Public Key Infrastructures

Andreas Hülsing
X.509 Revocation
Certificate revocation

- Abortive ending of the binding between
  - subject and key (public key certificate)
  OR
  - subject and attributes (attribute certificate)

- The revocation is initiated by
  - the subject
  OR
  - the issuer

- Typical frequency (assumption):
  - 10% of the issued certificates will be revoked  
    (See: “Selecting Revocation Solutions for PKI” by Årnes, Just, Knapskog, Lloyd and Meijer)
Revocation reasons (in X.509)

CRLReason ::= ENUMERATED {
    unspecified (0),
    keyCompromise (1),
    cACompromise (2),
    affiliationChanged (3),
    superseded (4),
    cessationOfOperation (5),
    certificateHold (6),
    removeFromCRL (8),
    privilegeWithdrawn (9),
    aACompromise (10)
} -- value 7 is not used
Revocation requirements

- Revocation information is publicly available
- Authenticity can be checked by everyone
- Revoked certificate is unambiguously identified
- Information about the time of the revocation

Optional:
- revocation reason
- temporary revocation (on hold / suspended)
Revocation mechanisms

• Dedicated infrastructure for dissemination of authentic revocation information

• Certificate Revocation Lists (CRL)
  • various types (e.g. Full, Delta, etc.)

• Online Certificate Status Protocol (OCSP)

• Certificate Revocation System (CRS)

• Certificate Revocation Trees (CRT)

• Alternative: very short certificate validity period, therefore no revocation (e.g. nPA).
Certificate Revocation List (CRL)

- Signed list of revoked certificates
- “Blacklist”, i.e. no positive information about the validity of a certificate
- Standard mechanism (e.g. X.509)
- Wide-spread mechanism
### Structure of a CRL (X.509)

<table>
<thead>
<tr>
<th>Version</th>
<th>Signature ID</th>
<th>Issuer</th>
<th>This Update</th>
<th>Next Update</th>
<th>List of revoked certificates</th>
<th>CRL-extensions (version 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>currently version 2</td>
<td>signature algorithm</td>
<td>issuer name</td>
<td>issuance time of the CRL</td>
<td>date of the next update</td>
<td>sequence of CRL entries</td>
<td>extensions for the whole CRL</td>
</tr>
</tbody>
</table>

**Signature**
CRLEntry (X.509)

- userCertificate
- revocationDate
- CRLEntry-extensions (version 2)

- serial number of the certificate
- revocation time for this certificate
- extensions regarding this CRL entry only
CertificateList ::= SEQUENCE {
  tbsCertList TBSCertList,
  signatureAlgorithm AlgorithmIdentifier,
  signatureValue BIT STRING }

ASN.1-Definition of the CRL
TBSCertList ::= SEQUENCE {
    version            Version OPTIONAL -- if present, MUST be v2
    signature          AlgorithmIdentifier,
    issuer             Name,
    thisUpdate         Time,
    nextUpdate         Time   OPTIONAL,
    revokedCertificates SEQUENCE OF SEQUENCE {
        userCertificate CertificateSerialNumber,
        revocationDate    Time,
        crlEntryExtensions Extensions OPTIONAL -- if present, MUST be v2
    } OPTIONAL,
    crlExtensions [0] EXPLICIT Extensions OPTIONAL -- if present, MUST be v2
}
Example

Certificate Revocation List (CRL):

Version 2 (0x1)
Signature Algorithm: sha1WithRSAEncryption
Issuer: /C=DE/ST=Hessen/L=Darmstadt/O=Technische Universitaet Darmstadt/CN=TUD-CCA/emailAddress=tud-cca@hrz.tu-darmstadt.de
Last Update: Apr 26 14:05:56 2007 GMT
Next Update: May 26 14:05:56 2007 GMT
CRL extensions:
  X509v3 CRL Number: 278
  X509v3 Authority Key Identifier:

Revoked Certificates:
  Serial Number: 4670
    Revocation Date: Jul 07 11:59:16 2006 GMT
  Serial Number: 207A
    Revocation Date: Aug 22 12:13:05 2006 GMT
  Serial Number: 09
    Revocation Date: Aug 29 12:07:51 2005 GMT

Signature Algorithm: sha1WithRSAEncryption
CRL: Publicly available
### Certificate Revocation List Information

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>V2</td>
</tr>
<tr>
<td>Issuer</td>
<td>RBG CA, FB Informatik, TU Darmst...</td>
</tr>
<tr>
<td>Signature algorithm</td>
<td>sha1RSA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial number</td>
<td>05 35</td>
</tr>
<tr>
<td>Revocation date</td>
<td>Montag, 22. November 2004 15:58:04</td>
</tr>
<tr>
<td>CRL Reason Code</td>
<td>Superseded (4)</td>
</tr>
</tbody>
</table>

### Revoked certificates:

<table>
<thead>
<tr>
<th>Serial number</th>
<th>Revocation date</th>
</tr>
</thead>
<tbody>
<tr>
<td>08 16</td>
<td>Donnerstag, 15. Juli 20...</td>
</tr>
<tr>
<td>0b f7</td>
<td>Mittwoch, 17. November...</td>
</tr>
<tr>
<td>0d bd</td>
<td>Montag, 9. Mai 2005 09...</td>
</tr>
<tr>
<td>05 35</td>
<td>Montag, 22. November...</td>
</tr>
<tr>
<td>0d De</td>
<td>Dienstag, 5. April 2005 ...</td>
</tr>
<tr>
<td>00 22</td>
<td>Ersttag, 23. Februar 20...</td>
</tr>
</tbody>
</table>
CRL: Authentic

Certificate Revocation List Information

Field | Value
--- | ---
Version | V2
Issuer | BAK Test CRL Signer 2:PN, Bundes
Effective date | Donnerstag, 8. März 2007 16:00:4
Next update | Mittwoch, 6. Juni 2007 16:00:49
Signature algorithm | sha1RSA
CRL Number | 21
Authority Key Identifier | KeyID=f7 bb bb 61 68 82 c9 8f f9
Issuing Distribution Point | , Only Contains User Certs=No, 0,
CRL: Certificate identification

OID 2.5.29.29 = Certificate Issuer, not CRL issuer => see indirect CRLs
CRL: Revocation date

<table>
<thead>
<tr>
<th>Serial number</th>
<th>Revocation date</th>
</tr>
</thead>
<tbody>
<tr>
<td>08 16</td>
<td>Donnerstag, 15. Juli 2005</td>
</tr>
<tr>
<td>0b f7</td>
<td>Mittwoch, 17. November 2005</td>
</tr>
<tr>
<td>0d bd</td>
<td>Montag, 9. Mai 2005 09:20:50</td>
</tr>
<tr>
<td>05 35</td>
<td>Montag, 22. November 2005</td>
</tr>
<tr>
<td>0d 0e</td>
<td>Dienstag, 5. April 2005</td>
</tr>
<tr>
<td>00 22</td>
<td>4dicht, 27. Februar 2005</td>
</tr>
</tbody>
</table>

**Revocation entry**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial number</td>
<td>0d bd</td>
</tr>
<tr>
<td>Revocation date</td>
<td>Montag, 9. Mai 2005 09:20:50</td>
</tr>
<tr>
<td>CRL Reason Code</td>
<td>Affiliation Changed (3)</td>
</tr>
<tr>
<td>Value</td>
<td>Montag, 9. Mai 2005 09:20:50</td>
</tr>
</tbody>
</table>
CRL: Revocation reason

Certificate Revocation List

Revoked certificates:

<table>
<thead>
<tr>
<th>Serial number</th>
<th>Revocation date</th>
</tr>
</thead>
<tbody>
<tr>
<td>08 16</td>
<td>Donnerstag, 15. Juli 2020</td>
</tr>
<tr>
<td>0b f7</td>
<td>Mittwoch, 17. November 2020</td>
</tr>
<tr>
<td>0d bd</td>
<td>Montag, 9. Mai 2005 09:20:50</td>
</tr>
<tr>
<td>06 35</td>
<td>Montag, 22. November 2005</td>
</tr>
<tr>
<td>0d 0e</td>
<td>Dienstag, 5. April 2005</td>
</tr>
<tr>
<td>09 22</td>
<td>Donnerstag, 27. Februar 2005</td>
</tr>
</tbody>
</table>

Revocation entry

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial number</td>
<td>0d bd</td>
</tr>
<tr>
<td>Revocation date</td>
<td>Montag, 9. Mai 2005 09:20:50</td>
</tr>
<tr>
<td>CRL Reason Code</td>
<td>Affiliation Changed (3)</td>
</tr>
</tbody>
</table>

Value:

Affiliation Changed (3)
CRL Extensions

- Affect the CRL as a whole

or maybe

- Each single CRL entry (all of them)
CRL Extensions: AKI / IAN

Authority Key Identifier
Issuer Alternative Name

As in X.509 certificates
The CRL number is a non-critical CRL extension which conveys a monotonically increasing sequence number for a given CRL scope and CRL issuer. This extension allows users to easily determine when a particular CRL supersedes another CRL. CRL numbers also support the identification of complementary complete CRLs and delta CRLs. CRL issuers conforming to this profile MUST include this extension in all CRLs. If a CRL issuer generates delta CRLs in addition to complete CRLs for a given scope, the complete CRLs and delta CRLs MUST share one numbering sequence. If a delta CRL and a complete CRL that cover the same scope are issued at the same time, they MUST have the same CRL number and provide the same revocation information. That is, the combination of the delta CRL and an acceptable complete CRL MUST provide the same revocation information as the simultaneously issued complete CRL. If a CRL issuer generates two CRLs (two complete CRLs, two delta CRLs, or a complete CRL and a delta CRL) for the same scope at different times, the two CRLs MUST NOT have the same CRL number.

CRLNumber ::= INTEGER (0..MAX)
The issuing distribution point is a critical CRL extension that identifies the CRL distribution point and scope for a particular CRL, and it indicates whether the CRL covers revocation for end entity certificates only, CA certificates only, attribute certificates only, or a limited set of reason codes.
CRL Extension:
Issuing Distribution Point

issuingDistributionPoint ::= SEQUENCE {
  distributionPoint [0] DistributionPointName OPTIONAL,
  onlyContainsUserCerts [1] BOOLEAN DEFAULT FALSE,
  onlyContainsCACerts [2] BOOLEAN DEFAULT FALSE,
  onlySomeReasons [3] ReasonFlags OPTIONAL,
  indirectCRL [4] BOOLEAN DEFAULT FALSE,
  onlyContainsAttributeCerts [5] BOOLEAN DEFAULT FALSE }

TU/e
CRL Entry Extensions

• Affect the current CRL entry

and maybe

• Some following ones (but not necessarily all of them)
The reasonCode is a non-critical CRL entry extension that identifies the reason for the certificate revocation.

**CRLReason** ::= ENUMERATED {
  unspecified (0),
  keyCompromise (1),
  cACompromise (2),
  affiliationChanged (3),
  superseded (4),
  cessationOfOperation (5),
  certificateHold (6),
  -- value 7 is not used
  removeFromCRL (8),
  privilegeWithdrawn (9),
  aACompromise (10) }
The hold instruction code is a non-critical CRL entry extension that provides a registered instruction identifier which indicates the action to be taken after encountering a certificate that has been placed on hold.

holdInstructionCode ::= OBJECT IDENTIFIER

Standard codes:
• id-holdinstruction-none   OBJECT IDENTIFIER ::= {holdInstruction 1}
  • Equivalent to extension being absent …

• id-holdinstruction-callissuer   OBJECT IDENTIFIER ::= {holdInstruction 2}
  • Contact issuer or reject certificate

• id-holdinstruction-reject   OBJECT IDENTIFIER ::= {holdInstruction 3}
  • Reject certificate
The invalidity date is a non-critical CRL entry extension that provides the date on which it is known or suspected that the private key was compromised or that the certificate otherwise became invalid. This date may be earlier than the revocation date in the CRL entry, which is the date at which the CA processed the revocation. When a revocation is first posted by a CRL issuer in a CRL, the invalidity date may precede the date of issue of earlier CRLs, but the revocation date SHOULD NOT precede the date of issue of earlier CRLs. Whenever this information is available, CRL issuers are strongly encouraged to share it with CRL users.

\[\text{invalidityDate} ::= \text{GeneralizedTime (type)}\]
CRL Properties

- Can be used offline (CRL caching)
- Easy implementation
- Easy management
- High information content (extendable!)

- The CRL (Full CRL) contains information about all revoked certificates

⇒ Size increases monotonically

All information is transferred at the same time
- High load (peak) at “nextUpdate” time
- Long validity period ⇒ bad timeliness
- Short validity period ⇒ bad performance
Load at the *nextUpdate*

**Figure 1: Request Rate following CRL issuance**

Source: ( “Selecting Revocation Solutions for PKI” by Årnes, Just, Knapskog, Lloyd and Meijer)
Download of CRLs

• Most common
  • Web pages (HTTP)
    − http://www.telesec.de/pki/roots.html
  • LDAP

• Other possibilities
  • File transfer (FTP)
  • CRL Push Services (Broadcasts)
  • ...
### CRL Reference Table

<table>
<thead>
<tr>
<th>Type</th>
<th>CA Name</th>
<th>CRL URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>VTN Root CAs</td>
<td>VeriSign Class 1 PCA</td>
<td><a href="http://crl.verisign.com/pca1.1.1.crl">http://crl.verisign.com/pca1.1.1.crl</a></td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="http://crl.verisign.com/pca1.crl">http://crl.verisign.com/pca1.crl</a></td>
</tr>
<tr>
<td></td>
<td>VeriSign Class 2 PCA</td>
<td><a href="http://crl.verisign.com/pca2.1.1.crl">http://crl.verisign.com/pca2.1.1.crl</a></td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="http://crl.verisign.com/pca2.crl">http://crl.verisign.com/pca2.crl</a></td>
</tr>
<tr>
<td></td>
<td>VeriSign Class 3 PCA</td>
<td><a href="http://crl.verisign.com/pca3.1.1.crl">http://crl.verisign.com/pca3.1.1.crl</a></td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="http://crl.verisign.com/pca3.crl">http://crl.verisign.com/pca3.crl</a></td>
</tr>
<tr>
<td></td>
<td>VeriSign Class 4 PCA</td>
<td>N/A – This CA has not yet issued any Certificates.</td>
</tr>
<tr>
<td></td>
<td>VeriSign Class 1 PCA – G2</td>
<td><a href="http://crl.verisign.com/pca1-g2.crl">http://crl.verisign.com/pca1-g2.crl</a></td>
</tr>
<tr>
<td></td>
<td>VeriSign Class 2 PCA – G2</td>
<td><a href="http://crl.verisign.com/pca2-g2.crl">http://crl.verisign.com/pca2-g2.crl</a></td>
</tr>
<tr>
<td></td>
<td>VeriSign Class 3 PCA – G2</td>
<td><a href="http://crl.verisign.com/pca3-g2.crl">http://crl.verisign.com/pca3-g2.crl</a></td>
</tr>
<tr>
<td></td>
<td>VeriSign Class 4 PCA – G2</td>
<td>N/A – This CA has not yet issued any Certificates.</td>
</tr>
<tr>
<td></td>
<td>VeriSign Class 1 PCA – G3</td>
<td><a href="http://crl.verisign.com/pca1-g3.crl">http://crl.verisign.com/pca1-g3.crl</a></td>
</tr>
</tbody>
</table>

**HTTP**

https://www.verisign.com/repository/crl.html
CRL Push Service

- The CRLs are delivered to registered clients
- Searching for a CRL is unnecessary
- Can only be used online.
- Suitable for e.g.:
  - Computer in intranet
  - Servers
- Covers only the certificates of a few PKIs.
Locating a CRL

• Using the policy:
  • The policy of the issuer names places where its CRLs are published.

• Using the certificate:
  • CRLDistributionPoints Extension (CRLDP)
  • Pointer to the places where the CRL will be located (usually as a URL)
  • Realized by the most typical applications.
CRLDistributionPoints extension

• CERTIFICATE extension

• Identifies how CRL information is obtained

• Non-critical

• Usage recommended
CRLDistributionPoints extension

CRLDistributionPoints ::= SEQUENCE SIZE (1..MAX) OF DistributionPoint

DistributionPoint ::= SEQUENCE {
  distributionPoint [0]   DistributionPointName OPTIONAL,
  reasons [1]            ReasonFlags OPTIONAL,
  cRLIssuer [2]          GeneralNames OPTIONAL }

DistributionPointName ::= CHOICE {
  fullName [0]           GeneralNames,
  nameRelativeToCRLIssuer [1] RelativeDistinguishedName }
Modifications

• Lean CRLs
  • Expired certificates are removed from the CRL
  • Expired certificates cannot be checked anymore

• (Details on the following slides):
  • Over-Issued CRLs
  • Delta CRLs
  • Indirect CRLs
  • Segmented CRLs
  • Redirect CRLs
Over-Issued CRLs

- CRLs are issued more frequently than nextUpdate requires
  - e.g. in a regular basis or with every certificate revocation
- Improved timeliness
- Frequency of the updates is chosen by the client
- Better load distribution
Over-Issued CRLs

CRL #123
CRL #124
CRL #125
CRL #126
CRL #127

time
Delta CRL

- Format like a “normal” CRL + Delta CRL Indicator Extension
- Associated to BaseCRL with the BaseCRLNumber
- Contains ALL changes since BaseCRL was issued
- Better network load, better scalability
- Slightly increases the administration costs (client and server)
- Can be combined with over-issued CRLs:
  - Together with each FullCRL also Deltas to the still valid CRLs are issued.
Delta CRL

CRL#14:
02
23
34

Δ-CRL
56

Δ-CRL
03
45
56

CRL#17:
02
03
23
34
45
56

BaseCRL

revocation certificate 56
revocation certificate 45
revocation certificate 03
CRL Extension: Delta CRL Indicator

The delta CRL indicator is a critical CRL extension that identifies a CRL as being a delta CRL.

The delta CRL indicator extension contains the single value of type `BaseCRLNumber`. The CRL number identifies the CRL, complete for a given scope, that was used as the starting point in the generation of this delta CRL. A conforming CRL issuer MUST publish the referenced base CRL as a complete CRL.

```
BaseCRLNumber ::= CRLNumber (type)
```
The freshest CRL extension identifies how delta CRL information for this complete CRL is obtained. The extension MUST be non-critical. This extension MUST NOT appear in delta CRLs.

FreshestCRL ::= CRLDistributionPoints (type)
Indirect CRLs

• The issuer of the CRL is not the issuer of the certificates

• Revocation can be delegated

• The revocation instance can operate online even if certificate issuer is offline

• Reflects the different security requirements on the keys that are used for signing certificates and the ones that are used for signing CRLs.
CRL Entry Extension: Certificate Issuer

If used by conforming CRL issuers, this extension MUST always be critical. If an implementation ignored this extension it could not correctly attribute CRL entries to certificates.

This CRL entry extension identifies the certificate issuer associated with an entry in an indirect CRL, that is, a CRL that has the indirectCRL indicator set in its issuing distribution point extension. If this extension is not present on the first entry in an indirect CRL, the certificate issuer defaults to the CRL issuer. On subsequent entries in an indirect CRL, if this extension is not present, the certificate issuer for the entry is the same as that for the preceding entry.

certificateIssuer ::= GeneralNames
Indirect CRL Example

Certificate Information

This certificate is intended for the following purpose(s):

- 1.3.6.1.4.1.24796.1.1
- All application policies

Refer to the certification authority’s statement for details.

Issued to: BÄK Test CRL Signer 2:PN

Issued by: 2R BÄK Test CA 1:PN

Valid from 30.05.2006 to 30.01.2011
CRL Segmentation

- The revocation information is separated into multiple CRLs (segmentation)

- Possibility 1: Multiple CRLDistributionPoints
  - Disjoint sets of certificates

- Possibility 2: CRLDistributionPoints points to a special Redirect CRL (see next slide)
  - Set of pairs (CRLDistributionPoint, Scope)
  - The scope describes a set of certificates
  - Advantage: can be changed later
Redirect CRLs

Certificate

CRL Distribution Point Extension

RCRL

Some scope

Other scope

description of the certificate e.g. based on the certificate serial number

CRL

List of revoked certificates

CRL

List of revoked certificates
Online Certificate Status Protocol (OCSP)

- Client-server architecture
- Clients
  - request the status of a certificate from an OCSP responder,
  - communicate online, in real time
  - can request the status of multiple certificates inside a single query
OCSP - Responder

- Responder
  - Provides signed answers
  - Has a certificate with the extension extendedKeyUsage = OCSPSigning

- Possible responses (basic version):
  - Unknown (nothing known about the certificate)
  - Revoked (certificate revoked)
  - Good (certificate not revoked)
    - Caution: Good means that the certificate is not revoked, but it may be expired or even not exist at all.

- The signed answer can be stored as a proof of validity at a given point in time.
OCSP Request – ASN.1

OCSPRequest ::= SEQUENCE {
  tbsRequest
  optionalSignature [0] EXPLICIT TBSRequest,
  Signature OPTIONAL }

TBSRequest ::= SEQUENCE {
  version [0] EXPLICIT Version DEFAULT v1,
  requestorName [1] EXPLICIT GeneralName OPTIONAL,
  requestList
  requestExtensions [2] EXPLICIT Extensions OPTIONAL }
OCSP Request – ASN.1 (cont.)

Request ::= SEQUENCE {
  reqCert
  singleRequestExtensions [0] EXPLICIT CertID,
  Extensions OPTIONAL
}

CertID ::= SEQUENCE {
  hashAlgorithm AlgorithmIdentifier,
  issuerNameHash OCTET STRING, -- Hash of Issuer's DN
  issuerKeyHash OCTET STRING, -- Hash of Issuer’s pub. key
  serialNumber CertificateSerialNumber
}
OCSP Request - Example

OCSP Request Data:
  Version: 1 (0x0)

Requestor List:
  Certificate ID:
    Hash Algorithm: sha1
    Issuer Name Hash:
      416AFF32B78A3CB75DECEA9EBDF8B26003683126
    Issuer Key Hash:
      C3CF75EAC011534513FE9765630069530296B964
    Serial Number: 31

Request Extensions:
  OCSP Nonce:
    02F2666CC11B571427268E0FEE158C3C
OCSP Response – ASN.1

BasicOCSPResponse ::= SEQUENCE {
  tbsResponseData ResponseData,
  signatureAlgorithm AlgorithmIdentifier,
  signature BIT STRING,
  certs [0] EXPLICIT SEQUENCE OF Certificate OPTIONAL }

ResponseData ::= SEQUENCE {
  version [0] EXPLICIT Version DEFAULT v1,
  responderID ResponderID,
  producedAt GeneralizedTime,
  responses SEQUENCE OF SingleResponse,
  responseExtensions [1] EXPLICIT Extensions OPTIONAL }
OCSP Response – ASN.1 (cont.)

ResponderID ::= CHOICE {
    byName [1] Name,
    byKey [2] KeyHash }

SingleResponse ::= SEQUENCE {
    certID CertID,  
certStatus CertStatus,  
thisUpdate GeneralizedTime,  
nextUpdate [0] EXPLICIT GeneralizedTime OPTIONAL,  
singleExtensions [1] EXPLICIT Extensions OPTIONAL}

CertStatus ::= CHOICE {
    good [0] IMPLICIT NULL,  
    revoked [1] IMPLICIT RevokedInfo,  
    unknown [2] IMPLICIT UnknownInfo }
OCSP Response Data:
  OCSP Response Status: successful (0x0)
  Response Type: Basic OCSP Response
  Version: 1 (0x0)
  Responder Id: C = DE, O = Bundesnetzagentur, CN = 10R-OCSP 3:PN
  Produced At: Apr 30 15:55:17 2007 GMT
  Responses:
  Certificate ID:
    Hash Algorithm: sha1
    Issuer Name Hash:
    416AFF32B78A3CB75DECEA9EBDF8B26003683126
    Issuer Key Hash: C3CF75EAC011534513FE9765630069530296B964
    Serial Number: 31
    Cert Status: good
....
Signed Response Acceptance Requirements

1. The certificate identified in a received response corresponds to that which was identified in the corresponding request

2. The signature on the response is valid

3. The identity of the signer matches the intended recipient of the request

4. The signer is currently authorized to sign the response

5. The time at which the status being indicated is known to be correct (thisUpdate is sufficiently recent)

6. When available, the time at or before which newer information will be available about the status of the certificate (nextUpdate) is greater than the current time.

OCSP Extensions

• Based on the extension model employed in X.509 version 3 certificates see [RFC5280].

• Support for all extensions is optional for both clients and responders.

• For each extension, the definition indicates its syntax, processing performed by the OCSP Responder, and any extensions which are to be included in the corresponding response.
OCSP Extensions: CRL Entry Extensions

- All CRL Entry extensions are supported by OCSP:
  - Reason Code
  - Hold Instruction Code
  - Invalidity Date
  - Certificate Issuer
OCSP Extension: Nonce

The nonce cryptographically binds a request and a response to prevent replay attacks.

The nonce is included as one of the requestExtensions in requests, while in responses it would be included as one of the responseExtensions. In both the request and the response, the nonce will be identified by the object identifier id-pkix-ocsp-nonce, while the extnValue is the value of the nonce.

\[
\text{id-pkix-ocsp-nonce OBJECT IDENTIFIER ::= \{ id-pkix-ocsp 2 }\]

OCSP Extension: CRL References

It may be desirable for the OCSP responder to indicate the CRL on which a revoked or onHold certificate is found. This can be useful where OCSP is used between repositories, and also as an auditing mechanism. The CRL may be specified by a URL (the URL at which the CRL is available), a number (CRL number) or a time (the time at which the relevant CRL was created).

CrlID ::= SEQUENCE {  
crlUrl [0] EXPLICIT IA5String OPTIONAL,  
crlNum [1] EXPLICIT INTEGER OPTIONAL,  
An OCSP client MAY wish to specify the kinds of response types it understands. To do so, it SHOULD use an extension with the OID id-pkix-ocsp-response, and the value AcceptableResponses. This extension is included as one of the requestExtensions in requests. The OIDs included in AcceptableResponses are the OIDs of the various response types this client can accept (e.g., id-pkix-ocsp-basic).

AcceptableResponses ::= SEQUENCE OF OBJECT IDENTIFIER
An OCSP responder MAY choose to retain revocation information beyond a certificate's expiration. The date obtained by subtracting this retention interval value from the producedAt time in a response is defined as the certificate's "archive cutoff" date. OCSP-enabled applications would use an OCSP archive cutoff date to contribute to a proof that a digital signature was (or was not) reliable on the date it was produced even if the certificate needed to validate the signature has long since expired. OCSP servers that provide support for such historical reference SHOULD include an archive cutoff date extension in responses.

ArchiveCutoff ::= GeneralizedTime
if archive cutoff < cert. expiration
OCSP response accurate
else
OCSP response inaccurate

Archive cutoff
OCSP response: revoked

Retention interval = 1 day

Day 1: cert. creation
Day 2: cert. revocation
Day 3: cert. expiration
Day 4: revok. info deletion

Archive cutoff
OCSP response: good
An OCSP server may be operated in a mode whereby the server receives a request and routes it to the OCSP server which is known to be authoritative for the identified certificate. The serviceLocator request extension is defined for this purpose.

ServiceLocator ::= SEQUENCE { 
  issuer Name, 
  locator AuthorityInfoAccessSyntax OPTIONAL } 

AuthorityInfoAccessSyntax ::= 
  SEQUENCE SIZE (1..MAX) OF AccessDescription 

AccessDescription ::= SEQUENCE { 
  accessMethod OBJECT IDENTIFIER, 
  accessLocation GeneralName } 

Values for these fields are obtained from certificate.
Authorized Responders

[Clients] MUST reject the response if the certificate required to validate the signature on the response fails to meet at least one of the following criteria: The OCSP signature certificate

1. matches a local configuration of OCSP signing authority for the certificate in question, or
2. is the certificate of the CA that issued the certificate in question, or
3. includes a value of id-ad-ocspSigning in an ExtendedKeyUsage extension and is issued by the CA that issued the certificate in question

RFC 2560  http://www.ietf.org/rfc/rfc2560.txt
OCSP server revocation

• Problem: is the certificate of the OCSP valid?

• An approach:
  • No revocation for the certificates of OCSP-responders.
    − Special extension ocsp nocheck
    − Short validity period

• Use other methods:
  • e.g. CRL
Certificate extension: Authority Information Access

- Certificate extension, MUST be non-critical

- Indicates how to access CA information and services for the issuer of the certificate in which the extension appears. Information and services may include online validation services and CA policy data.

- This extension does not contain CRL location information

- Two standard access methods defined in RFC 5280
  - calissuers (alternative certificates of the issuer)
  - ocsp (location of the corresponding OCSP responder for validating the issuer’s certificate)
OCSP Stapling


- Chapter 8 defines *Certificate Status Request* Extension for TLS

- During the TLS handshake, servers may return a suitable certificate status response along with their certificate.
  
  - Servers can cache OCSP responses and reuse them (until nextUpdate time)
  - No additional OCSP request by the client required
  - May reduce load for OCSP servers
Honest Achmed

- https://bugzilla.mozilla.org/show_bug.cgi?id=647959