EVERTRUST

EverTrust Stream documentation v1.0

Installation Guide

EVERTRUST

Table of Contents

1. Introduction

1.1. Description

Stream is EverTrust Certificate Authority solution and is powered up by:

- Akka
- BouncyCastle
- MongoDB
- Kamon
- Play! Framework
- Scala
- NGINX
- Vue.js
- Quasar

This document is specific to Stream version **1.0**, and may apply to follow-up minor releases.

1.2. Scope

This document is an installation procedure detailing how to install and bootstrap Stream on a server running **EL 7.x/8.x x64**.

1.3. Out of Scope

This document does not describe how to configure and operate a Stream instance. Please refer to the administration guide for administration related tasks.

2. Pre-requisites

This section describes the system and software pre-requisites to install Stream.

2.1. System pre-requisites

The following elements are considered as system pre-requisites:

- A server running EL [7.x-8.x] x64 (CentOS / RHEL) with the network configured and **SELinux** disabled;
- Base and EPEL CentOS / RHEL [7.x-8.x] x64 repositories activated;
- An access with administrative privileges (root) to the server mentioned above;

2.2. Software pre-requisites

The following elements are considered as software pre-requisites:

- The Stream installation package: '***stream-1.0-1.noarch.rpm***';
- The MongoDB Community Edition package available from the MongoDB web site;
- EPEL repository activated.

NOTE

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

```
yum install epel-release
```

3. Installation Procedure

This section details how to install Stream.

3.1. Installation Procedure

This section details how to install Stream.

3.2. Installing MongoDB Community Edition

NOTE	Mongo DB version 4.2.x to 5.x.x are supported by Stream

Download the last version of the following Mongo DB 5.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 mongosh 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 version 5.0.1, run the following command from the folder containing the RPMs:

```
# yum install mongodb-org*
# yum 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:

```
# mongo
```

NOTE You can disconnect from the shell with ^D

3.3. Installing NGINX

Step 1: Access the server through SSH with an account with administrative privileges;

Step 2: Install the NGINX web server using the following command:

```
# yum install nginx
```

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

```
# systemctl enable nginx
```

Step 4: Stop the NGINX service with the following command:

```
# systemctl stop nginx
```

3.4. Installing Stream

3.4.1. Installation from the EverTrust repository

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

```
[stream]
enabled=1
name=Stream Repository
baseurl=https://repo.evertrust.io/repository/stream-rpm/
gpgcheck=0
username=<username>
password=<password>
```

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

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

```
# yum install stream
```

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

adding

```
exclude=stream
```

at the end of the `/etc/yum.repos.d/stream.repo` file after installing Stream.

3.4.2. Installing from RPM

Upload the file '***stream-1.0-1.noarch.rpm***' through SCP under '***/root***';

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

Install the Stream package with the following command:

```
# yum localinstall /root/stream-1.0-1.noarch.rpm
```

NOTE

Installing the Stream package will install the following dependencies:

- '***dialog***'
- '***java-11-openjdk-headless***'

Please note that these packages may have their own dependencies.

3.5. Configuring the Firewall

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
```

Stream also needs HTTP traffic allowed since it is required to set up the CRLDPs :

```
# firewall-cmd --permanent --add-service=http
```

To make the change effective, you need to restart the firewall service:

```
# systemctl restart firewalld
```

4. Configuration

4.1. Stream Initial Configuration

4.1.1. Generating a Tink keyset

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

- A plaintext keyset (not protected);
- A GCP keyset (protected by a master key in a GCP KMS);
- An AWS keyset (protected by a master key in an AWS KMS).

Stream comes with '**tinkey**' client to manage the generation of a tink keyset.

Here is how to generate a tink keyset:

```
# # Generating a plaintext keyset
# /opt/stream/sbin/tinkey generate-keyset --out=/opt/stream/etc/stream.keyset

# # Generating a GCP protected keyset
# /opt/stream/sbin/tinkey generate-keyset --out=/opt/stream/etc/stream.keyset --master
-key-uri=gcp-kms://<GCP master key path>

# # Generating an AWS protected keyset
# /opt/stream/sbin/tinkey generate-keyset --out=/opt/stream/etc/stream.keyset --master
-key-uri=aws-kms://<AWS master key path>
```

Once the keyset is generated, the following commands need to be run:

```
# chown stream:stream /opt/stream/etc/stream.keyset
```

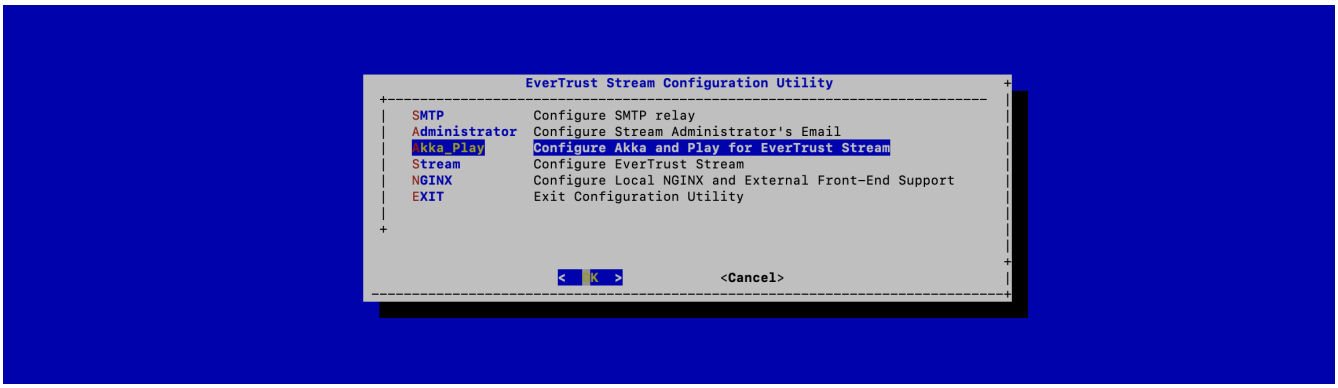
4.1.2. Generating a Play secret

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

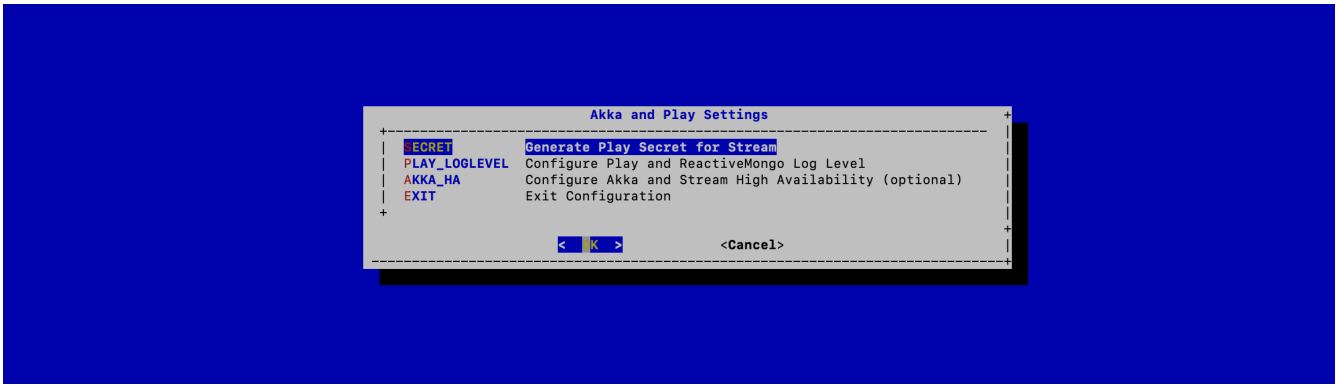
Start the Stream configuration utility by running:

```
# /opt/stream/sbin/stream-config
```

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

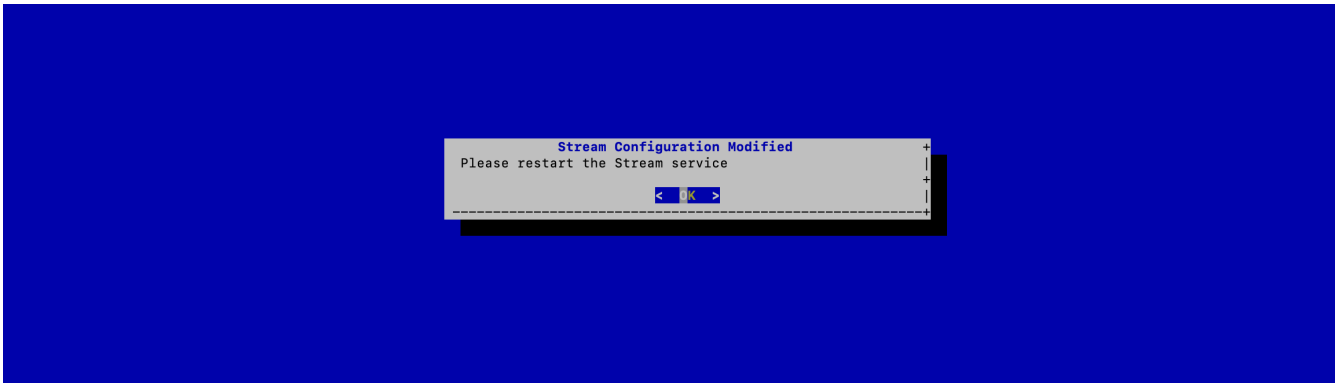


In the Akka_Play menu, select '**SECRET**':



Validate the new Stream Application Secret:

The Stream configuration is updated:



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

```
# systemctl restart stream
```

4.1.3. JVM Configuration

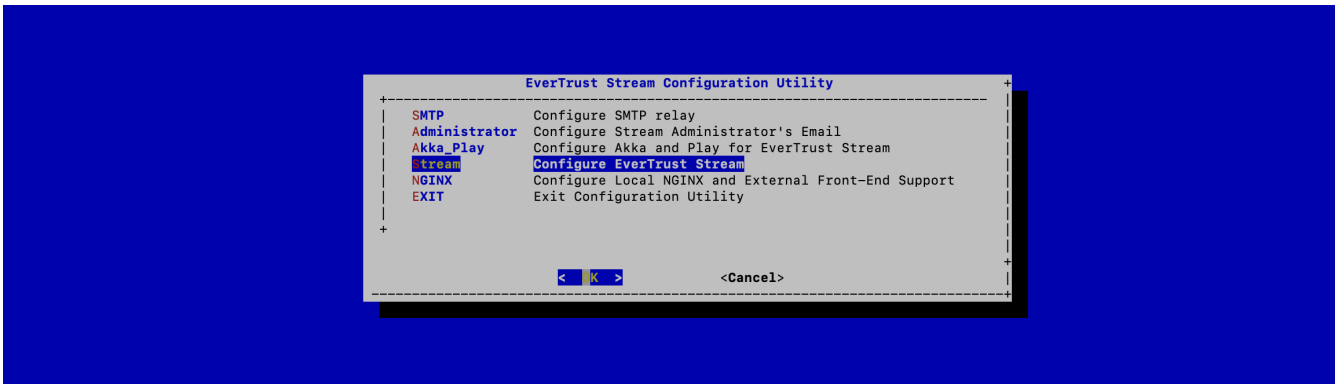
Stream allows you to configure the ***Xms*** (minimum memory allocation pool) and ***Xmx*** (maximum memory allocation pool) parameters of the JVM running Stream using the configuration tool.

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

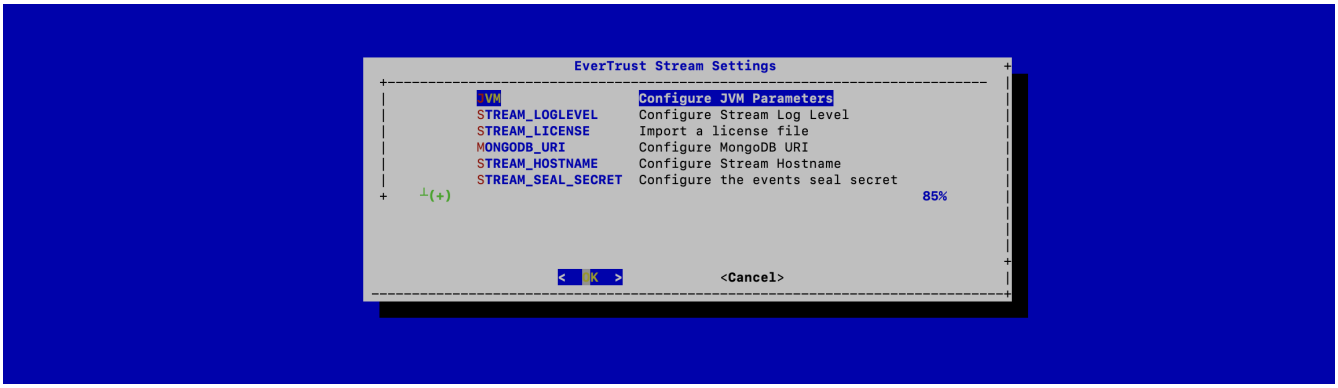
Start the Stream configuration utility by running:

```
# /opt/stream/sbin/stream-config
```

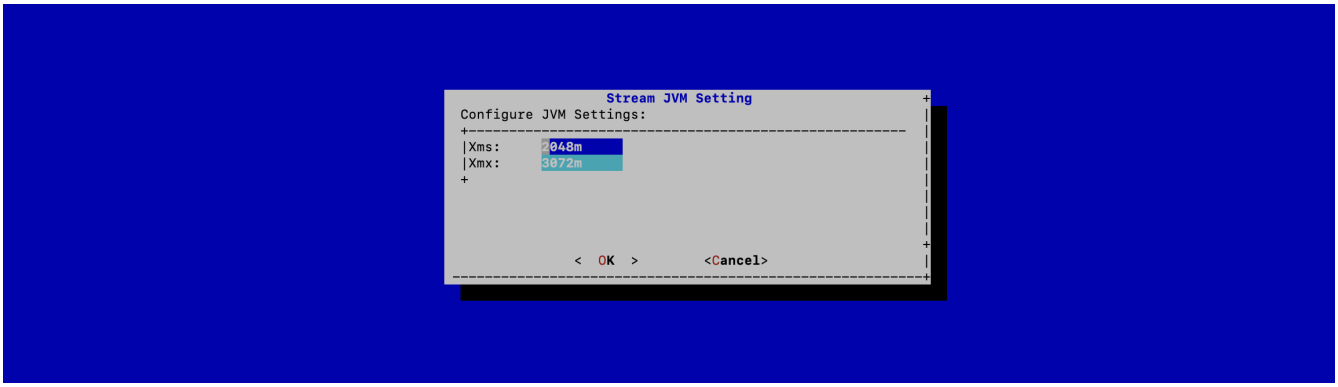
In the configuration menu, select **Stream**:



In the Stream 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 Stream service by running:

```
# systemctl restart stream
```

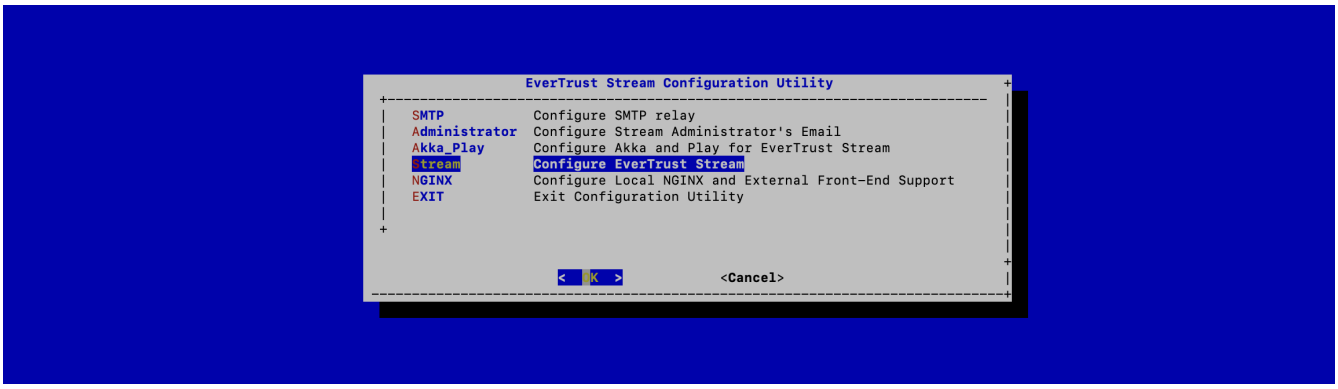
4.1.4. MongoDB URI Configuration

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

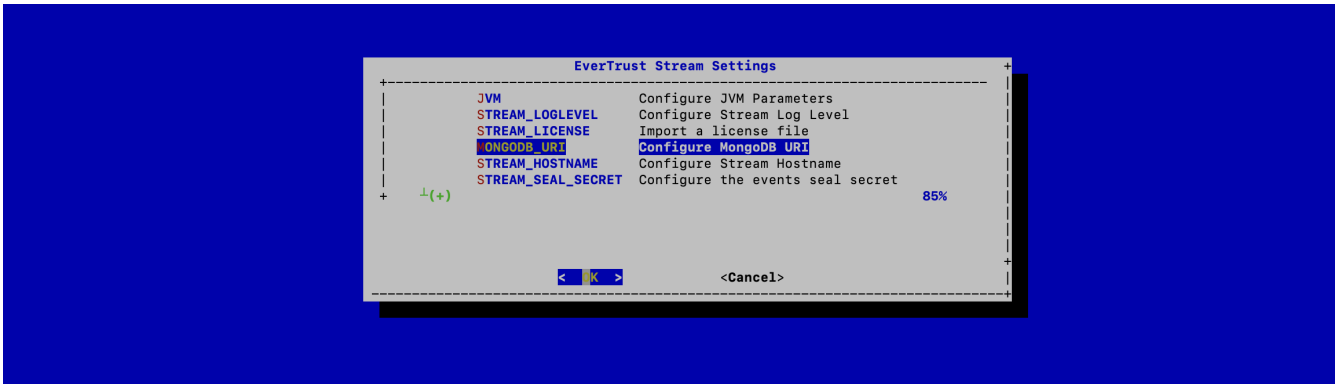
Start the Stream configuration utility by running:

```
# /opt/stream/sbin/stream-config
```

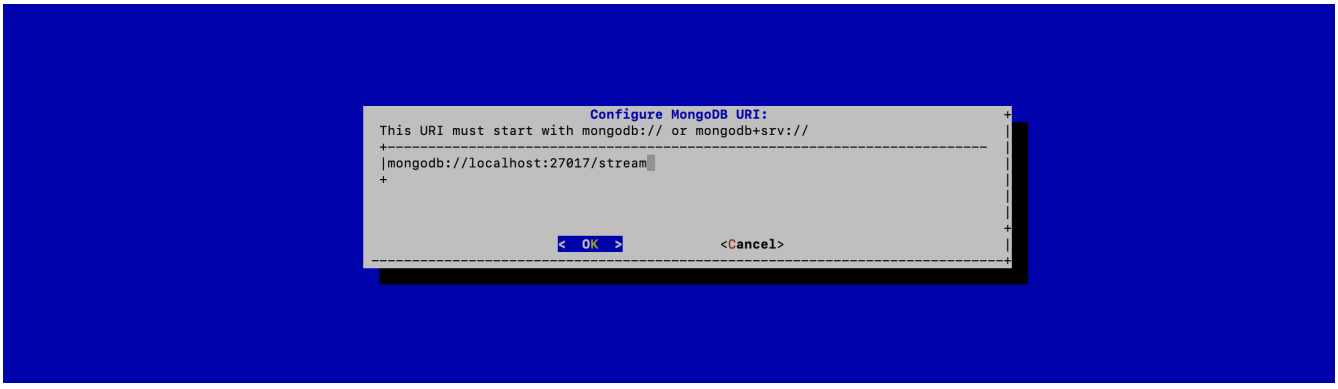
In the main menu, select **Stream**:



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



Specify the MongoDB URI to target your MongoDB instance:



NOTE	Stream 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>/stream

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

The MongoURI is configured.

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

```
# systemctl restart stream
```

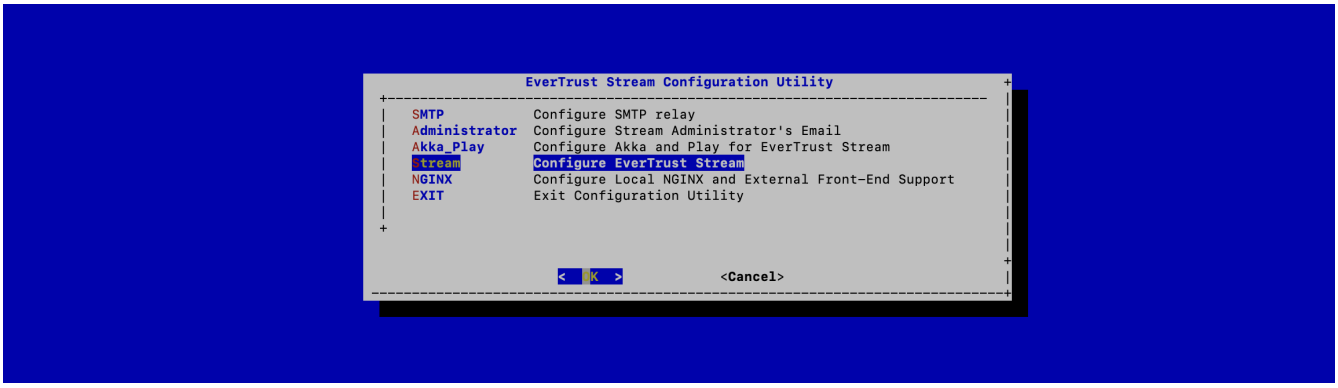
4.1.5. Stream Hostname Configuration

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

Start the Stream configuration utility by running:

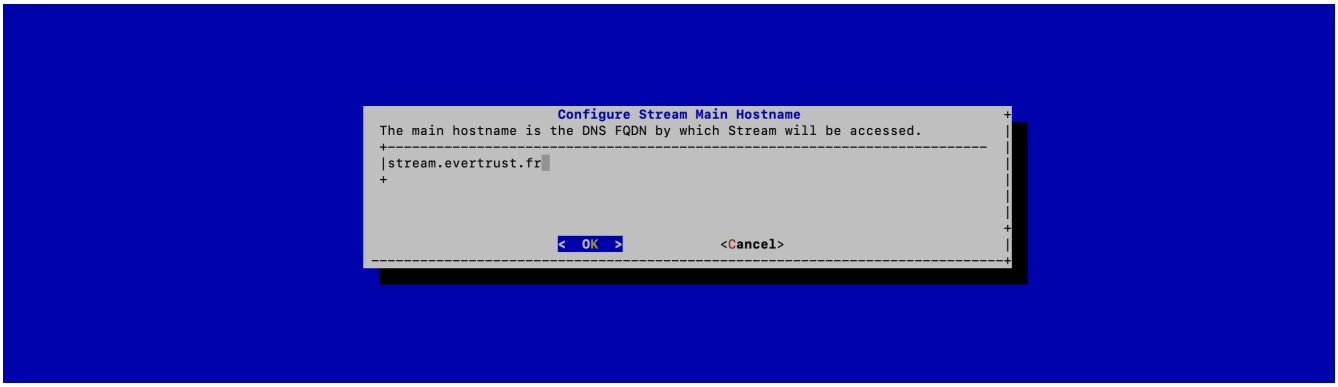
```
# /opt/stream/sbin/stream-config
```

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

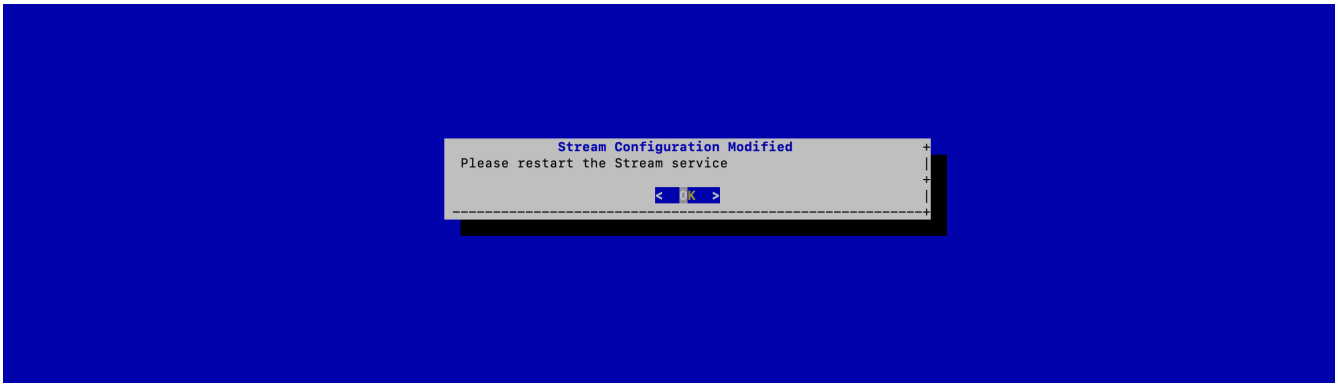


In the Stream configuration menu, Select **STREAM_HOSTNAME**:

Specify the DNS FQDN by which Stream will be accessed:



The Stream Hostname is configured:



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

```
# systemctl restart stream
```

4.1.6. Generating an event seal secret

Stream will generate functional events when using the software.

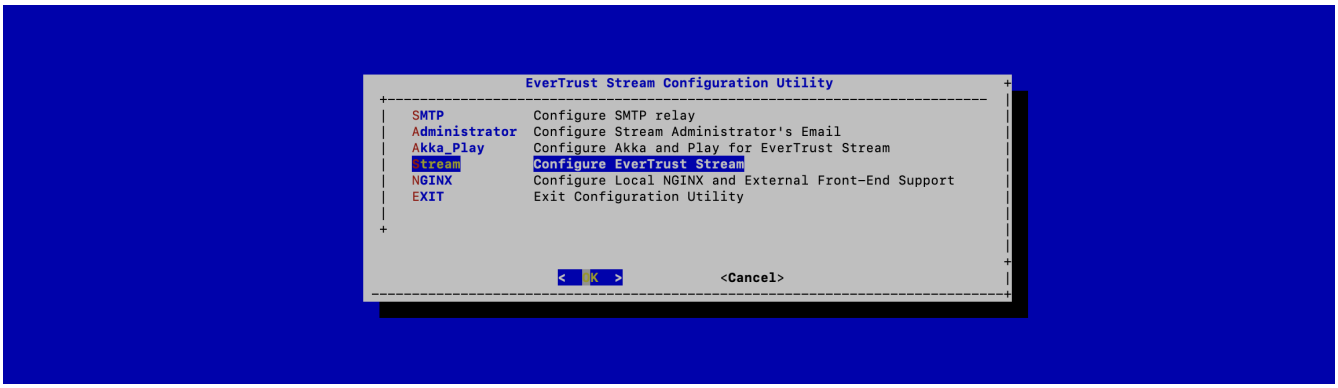
These events are typically signed and chained to ensure their integrity. Therefore, you must specify a sealing secret for this feature to work properly.

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

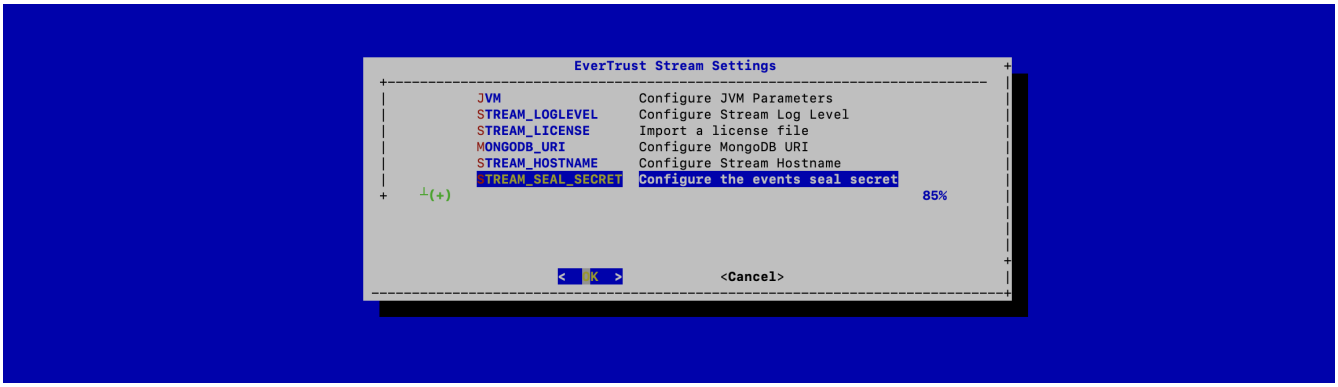
Start the Stream configuration utility by running:

```
# /opt/stream/sbin/stream-config
```

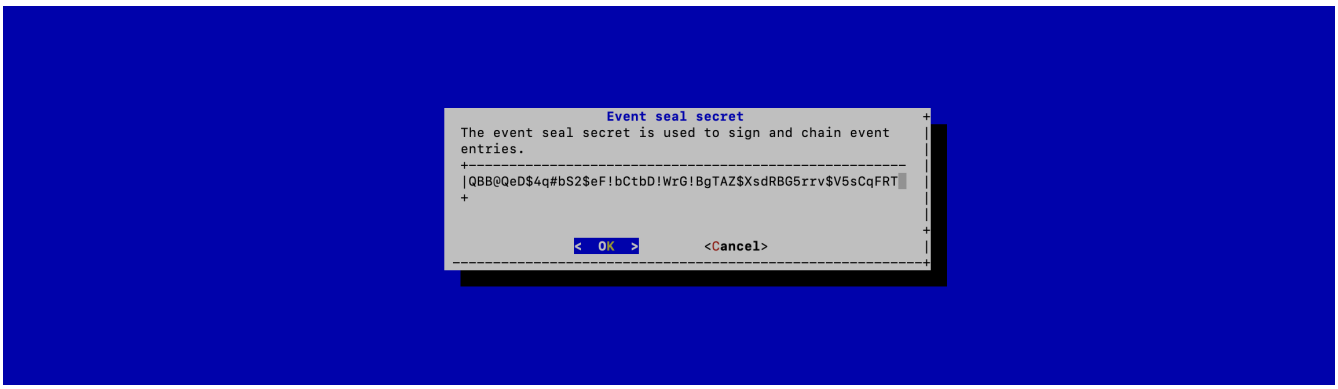
In the main menu, select '**Stream**':



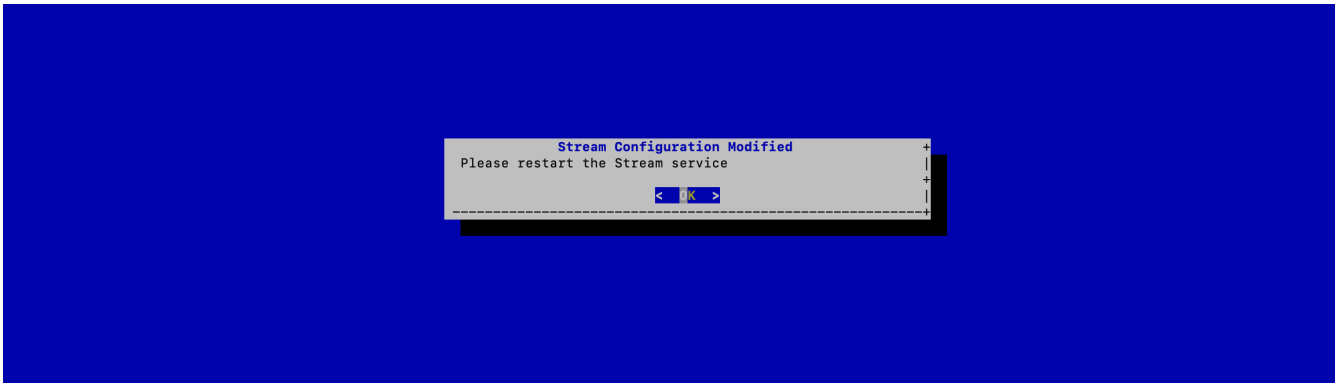
In the Stream menu, select '**STREAM_SEAL_SECRET**':



Validate the new event seal secret:



The even seal secret is now configured:



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

```
# systemctl restart stream
```

4.1.7. Installing the Stream license

NOTE	You should have been provided with a '***stream.lic***' file. This file is a license file and indicates an end of support date.

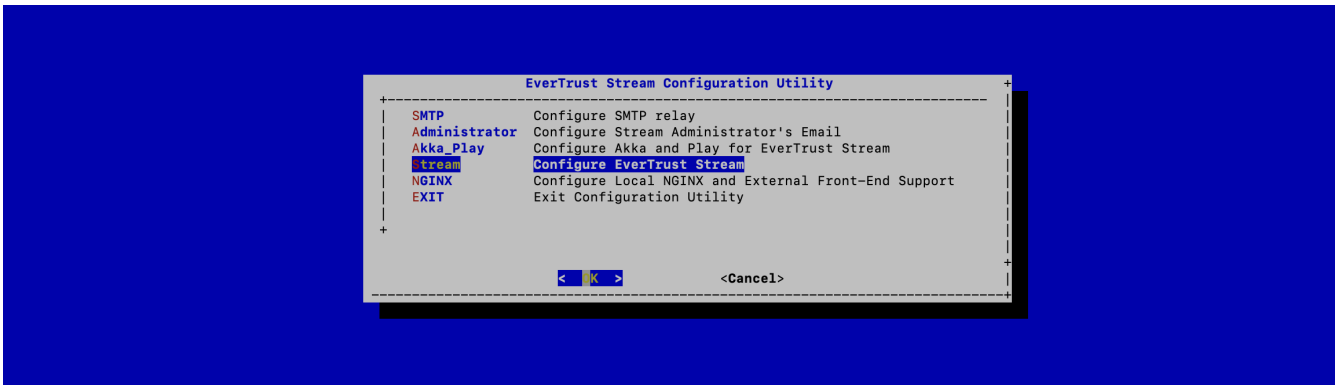
Upload the '***stream.lic***' file through SCP under '***/tmp/stream.lic***';

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

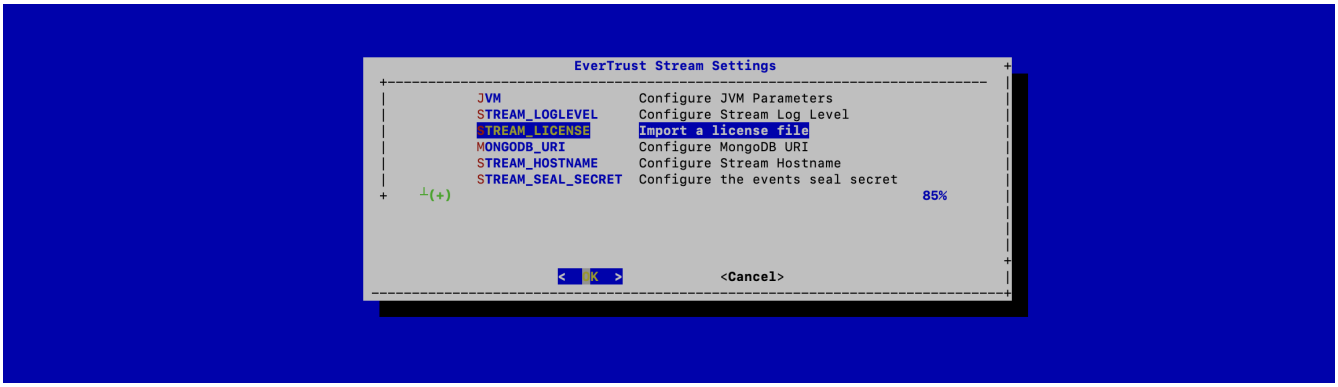
Start the Stream configuration utility by running:

```
# /opt/stream/sbin/stream-config
```

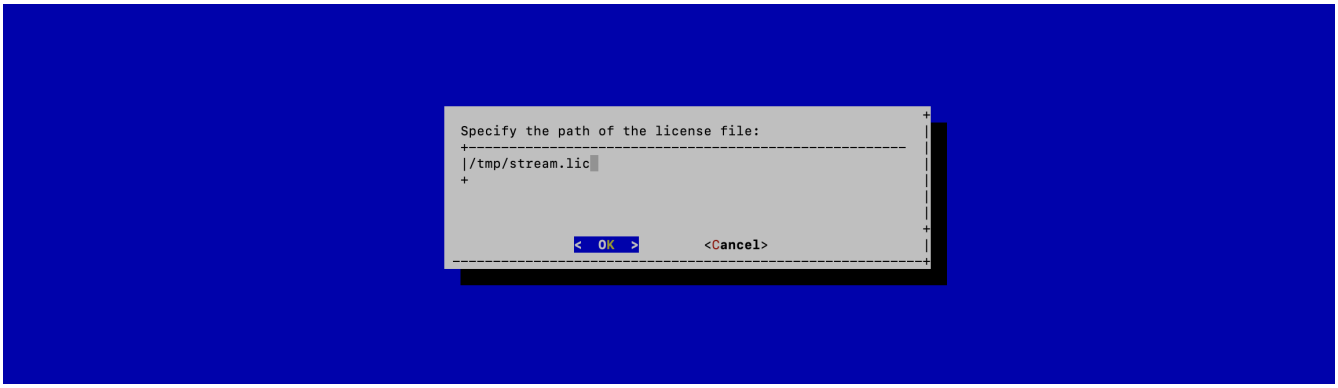
In the main menu, select **Stream**:



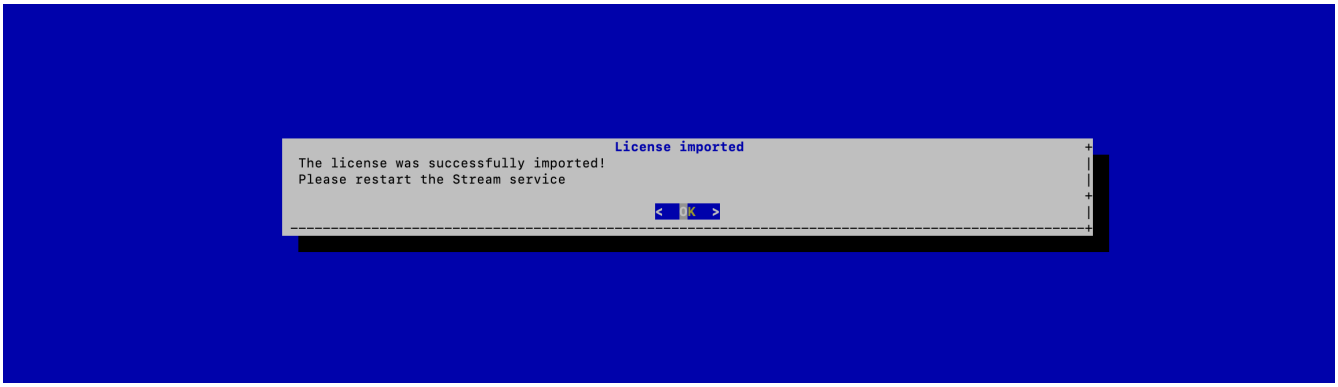
In the Stream configuration menu, Select **STREAM_LICENSE**:



Specify the path '***/tmp/stream.lic***' and validate:



The Stream License is configured:



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

```
# systemctl restart stream
```

4.2. Installing a Server Authentication Certificate

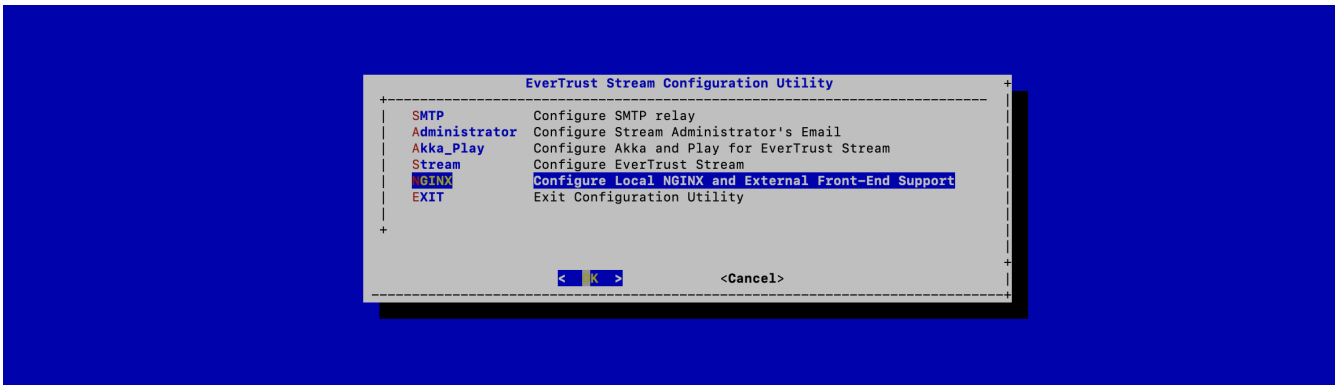
4.2.1. Issuing a Certificate Request (PKCS#10)

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

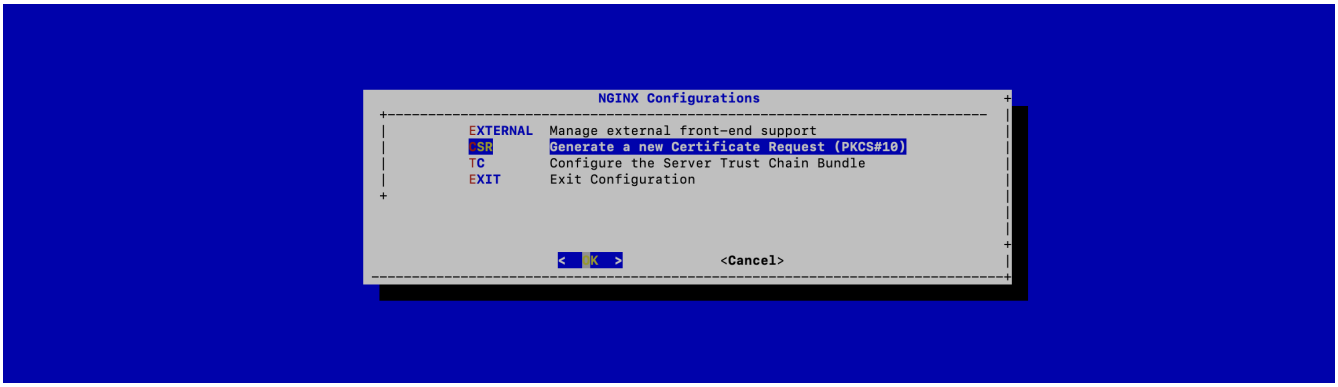
Start the Stream configuration utility by running:

```
# /opt/stream/sbin/stream-config
```

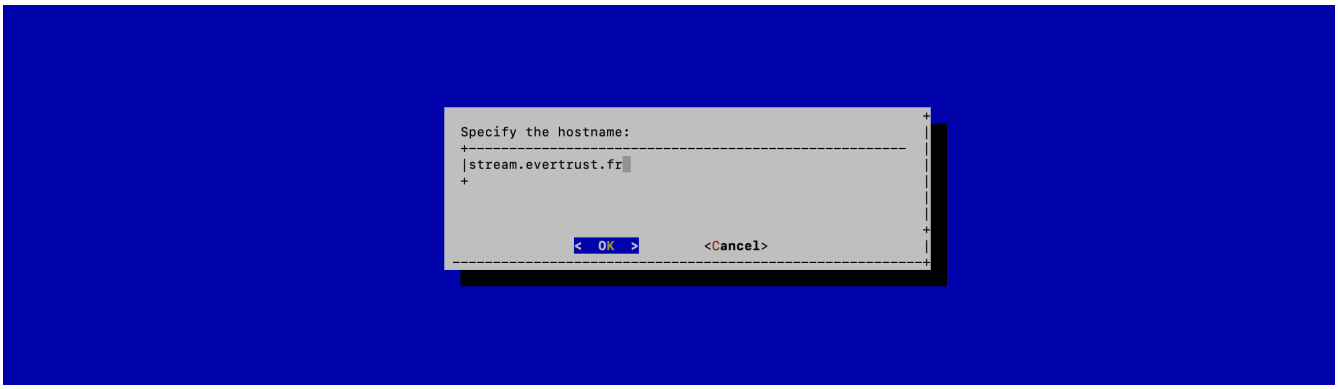
In the main menu, select '**NGINX**':



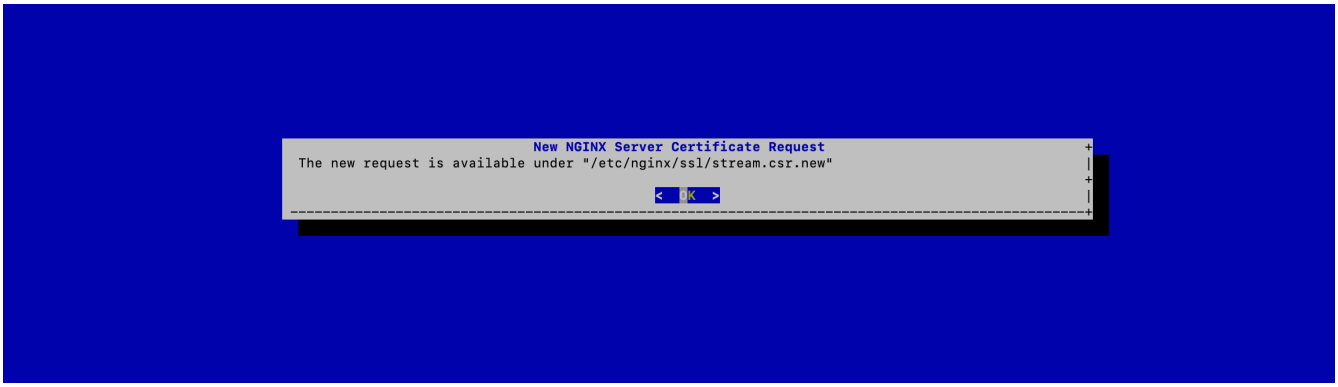
In the NGINX menu, select '**CSR**':



Specify the DNS Name of the Stream server (the same that you used as Stream hostname previously):



The certificate request is generated and available under '***/etc/nginx/ssl/stream.csr.new***':



4.2.2. Signing the server certificate

Signing using an existing PKI

If you desire to sign your Stream web server certificate using an existing PKI, you need to provide your certificate authority with the '***/etc/nginx/ssl/stream.csr.new***' file that was generated at the previous step. You will then need to upload the signed certificate via SCP under '***/tmp/stream.crt***' (PEM and DER formats are supported).

Self-signing the certificate

If you plan on using the Stream PKI to manage the Stream web server certificate, you must self-sign it for configuration purposes, then refer to the administration guide to replace it later on.

To self-sign it using openssl, run the following commands:

```
# cd /etc/nginx/ssl
# openssl x509 -req -days 365 -in stream.csr.new -signkey stream.key.new -sha256 -out
/tmp/stream.crt
```

4.2.3. Installing the Server Certificate

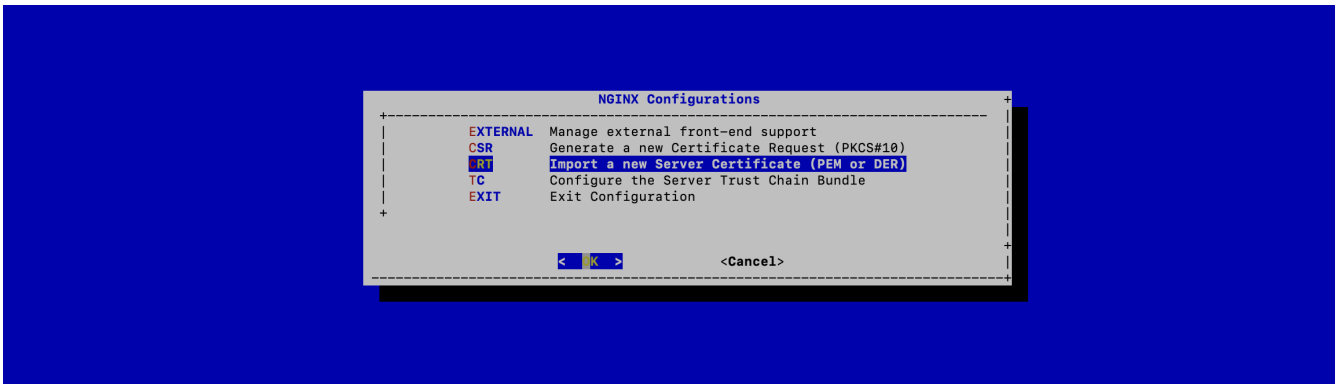
Upload the signed server certificate (in PEM format) on the Stream server under '***/tmp/server.crt***' through SCP;

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

Start the Stream configuration utility by running:

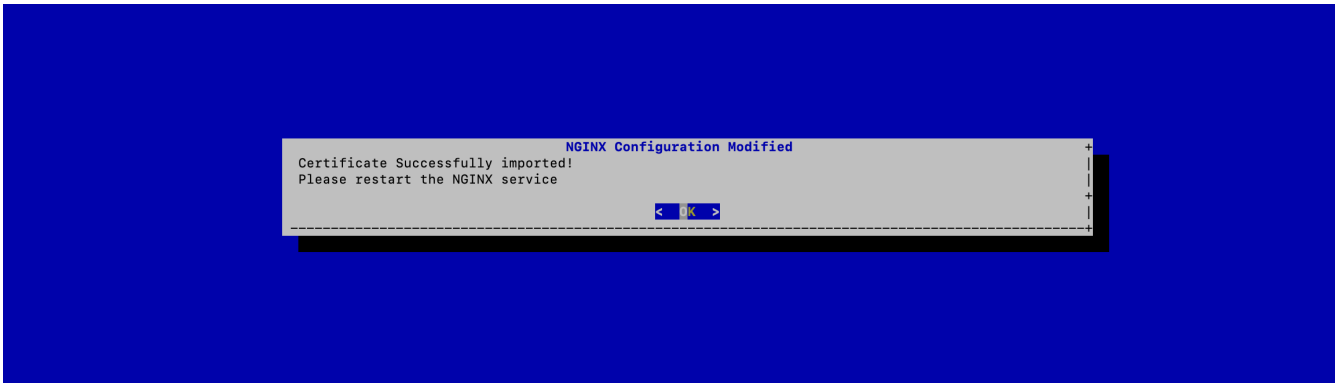
```
# /opt/stream/sbin/stream-config
```

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



Specify the path '***/tmp/stream.crt***' and validate:

The server certificate is successfully installed:



4.2.4. Installing the Server Certificate Trust Chain

NOTE	You must follow this section only if you signed the server certificate with an existing PKI. If you self-signed the server certificate, you do not need to follow this step.

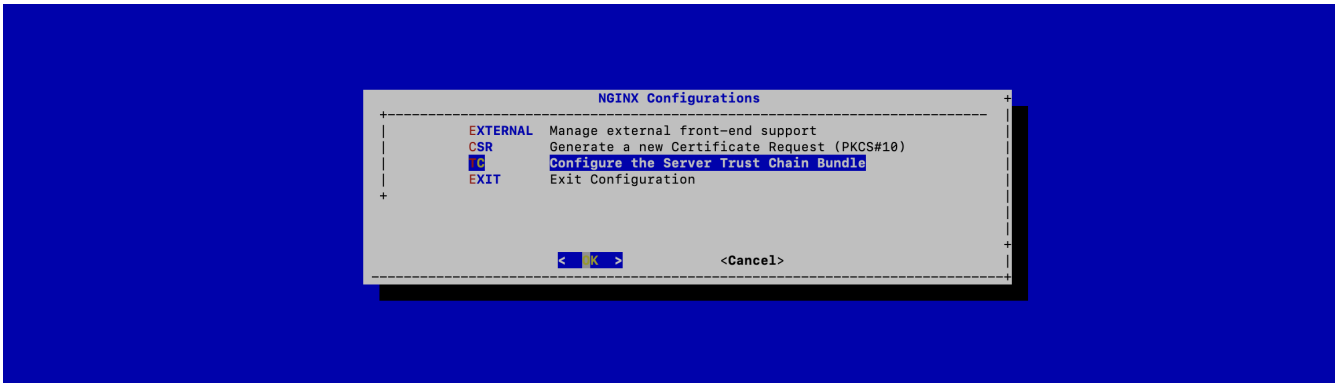
Upload the server certificate trust chain (the concatenation of the Certificate Authority certificates in PEM format) on the Stream server under '***/tmp/server.bundle***' through SCP;

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

Start the Stream configuration utility by running:

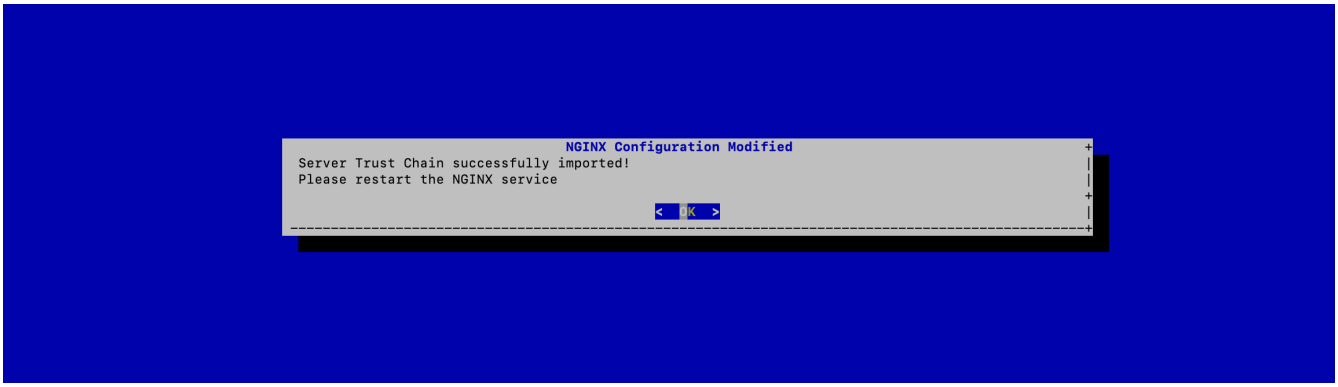
```
# /opt/stream/sbin/stream-config
```

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



Specify the path '***/tmp/server.bundle***' and validate:

The server bundle is successfully installed:


NGINX Configuration Modified
Server Trust Chain successfully imported!
Please restart the NGINX service
< OK >

Verify the NGINX configuration with the following command:

```
# nginx -t
```

Restart the NGINX service with the following command:

```
# systemctl restart nginx
```

4.3. Adding an initial administrator account

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

Run the following command to create the initial administrator:

```
# mongosh
# use stream
#
db.security_accounts.insertOne({"identifier":"administrator","secret":"$6$96ZV/UmX1oMP
UVA3$U5MejjbJ9S3jhqq1TDqhZMwVOcDX5BAWY3DL2nsxUHlpHj0LOfPuswy4nWjkMLify4FvKGKhEfADzljy7
FGc8.","permissions":[{"value":"configuration:*"},{"value":"lifecycle:*"}],"roles":[],
"type":"local"})
```

You can then exit the shell using CTRL + C twice.

NOTE

The administrator credentials are:

- Login: '**administrator**'
- Password: '**stream**'

5. Initial Stream Access

5.1. Starting the Stream services

Step 1: Access the server through SSH with an account with administrative privileges;

Step 2: Start the stream service with the following command:

```
# systemctl start stream
```

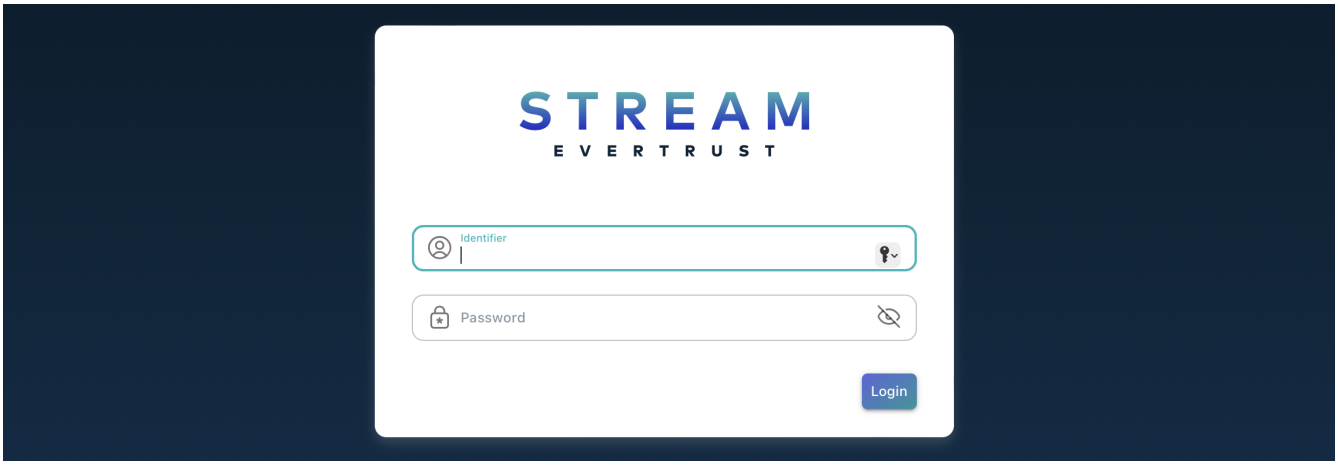
Step 3: Start the nginx service with the following command:

```
# systemctl start nginx
```

5.2. Accessing the Stream Web Interface

Step 1: Launch a web browser;

Step 2: Browse to '***https://[IP or DNS Name of the Stream component]/ui#***':

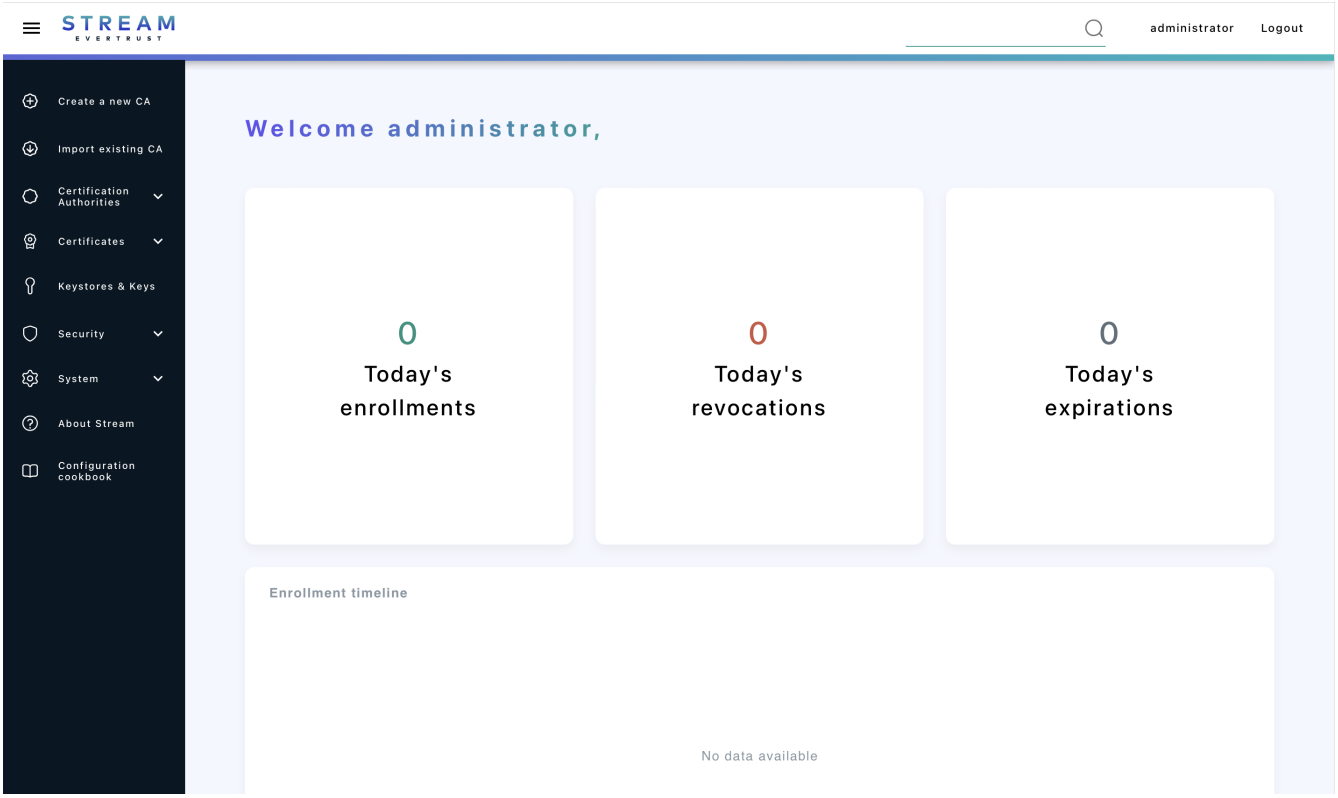


NOTE

The default administration credentials are:

- Login: '**administrator**'
- Password: '**stream**'

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



CAUTION

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

6. Upgrade Procedure

6.1. Upgrade the Stream installation

The first step in the upgrade procedure is to upgrade Horizon component itself.

6.1.1. If Stream was installed using a repository

If you installed Stream using our repository (as described in the installation section), you should:

- Unpin the Stream version by commenting out any line excluding the `stream` package in the `/etc/yum.repos.d/stream.repo` repository file :

```
[stream]
enabled=1
name=Stream Repository
# exclude=stream
```

- Run `yum update stream`

Don't forget to pin the version again by uncommenting the line that was previously commented.

6.1.2. If Stream was installed manually

You must retrieve the latest Stream RPM from the EverTrust repository manually using the user credentials you were provided.

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

Install the Stream package with the following command:

```
# yum localinstall stream-1.0-1.noarch.rpm
```

6.2. Upgrade the database schema

Some Stream versions require that you run migration scripts against your database. Stream comes bundled with an `stream-upgrade` script that handles this migration logic.

Therefore, after each upgrade, you should run `stream-upgrade` to check whether new migrations should be run.

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

Run the following command:

```
# /opt/stream/sbin/stream-upgrade -t <target version>
```

In most cases, `stream-upgrade` can detect the version you're upgrading from by checking the database. if the source version is not automatically detected, you will encounter the following error:

```
*** Unable to infer the source version from your database. Specify it explicitly with
the -s flag. ***
```

You'll have to explicitly tell `stream-upgrade` which version you are upgrading from. To do that, simply set the source version explicitly with the -s flag :

```
# /opt/stream/sbin/stream-upgrade -t <target version> -s <source version>
```

Similarly, `stream-upgrade` will try to use the MongoDB URI that was configured by the Stream configuration utility. If it fails to auto-detect your database URI or you wish to migrate another database, specify the URI explicitly using the `-m` flag:

```
# /opt/stream/sbin/stream-upgrade -t <target version> -m "<mongo uri>"
```

NOTE The upgrade script requires a MongoDB client to connect to your database (either `mongo` or `mongosh`). If no client is installed on the host where Stream is running, consider installing the standalone `mongosh` client or running the upgrade script from another host that has access to the database.

7. Uninstallation Procedure

WARNING	Before uninstalling, please ensure that you have a **proper backup of the Stream component**. Once uninstalled, all Stream data will be **irremediably lost**!

NOTE	Uninstalling Stream consists in uninstalling: • The Stream service; • The MongoDB service; • The NGINX service.

7.1. Uninstalling Stream

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

Uninstall Stream with the following commands:

```
# systemctl stop stream
# yum remove stream
# rm -rf /opt/stream
# rm -rf /var/log/stream
# rm -f /etc/default/stream
```

7.2. Uninstalling NGINX

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

Uninstall NGINX with the following commands:

```
# systemctl stop nginx
# yum remove nginx
# rm -rf /etc/nginx
# rm -rf /var/log/nginx
```

7.3. Uninstalling MongoDB

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

Uninstall MongoDB 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
```

8. Using Stream Doctor

Stream doctor is a tool that performs checks on your Stream installation as well as its dependencies to ensure that everything is configured properly. Note that the tool requires root permissions to run.

8.1. Checks performed

At the moment, Stream doctor checks for :

8.1.1. OS checks

- Checks for installed Stream version, MongoDB version, Java version, Nginx Version and OS version.
  - If the OS is a RedHat distribution, checks for RedHat subscription
  - If Mongo is not installed locally, it notices it as an information log
- Checks for **SELinux**'s configuration (throws a warning if SELinux is enabled)
- Checks for the status of the necessary services: **mongod**, **nginx** and **stream**.
- Checks how long the **stream 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.

8.1.2. 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 stream:stream.
- 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 stream:stream.
- Checks for existence and permissions of the **keyset** file: the permissions are expected to be exactly 600 and the file is supposed to belong to stream:stream.
- Checks for existence and permissions of the Stream directory (default : /opt/stream) : 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 Stream and informs the user if the default ones are used (min = 2048 and max = 3072).
- Retrieves the **Stream DNS hostname** and raises an error if it has not been set.
- 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 **licence file** to retrieve its expiration date.

8.1.3. 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 *stream*; throws an error if no database is specified in the URI).
- Checks for **AKKA 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 8558), 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 25520) and the management port (default is 8558) 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 on each network interface and raises a warning if it is the case (with the interface name).

8.1.4. TLS checks

- Checks for existence and permissions of the **Stream 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 **Stream 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 **Stream server certificate** file and checks if the **Stream hostname** is present in the **SAN DNS names** of the certificate, throws an error if it is not there.

8.2. Log packing option

If the Stream doctor is launched with the *-l option*, it will pack the logs of the last 7 days (in */opt/stream/var/log*) as well as the startup logs (the */var/log/stream/stream.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 **Stream 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 :

```
# stream-doctor -l
```

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

```
# stream-doctor -l 30
```

8.3. Saving the doctor's output

If the Stream 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 commandline 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 *stream-doctor.out* instead of the stdout :

```
# stream-doctor -o stream-doctor.out
```

8.4. Direct fixes

The Stream doctor is able to fix the following issues directly by itself if you use the --fix flag with the script:

- If the application secrets (play secret and event seal secret) have not been changed, the doctor will generate random application secrets and provide them to Stream directly (requires you to manually restart Stream afterwards);
- If firewalld is not allowing HTTP and HTTPS traffic, the doctor will change the firewall settings to allow **both** protocols and then restart the firewall by itself;
- If some permissions for the configuration file, the license file or the keyset file are not what they should be, the doctor will change these permissions (file owner and rwx permissions) to be

what they should.

8.5. Help menu

To display Stream doctor's help menu, use the *-h option*.