Validation Server

Introduction

Swiss IT Security AG
Etzelmatt 3
5430 Wettingen
Schweiz

+41 848 088 088
www.sits.ch
info@sits.ch

1. Validation Server

2. Introduction

The Validation Server provides OCSP validation based on multiple sources. It consists of a server application handling the validation requests and an administration application for the configuration of the server application. Depending on the chosen deployment, both applications run on the same server or on different machines.

The Validation Server is a modularly built OCSP V1 (RFC 2560) compliant responder for the online validation of certificates. OCSP V1 compatible clients can send OCSP V1 requests to the Validation Server to verify the status of a certificate without having to use client-side CRLs (certificate revocation lists). The Validation Server fulfills all MUST criteria described in RFC 2560.

In addition to RFC 2560, the Validation Server also supports *The Lightweight Online Certificate Status Protocol (OCSP) Profile for High-Volume Environments* according to RFC 5019.

3. What is OCSP?

OCSP allows querying of certificate status information based on a unique CA identification and the serial number of the certificate to be checked at the current time. The certificate itself is not transmitted to the server.

Any number of certificate statuses can be requested in a request. The certificate path is formed on the client.

The answer is a kind of signed mini CRL which contains the status of the requested certificates.

OCSP knows three different states

- **UNKNOWN**: The CA is unknown or the CA has no valid revocation information of the certificate
- **GOOD**: The certificate is not revoked
- **REVOKED**: The certificate is revoked. The information about the reason and the time of the relocation will be returned

4. High-Level Architecture

The high-level architecture of the Validation Server is as follows:

The Validation Server Responder can also be operated as a standalone server. See Deployment for the different deployment possibilities.

4.1. Admin

The control (start/stop) and the configuration of the Validation Server takes place via a web interface. The administrator only needs a standard web browser and a client certificate.

Users can be assigned to groups to which appropriate administration rights are assigned via an access control management.

The database password can be stored locally encrypted via the Administration Service, so that the Validation Server can also be started automatically (unattended).

Trusted root certificates can be installed via the web interface so that after the initial setup the entire configuration is realized via the web interface. Configurations can be exported and imported.

4.2. Web Server/Servlet Engine

All common protocols for certificate validation (OCSP, SCVP, OCSPv2) use the HTTP protocol to transport the messages. As a basis for the Validation Server, a web server with Servlet Engine can therefore be used which takes over the connection handling. Whether a combination of web server/servlet engine or just a servlet engine is used does not matter from the point of view of the Validation Server, but can have an influence on the performance of the system.

Alternatively, the Validation Server can also be operated as a standalone server. A built-in, minimal HTTP server is used in this case.

4.3. Validation Server Responder

The responder is designed as a web application and as a standalone application. The detailed architecture of the responder is described in Responder.

4.4. Validation Server Administration

The administration of the Validation Server is run as a web application. The web application is compatible with the common J2EE application servers.

An application server (Tomcat) is supplied if no application server is available.

4.5. JNDI PKCS #11

To control the optional Hardware Security Module (HSM) from Java, a Java ↔ PKCS #11 Bridge is included.

4.6. Hardware Security Module (HSM)

The HSM is used to accelerate cryptographic operations and securely store keys and is controlled via the PKCS #11 V2.01. The key to sign the OCSP responses is generated in the HSM and the signing with the key is done on the HSM itself.

If no HSM is used, the key is placed in a PKCS #12 file protected with a randomly generated, strong password.

4.7. Why this Architecture?

Below are the main reasons for the choice of architecture:

Web Server/Servlet Architecture

- Proven technology, which is also used for large installations.
- There are a variety of technologies for the scaling and load distribution of web applications.
- The administration and configuration is performed via a web interface and does not require direct access to the server.
- The deployment of web applications is easy.
- The implementation is not dependent on a particular operating system. (Exceptional native code for the HSM connection).
- Can co-exist with other web applications and does not require a dedicated web server.

Java

- Stable framework, no memory leaks in server applications.
- Exception handling allows a clean treatment of errors which is important for a non-stop operation.
- No risk of buffer overflows or other attacks.
- Through consistent object-oriented modeling, the server can be easily extended at any time.
- High computing power is primarily required for the cryptographic operations, but these are carried out by the HSM or the WebServer (SSL)

The advantages of this architecture over a realization as a native application outweigh the drawbacks by far.

5. Administration

5.1. Web interface

The administration takes place completely via a web interface which is secured via TLS with client authentication.

5.2. Rights Management

Administrators are identified via the DN of their certificate. The users are assigned to groups that define the rights of the users. The rights can be assigned in detail per group:

Users of a group with only read permissions, for example, can view the configuration and the log files via the web interface but cannot make any changes to the configuration itself.

The rights management allows a clear separation between security administration, operation and audit.

6. Responder

The Validation Server has a highly modular and expandable internal structure:

Validation requests are processed as follows:

1. The request is accepted by the appropriate handler
2. The request is checked for validity and format
3. The information about the certificate to be verified is extracted from the request
4. Optionally, request extensions can be handled
5. The revocation information for the desired certificate is determined. The revocation information of the issuers is also determined.
6. The policies for the certificates in the path are assigned and checked*
7. Optionally, response extensions are added
8. The response is created
9. The response is signed
10. The response is returned to the client

The following chapters explain the technical specifications of the basic configuration of the responder.

*Only with Option Policy Engine and if the validation protocol used transmits the complete certificate to be checked. Not for OCSPv1.

6.1. OCSP Responder

The basic configuration includes an OCSP responder according to RFC 2560. Signed requests are not supported because they are only of interest for the provision of paid validation services in order to restrict the user group.

Requests according to RFC 5019 (The Lightweight Online Certificate Status Protocol (OCSP) Profile for High-Volume Environments) are supported.

6.2. Unlimited Number of Trusted CAs

The number of supported CAs is not limited. The source of the revocation information is independently defined per CA. Each CA can be assigned any number of sources for the revocation information.

CA hierarchies are supported.

The maximum number of CAs and CRL sources is defined by the available memory of the server.

6.3. CRL Support

CRLs according to RFC 3280 are supported. Except for delta CRLs and unknown critical extensions, all CRL formats defined according to X.509 are supported.

CRLs can be obtained in binary format (DER) using the following protocols:

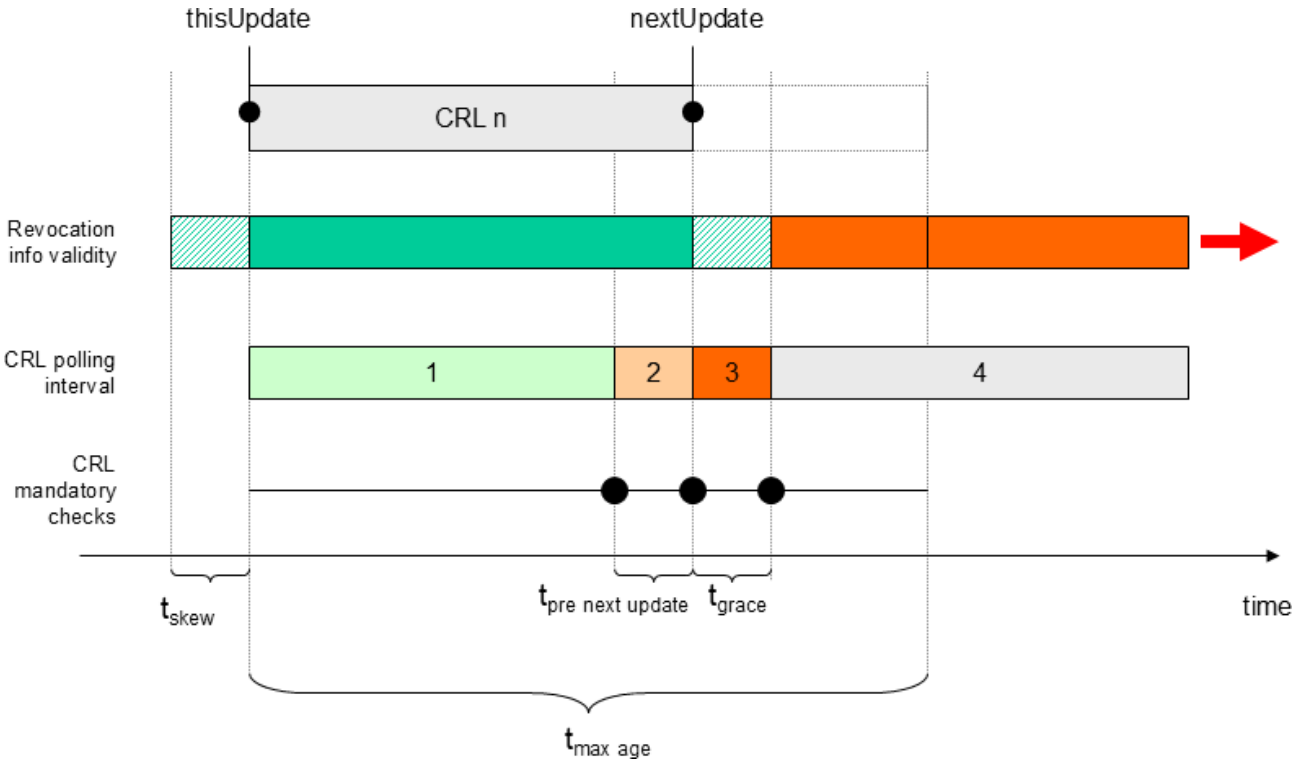
- LDAPv2 and LDAPv3 (without authentication)
- HTTP (also via Proxy, Basic Authentication is supported for server and proxy)
- HTTPS (also via Proxy, Basic Authentication is supported for server and proxy)
- Directly from local files

When using CRL Distribution Points, the CRLs can be dynamically defined by wildcard, i.e. not all CRLs need to be configured manually. An OCSP request does not contain the necessary information to determine the CRL, i.e. the CRLDP extension.

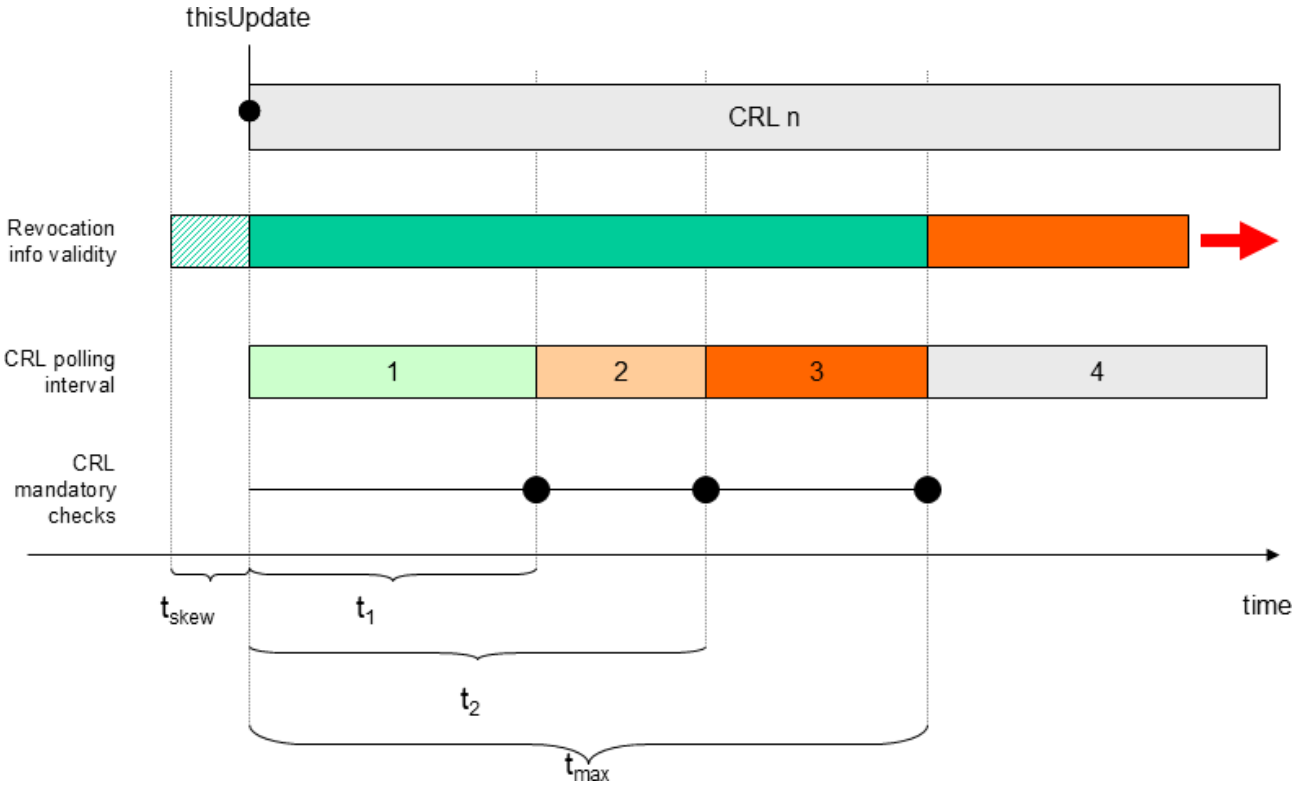
The intervals for the CRL update are freely configurable. Before the CRLs expire, they are automatically updated regardless of the configured interval. The update is done in the background and does not block the responder.

The following graphics provide an overview of the configuration options of the update intervals and times:

Example of configurable times and intervals if nextUpdate is present in the CRL



Example of configurable times and intervals if nextUpdate is not present in the CRL



The information supplied in the OCSP response is based on the underlying revocation information (*thisUpdate* is adopted by the CRL) and the next update of this information based on the configuration (*nextUpdate*).

The configurable times and intervals are created as templates that are assigned to the CRL sources. This means that uniform standards can be enforced, for example, for internal as well as external CAs.

6.4. Local CRL Cache

All current and valid CRLs are stored locally in a file-based cache to be immediately available for validation requests, independent of other components in the event of a restart.

During a system restart, the most current CRLs are downloaded immediately, so the cache is only used to bridge the time until the most recent CRLs are available.

6.5. Determine Revocation Status via OCSP

Instead of CRLs as the source of revocation information, OCSP can be also used to determine revocation status when the respective CAs have been appropriately configured.

6.6. Query CA Database Whether Certificate has been Issued

Optionally, a database can be configured for each CA and queried whether the verified serial number exists in the database.

6.7. Signature Key in Hardware

When using an HSM, the key with which the responses are signed is generated and used in the HSM. When operating without an HSM (e.g. in a test environment), the key can also be generated and used in software.

It should be noted that the certificate issued for the Responder has the necessary extensions and key usages for an OCSP responder.

The admin interface allows the creation of certificate signing requests according to PKCS #10 as well as the issuing of self-signed certificates.

6.8. Configuration

The configuration is cryptographically protected (MAC) and is created and modified via the administration web application. The DNs of the authorized administrators are initially set locally and can later be modified via the admin interface.

The communication of the admin client with the responder is encrypted and authenticated via standard TLS.

6.9. Log

A log with all relevant events is created. The events are classified (info, errors, etc.) so that an automatic evaluation is possible for monitoring.

Any number of log files can be configured and assigned to individual elements (CAs, CRL source, etc.). Optionally, the log files are renamed daily to allow for archiving.

The admin interface allows you to view the logs and search for specific entries (text, log level).

7. Emergency Measures

In emergencies, the Validation Server can override the revocation status or expiry of a certificate (i.e. considering it still valid) to enable the clients to continue operating.

7.1. Accepting Revocation Information

CRLs can be configured to be considered valid despite their expiration. This makes it possible to ensure operation despite failure of a CRL source (directory, CA issuing CRLs). This must be configured by the administrator per CRL source.

The use of templates for the configuration of times and intervals allows to create a template for emergencies and to assign it to the sources if necessary.

7.2. Overriding Revocation Information

The following measures allow a manual override of revocation information:

- Per CA, a blacklist of certificate serial numbers is kept which are to be considered revoked. The administrator can add new serial numbers to this list and remove existing serial numbers. The status of the listed certificates is returned as revoked with cause *certificateHold* and hold instruction *id-holdinstruction-reject*.
- Per CA, a whitelist with certificate serial numbers is kept which are to be considered valid. The administrator can add new serial numbers to this list or remove existing serial numbers. Thus, in the event of an unintentional revocation, the operation of a critical component can be maintained until a new certificate is issued. This feature can be removed if there are security concerns. The status of the listed certificates is returned as revoked with cause *certificateHold* and hold instruction *id-holdinstruction-reject*.

- Trusted root certificates can be marked as revoked by an administrator, which automatically sets the status for all certificates issued by this CA to revoked with the current date and cause *cACompromise*. The revocation of the CA in the Validation Server cannot be reversed.

8. Performance

The performance of the Validation Server depends on a variety of factors:

- Server hardware and software
- Web server
- Servlet engine
- OCSP features (e.g. without NONCE, the responder can reuse responses at the expense of security)
- Use of an HSM (hardware security module)
- Size of signature key
- Configured log levels
- Availability of CRLs
- Configuration of external OCSP servers for status query

The actual performance can therefore only be determined in the direct target infrastructure with the desired setting.

Tests with a production infrastructure and a LunaSA 4 as HSM have shown that with an instance of the Validation Server, more than 1,000 OCSP requests per second could be answered, with each request querying the status of two certificates. This results in a capacity of 3,600,000 OCSP queries per hour or 86.4 million queries per day.*

By using several machines, the performance can in principle be increased as desired, especially since a simple load balancing is possible as with normal web servers.

*Non-binding indication (limit for a LunaSA 4 is 1,200 signatures per second)

9. Operating Systems

Except for the PKCS #11 connection, the Validation Server is completely implemented in Java and therefore operating system independent. The Java Runtime environment 1.6.x is required, the Servlet Engine must support the Servlet Specification 2.5.

For the following platforms a PKCS #11 connection is delivered:

- Solaris 9, 10 (Sparc, x86)
- Windows 2003, 2008, 2008R2, 2012, 2012R2 (x86, x64)
- Linux systems with kernel 2.4 or higher (x86, amd64, x64)

Porting to other operating systems is possible.

10. Options

The modular design allows for a variety of options. To meet the minimum requirements for OCSPv1, the following options are not required. However, for the long-term securing of the investment, it is important that the Validation Server can be expanded at any time.

10.1. Support of Policies

After the creation of the certification path, it is checked according to the PKIX policy (e.g. path length restrictions, keys usage). Custom policies can optionally be implemented (e.g. the evaluation of Certificate Policies Extensions).

10.2. OCSPv2, SCVP, XKMS Support

OCSPv2 and SCVP responders (both protocols are not yet definitively adopted) or an XKMS responder for the validation allow the use of clients that support these protocols.

10.3. CRL Archiving

CRL archiving allows the CRLs to be stored in a structured form. These CRLs can then also be used to determine the certificate status for times in the past, which can be of particular interest to understand a certificate suspension.

10.4. Alerting

Alarm modules can be implemented, which, for example, in the event of a fault (CRL cannot be picked up etc.) send an email or initiate external applications.

11. Deployment

11.1. In the Form of a Web Application

The validation responder can run directly in a servlet engine along with the administration application.

11.2. As a Standalone Server

The validation responder can also be used in standalone operation with the help of the built-in minimal HTTP server. However, a servlet engine is still required for the administration application.