volumes:
        - name: chisel-key
          secret:
            secretName: chisel-key
---
apiVersion: v1
kind: Secret
metadata:
  name: chisel-key
```

```
type: Opaque
data:
  chisel_key: <base64-encoded-chisel-key>
---
```

Apply the manifest to your cluster:

```
$ kubectl apply -f chisel-client.yaml
```

Reference tunnel in Horizon

Now, you'll be able to reference the tunnel in Horizon. For example, if you want to push a certificate to a third-party service that is accessible through the tunnel, you can configure the connector in Horizon to point to the tunnel instead of directly to the third-party service.

For instance, if your server is running on `http://chisel:9000` in your Kubernetes cluster, you can use this address in the connector configuration.