EVERTRUST

EverTrust Stream documentation v1.1

Administration 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 **1.1**, 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".

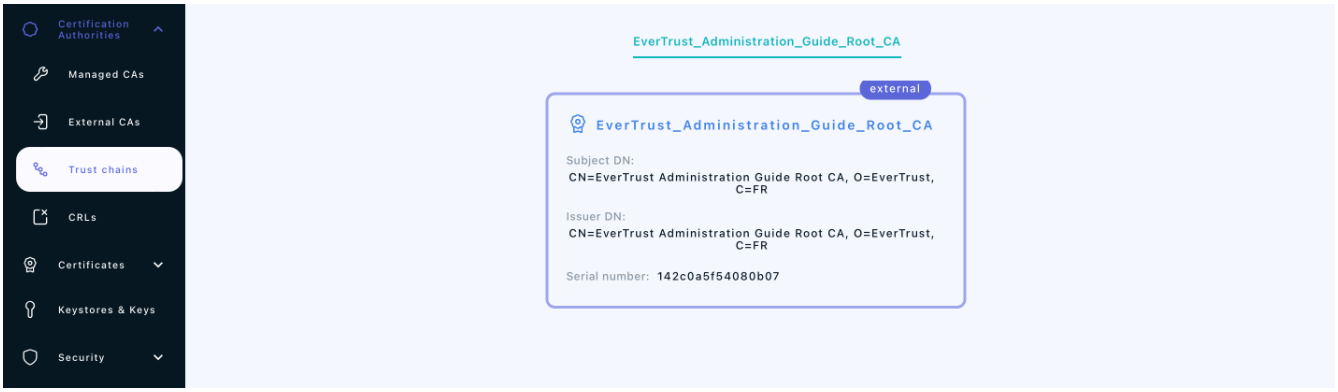
4. In the **Details** tab, check if the details that were parsed from the certificate match those of the CA you wish to import. If it does, click "Next".

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- Specify the **Outdated Revocation Status Policy**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.

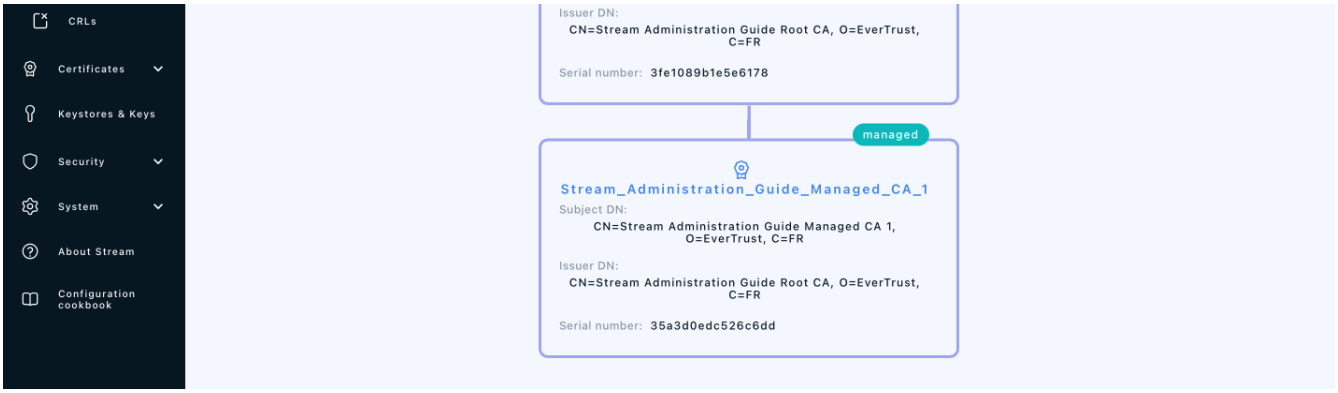
4. Scroll down to the bottom of the page and check the certificate's information. If everything is correct, click "Next".

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'd 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.

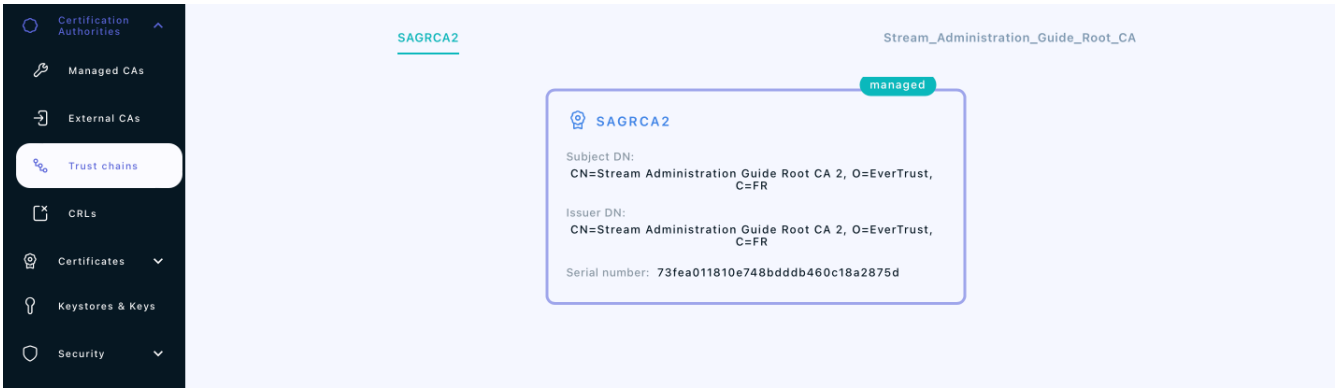
5. Select **Selfsigned** as a signing method, and pick the hash algorithm of your choice.

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, your 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'd 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.

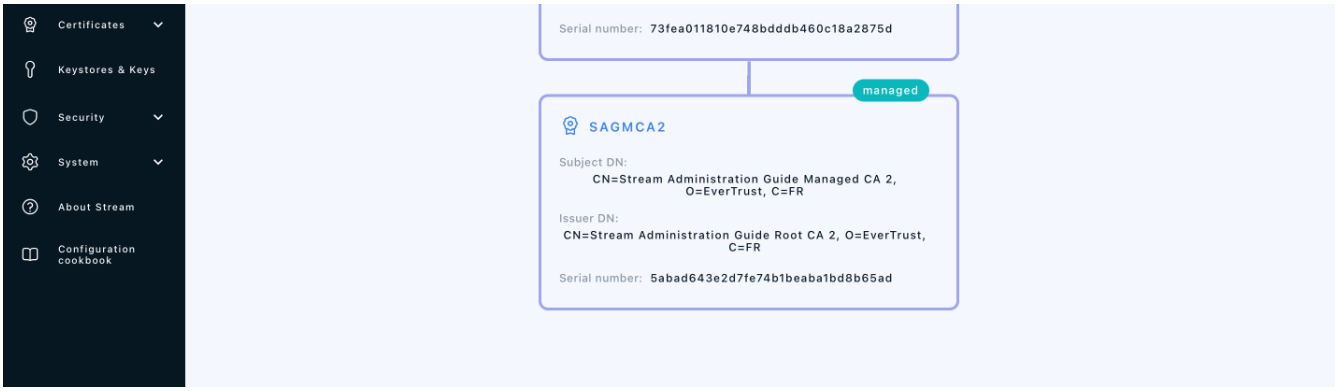
6. Select the **Managed CA** you want to sign the certificate with.

7. Set the **lifetime** or 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 "Add".

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**:



2.4.2. 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'd 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.

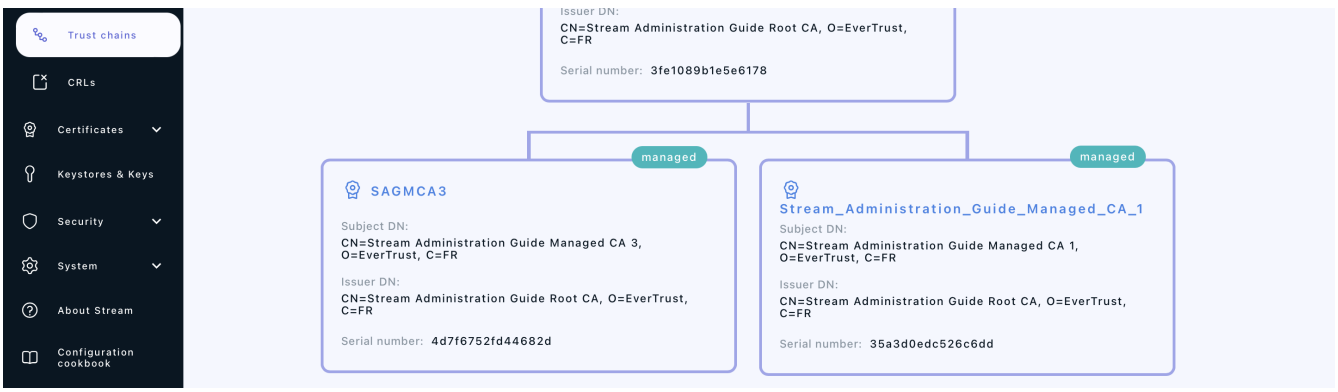
6. Click the link in the **Export** section to download the **CSR** for your CA, then sign it using your external CA and export the signed certificate under PEM or DER format.

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.

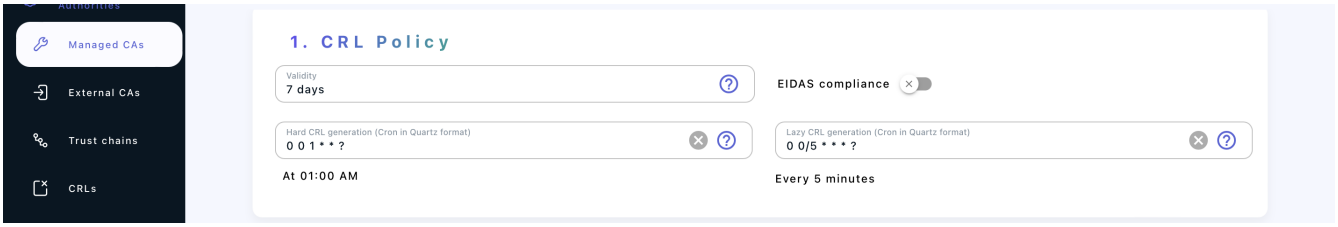
3. Go under the **CRL** tab.

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:

- 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

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 a credential for a cloud provider if you want to use a cloud storage solution.

To configure an external 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 :

- 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.

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.

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:

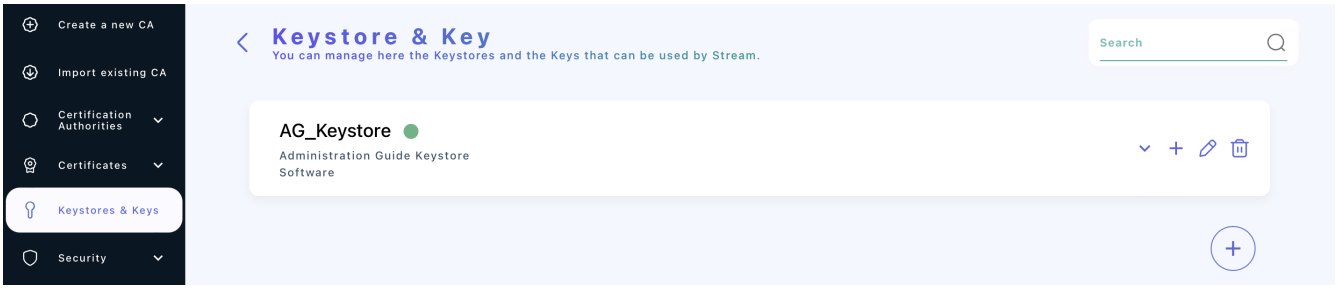
1. Log in to the Stream Administration Interface.

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.

If everything was good, 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.

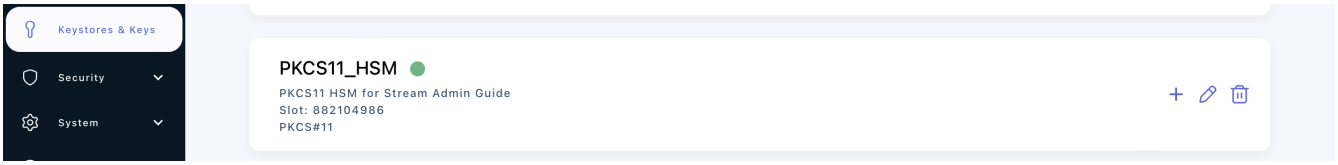
2. Go to **Keystores and keys** and click ⊕.

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 succesfully 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**. Then click "Add".

If everything was good, 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 ⊕.

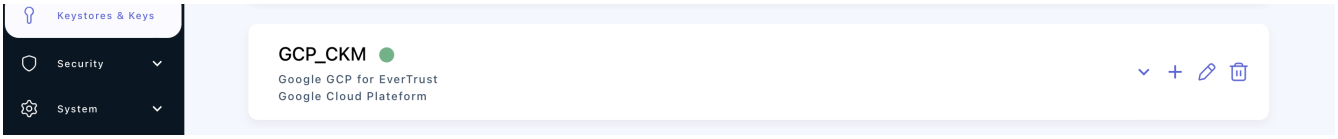
3. In **Type**, select **Google Cloud Platform**. In **Name**, set the name you want to give to your keystore. Optionally, you can add a description to your keystore.

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".

If everything was good, 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 ⊕.

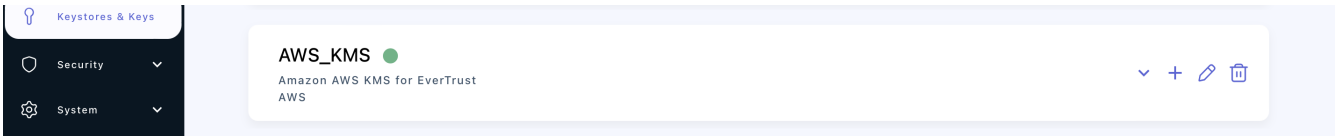
3. In **Type**, select **AWS**. In **Name**, set the name you want to give to your keystore. Optionally, you can add a description to your keystore.

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".

If everything was good, 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.

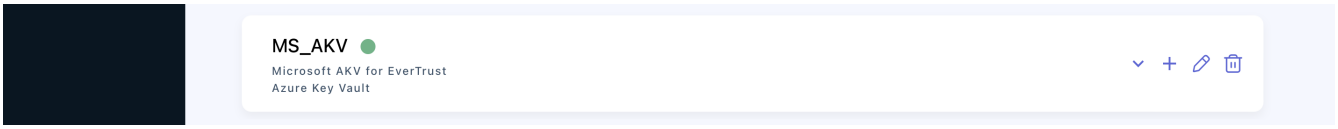
2. Go to **Keystores and keys** and click ⊕.

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".

If everything was good, 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.

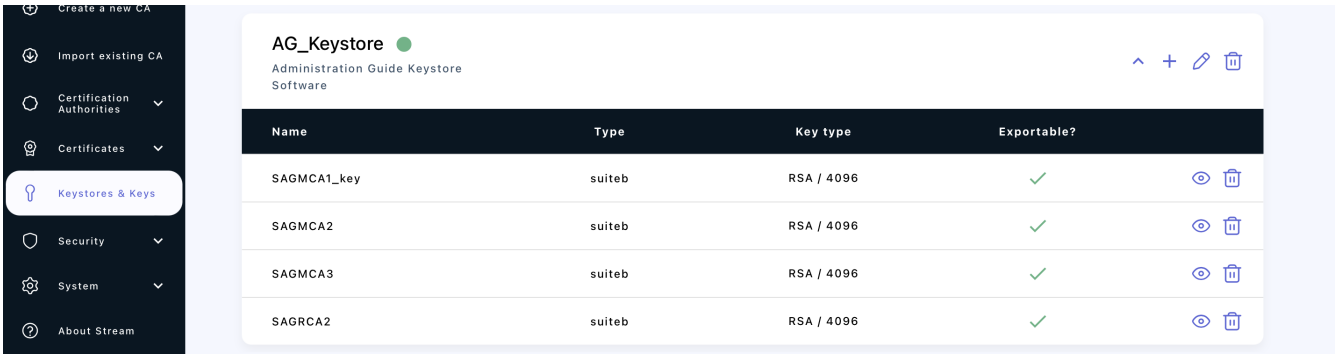
2. Go to **Keystores and keys** and click + on the keystore you want to add the key into.

3. Set the name of the key as well as the key type (RSA or ECDSA) and the key size (for RSA)/key parameter(for ECDSA).

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".

If everything went good, 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:

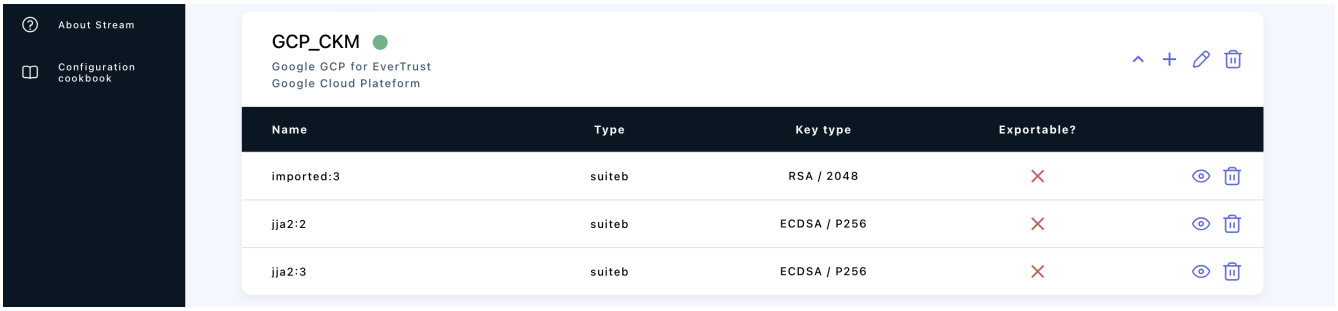


4.5.2. Viewing the keys of a keystore

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.

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 **Policy, CRLDP & AIA** tab, you can edit the **CRLDPs**, **AIA** and **Policy** 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.

6. In the **DN & Extensions** 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** swiches 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**) and 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.

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 Cloud KMSs in Stream, 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 or Google GCP);

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;

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

7. Managing Certificate Lifecycle

7.1. Enroll

At the moment, enrolling a certificate via Stream is only possible through the API call. Please refer to the developer guide for more information.

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. Overridable configuration parameters

This page presents the overridable parameters from the Stream configuration.

8.1. Overriding the parameters

To override one of these parameters, simply :

1. Access the EverTrust Stream server through SSH with an account with administrative privileges;

2. With an editor like vi, open the **/etc/default/stream** file and go at the bottom of it;

3. Add this line at the end of the file :

```
JAVA_OPTS="$JAVA_OPTS -D<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 :

```
JAVA_OPTS="$JAVA_OPTS -Dstream.ca.tiemout="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.

8.2. 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

8.3. Bootstrapping parameters

Parameter	Default value	Description
stream.bootstrap.timeout	"1 minute"	Duration after which the boostrap 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

8.4. 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

8.5. 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

8.6. 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)

8.7. 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 throug 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	AlgorithmIdentifiers.HMAC_SHA512	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

8.8. 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
stream.crl.queue.size	100	Number of CRL signature requests that can be queued before starting to discard them