



# EverTrust Stream documentation v1.3

## *Administration Guide*

EVERTRUST

# Table of Contents

1. Introduction	1
1.1. Description	1
1.2. Scope	1
1.3. Out of Scope	1
2. Managing Certification Authorities	2
2.1. Importing an External Certification Authority	2
2.2. Importing an existing Managed Certification Authority	2
2.3. Issuing a new Root Certification Authority	3
2.4. Issuing a subordinate Certification Authority	4
2.5. Note on CRLDP and AIA settings	6
3. Managing Certificate Revocation Lists	7
3.1. Configuring Certificate Revocation Lists for an External CA	7
3.2. Configuring Certificate Revocation Lists for a Managed CA	7
3.3. Viewing CRLs	8
3.4. Downloading CRLs	9
3.5. Configuring an external storage for your CRLs	10
4. Managing Keystores & Keys	12
4.1. Keystores in Stream	12
4.2. Software keystore	12
4.3. PKCS#11 HSM	13
4.4. Cloud KMS	14
4.5. Managing keys in Stream	15
5. Managing Certificate Templates & EKUs	17
5.1. Certificate Templates	17
5.2. Extended Key Usage	18
6. Managing Security	19
6.1. Permissions	19
6.2. Accounts	19
6.3. Roles	20
6.4. Credentials	21
6.5. Enforce Certificate Authentication	21
6.6. Events	24
7. Managing Certificate Lifecycle	27
7.1. Enroll	27
7.2. Revoke	27
7.3. Search	27
8. Backup and Restore	29
8.1. Backup Procedure	29
8.2. Restoration Procedure	31

9. Overridable configuration parameters .....	32
9.1. Overriding the parameters .....	32
9.2. Customizing trust chain colors .....	32
9.3. Bootstrapping parameters .....	33
9.4. Timeout parameters .....	33
9.5. HTTP Header parameters .....	34
9.6. Search queries parameters .....	34
9.7. Security parameters .....	34
9.8. Queue parameters .....	35

# 1. Introduction

## 1.1. Description

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

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

This document is specific to Stream version **1.3**, and may apply to follow-up minor releases.

## 1.2. Scope

This document is an administration guide detailing how to configure and operate Stream.

## 1.3. Out of Scope

This document does not describe how to install and bootstrap a Stream instance. Please refer to the installation guide for installation related tasks.

## 2. Managing Certification Authorities

### 2.1. Importing an External Certification Authority

1. Log in to the Stream Administration Interface.

2. Go to **Certification Authorities** > **External CAs** and click on .

3. You need to provide the X509 CA Certificate, either by pasting it directly into the box or by importing the file. **PEM** and **DER** formats are supported. Then click "Next".

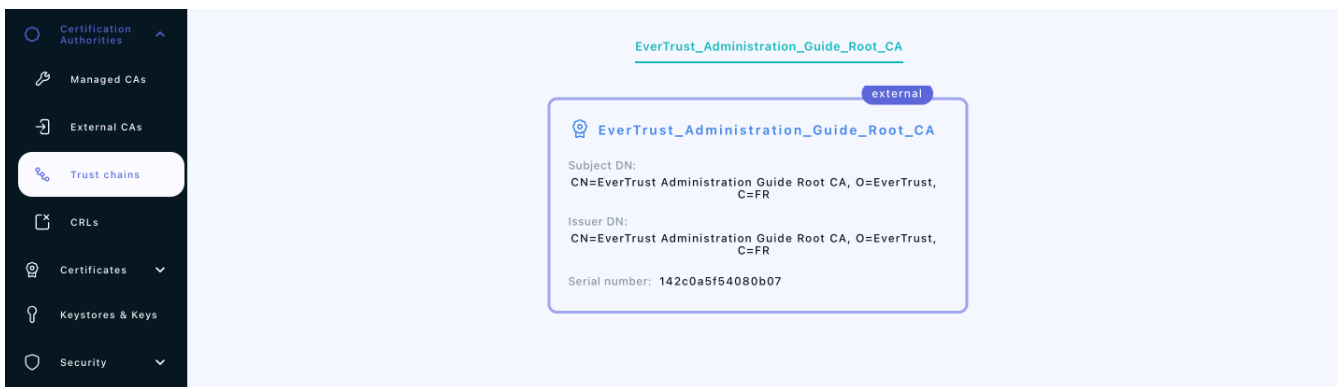
4. In the **Details** tab, check if the details that were parsed from the certificate match those of the CA you wish to import. If it does, click "Next".

5. In the **Configuration** tab, you can

- Add a **CRL**
- Edit the **Refresh period**
- Edit the **Timeout timer**
- Configure a **proxy**
- Toggle whether the external CA should be trusted for **server authentication** or **client authentication**
- Specify the **Outdated Revocation Status Policy**

6. You can finally click the "Import" button in the bottom right corner to import your CA.


If everything was ok, you should see your CA marked as *external* if you go to **Certification Authorities** > **Trust chains**:



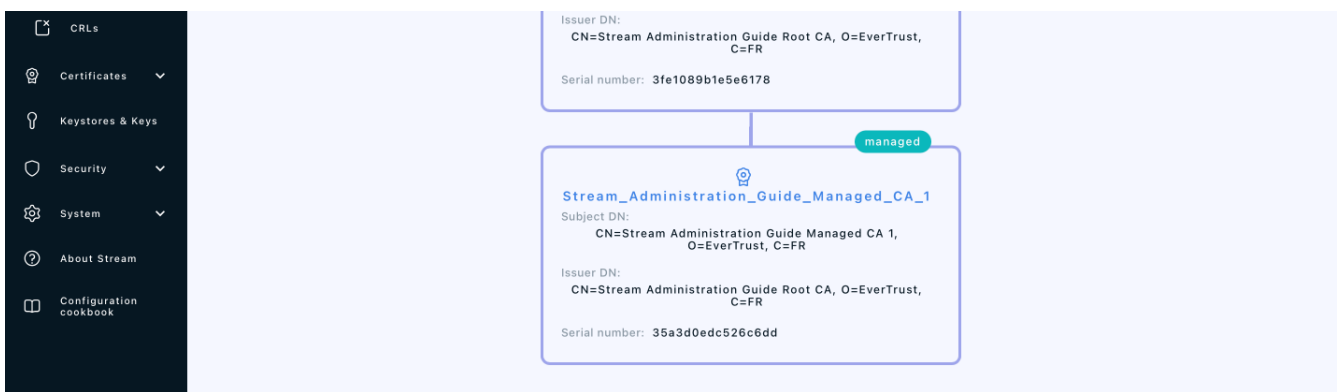
### 2.2. Importing an existing Managed Certification Authority

1. Log in to the Stream Administration Interface.



2. Go to **Import existing CA** from the menu on the left

3. Import your CA certificate file or paste the content of the file in the *Copy/paste the certificate* box. If you decide to paste the file's content, don't forget to click the parse button  on the right before continuing.
4. Scroll down to the bottom of the page and check the certificate's information. If everything is correct, click "Next".
5. Select the Keystore where your CA's key is stored. If you do not have a keystore set up yet, please refer to the *Managing Keystores & Keys* section.
6. Select the key that was used to generate the CA from the selected keystore and click "Next".
7. Upload your CA's CRL file and click "Add".

If everything was ok, you should see your CA marked as *managed* if you go to **Certification Authorities > Trust chains**:

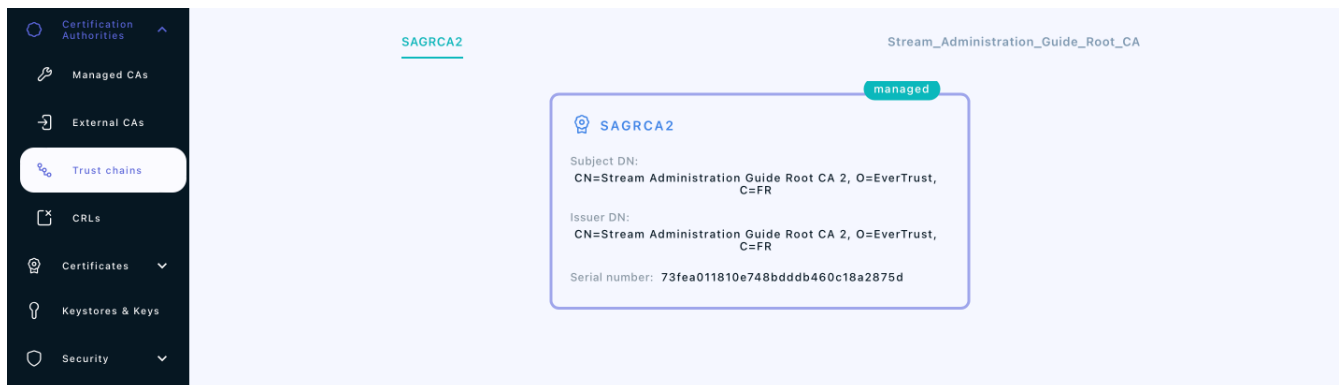


## 2.3. Issuing a new Root Certification Authority

1. Log in to the Stream Administration Interface.
2. Go to **Create a new CA** from the menu on the left.
3. Input your CA's **internal name** and manage the DNs that you want to add (using the  button on the top right corner) or to remove (using the  icon).
4. Select the **Keystore** that contains the key you want to use to generate this CA, then select the key that you want to use. If you do not have a keystore set up yet, please refer to the *Managing Keystores & Keys* section.
5. Select **Selfsigned** as a signing method, and pick the hash algorithm of your choice. Optionally, if you picked a PKCS#11 Keystore and an RSA key, you have the ability to use a PSS signature instead of the classic PKCS#1 one : if you wish to do so, just turn on the toggle. Note that your HSM must support the **CKM\_RSA\_PKCS\_PSS** mechanism.
6. Set the **lifetime** of your CA in days. Optionally, you can set up a **backdate** and a **path length**. Once you are done, click "Add".
7. You can directly configure your CA from this menu, by turning on or off **enrollment**, trusting the



CA for **client authentication** or **server authentication** or **enforcing key unicity**. Once you're satisfied with your settings, click "Add".

If everything was ok, you should see your CA marked as **managed** on a new trust chain if you go to **Certification Authorities > Trust chains**:

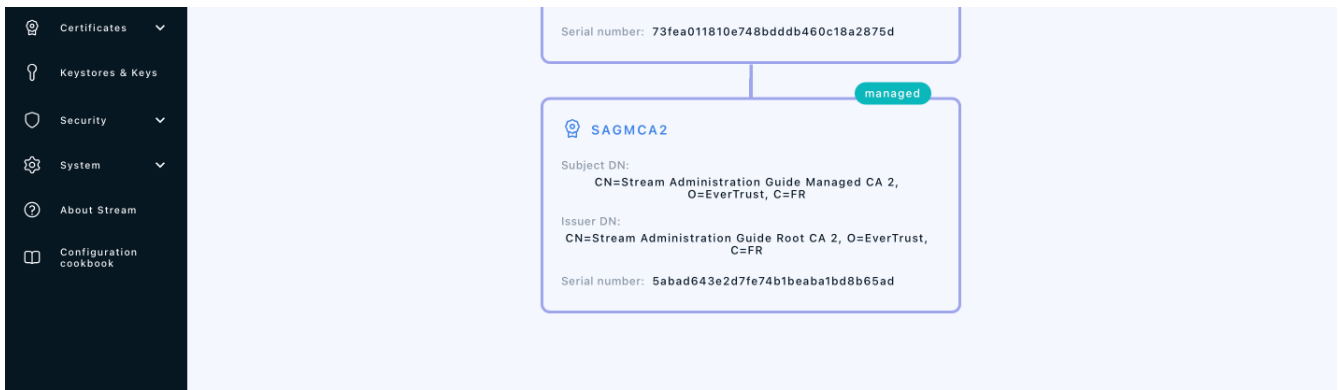


## 2.4. Issuing a subordinate Certification Authority



### 2.4.1. Signed locally

1. Log in to the Stream Administration Interface.
2. Go to **Create a new CA** from the menu on the left.
3. Input your CA's **internal name** and manage the **DNs** that you want to add (using the  button on the top right corner) or to remove (using the  icon).
4. Select the **Keystore** that contains the key you want to use to generate this CA, then select the key that you want to use. If you do not have a keystore set up yet, please refer to the **Managing Keystores & Keys** section.
5. Select **Signed with an internal CA** as the signing method.
6. Select the **Managed CA** you want to sign the certificate with.
7. Set the **lifetime** of your CA in days. Optionally, you can set up a **backdate** and a **path length**.
8. Optionally, you can set up an **OID Policy**, a **CPS Pointer**, add **CRLDPs** and the CA's **AIA**. Once you are finished with the settings, click "Issue CA".
9. You can directly configure your CA from this menu, by turning on or off **enrollment**, trusting the CA for **client authentication** or **server authentication** or **enforcing key unicity**. Additionally, if you issued this CA using an RSA key from a PKCS#11 keystore, you can choose to use the PSS signature algorithm instead of the default PKCS#1 one to sign new certificates. To do so, simply turn on the toggle. Note that your HSM must support the **CKM\_RSA\_PKCS\_PSS** mechanism. Once you're satisfied with your settings, click "Add".

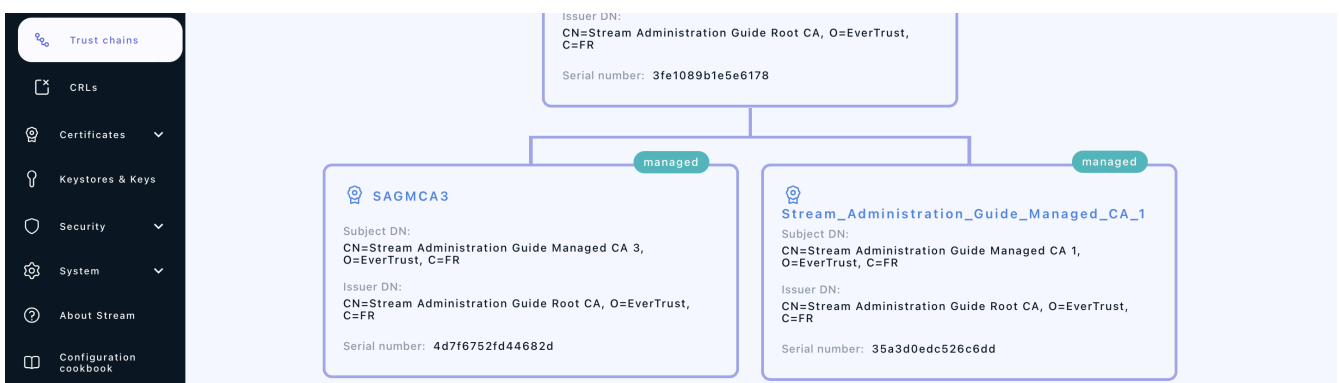
If everything was ok, you should see your CA marked as **managed** on a new trust chain if you go to **Certification Authorities > Trust chains**:



## 2.4.2. Signed externally

1. Log in to the Stream Administration Interface.
2. Go to **Create a new CA** from the menu on the left.
3. Input your CA's **internal name** and manage the **DNs** that you want to add (using the  button on the top right corner) or to remove (using the  icon).
4. Select the **Keystore** that contains the key you want to use to generate this CA, then select the key that you want to use. If you do not have a keystore set up yet, please refer to the **Managing Keystores & Keys** section.
5. Select **Signed with an external CA** as the signing method.
6. Click the link in the **Export** section to download the **CSR** for your CA, then sign it using your external CA and export the signed certificate under PEM or DER format.
7. Upload the signed certificate in the **Import** section.
8. Scroll down to the bottom of the page and check the certificate's information. If everything is correct, click "Next".
9. You can directly configure your CA from this menu, by turning on or off **enrollment**, trusting the CA for **client authentication** or **server authentication** or **enforcing key unicity**. Once you're satisfied with your settings, click "Add".

If everything was ok, your should see your CA marked as **managed** on a new trust chain if you go to **Certification Authorities > Trust chains**:







## 2.5. Note on CRLDP and AIA settings

### NOTE

Regardless of the CA type, the setting "CRLDP" refers to the CRL of the CA you are configuring, and **NOT** the one of the issuing CRL. Same goes for the AIA: you need to specify the certificate of the CA you are configuring, and not the certificate of its issuing CA.

## 3. Managing Certificate Revocation Lists


### 3.1. Configuring Certificate Revocation Lists for an External CA

1. Log in to the Stream Administration Interface ;
2. Go to **Certification Authorities** > **External CAs** and click on  next to the name of the CA you want to import the CRL of ;
3. Select a valid CRL file that has been signed by your CA ;
4. If everything went through correctly, the CRL of that external CA should be available to download from Stream ;
5. Additionally, if you want to push the CRL into S3 Buckets, click  on the external CA ;
  - 5.1 In the **Configuration** tab, select one or several previously created external storage buckets from the drop-down menu ;
  - 5.2 Click the **Save** button at the top.

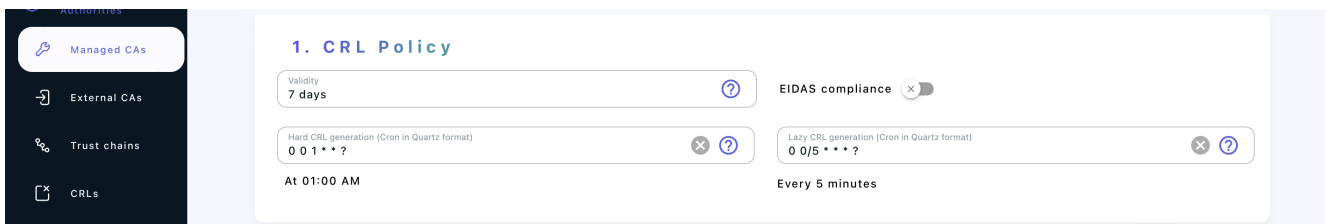
The CRL should now also be pushed in the S3 Bucket(s) whenever you manually import it into Stream. Note that the CRL will still be accessible from the standard Stream CRLDP.

### 3.2. Configuring Certificate Revocation Lists for a Managed CA

To manage the **CRLs** of a managed CA, you first need to set up a **CRL Policy**:


1. Log in to the Stream Administration Interface.
2. Go to **Certification Authorities** > **Managed CAs** and click on  next to the name of the CA you want to edit the CRL policy of.
3. Go under the **CRL** tab.
4. First, you need to define the validity period of your CRL, i.e. the period of time while your CRL is considered valid. The countdown starts at the moment the CRL is generated. If you want your CRLs to be valid for a week, you can type **7 days**.
5. You can then automate the **CRL generation** using either the **Hard CRL generation**, the **Lazy CRL generation** or both of them in combination:
  - The **Hard CRL generation** parameter takes a cron expression in Quartz format and generates the CRL every time that cron expression is valid, without any condition. It is recommended to generate the **CRLs** every day. To generate a new **CRL** every day at 1 A.M., the cron expression is:  
0 0 1 \* \* ?

- The **Lazy CRL generation** parameter takes a cron expression in Quartz format and checks if the CRL needs to be updated, i.e. if a certificate has been revoked, since the last CRL generation. If a certificate has been revoked since the last generation then a new CRL will then be generated, otherwise it will do nothing. It is recommended to have a short time span for the lazy generation so that the CRL always stays up to date. To check for possible CRL updates every 5 minutes, the cron expression is: `0 0/5 * * * ?`



6. Click the **Save** button at the top of the page.

Now your CRL policy has been configured, and you've been redirected to the Managed CAs page.

You can then generate manually the CA's first CRL using the  button next to the CA's name that you just configured. If you configured the **Hard** or the **Lazy** generation, your CRL will then automatically be updated according to the cron quartz expression you specified.

7. Additionally, if you want to push the CRL into S3 Buckets, click  on the managed CA ;

7.1 In the **Configuration** tab, select one or several previously created external storage buckets from the drop-down menu ;

7.2 Click the **Save** button at the top.

The CRL should now also be pushed in the S3 Bucket(s) whenever Stream generates it, based on the policy settings you defined above. Note that the CRL will still be accessible from the standard Stream CRLDP.

### 3.3. Viewing CRLs




1. Log in to the Stream Administration Interface.

2. Go to **Certification Authorities > CRLs**.

3. You can then see information regarding your CAs' CRLs that are going to be detailed below:

CA	Number	Last update	Next update	Next refresh	
SAGMCA2	15	Oct 3, 2022 10:17 AM +02:00	Oct 10, 2022 10:17 AM +02:00	Oct 3, 2022 10:20 AM +02:00	↓ ↻
SAGMCA3	c	Oct 3, 2022 10:17 AM +02:00	Oct 10, 2022 10:17 AM +02:00	Oct 3, 2022 10:20 AM +02:00	↓ ↻
SAGRCA2	1e	Oct 3, 2022 10:16 AM +02:00	Oct 10, 2022 10:16 AM +02:00	Oct 3, 2022 10:20 AM +02:00	↓ ↻

Records per page: 20 |< < 1/1 > >|

- The **CA** column indicates the name of the CA whose CRL is detailed in the line
- The **Number** column indicates the serial number of the CRL. It starts at 1 for the very first CRL generated and is incremented by 1 at each generation. It is displayed in hexadecimal format.
- The **Last update** column indicates the date and time when the current CRL was generated.
- The **Next update** column indicates the date and time when the current CRL will expire. It should be equal to *Last update* + the validity period you set in the **CRL policy** field.
- The **Next refresh** column indicates the date and time when the current CRL will be refreshed. It should be equal to the nearest date matching either cron quartz expression you set in the **CRL policy** field (lazy or hard).
- The **download**  **button** allows you to download your CRL. It also serves as a CRLDP. For more information about CRLDPs in Stream, please refer to next section.
- The **generate**  **button** allows you to manually refresh the CRL and generates a new one.
- The **refresh**  **button** refreshes the information displayed in the tab, in case a generation happened in between. It **does not** refresh the CRLs, only the displayed information.

### 3.4. Downloading CRLs

Stream allows you to download the **CRLs** of the **CAs** it manages. The standard download URL format is `http(s)://[stream_url]/crls/CA_internal_name`. This URL can be accessed by anyone without prior authentication, either through HTTP or HTTPS.

You need to specify the **Internal name** of the CA to download its **CRL** and not its **Common Name (CN)**.

**CRLs** are by default generated and thus downloaded in **DER** format. You can specify `?form=PEM` at the end of the previously given URL to download the CRL in PEM format.

As an example, here are the **CRLDPs** of 2 different CAs that were set up through this guide:


- `https://stream.evertrust.fr/crls/SAGMCA2` will download the CRL for SAGMCA2 through HTTPS in DER format
- `http://stream.evertrust.fr/crls/SAGMCA3?form=PEM` will download the CRL for SAGMCA3 through HTTP in PEM format

## 3.5. Configuring an external storage for your CRLs

### 3.5.1. Creating an S3 External CRL Storage

Stream allows you to push your CRLs into S3 buckets upon generation, but it implies to configure an external storage first. This section also assumes you have already configured credentials for a cloud provider if you want to use a cloud storage solution.

To configure an external S3 CRL storage:


1. Log in to the Stream Administration Interface ;
2. Go to **Certification Authorities** > **External CRL Storage** and click on  ;
3. Fill in the information :
  - Type (*select*): The type of external CRL storage (**mandatory, select Amazon S3**)
  - Name (*string input*) : The name to give to that external storage (**mandatory**)
  - Description (*string input*) : An optional description for that external storage
  - Bucket (*string input*) : The name of the S3 bucket to store CRLs into (**mandatory**)
  - Credential (*select*) : The credential to use to connect to the S3 server (AWS format)
  - Role Arn (*string input*) : The RoleArn to use when connecting to the S3 provider (only applicable for AWS)
  - Region (*string input*) : The cloud region to use if the S3 is in the cloud (AWS, GCP)
  - Proxy (*select*) : The proxy to use to connect to the external storage, if any
  - Endpoint (*string input*) : The S3 endpoint to use (if not using an AWS S3 Bucket)
  - Force path style (*boolean*) : If turned on, forces path style in URL name
4. Once you've filled all the information, click "Add"

The External CRL Storage is now created and can be used in CA details.

### 3.5.2. Creating an LDAP External Storage

Stream allows you to push your CRLs into LDAP directories upon generation, but it requires to create an external LDAP storage in the product first. This section also assumes that you have already configured credentials for the desired LDAP directory.

To configure an external LDAP CRL storage:

1. Log in to the Stream Administration Interface ;
2. Go to **Certification Authorities** > **External CRL Storage** and click on  ;
3. Fill in the information :

- Type (*select*): The type of external CRL storage (**mandatory, select LDAP**)
- Name (*string input*) : The name to give to that external storage (**mandatory**)
- Description (*string input*) : An optional description for that external storage
- Host (*string input*) : The hostname of the LDAP server where the CRL will be pushed into. Don't add "ldap://" or "ldaps://" in the beginning (**mandatory**)
- Port (*int input*) : The port on which the LDAP server is running (default is 389 for LDAP and 636 for LDAPS) (**mandatory**)
- Base DN (*string input*) : The LDAP base DN where to publish the CRL into (**mandatory**)
- Filter (*string input*) : The LDAP search filter to find the resource where to publish the CRL into (**mandatory**). **Example** : (*objectclass=cRLDistributionPoint*)
- CRL Attribute (*string input*) : The resource attribute to publish the CRL into (**mandatory**)
- Follow referrals (*boolean*) : Whether to allow Stream to follow LDAP referral URLs
- Secure (*boolean*) : Whether to use the LDAPS protocol instead of the regular LDAP protocol
- Disable hostname validation (*boolean*) : Whether to allow Stream to connect to the LDAP server in LDAPS even if the server certificate does not have the specified hostname as a DNS SAN (**only if Secure is turned on**)

4. Once you've filled all the information, click "Add"

The External CRL Storage is now created and can be used in CA details.

## 4. Managing Keystores & Keys

### 4.1. Keystores in Stream

In Stream, keys are grouped in key containers called **Keystores**.

Stream handles 3 types of Keystores: Software keystores, PKCS#11 HSMs and Cloud KMS. Note that some restrictions apply regarding the supported key types of the HSMs, namely:

- The software keystore supports:
  - RSA key sizes above 512 bits (the web administration console only offers RSA 2048, RSA 3072, RSA 4096 and RSA 8192);
  - 3 elliptic curves: ECC NIST P-256, ECC NIST P-384 and ECC NIST P-521;
  - 2 Edward curves: ED-448 and ED-25519;
- The PKCS#11 keystore crypto capabilities are entirely reliant on the HSM that is used. Generally, RSA keys are all supported, while elliptic curves are not all supported by every HSM vendor. Currently, Edward curves are also not supported by some HSM vendors;
- Stream can consume the following key types from an AWS KMS instance:
  - RSA 2048, RSA 3072, RSA 4096;
  - ECC NIST P-256, ECC NIST P-384, ECC NIST P-521;
  - The AWS KMS currently does not support Edward Curves;
  - Stream currently does not support the ECC SECG P-256k1;
- Stream can consume the following key types from an AKV instance:
  - RSA 2048, RSA 3072, RSA 4096;
  - ECC NIST P-256, ECC NIST P-384, ECC NIST P-521;
  - Azure Key Vaults (even the Premium ones) currently do not support Edward Curves;
  - Stream currently does not support the ECC SECG P-256k1;
- Stream can consume the following key types from a GCP CKM instance:
  - RSA 2048, RSA 3072, RSA 4096;
  - ECC NIST P-256 and ECC NIST P-384;
  - The GCP CKM currently does not support Edward Curves.

### 4.2. Software keystore

Stream comes installed with a software keystore that can be used to generate RSA and ECDSA keys. To set up a software keystore:

1. Log in to the Stream Administration Interface.

2. Go to **Keystores and keys** and click  .

3. In **Type**, select **Software**. In **Name**, set the name you want to give to your keystore. Optionally, you can add a description to your keystore.

4. Click the **Add** button.

Your keystore should appear in your keystores list with a green circle next to its name.

**NOTE**

When using the software keystore, private keys are at some point stored in memory in **plain text**. That represents a huge security flaw since it would just take a memory dump of the Stream machine to be able to recover the private keys.

**CAUTION**

It is **not recommended** to use the software keystore except for testing or development purposes due to the safety reasons detailed above.

## 4.3. PKCS#11 HSM

Stream supports key management through **PKCS#11 HSMs**.

Stream has been qualified to work with the following **HSMs** but should be working with any **PKCS#11 HSM**:


- Entrust nShield Solo, Entrust nShield Connect, Entrust nShield as a Service
- Atos Proteccio
- Thales Luna (including DPoD), Thales Protect Server
- Utimaco CryptoServer

To set up a PKCS#11 keystore:

1. Log in to the Stream Administration Interface.

2. Go to **Keystores and keys** and click  .

3. In **Type**, select **PKCS#11**. In **Name**, set the name you want to give to your keystore. Optionally, you can add a description to your keystore.

4. Input the **full path** of the **PKCS#11 library** (ending in `.so`) of your HSM, then click the parse  button. If your HSM's library was successfully loaded into Stream, you should be seeing your HSM's information. If you get an HSM error, please check the configuration of your HSM. Click "Next".

5. Select the **HSM slot** that you will be using on your HSM for this keystore and input its **PIN code**;

6. Optionally, you can set a Pool Size to your PKCS#11 interface. If disabled, Stream will open a PKCS#11 session every time it needs to sign a certificate, then close it afterwards. If enabled, Stream will open the number of connections specified in the pool size value and maintain them open as long as Stream is running, to be able to directly sign certificates without having to open a PKCS#11 session. This feature comes particularly handy whenever working with a slow HSM, where opening a session is a pretty long operation that can completely ruin performance.




Once you are done, click "Save". Your keystore should appear in your keystores list with a green circle next to its name.

## 4.4. Cloud KMS


Stream supports 3 types of Cloud KMS: Google Cloud Platform (GCP), AWS Key Management Service (KMS) and Microsoft Azure Key Vault (AKV).

### 4.4.1. Setting up a Google Cloud Key Management (GCP CKM)

1. Log in to the Stream Administration Interface.
2. Go to **Keystores and keys** and click  .
3. In **Type**, select **Google Cloud Platform**. In **Name**, set the name you want to give to your keystore. Optionally, you can add a description to your keystore.
4. Select the GCP credential to use to connect to the Cloud Key Management service. If you do not have your GCP CKM credentials set up in Stream yet, please refer to the *Credentials* part of the *Managing Security* section.
5. Input the **GCP Project name** in **Project**, the **GCP Server location** to use and the **GCP Key Ring** to use. Additionally, you can specify the **proxy** to use as well as the **timeout period**. Once you are done, click "Add".

Your keystore should appear in your keystores list with a green circle next to its name.

### 4.4.2. Setting up an AWS Key Management Service (AWS KMS)


1. Log in to the Stream Administration Interface.
2. Go to **Keystores and keys** and click  .
3. In **Type**, select **AWS**. In **Name**, set the name you want to give to your keystore. Optionally, you can add a description to your keystore.
4. Select the AWS credential to use to connect to the AWS Key Management Service. If you do not have your AWS KMS credentials set up in Stream yet, please refer to the *Credentials* part of the *Managing Security* section.
5. Input the **AWS server's region** in **AWS Region**. Optionally, you can specify which AWS Role ARN that should be impersonated for that KMS. Additionally, you can specify the **proxy** to use as well as the **timeout period**. Once you are done, click "Add".

#### NOTE

To make Stream able to use the keys in the AWS KMS for signature, you need to give it the proper permissions in the AWS console. For more information regarding this topic, please refer to [this link](#), under the "Asymmetric KMS keys for signing and verification".

Your keystore should appear in your keystores list with a green circle next to its name.

### 4.4.3. Microsoft Azure Key Vault (AKV)


1. Log in to the Stream Administration Interface.
2. Go to **Keystores and keys** and click  .
3. In **Type**, select **Azure Key Vault**. In **Name**, set the name you want to give to your keystore. Optionally, you can add a description to your keystore.
4. Select the AKV credential to use to connect to the Microsoft Azure Key Vault. If you do not have your Microsoft AKV credentials set up in Stream yet, please refer to the *Credentials* part of the *Managing Security* section.
5. Specify your Azure vault URL in the **Vault URL** box and the Azure tenant in the **Azure Tenant** box. Additionally, you can specify the **proxy** to use as well as the **timeout period**. Once you are done, click "Add".

Your keystore should appear in your keystores list with a green circle next to its name.

## 4.5. Managing keys in Stream


Regardless of the type of keystores you set up, you can manage the keys through Stream the same way

### 4.5.1. Adding a key into a keystore

1. Log in to the Stream Administration Interface.
2. Go to **Keystores and keys** and click  on the keystore you want to add the key into.
3. Set the name of the key as well as the key type (RSA, ECDSA or EDDSA) and the key size (for RSA)/key parameter (for ECDSA/EDDSA).
4. For the **Cloud KMSs**, you can set the key to be **Hardware protected** through the dedicated toggle. For the **PKCS#11 HSM**, you can set the key to be **exportable** through the dedicated toggle.
5. Once you set up the key parameters as you want them, click "Add".

The page should refresh and show you the list of keys for the keystore you pushed the key into, where you should see the key you just added.

### 4.5.2. Viewing the keys of a keystore

1. Log in to the Stream Administration Interface.
2. Go to **Keystores and keys** and click  on the keystore you want to view.

3. You should see the list of keys on your keystore.

You can then see information about the keys in the keystore:

- The **name** column where you can see the name of the key ;
- The **type** column where you can see the type of algorithm that was used to generate the key. Both RSA and ECDSA are part of the *suiteb* type algorithms ;
- The **key type** column where you can see the algorithm that was used to generate the key as well as the key size/parameter ;
- The **exportable** column indicates if the key is exportable or not.

### 4.5.3. Deleting a key from a keystore

1. Log in to the Stream Administration Interface.

2. Go to **Keystores and keys** and click  on the keystore you want to delete the key from.

3. Click the  icon on the key that you want to delete and click "Confirm" on the prompt.

**NOTE**

You cannot delete a key from a keystore if this key is currently used by a CA in Stream. You must first delete the CA that references it and then go over the deleting procedure.

# 5. Managing Certificate Templates & EKUs

## 5.1. Certificate Templates

Stream uses the notion of **Certificate Templates** to add additional verifications when enrolling a certificate.

To define a new certificate template:

1. Log in to the Stream Administration Interface.

2. Go to **Certificates > Templates** and click  .

3. In the **General** tab, you can set the template's name, the **path length** it will tolerate, turn the template on or off and check for proof of possession when enrolling with a CSR. In the **Duration** part of the tab, you can edit the lifetime of the certificates that will enroll on this template, as well as backdate them should you need to. In the **Private Key policy** part of the tab, you can choose whether to enforce a usage period for the private key that is detached from the validity of the certificate. Should it be defined, this period must be within the validity period of the certificate. This field is optional in the RFC 5280 but mandatory in the ICAO MRTD 9303 norm (section 7.1.1) and should only be used for signature certificates.




4. In the **KU & EKU** tab, you can set the **Key Usages** and **Extended Key Usages** of the certificates that will enroll on this template. You can also use your own **EKUs** here. If you want to set up your own **EKUs**, please refer to the *Extended Key Usages* part of this section.

5. In the **Extensions** tab, you can edit the **CRLDPs**, **AIA**, **Authority Information Access**, **Policy**, **Qualified Certificate Statement** of the certificates that will enroll on this template. If you want to, the certificates could use the information of the **CA** they will enroll on, otherwise, you can set specific values in the template. These values will then override those retrieved from the CA.

- If you want to issue Qualified Certificates:
  - ETSI QC Compliance Statement declares that the certificates is a Qualified Certificate.
  - ETSI QC SSCD Statement declares that the private key related to the certified public key resides in a Secure Signature Creation Device.
  - ETSI Retention Period Statement indicates the duration of the retention period of material information.
  - ETSI QC Type Statement indicates which type of document can be signed by the certificate (possible values are: ESEAL, ESIGN, WEB, NONE).
  - ETSI Transaction Limit Statement indicates the limits of the transactions, you must fill every field if enabled.
  - ETSI QC PDS Statement is the PKI Disclosure Statement URI for a specified language.
  - ETSI QC Legislation Statement is an array of country codes.

6. In the **Data Fields** tab, you can enforce your **DNs**, **SANs** and **Extensions** to match certain criteria

that can be defined in this section. By default, everything is accepted, meaning that any type and amount of **DNs**, **SANs** and **Extensions** can be used in the certificates and it would successfully enroll on the template.

- If you want to enforce a **Subject DN** policy, then click  in **Subject DN composition**, then select the DN element that you want to put a policy on. You can set this element to be mandatory or not, to use a default value for that element that can be editable or not, you can also add a whitelist of elements that are accepted values for this DN, or you can instead use a regex to match the DN values that are accepted for this element.
- If you want to enforce a **Subject Alternate Names** policy, you can either click **None** to forbid the use of **SANs** in certificates or you can click **Some** to configure the policy. If you clicked **Some**, click  and select the **SAN element** that you want to enforce a policy upon. You can then input a minimum and maximum number of this **SAN element** to be present in the certificate that will enroll: as an example, if you want to make the use of at least one **DNS SAN** mandatory, use 1 as a minimum number. Finally, you can enforce your **SANs** to match a regex to be considered valid on a certificate.
- If you want to enforce an **Extension** policy, you can either click **None** to forbid the use of **Extensions** in certificates or you can click **Some** to configure the policy. If you clicked **Some**, click  and select the **Extension** that you want to enforce a policy upon. You can then set it mandatory or not, and if supported, give it a default value that can be edited or not.

7. Once you've configured your template, you can click **Save** at the top of the page.


**NOTE**

As mentioned previously, if you want your certificates to inherit the **CRLDP**, the **AIA** and the **Policy** from the CA, you must toggle on the **Get from CA** switches and not specify any policy, CRLDP or AIA in the template.

## 5.2. Extended Key Usage

Stream allows you to create and manage your own EKUs as long as you have an OID for it.

To create a custom EKU:

1. Log in to the Stream Administration Interface.
2. Go to **Certificates** > **EKU** then go at the bottom of the page and click  .
3. Specify the name you want to give to your custom EKU as well as its OID in the menu, then click "Add".

The EKU should show in the list with the custom switch turned on, as opposed to the standard EKUs that have the custom switch turned off.

# 6. Managing Security

## 6.1. Permissions

Stream allows you to manage 2 types of permissions: configuration and lifecycle. Stream uses wildcard permissions which means you can configure the permissions very thoroughly.

For configuration permissions, you can specify:

- the **Section** (ex: Security)
- the concerned **Module** (only for **Security** and **System**)
- the type of permission: **Audit** (read-only) or **Manage** (read-write, equivalent to *All*).

For lifecycle permissions, you can specify the concerned **CA** and the concerned **Template** then the type of permission: **Enroll**, **Revoke**, **Search** or **All** of these.


### CAUTION

The following permissions allow the affected user (or users for roles) to manipulate permissions:

- All permissions of all type
- Security > All > Manage
- Security > Accounts > Manage
- Security > Roles > Manage

## 6.2. Accounts

### 6.2.1. Adding an account






1. Log in to the Stream Administration Interface.
2. Go to **Security > Accounts** and click  ;
3. Select the type of account to create: a **local account** connects to Stream using a **login and a password**, while an **X509 account** uses a **user authentication certificate** to connect to Stream;
4. Set the **identifier** (login) of the account. Optionally, you can add a **description** to the account.
5. Select the **roles** you want to assign to this account (not mandatory). If you do not have a role set up for the account, please refer to the **Roles** section of this guide.
6. Specify the **straight configuration and lifecycle permissions** you want to give the account (not mandatory).

Once everything is set up, you can click **Save**.

### NOTE

It is highly recommended to configure and use Roles rather than using straight permissions.




## 6.2.2. Managing accounts

1. Log in to the Stream Administration Interface.
2. Go to **Security** > **Accounts**. From there, you can see all Stream accounts and their associated information.
  - The **identifier** column represents the account's internal name and serves as login for a local account;
  - The **type** column indicates whether the account is **local** or **X509**;
  - The **description** column shows the account's description;
  - The **permissions** column shows the **straight permissions** that are built into the account. If the account has any **configuration** permission, it will display  and if it has any lifecycle permission it will display .
  - The **roles** column shows the **roles** that have been assigned to that account;
  - You can **reset the password** of local accounts using the  button;
  - You can **delete an account** using the  button;
  - You can **edit an account's** information using the  button.

## 6.3. Roles

Roles are a way to factor permissions making it easier to configure accounts and track permissions.

### 6.3.1. Creating a new role






1. Log in to the Stream Administration Interface.
2. Go to **Security** > **Roles** and click  ;
3. Set the **name** of the role you want to create. Optionally, you can add a **description** to the role.
4. Add the **configuration permissions** you want the members of this role to have using the  from **Configuration permissions**. If the role is supposed to have no configuration permission, leave this section empty.
5. Add the **lifecycle permissions** you want the members of this role to have using the  from **Lifecycle permissions**. If the role is supposed to have no lifecycle permission, leave this section empty.

Once everything is set up, you can click **Save**.

### 6.3.2. Managing roles

1. Log in to the Stream Administration Interface.

2. Go to **Security > Accounts**. From there, you can see all Stream roles and their associated information.

- The **name** column displays the role's name;
- The **description** column displays the role's description;
- The **permissions** column shows the **straight permissions** that are set up for the role. If the account has any **configuration** permission, it will display  and if it has any lifecycle permission it will display .
- You can **view all the members of a role** using the  button;
- You can **delete a role** using the  button;
- You can **edit a role's** information using the  button.

## 6.4. Credentials

To use external services in Stream (Cloud KMS, LDAP Directory), you will need to create a credential for the corresponding service:

1. Log in to the Stream Administration Interface.

2. Go to **Security > Credentials** and click  ;

3. In **Target**, select the service you want to create the credential for (Amazon AWS, Microsoft AKV, Google GCP or LDAP);

4. In **Name**, specify the internal name of the credential you are creating (example: aws-account). Optionally, you can add a description for the credential;

5.1 To connect to **AWS** and **AKV**, input the login and password of your **AWS** or **AKV** account;

5.2 To connect to **GCP**, paste the **JSON token** of your **GCP** account in the **JSON Token** box;

5.3 To connect to an **LDAP Directory**, specify the login DN to use as well as its password.

Once everything is set up, you can click "Add".

## 6.5. Enforce Certificate Authentication

It is possible to enable `x509_enforcing` parameter in order to authorize only certificate authentication.

**WARNING** | This means local accounts will no longer be able to connect on Stream.

**CAUTION** | When logging in using an X509 certificate, there is no logout option, meaning that the only way to log out is to change the presented certificate in your browser, or to switch to private browsing.



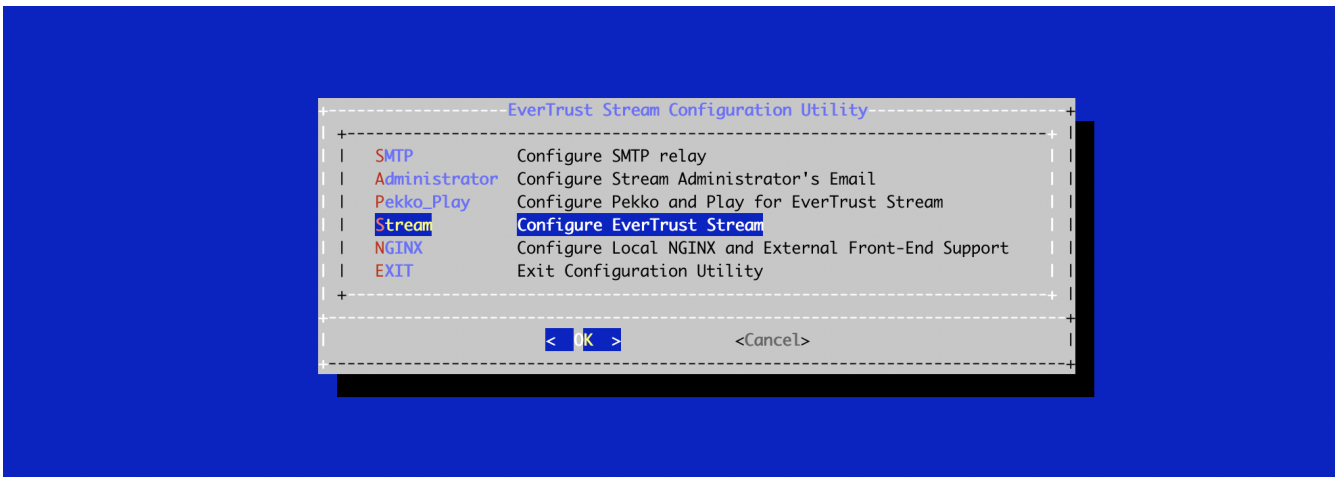
## 6.5.1. Using Stream configuration utility

Access the server with an account with administrative privileges;

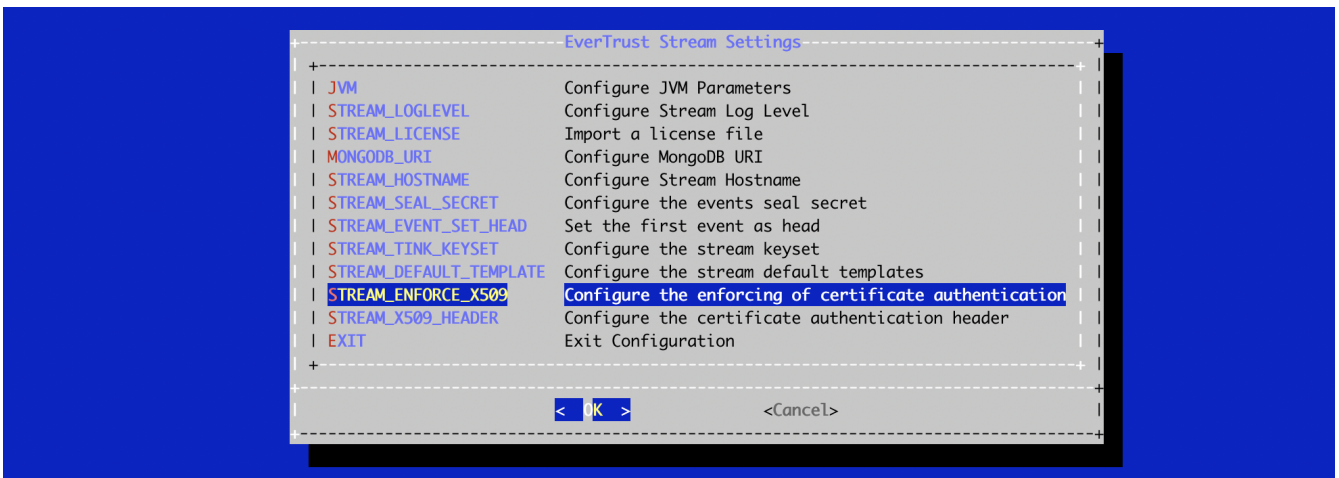
Start the Stream configuration utility by running:

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

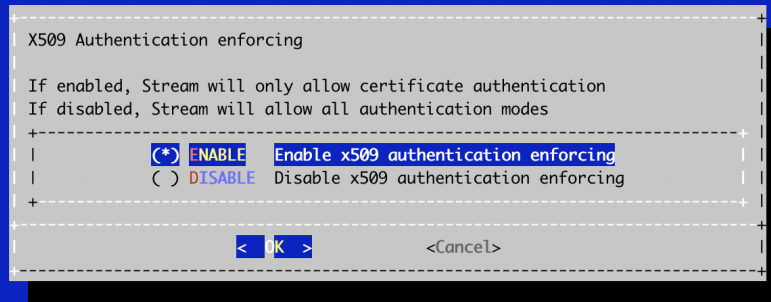
In the main menu, select '**Stream**':



In the Stream menu, select '**STREAM\_ENFORCE\_X509**':



In the X509 Authentication Enforcing menu, select '**ENABLE**':



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

```
# systemctl restart stream
```

X509 Authentication is now enforced.

## 6.5.2. Re-enable local authentication

### CAUTION

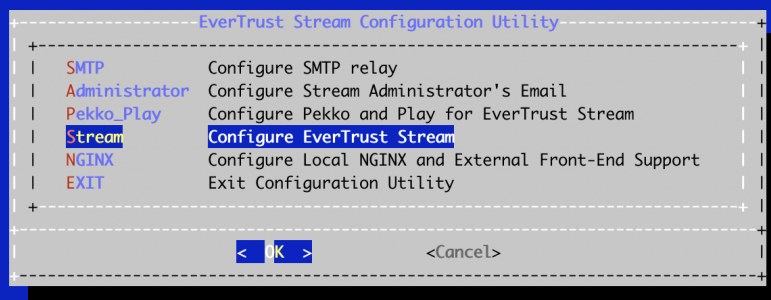
This should be done in a confined and secure environment, during the execution of the **Recover access** steps of the installation guide.

If you lose all available authentication certificates to Stream and want to re-gain access to the administration console, please follow these steps:

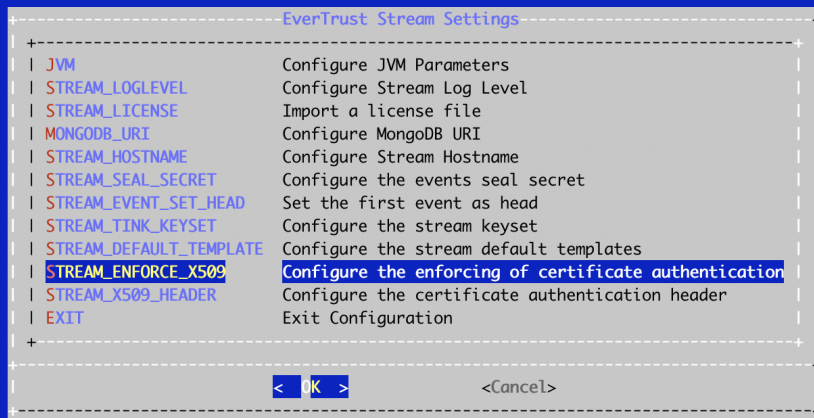
Access the server with an account with administrative privileges;

Start the Stream configuration utility by running:

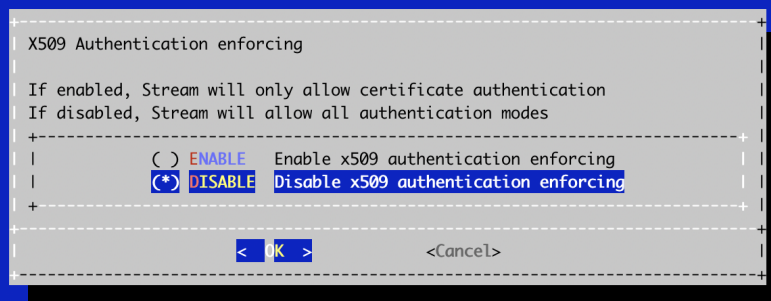
```
# /opt/stream/sbin/stream-config
```



In the Stream menu, select **'STREAM\_ENFORCE\_X509'**:



In the X509 Authentication Enforcing menu, select '**DISABLE**':



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

```
# systemctl restart stream
```

Now that the X509 enforcing is disabled, you can log in with Stream local accounts.

## 6.6. Events

The event system exists to overview the actions happening on Stream.


By default, the events are chained by the following rule: event  $n$  references event  $n-1$ . They are signed with the event seal secret set up during the stream installation.

To consult them:

1. Log in to the Stream Administration Interface.
2. Go to **System > Events**.

## 6.6.1. Event integrity reports

To check the integrity of the events, you can run an event integrity report:

1. Log in to the Stream Administration Interface.
2. Go to **System > Events Integrity Reports**.
3. Click ;
4. Click **Run**

The integrity of the event chain is checked and can take some time depending on the number of events in the database. Once finished, the report may have different status:

- **Running**: the integrity of the events is currently being checked.
- **Verified**: the event chain is not compromised.
- **Report integrity failure**: the report signature has been compromised.
- **Event integrity failure**: the event chain has been compromised, one event could have been modified or deleted. The event integrity report error provides details about the cause of the integrity failure.

### WARNING

Any compromised object means an account with enough permission to write in the database has been compromised.

## 6.6.2. Purging/Backup event database

### NOTE

Manual actions regarding the events manipulation should be done with stream turned off and in a confined environment.

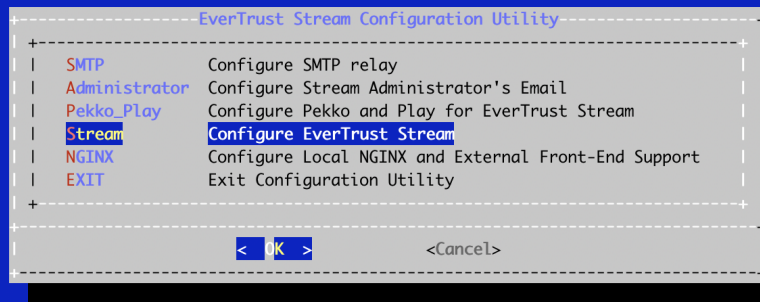
Follow the [Backup guide](#) to save your database. Once done, you might want to delete the events in database.

Deletion of events can only be made from the oldest to the newest since events are chained. For example, you might want to delete every event before a date:

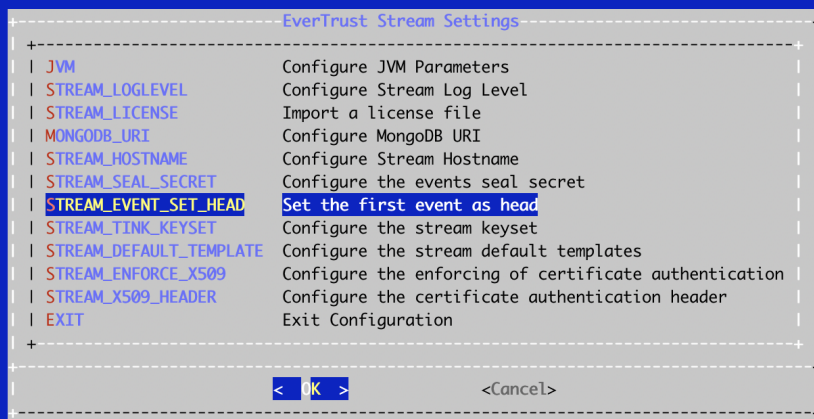
```
use stream;
db.events.deleteMany({"timestamp":{$lt: ISODate("2023-09-20")}});
```

After the deletion of events, the Head is still chained to a deleted event. In order to fix that, you will need to run the **Set the first event as head** in `/opt/stream/sbin/stream-config`:

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



In the Stream menu, select '**STREAM\_EVENT\_SET\_HEAD**':



### 6.6.3. Integrity compromised

If an event or event integrity report has been compromised, it means that someone had database access to Stream or one of its backups and manually edited the events to hide specific actions.

You should close all network access to the server and, if necessary, turn off stream. Once confined, you should follow these steps:

1. Follow the Backup guide to back up your database. It may be used to investigate the problem.
2. Analyze the logs (you may use an older verified backup to assess modifications).

#### NOTE

Since the database has been compromised, every event should be considered as a non trusted information

3. Based on your assessments, take the appropriate actions. This could mean changing the mongod password, changing the server password, revoking stream access certificates or other actions.
4. To resume a normal state, remove every corrupted event following the steps in the event purge guide.

# 7. Managing Certificate Lifecycle

## 7.1. Enroll

### NOTE

Stream's RA is not supposed to be a comprehensive registration authority and should only be used when necessary. This simple RA is made for "on the fly" generation only. If you want more advanced RA features to manually enroll certificates, you should consider using Horizon's Web RA.


To enroll a certificate via Stream:

1. Log in to the Stream Administration Interface.
2. Go to **Certificates > Enroll**. You'll be prompted to fill the following information:
  - **CA** (*select*): The CA that will issue the certificate. The CA must be managed by Stream;
  - **Template** (*select*): The Stream certificate template to use to issue the certificate;
  - **CSR type**: Whether the CSR to sign is in a dedicated file (**File** option) or in the clipboard (**Text** option);
  - **CSR field**: The CSR to sign (file or PEM-string).
3. Click the **Enroll** button.

Your certificate should now be visible in the Stream search engine.

## 7.2. Revoke

To revoke a certificate in Stream:

1. Log in to the Stream Administration Interface.
2. Go to **Certificates > Search** then find the certificate you want to revoke.
3. Click  on the certificate you want to revoke. Alternatively, you can click on the certificate's DN then click **Action > Revoke**.

Your certificate status should turn red.

## 7.3. Search

To search for certificates in Stream, log in to the Stream Administration Interface and then go to **Certificates > Search**.

Here are all the search criteria you can use:

- **CA**: the issuing certificate authority
- **Status**: the validity status of the certificate (valid, revoked or expired)

- **Template:** the certificate template the certificate has been enrolled on
- **Certificate DNs:** information regarding the certificate's DNs
- **Expiration date:** the date when the certificate will expire
- **Issuer:** information regarding the certificate issuer's DNs
- **Serial:** the certificate's serial number

You can combine any number of them to refine your search.

## 8. Backup and Restore

This section details how to use the provided EverTrust Tools to back-up and restore Stream if deployed using the RPM package. If you deployed Stream using Docker/Kubernetes, the configuration should be backed-up using the Docker/Kubernetes management platform, and the database should be backed-up using MongoDB tools.

### 8.1. Backup Procedure

This section details how to back up Stream configuration elements (the `/opt/stream/etc` folder, that includes the Nginx configuration, and the `/etc/default/stream` configuration file) and the Stream MongoDB database.

The backup tool allows backing up these elements independently.

```
# /opt/stream/sbin/stream-backup --help
```

Stream Backup tool usage: `stream-backup [-cdeho:q]`

`-c` | `--conf` Backup the Stream configuration files

`-d` | `--db` Backup the Stream MongoDB Database

`-e` | `--encrypt` Encrypt the backup files with the specified passphrase

`-h` | `--help` Display the 'stream-backup' help

`-o` | `--output` [path] Specify the Stream backup output folder (default: `/opt/stream/var/backup`)

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

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

```
# /opt/stream/sbin/stream-backup -c
```

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

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

```
# /opt/stream/sbin/stream-backup -d
```

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

To run a complete backup, execute the following command:

```
# /opt/stream/sbin/stream-backup -c -d
```

#### NOTE

- 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
- If you want to encrypt your back-up files, use the `-e` | `--encrypt` parameter. The backup tool will prompt you for a passphrase. The back-up will be encrypted using AES-256.

## 8.2. Restoration Procedure

This section details how to restore a Stream back-up that was generated using the `stream-backup` tool. The restoration happens using the `stream-restore` tool.

```
# /opt/stream/sbin/stream-restore --help
```

Stream restore tool usage: stream-restore

- a | --archive [filepath] The encrypted backup file to restore
- c | --conf [filepath] The path to the Stream configuration backup file
- d | --db [filepath] The path to the Stream database backup file
- m | --mongo\_uri [MongoDB URI] The MongoDB URI to back-up the database into (optional)
- h | --help Displays the 'stream-restore' help
- q | --quiet Quiet mode

Whenever trying to restore a backup, you need to stop the Stream service first:

```
# systemctl stop stream
```

To restore an unencrypted configuration backup, run the following command:

```
# /opt/stream/sbin/stream-restore -c [configuration backup archive path]
```

To restore an unencrypted MongoDB database backup, run the following command:

```
# /opt/stream/sbin/stream-restore -d [MongoDB backup archive path] -m [MongoDB URI]
```

The MongoDB URI is optional: if not provided, the script will try to infer it from the `/etc/default/stream` file. If it cannot be inferred and none is provided, the restore will fail.

To restore an encrypted backup archive, run the following command:

```
# /opt/stream/sbin/stream-restore -a [encrypted backup archive path] -m [MongoDB URI]
```

The restoration tool will prompt you for the passphrase that was used to encrypt the backup. If the archive contains only a configuration backup, the script will perform the equivalent of the `-c` parameter. If the archive contains only a database backup, the script will perform the equivalent of the `-d` parameter, and you might need to provide the MongoURI through the `-m` parameter. If the archive contains both a database and a configuration backup, both of them will be restored.

When the restoration is complete, you can start Stream again using the following command:

```
# systemctl start stream
```

## 9. Overridable configuration parameters

This page presents the overridable parameters from the Stream configuration.

### 9.1. Overriding the parameters

To override one of these parameters, simply :

1. Access the EverTrust Stream server with an account with administrative privileges;
2. With an editor like vi, open the `/opt/stream/etc/conf.d/stream-extra.conf` file;
3. Add this line at the end of the file :

```
<option name>=<option value>
```

As an example, if you want to modify the CA timeout in Stream and bump it up from 60 seconds to 300 seconds, you need to add this :

```
stream.ca.timeout="300 seconds"
```

4. Save your modifications and restart the Stream service :

```
# systemctl restart stream
```

#### NOTE

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.

You'll find below an exhaustive list of overridable parameters.

### 9.2. Customizing trust chain colors

Parameter	Default value	Description
stream.trustchain.ca.online.root.operational	"#08907B"	Displayed color of online operational root CAs in the trust chain viewer
stream.trustchain.ca.online.root.non_operational	"#76A2A0"	Displayed color of online non-operational root CAs in the trust chain viewer
stream.trustchain.ca.offline.root.non_operational	"#08907B"	Displayed color of offline non-operational root CAs in the trust chain viewer
stream.trustchain.ca.online.subordinate.operational	"#187EC7"	Displayed color of online operational subordinate CAs in the trust chain viewer

Parameter	Default value	Description
stream.trustchain.ca.online.subordinate.non_operational	"#6892B1"	Displayed color of online non-operational subordinate CAs in the trust chain viewer
stream.trustchain.ca.offline.subordinate.non_operational	"#08907B"	Displayed color of offline non-operational subordinate CAs in the trust chain viewer

### 9.3. Bootstrapping parameters

Parameter	Default value	Description
stream.bootstrap.timeout	"1 minute"	Duration after which the bootstrap of Stream times out
stream.bootstrap.administrator.name	"administrator"	Default administrator account name
stream.bootstrap.administrator.password.path	"var/run/adminPassword"	Absolute path of the file where the initial admin password should be stored into
stream.bootstrap.administrator.password.length	24	Length (in bytes) of the initial admin password

### 9.4. Timeout parameters

Parameter	Default value	Description
stream.ca.timeout	"60 seconds"	Duration after which a signing request times out
stream.security.manager.timeout	"10 seconds"	Maximum duration that Stream can wait to get an answer from the actor that handles authentication
play.http.session.maxAge	"15 minutes"	Duration after which the authentication session expires
stream.crl.storage.timeout	"60 seconds"	Duration after which Stream times out when pushing a CRL to an external CRL storage
stream.queue.timeout	"5 seconds"	Duration that the Certificate Authority Manager actor will wait to retrieve all the existing queues in Stream before timing out
stream.trust.manager.timeout	"10 seconds"	Duration that the Trust Manager actor will wait to retrieve information about certificates (is it trusted ? its trust chain ?)

Parameter	Default value	Description
stream.trust.manager.cache.external.expireAfterAccess	"30 days"	Time during which an external CA CRL is kept in cache before being removed if nothing accesses it
stream.trust.manager.cache.managed.expireAfterAccess	"5 minutes"	Time during which a managed CA CRL is kept in cache before being removed if nothing accesses it
stream.keystore.timeout	"5 seconds"	Maximum duration for Stream to retrieve the signature of a CRL from the Content Signer

## 9.5. HTTP Header parameters

Parameter	Default value	Description
stream.http.header.realip	"X-Real-IP"	Name of the HTTP header to use as Real IP
stream.security.http.headers.xapi.id	"X-API-ID"	Name of the HTTP header to use as XAPI-ID
stream.security.http.headers.xapi.key	"X-API-KEY"	Name of the HTTP header to use as XAPI-KEY

## 9.6. Search queries parameters

Parameter	Default value	Description
stream.certificate.search.page.default_size	50	How many elements to retrieve in a certificate search query if no pageSize has been specified
stream.certificate.search.page.max_size	(no default value)	How big can the pageSize parameter be in a certificate search query ? (Must be a positive integer)
stream.event.search.page.default_size	50	How many elements to retrieve in an event search query if no pageSize has been specified
stream.event.search.page.max_size	(no default value)	How big can the pageSize parameter be in an event search query ? (Must be a positive integer)

## 9.7. Security parameters

Parameter	Default value	Description
stream.security.trustmanager.enforce_serverauth	false	Enforces TLS authentication for the Stream web application
stream.secret.manager.keyset.path	"/etc/stream.keyset"	Relative path (relative to /opt/stream/) to get the keyset file for Stream from
stream.event.ttl	(no default value)	(Optional) Duration after which Stream technical events will be removed from database. If not set manually, technical events will never be removed from database through the TTL mechanism but can still be removed if the events collection is capped in the Mongo database.
stream.event.chainsign	true	Specify whether to chain and sign the Stream events to ensure they haven't been tampered with
stream.event.seal.algorithm	"HS512"	Algorithm to use to sign the Stream events
stream.event.manager.interval	"5 seconds"	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 Stream
stream.account.secret.length	42	Length of random passwords generated when creating a local account or resetting one's password
stream.crl.storage.sync.interval	"15 minutes"	How often to push CRLs into external storages
stream.event.disable-stacktrace	true	If the stacktrace must be disabled in the event

## 9.8. Queue parameters

Parameter	Default value	Description
stream.queue.default.parallelism	5	Stream's default queue parallelism size (the number of concurrent signature requests that can be processed at once)
stream.queue.default.size	100	Number of signature requests that can be queued before starting to discard them

<b>Parameter</b>	<b>Default value</b>	<b>Description</b>
stream.crl.queue.size	100	Number of CRL signature requests that can be queued before starting to discard them