Signature Activation Module

Product description

ENTRUST
SECURING A WORLD IN MOTION

Table of Contents

Introduction

The EU eIDAS Regulation and other electronic signature regulations recently introduced a new use case for digital signing services: server signing, also known as remote signing service when operated by a trust service provider (TSP).

With server signing, signing keys and certificates are centralized on a server, with mechanisms to guarantee the signer retains sole control of the signing key. This model is different from local signing, where certificates and keys are distributed to the user's local machine(s).

Although the scope of a regulation is always specific to its jurisdiction (or grouping of jurisdictions), it's common practice to rely on a set of widely accepted technical standards. In this sense, it's worth recognizing the standardization work carried out in recent years by organizations such as ETSI and CEN.

ETSI and CEN standards

ETSI is primarily focused on defining the technical, policy, and security requirements of a server signing system. In contrast, CEN is mainly focused on the security requirements of the software and hardware products involved in supporting server signing systems.

In the context of EU eIDAS, the trust service provider (TSP) operating a server signing system is supervised to ensure that the requirements based on ETSI and CEN standards are met. The most important standards are:

Policy and security

- **ETSI EN 319 401:** Electronic Signatures and Infrastructures (ESI); General Policy Requirements for Trust Service Providers

- **ETSI TS 119 431-1:** Policy and security requirements for trust service providers; Part 1: TSP service components operating a remote QSCD/SCDev

- **ETSI TS 119 431-2:** Policy and security requirements for trust service providers; Part 2: TSP service components supporting AdES digital signature creation

Server signing support

Currently, CSC Remote Signing Protocol v0.1.7.9 is used. However, technology will progressively adopt the new versions of Remote Signing Protocol such as ETSI TS 119 432: Protocols for Remote Digital Signature Creation and Cloud Signature Consortium Remote Signing Protocol v1.0.

- **CEN EN 419 241-1:** Trustworthy Systems Supporting Server Signing Part 1: General System Security Requirements, dated 2018-02

ETSI and CEN standards (continued)

Server signing QSCD (Qualified Signature/Seal Creation Device)

- **CEN EN 419 241-2:** Trustworthy Systems Supporting Server Signing Part 2: Protection Profile for QSCD for Server Signing, dated 2018-05-11; this standard is a protection profile for the Signature Activation Module

- **CEN EN 419 221-5**: Protection Profiles for TSP Cryptographic Modules - Part 5 - Cryptographic Module for Trust Services; this standard is a protection profile for the HSM

Both the CEN EN 419 241-2 and the CEN EN 419 221-5 standards establish QSCD requirements in EU eIDAS Regulation Annex II. These two standards are expected to be added by the European Commission to the list of standards for the security assessment of information technology products in the next implementing act.

Currently, in the absence of listed QSCD standards for server signing, the conformity of QSCDs is certified by appropriate public or private bodies designated by member states. Once the next implementing act is released, the two CEN technical standards above are expected to become the default ones for QSCD conformity in the European Union and for an increasing number of geographies that are progressively referring to those standards.

The **Entrust Signature Activation Module** described in this document is software installed within a protected, tamper-evident module. Together with the cryptographic module, they form a QSCD aligned with CEN EN 419 241-2 and CEN EN 419 221-5.

Product description

The Entrust Signature Activation Module (SAM) is used to deploy a server-side endpoint that will be used by the server signing applications to get data signed (a document hash). The SAM receives the signer authentication data, the signer's signing key, and the data to be signed through a Signature Activation Protocol (SAP). It interacts with either the Entrust nShield Solo XC HSM (PCIe card) or the nShield Connect XC HSM (network-attached appliance) to return the encrypted data with the signing key. In technical documentations, the HSM is usually called cryptographic module or CM.

The design of the Entrust SAM is based on the Trustworthy Systems Supporting Server Signing (TW4S) architecture described in the CEN EN 419 241 standards. It implements the CEN EN 419 241-2 standard to be integrated with the Server Signing Application (SSA) as described in CEN EN 419 241-1.

The cryptographic module used with the Entrust SAM module can either be the Entrust nShield Connect XC HSM or the Entrust nShield Solo XC HSM. Both are Common Criteria certified for CEN EN 419 221-5 and FIPS 140-2 L3.

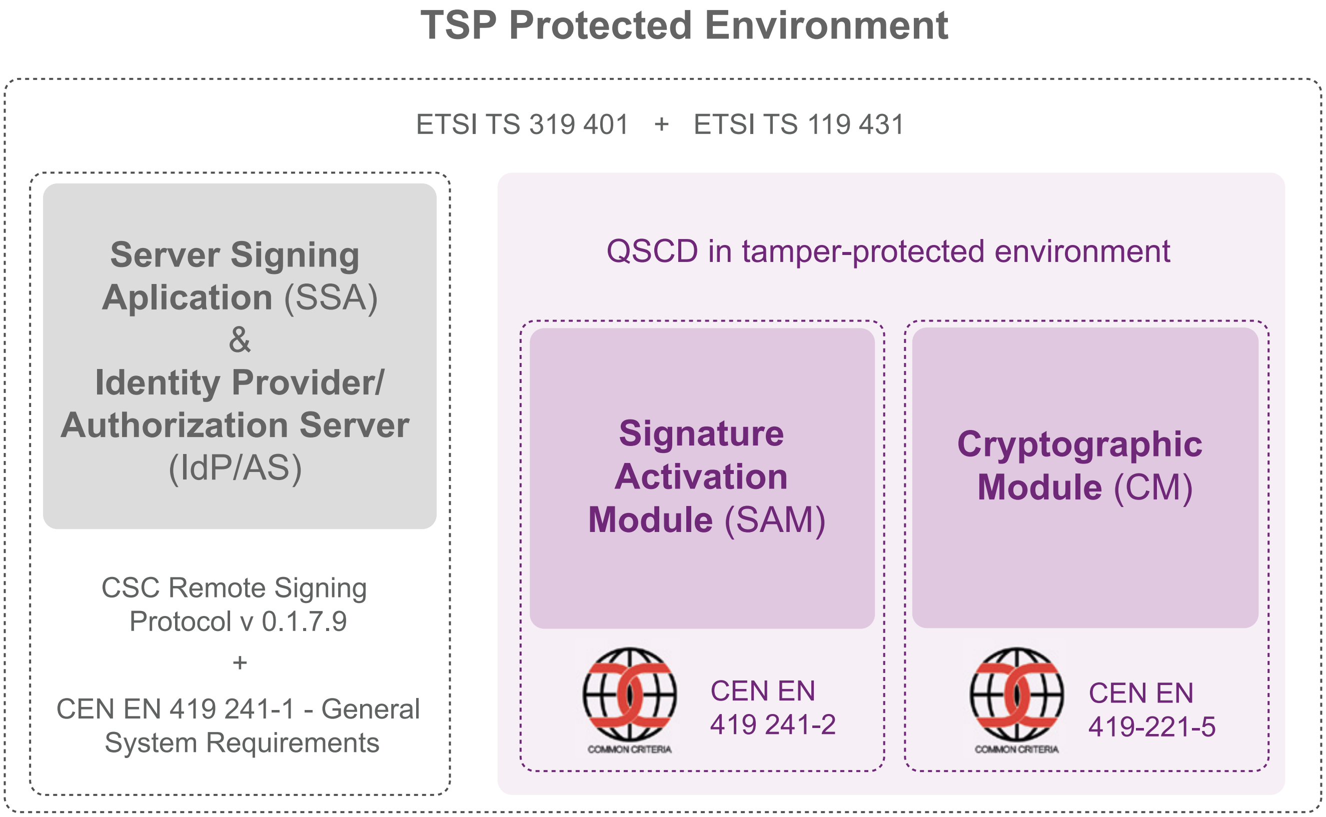

TSP Protected Environment
ETSI TS 319 401 + ETSI TS 119 431
Server Signing Aplication (SSA) & Identity Provider/ Authorization Server (IdP/AS)
CSC Remote Signing Protocol v 0.1.7.9
+
CEN EN 419 241-1 - General System Requirements
QSCD in tamper-protected environment
Signature Activation Module (SAM)
Cryptographic Module (CM)
COMMON CRITERIA
CEN EN 419 241-2
COMMON CRITERIA
CEN EN 419-221-5

Figure 1: General architecture of a Trustworthy System Supporting Server Signing (TW4S). The implementation can vary depending on the customer's requirements and existing infrastructure.

Functional description

The Entrust SAM carries out the signature operation authorization required to ensure the signer retains the sole control of their signing keys. The SAM activates a signing key within the CM via a Signature Activation Protocol (SAP) through which it receives data from the Server Signing Application (SSA) and an Identity Provider/Authorization Server (IdP/AS). This IdP/AS can be part of the SSA or not, and is in charge of authenticating the signer. The data element received by the SAM – which binds the signer authentication and the signing key[1], plus the hash of the documents to be signed – is called Signature Activation Data (SAD).

The Entrust SAM uses the SAD to guarantee with a high level of confidence that the signing keys are used under the signer's sole control. For this, as specified in the CEN EN 419 241-2 standard, the SAM must be deployed in a tamper-protected environment.

The Entrust SAM is used in combination with a CEN EN 419 221-5 certified Entrust nShield Solo XC or Connect XC HSM to support the deployment of a Trustworthy System Supporting Server Signing (TW4S) for generating qualified electronic signatures/seals. Specifically, it enables the deployment of a remote Qualified Signature/Seal Creation Device (QSCD) when operated by a Qualified Trust Service Provider (QTSP).

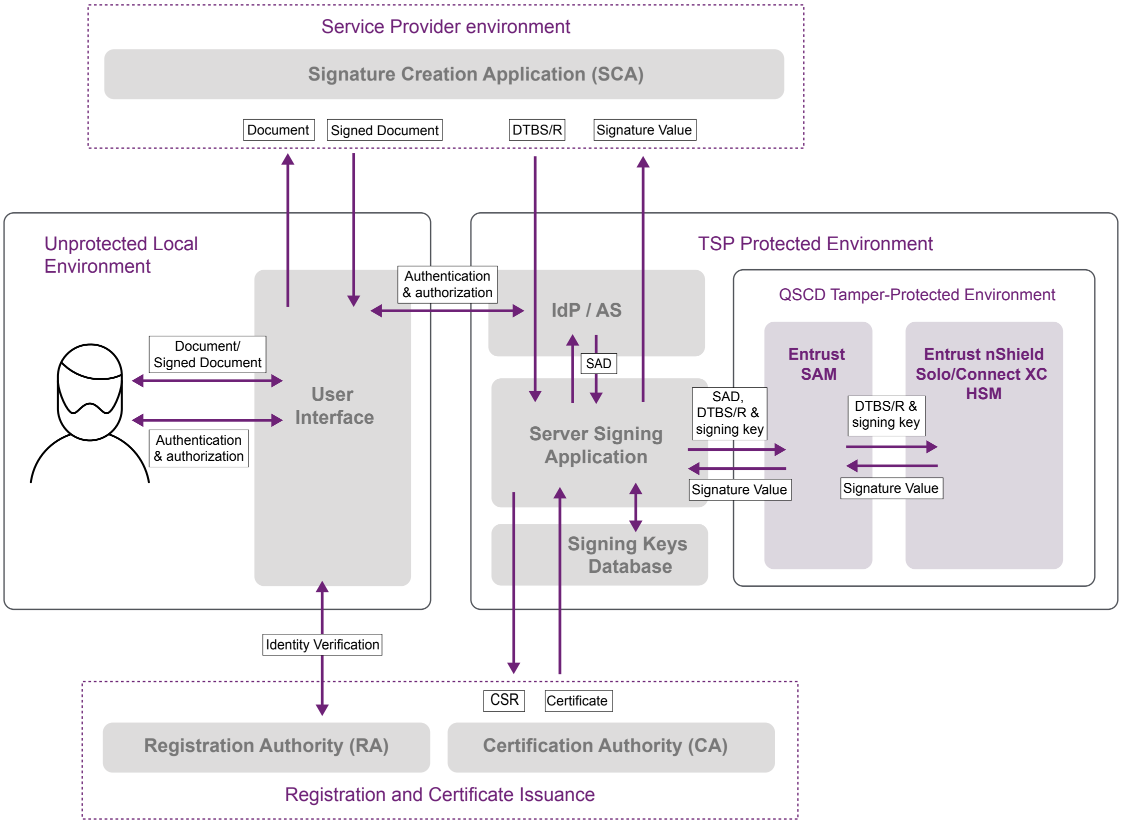

Service Provider environment
Signature Creation Application (SCA)
Document
Signed Document
DTBS/R
Signature Value
Unprotected Local Environment
Document/ Signed Document
Authentication & authorization
User Interface
Authentication & authorization
TSP Protected Environment
IdP / AS
SAD
Server Signing Application
Signing Keys Database
QSCD Tamper-Protected Environment
Entrust SAM
Entrust nShield Solo/Connect XC HSM
SAD, DTBS/R & signing key
Signature Value
DTBS/R & signing key
Signature Value
Identity Verification
CSR
Certificate
Registration Authority (RA)
Certification Authority (CA)
Registration and Certificate Issuance

Figure 2: Entrust SAM in the context of a Trustworthy System Supporting Server Signing (TW4S). The implementation described can vary depending on the customer's requirements and existing infrastructure.

[1] The signing key is sent in an encrypted state. It is decrypted once loaded into the CM by the SAM.

Functional description (continued)

The SAP is secured with TLS using mutual authentication in order to get the client strongly authenticated. In other words, the SAM (the server-side component) provides strong authentication to the SAM clients, which are called privileged user(s) of the signing service under the SAM terminology. There are two types of privileged users:

1. Operation privileged users (e.g., administrators)

2. Signing service privileged users (e.g., SSAs)

After the TLS handshake, a cryptographically secured channel is provided for all communications between the SAM and the clients.

The Identity Provider/Authorization Server (IdP/AS) component is recognized as a trusted entity by the SAM. It authenticates the signers and generates a signed authorization assertion (SAD) vouching for the correct authentication and the explicit consent of the signer to create a concrete signature.

Signers are strongly authenticated by using a delegated scheme or a partially delegated scheme. In the delegated scheme, all of the factors are authenticated by the IdP/AS; while in a partially delegated scheme, one user authentication factor can be managed by the SAM while the rest of the factor/s are delegated to the IdP/AS.

From a security point of view, the Entrust SAM has a whitelist of authorized AS and SSA:

- The AS is used to generate and digitally sign the SAD (also called authorization assertion)

- The SSA is used to securely connect to the SAM (using a TLS mutual authenticated connection)

Thanks to this method, only the SSAs explicitly registered in the SAM can connect and request signing services. Also, the SAM will only accept a SAD that is digitally signed by an AS it has previously registered as a trusted entity.

Functional description (continued)

The Data To Be Signed/Representation (DTBS/R) is the hash of the document computed by the Signature Creation Application (SCA). The SCA invokes the SSA for the signing operation by sending the DTBS/R to the SSA, which will then request a SAD from the AS, and send it to the SAM along with the DTBS/R and the signing key to use for the signature.

After the SAD has been checked, the SAM performs, as a minimum, the following tasks:

1. Loads the signing key into the cryptographic module (CM)[2]
2. Invokes the CM signing function on the DTBS/R
3. Gets the created signature
4. Removes the signing key from the CM
5. Generates a digitally signed audit log attesting that the signature has been generated
6. Sends back the created signature to the SSA

The SSA receives the PKCS#1 signature from the SAM, and the signature is returned back to the SCA as the result of the signing service.

The signing service ensures the safe creation of SAM clients (SSAs or "signing service privileged user" as described in the SAM terminology) through a process that maintains the segregation of roles. To summarize, these roles consist of the signers and some privileged roles (clients) who maintain and operate the product.

[2] The signing key (also called Signature Creation Data) is always in encrypted format. Signing keys can only be decrypted once loaded into the CM since they have all been encrypted by the CM itself during their creation/registration in the system.

Functional description (continued)

From a functionality point of view, the Entrust SAM is also used for the generation and deletion of the signers' key pairs. The key pair generation operation for a signer has two parts:

1. Generating a key pair
2. Assigning the key to a specific signer (after certifying the public key)

Regarding the key generation, the explicit authorization of the signer is not necessary, allowing use cases such as pre-generation of keys. A key pair assigned to a signer cannot be assigned to another user, thus respecting the uniqueness of the signers' keys.

Regarding the deletion of key pairs, both a specific privileged user (accepted in the system and authenticated by the TLS protocol with client authentication) and the signer who own the key pair can request it. When a key pair is deleted, the SAM invokes security mechanisms to ensure the keys must no longer be used.

The Entrust SAM generates audit records for all security events involved in its operations. The SAM issues and signs an event log for any given service request, and stores them in a permanent repository. Audit records can be verified and securely stored externally by the SSA. (NOTE: The SAM's external log or any SIEM database is not shown in figure 2).

API interface

The Entrust Signature Activation Module includes an API to invoke administration functions and service functions.

Administration functions

The administration functions are used to modify the dynamic configuration, which include parameters that can be changed without the need to restart the service. An administration operation can be performed using the following endpoint[3]: **PUT /sam/v1/configurations/dynamic**

This operation is generally used to register new SSAs (signing service privileged users in SAM terminology) and, less frequently, to register new IdP/AS.

In the request, the following JSON structure is sent:

```
{
   “version” : {number},
   “authorization_servers” : [
      {
         “id” : {string},
         “public_key” : {string}
      }
   ],
   “signing_service_privileged_users” : [
      {
         “id” : {string},
         “public_key” : {string}
         “allowed_authorization_servers” : [ {string} ]
      }
   ]
}
```

If the operation is successful, the response simply contains the following status line: **HTTP/1.1 204 No Content**

[3] For details, go to SAM product documentation and navigate to: API / Administration functions / Set the dynamic configuration

API interface (continued)

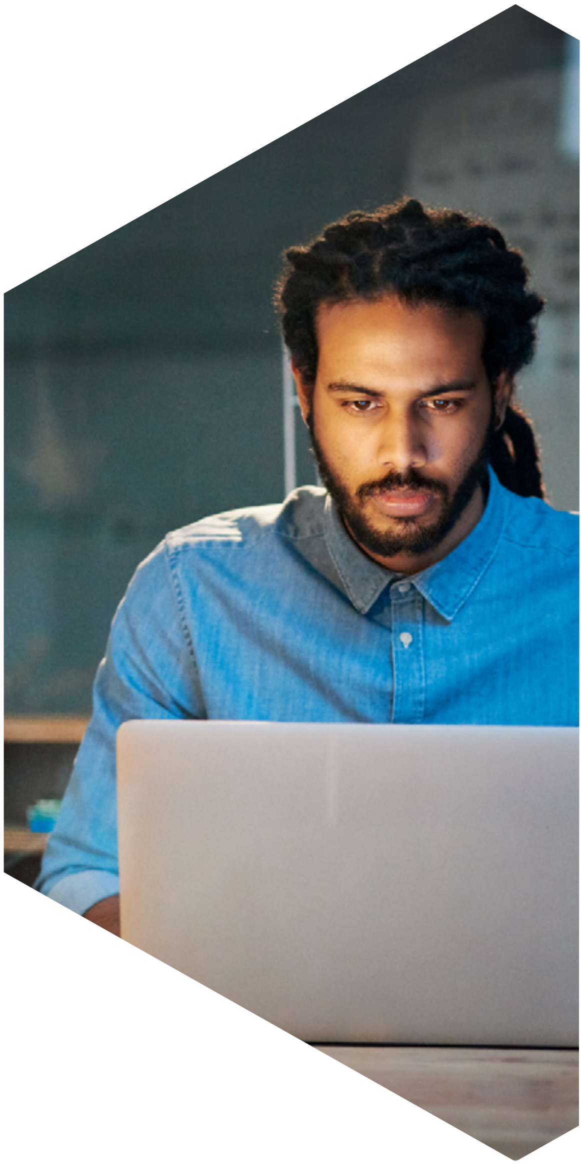
Service functions

Service functions are often used to create digital signatures on behalf of end-users (signers). However, signing keys and signing certificates for end-users have to be created first:

The first operation would be to create the signing keys[4]:
POST /sam/v1/key_pairs

In the request, a JSON structure is sent indicating the cryptographic algorithm (RSA or ECDSA) and other parameters necessary to create the signing keys:

```
{
   “key_template” : {
      “algorithm” : {string},
      “size” : {number},
      “named_curve” : {string}
   }
}
```



If the operation is successful, the response contains the following status line and a JSON structure with the created key pair: **HTTP/1.1 201 Created**

[4] For details, go to SAM product documentation and navigate to: API / Service functions / Generate signing key

API interface (continued)

The Entrust SAM is stateless, so it is the responsibility of the SSA to store the key pairs in their own databases and provide them again to the SAM in each operation that requires them. As for the security of the private keys, they are always encrypted with keys that are contained within the HSMs and therefore only usable when reloaded into the same HSMs.

At this point, the signing keys have been created but have not yet been assigned to an end-users. Once the user is known, the keys can be assigned to the user, who will be the sole owner throughout its lifecycle[5]:
POST /sam/v1/signers/$body/key_pairs

In the request, a JSON structure is sent that includes the information of the key pair that will be assigned to the user, and the information of the users themselves:

```
{
   “key_pair” : {object},
   “signer” : {object},
   “subject” : {string}
}
```

If the operation is successful, the response contains the following status line:
HTTP/1.1 200 OK

Some customer projects require the signing service to keep a record of the end-user’s consent for a signing key to be created for them, usually for the purpose of compliance with legal regulations. In this case, in the operation that we have just described above, Signature Activation Data (also called a SKAD) must be added to the request. It’s a structure signed by a trusted authorization server and contains information about the context in which the user gave the consent (see Resources | SKAD in the product documentation).

The next step is to create the signing certificate. It is very common for CAs to require proof of possession of the signing keys before proceeding to create the signing certificate, so you need to get this proof first and it needs to be signed by the SAM[6]: **POST /sam/v1/proofs_of_possession**

[5] For details, go to SAM product documentation and navigate to: API / Service functions / Assigning a signing key
[6] For details, go to SAM product documentation and navigate to: API / Service functions / Generate a proof-of-possession

API interface (continued)

The proof of possession is generated by the SSA; it is generally a CSR based on PKCS #10 and must be signed by the SAM before being sent to the CA. In the body of the signature request for the SAM, the CSR is referenced as "DTBS" (which is not related to DTBS/R).

```
{
   “key_pair” : {object},
   “request” {
      “dtbs” : {string},
      “signature_algorithm” : {string},
   }
}
```

If the operation is successful, the response contains the following status line and a JSON structure with the signature of the DTBS/R: **HTTP/1.1 200 OK**

Now you can contact the CA to have the signing certificate created. This request is made by the SSA and is outside of the scope of this document. Once the certificate is obtained, an operation has to be invoked to modify the information of the signing key, indicating the value *certified*[7]: **PUT /sam/v1/key_pairs/$body**

In the request, a JSON structure is sent that includes the information of the key pair and the user:

```
{
   “key_pair” : {object},
   “signer” : {object},
   “status” : {
      “certified” : true,
      “enabled” : {boolean}
   }
}
```

If the operation is successful, the response contains the following status line: **HTTP/1.1 200 OK**

[7] For details, go to SAM product documentation and navigate to: API / Service functions / Update a signing key

API interface (continued)

Finally, everything is ready to create digital signatures[8]:
POST /sam/v1/signatures

In the request, a JSON structure is sent. It includes an array of data to be signed (DTBS/R), which corresponds to the hash of the documents to sign (e.g., the hashes of several PDF documents that the user wants to sign):

```
{
   “signature_algorithm” : {string},
   “requests” : [
      “dtbs” : {string},
      “signature_algorithm” : {string}
    ],
   “key_pair” : {object},
   “signer” : {object}
}
```

If the operation is successful, the response contains the following status line and a JSON structure with the signature of the DTBS: **HTTP/1.1 200 OK**

In the above operation, Signature Activation Data (also called a SKAD) must always be added to the request (in the header) as evidence that the end-user has given their consent for the signatures to be created on their behalf. It is a structure signed by a trusted authorization server, which contains information about the context in which the user gave the consent.[9]

[8] For details, go to SAM product documentation and navigate to: API / Service functions / Generate a digital signature

[9] For details, go to SAM product documentation and navigate to: Resources / SAD

Architecture and deployment

The Entrust SAM is delivered in the form of software that is installed on a Linux operating system, specifically CentOS 7 or greater. Entrust SAM 1.0.2 also supports Linux RHEL, version 7 or later. A physical server is required, equipped with a baseboard management controller (BMC) chip for tamper detection. **An Entrust nShield HSM is required to work with the Entrust SAM – either Solo XC or Connect XC.**

When creating the signing keys, the SAM determines the cryptographic algorithms that are available through its APIs. The supported algorithms are:

- **RSA**: Key sizes can be 2048 or 4096 bits
- **ECDSA**: Supported curves are secp256r1, prime256v1, secp384r1, prime384v1, secp521r1, and prime521v1

Figure 3 below shows a set of service components that the Entrust SAM interacts with. To deploy the SAM in a high availability cluster, you can simply deploy multiple nodes and a load balancer in front of them (the SAM does not include any load balancing functionality), or implement load balancing in the SSA that the SAM interacts with, if available.

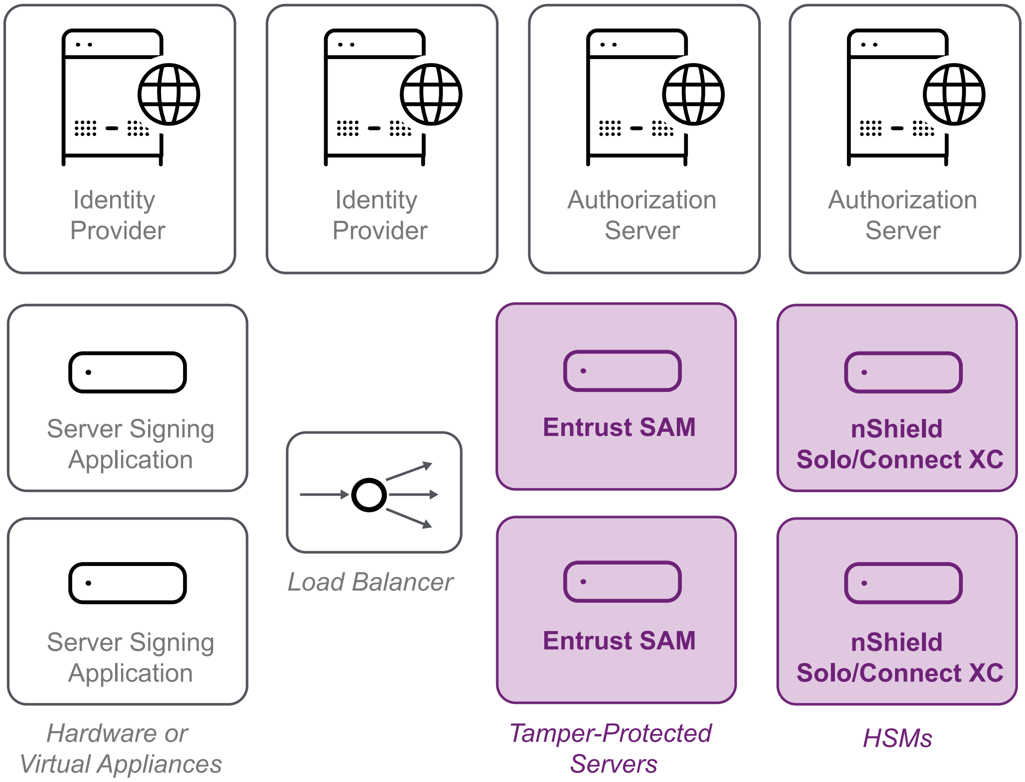


Figure 3: An example of Entrust SAM deployed in high availability. The implementation described can vary depending on the customer's requirements and existing infrastructure.

Architecture and deployment (continued)

The rest of the components shown in Figure 3 (identity provider, authorization server, network infrastructure elements, etc.) should also be deployed in high availability. Other PKI-related environment components, such as the CA that creates the signing certificates, could also be operated by the trust service provider itself, but in any case, they are not included as part of the Entrust SAM.

The SAM does not require a Database Management System (DBMS) to run, since all data is received from the SSA through the APIs. The SSA will most likely use a database to store details of the end-users and their signing keys, especially if it handles large volumes.

However, the SAM does need disk storage for logs. Specifically, the logs are recorded in two files[10]:

- One for service operations
- One for administration operations

These files will grow in size over time and you'll need to schedule some maintenance. One of the most common solutions is to use the logrotate tool that comes with most Linux operating systems. This way, you can configure how often you want to create a new log file, leaving the old files free to be archived off the server and perhaps deleted. The Entrust SAM is prepared to process queries sent by logrotate, and the file change will occur in a controlled manner without the risk of any data loss.

You may also want to forward a copy of the logs in real time to a centralized log server or SIEM. To do this, you can use the rsyslog tool that also normally comes in Linux operating systems, or another tool recommended by the SIEM manufacturer.

To deploy two or more nodes of the SAM module, the configuration files of the first node are needed. Three files are needed:

- **One file corresponding to the so-called private configuration (sam_infrastructure.json).** It is created automatically when initializing the first node with the command **admin – create**[11]

To import it into the rest of the nodes, use the command **admin – initialize**[12]

- **Two files corresponding to the so-called public configuration (static_config.json and dynamic_config.json).** They are created using a text editor following the instructions in the documentation and imported in all nodes in the same way, using the command **admin – import**[13]

[10] For details, go to SAM product documentation and navigate to: Resources / Default log files

[11] For details, go to SAM product documentation and navigate to: Administration / Create the private configuration

[12] For details, go to SAM product documentation and navigate to: Administration / Initialize the private configuration

[13] For details, go to SAM product documentation and navigate to: Administration / Import the public configuration

Planned certifications

The Entrust SAM's CC EAL4+ certification-conforming **CEN EN 419 241-2 – Protection Profile for QSCD Server Signing** is in progress, with estimated completion by the end of 2021. For more information, refer to the Entrust SAM webpage available on the website of the certification body.[14]

A QSCD operated with a CEN EN 419 241-2 CC EAL4+ certified SAM and a CEN EN 419 221-5 CC EAL 4+ certified HSM can only be considered as QSCD when duly operated by a qualified trust service provider in accordance with eIDAS Regulation (EU) 910/2014. For more information, consult Zentrum für sichere Informationstechnologie – Austria (A-SIT) at a-sit.at.

NOTE: The current estimated date of Common Criteria certification for the Entrust SAM is subject to change until the evaluation process carried out by the accredited laboratory is completed and approved by the certification body.

Product licensing

One-server deployments

Unless otherwise specified in your order(s), the Entrust SAM requires one license for each server where it is installed, or for each cryptographic module it will connect to, whichever is greater.

Clustered server deployments

Unless otherwise specified in your order(s), the Entrust SAM requires one license for each node (server) in the cluster, or each cryptographic module connected to the cluster, whichever is greater.

For example:

- A 2-node SAM cluster connected to a 2-node HSM cluster requires 2 SAM licenses
- A 3-node SAM cluster connected to a 2-node HSM cluster requires 3 SAM licenses
- A 3-node SAM cluster connected to a 4-node HSM cluster requires 4 SAM licenses
- A 2-node SAM cluster connected to a 5-node HSM cluster requires 5 SAM licenses

Approved cryptographic modules are:

- Entrust nShield Solo XC: Basic, Mid, and High
- Entrust nShield Connect XC: Basic, Mid, and High

The Entrust SAM license does not include the cryptographic module(s) or any other type of hardware(s) where the software is installed. It must be purchased separately. The Entrust SAM support and maintenance services are subject to additional contracts, purchased separately.

[14] The Entrust SAM was originally submitted as "TrustedX SAM" for the certification process. The name was changed to Entrust SAM shortly after submission, but the URL and description on the auditor's page do not reflect these changes yet.

Bibliography

Reference	Referenced document
[EU eIDAS Regulation]	REGULATION (EU) No 910/2014 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC
[Implementing Decision EU 2016/650]	Commission Implementing Decision (EU) 2016/650 of 25 April 2016 laying down standards for the security assessment of qualified signature and seal creation devices pursuant to Articles 30(3) and 39(2) of Regulation (EU) No 910/2014 of the European Parliament and of the Council on electronic identification and trust services for electronic transactions in the internal market (Text with EEA relevance)
[CEN EN 419 221-5]	CEN EN 419 221-5 Protection Profiles for TSP Cryptographic Modules; Part 5 - Cryptographic Module for Trust Services
[CEN EN 419 241-1]	CEN EN 419 241-1 Trustworthy Systems Supporting Server Signing; Part 1: General System Security Requirements
[CEN EN 419 241-2]	CEN EN 419 241-2 Trustworthy Systems Supporting Server Signing; Part 2: Protection Profile for QSCD for Server Signing
[ETSI TS 119 432]	Protocols for Remote Digital Signature Creation and Cloud Signature Consortium Remote Signing Protocol v1.0
[Cloud Signature Consortium]	CSC Remote Signing Protocol v0.1.7.9 and CSC Remote Signing Protocol v1
[ETSI TS 119 431-1]	Policy and security requirements for trust service providers; Part 1: TSP service components operating a remote QSCD / SCDev
[ETSI TS 119 431-2]	Policy and security requirements for trust service providers; Part 2: TSP service components supporting AdES digital signature creation
[ETSI EN 319 401]	Electronic Signatures and Infrastructures (ESI); General Policy Requirements for Trust Service Providers

Glossary of acronyms

CA: Certification Authority

CEN: European Committee for Standardization (French: Comité Européen de Normalisation)

CM: Cryptographic Module

CSC: Cloud Signature Consortium

DBMS: Database Management System

DTBS/R: Data To Be Signed/Representation

ETSI: European Telecommunications Standards Institute

JSON: JavaScript Object Notation

IdP/AS: Identity Provider/Authorization Server

QTSP: Qualified Trust Service Provider

QSCD: Qualified Signature/Seal Creation Device

RA: Registration Authority

SAD: Signature Activation Data

SAM: Signature Activation Module

SCA: Signature Creation Application

SCD: Signature Creation Data

SIEM: Security Information and Event Management

SSA: Server Signing Application

TSP: Trust Service Provider

TW4S: Trustworthy Systems Supporting Server Signing

For more information

888.690.2424
+1 952 933 1223
+44 (0) 118 953 3000
sales@entrust.com
entrust.com

ABOUT ENTRUST CORPORATION

Entrust keeps the world moving safely by enabling trusted identities, payments, and data protection. Today more than ever, people demand seamless, secure experiences, whether they're crossing borders, making a purchase, accessing e-government services, or logging into corporate networks. Entrust offers an unmatched breadth of digital security and credential issuance solutions at the very heart of all these interactions. With more than 2,500 colleagues, a network of global partners, and customers in over 150 countries, it's no wonder the world's most entrusted organizations trust us.

Learn more at
entrust.com

ENTRUST

Global Headquarters
1187 Park Place, Minneapolis, MN 55379
U.S. Toll-Free Phone: 888 690 2424
International Phone: +1 952 933 1223
info@entrust.com entrust.com/contact