Public Key Infrastructures

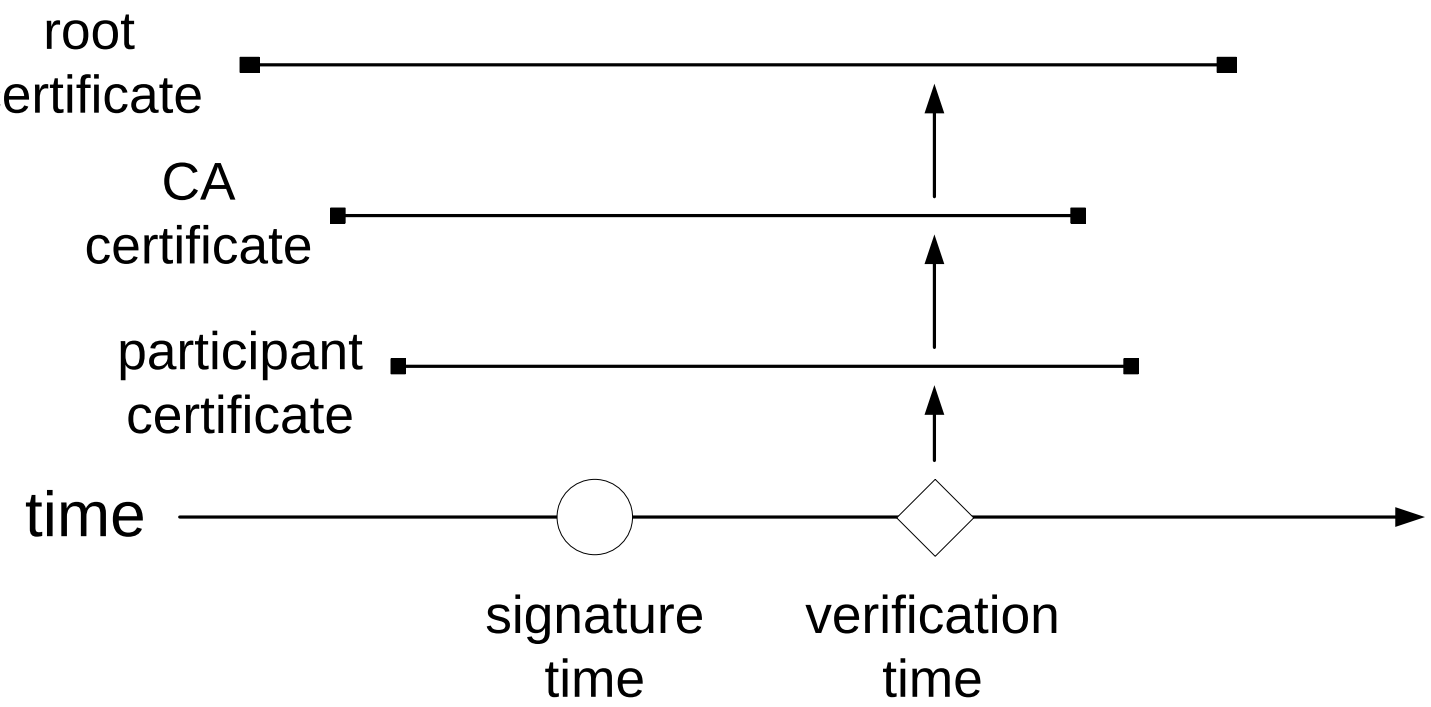
Andreas Hülsing

TU/e Technische Universiteit Eindhoven University of Technology

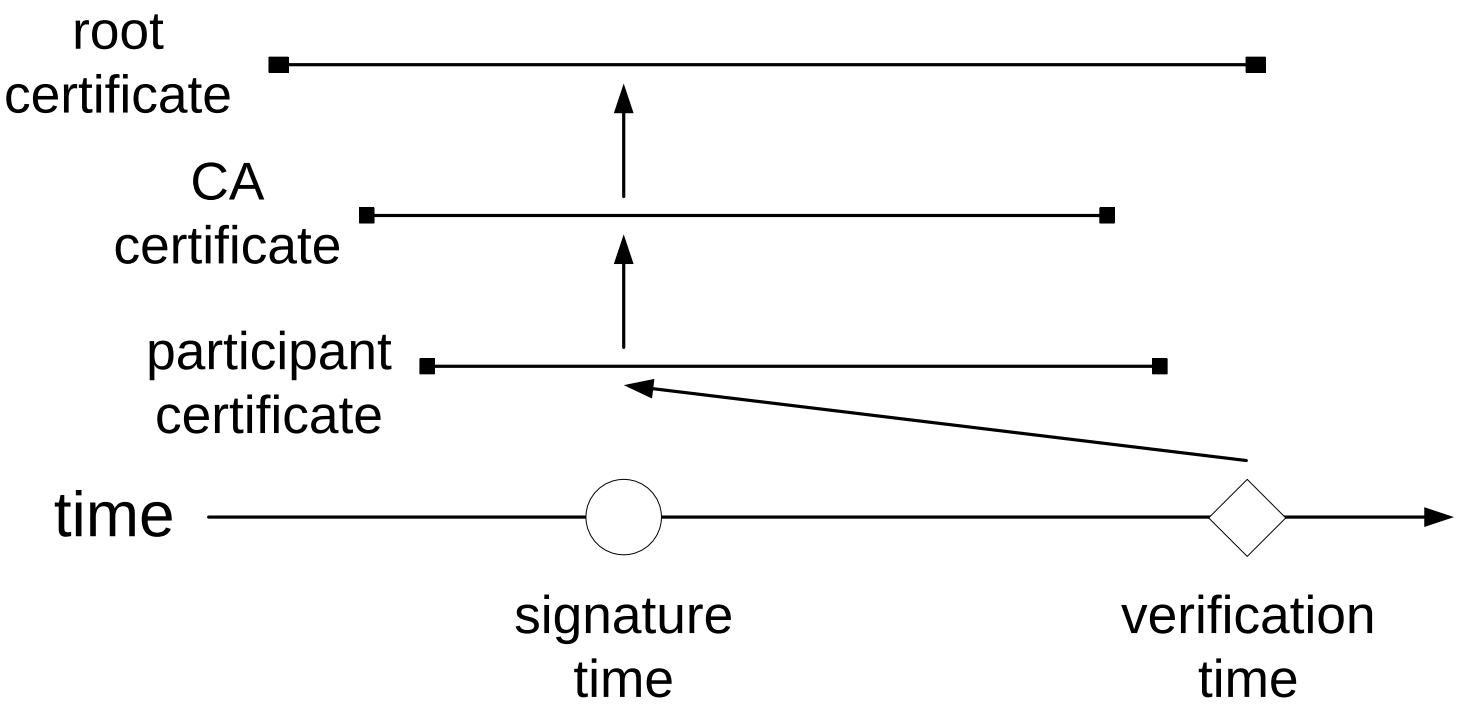
Where innovation starts

Certification Path Validation

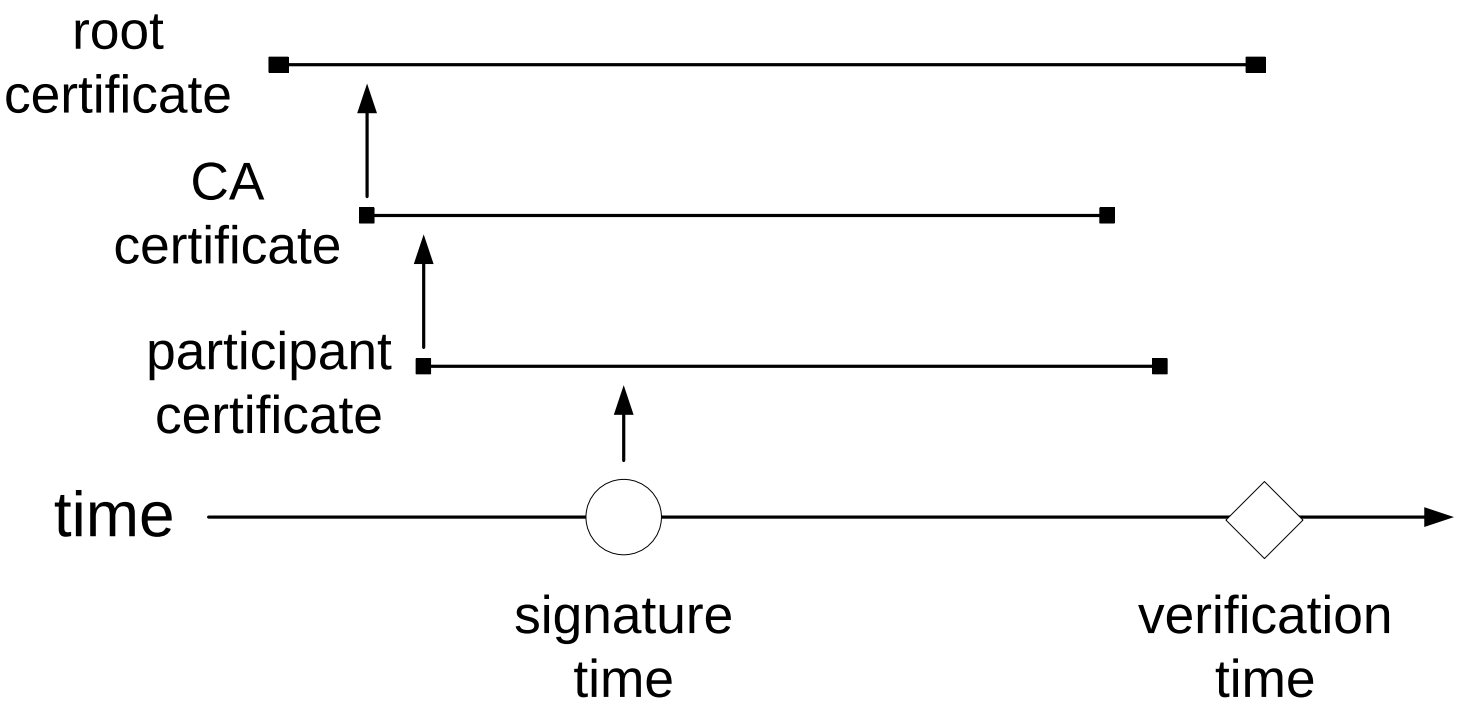
Shell model

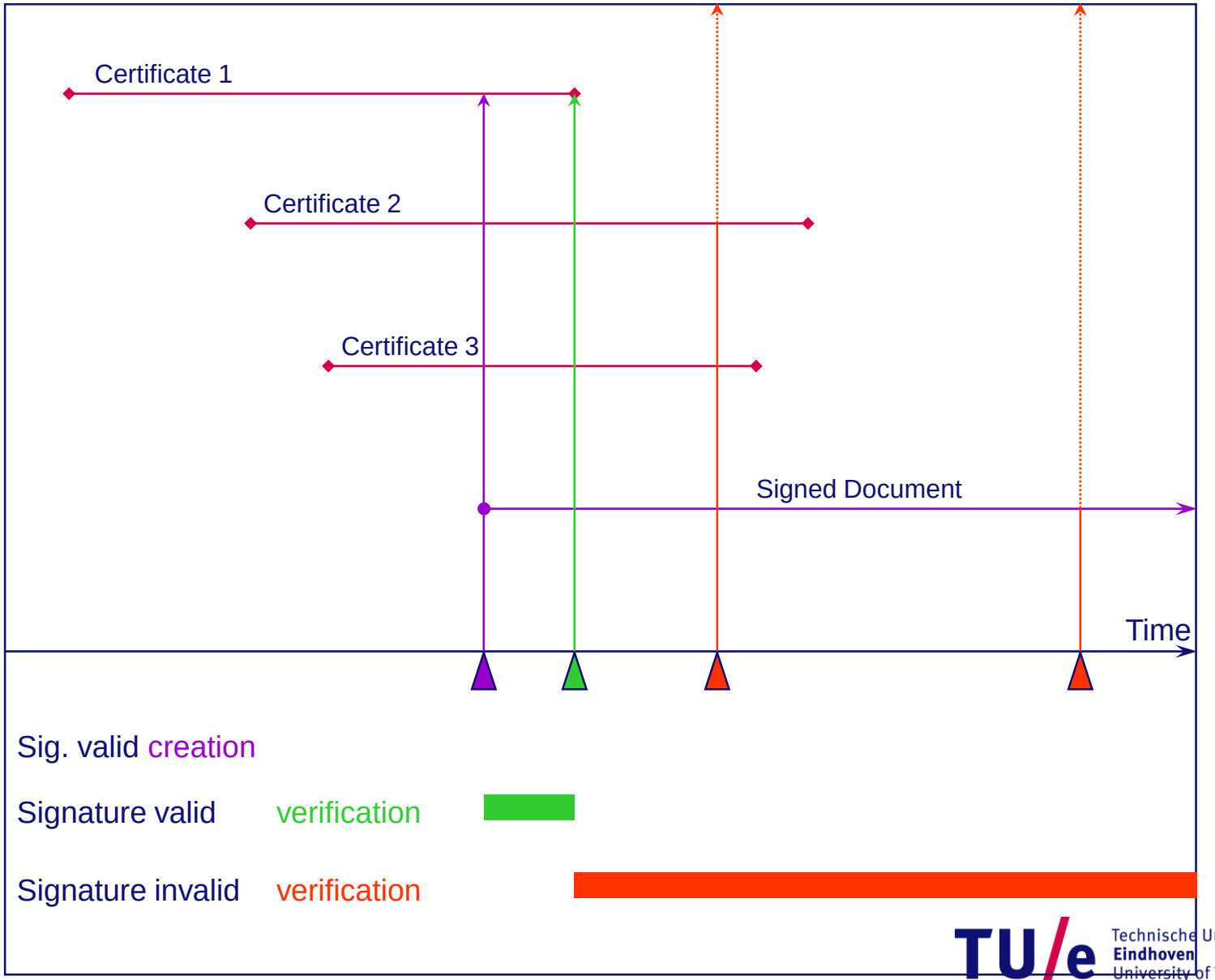

root
certificate
CA
certificate
participant
certificate
time
signature
time
verification
time

Modified or hybrid model



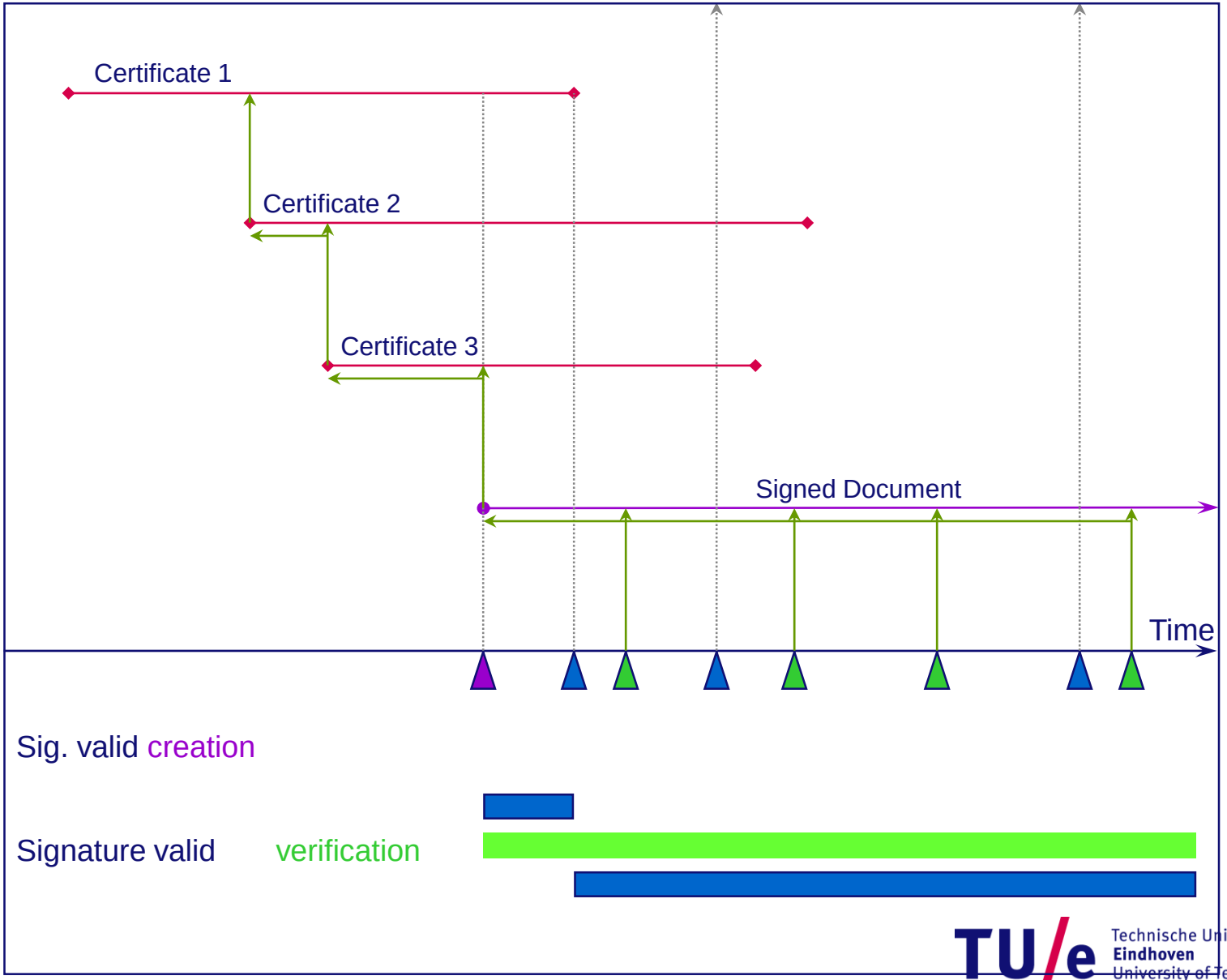
Chain model



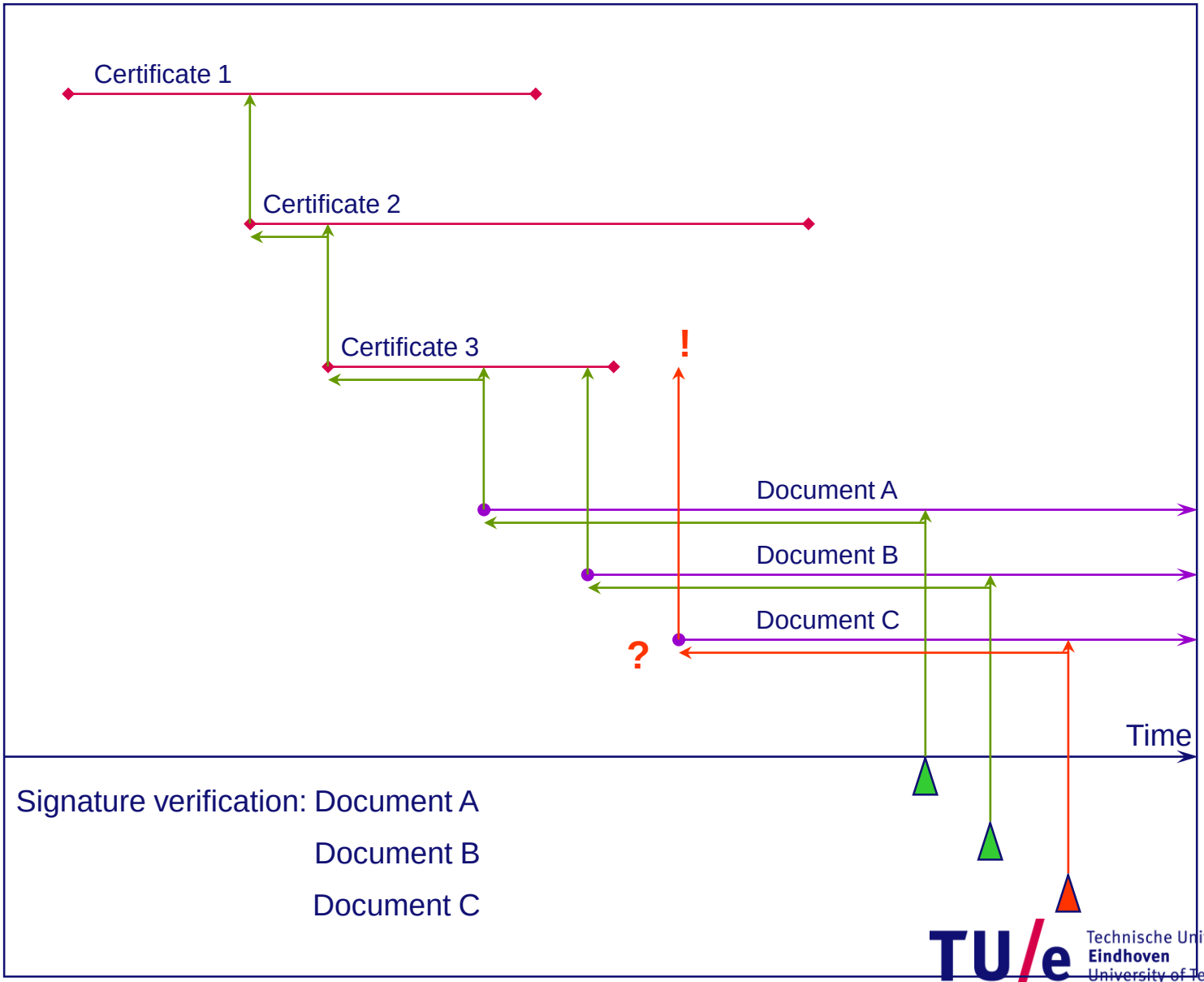

Certificate 1
Certificate 2
Certificate 3
Signed Document
Time
Sig. valid creation
Signature valid verification
Signature invalid verification

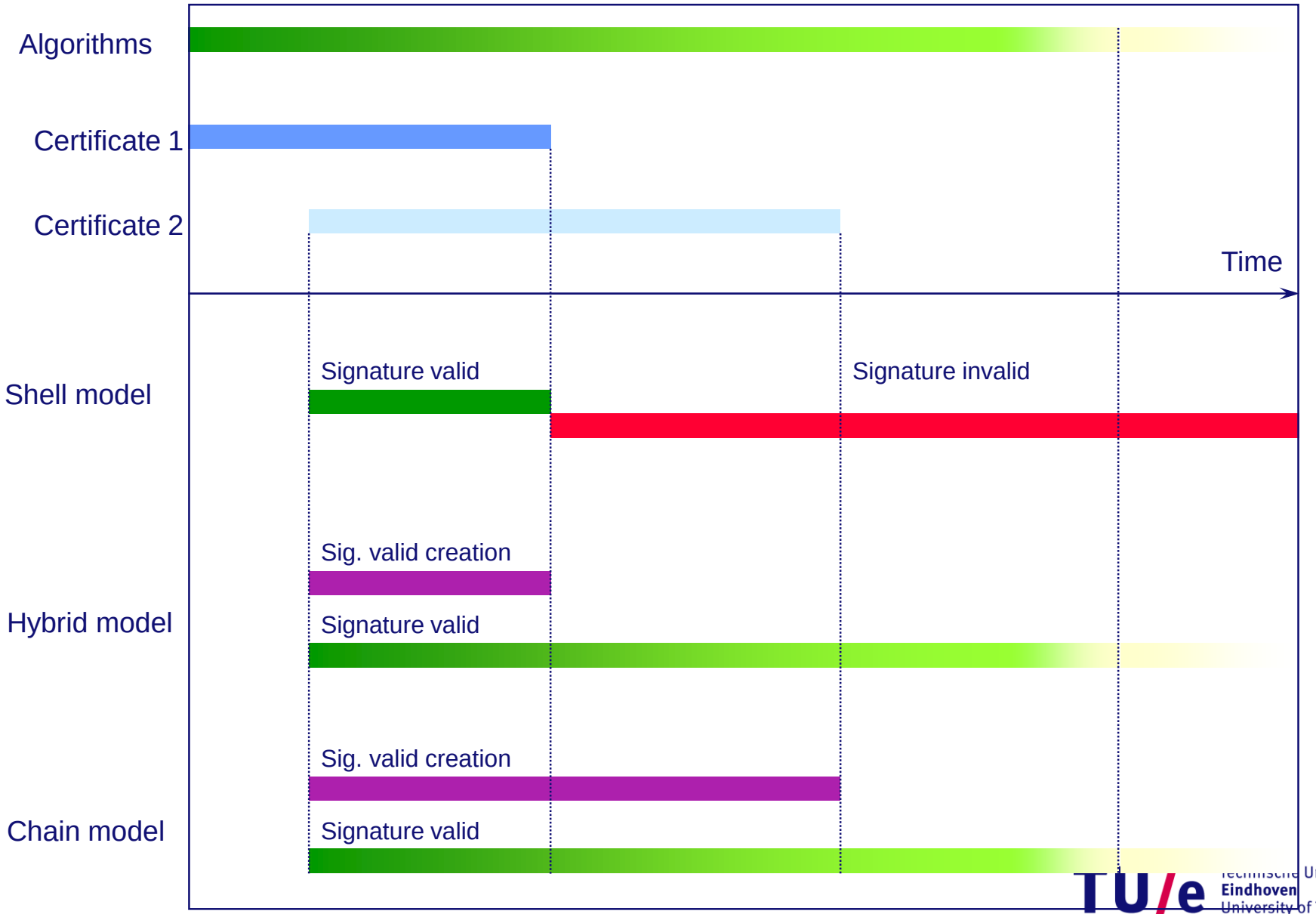
Shell model

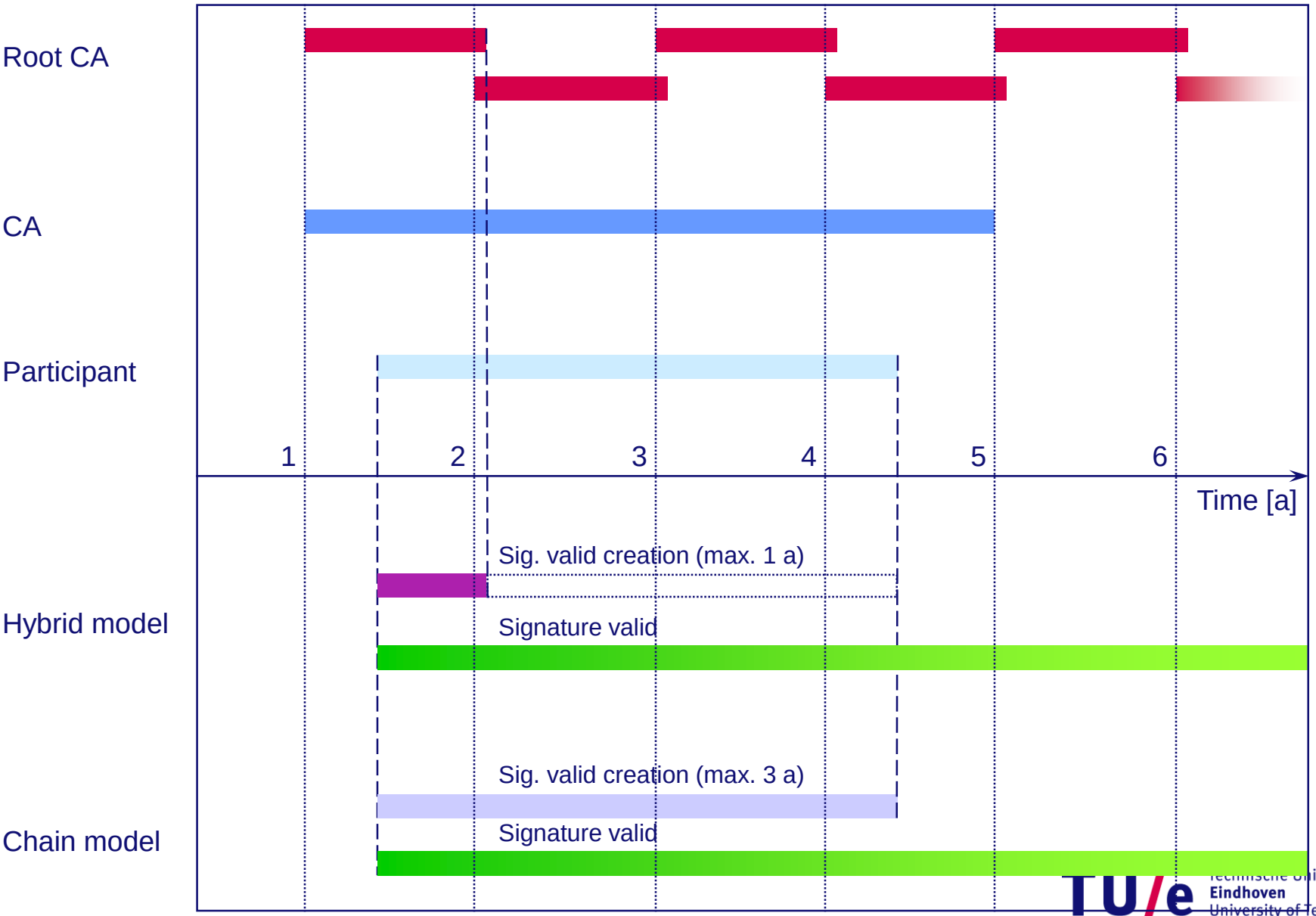
Chain model


Certificate 1
Certificate 2
Certificate 3
Signed Document
Time
Sig. valid creation
Signature valid
verification

Chain model: multiple-validation




Algorithms
Certificate 1
Certificate 2
Time
Shell model
Signature valid
Signature invalid
Hybrid model
Sig. valid creation
Signature valid
Chain model
Sig. valid creation
Signature valid

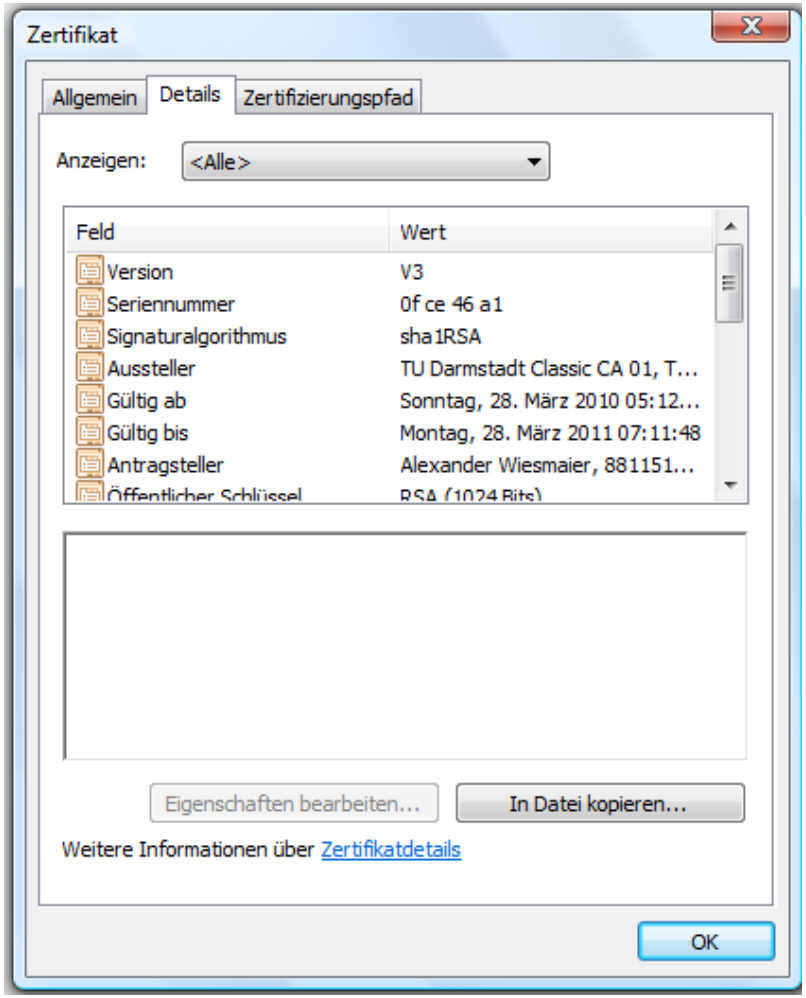

Root CA
CA
Participant
1
2
3
4
5
6
Time [a]
Sig. valid creation (max. 1 a)
Signature valid
Hybrid model
Sig. valid creation (max. 3 a)
Signature valid
Chain model

X.509 Certificates

X.509 Certificates

- Relevant Standard:
  - X.509 (ITU-T)
  - PKIX (RFC 5280)
- Encoding:
  - Abstract Syntax Notation Nr.1: ASN.1
  - Distinguished Encoding Rules: DER
- Content (excerpt):
  - Name / Pseudonym of the holder
  - Public Key (and algorithm) of the holder
  - Unique ID of the certificate
  - Validity period of the certificate
  - Identity of the certificate issuer
  - Key usage limitation for the public keys

X.509 Certificates



X.509 Certificates: Contents

Version 1 (1988)

- Version (0=v1, 1=v2, 2=v3)
- Serial Number (Unique within PKI)
- Certificate Signature Algorithm
- Issuer
- Validity Period
- Subject
- Subject Public Key Info

Version 2 (1993)

- Subject Unique ID (worldwide unique)
- Issuer Unique ID (worldwide unique)

Version 3 (1997)

- Extensions

X.509 Extensions: Properties

- **Assignment of extra attributes to**
  - **the owner**
  - **public or private key**
  - **issuer**
- **Support for better certificate management**
- **Arbitrary extensions ⇒ Bad interoperability**

X.509 (Non)critical extensions

	Critical	Non-Critical
Known	valid	valid
Unknown	invalid	valid

Key Usage

Defines the purpose of the key contained in the certificate.

```
KeyUsage ::= BIT STRING {
    digitalSignature        (0),
    nonRepudiation          (1),
    keyEncipherment         (2),
    dataEncipherment        (3),
    keyAgreement            (4),
    keyCertSign             (5),
    cRLSign                 (6),
    encipherOnly            (7),
    decipherOnly            (8) }
```

http://www.ietf.org/rfc/rfc5280.txt (pp 29ff)

Extended Key Usage (1)

Indicates one or more purposes for which the certified public key may be used, in addition to or in place of the basic purposes indicated in the key usage extension

For example:

- **Code signing**
- **OCSP signing**
- **Timestamping**

ExtKeyUsageSyntax ::= SEQUENCE SIZE (1..MAX) OF KeyPurposeId
KeyPurposeId ::= OBJECT IDENTIFIER

Extended Key Usage (2)

If a certificate contains **both** a **key usage extension and** an **extended key usage extension**, then both extensions **MUST be processed independently** and the **certificate MUST only be used for a purpose consistent with both extensions**. If there is no purpose consistent with both extensions, then the certificate MUST NOT be used for any purpose.

Source: RFC 4334

X.509 Revocation

Certificate revocation

- **Abortive ending of the binding between**
  - **subject and key (public key certificate)**

  OR
  - **subject and attributes (attribute certifica**

- **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)
- Novomodo

- Alternative: very short certificate validity period, therefore no revocation (e.g. nPA).

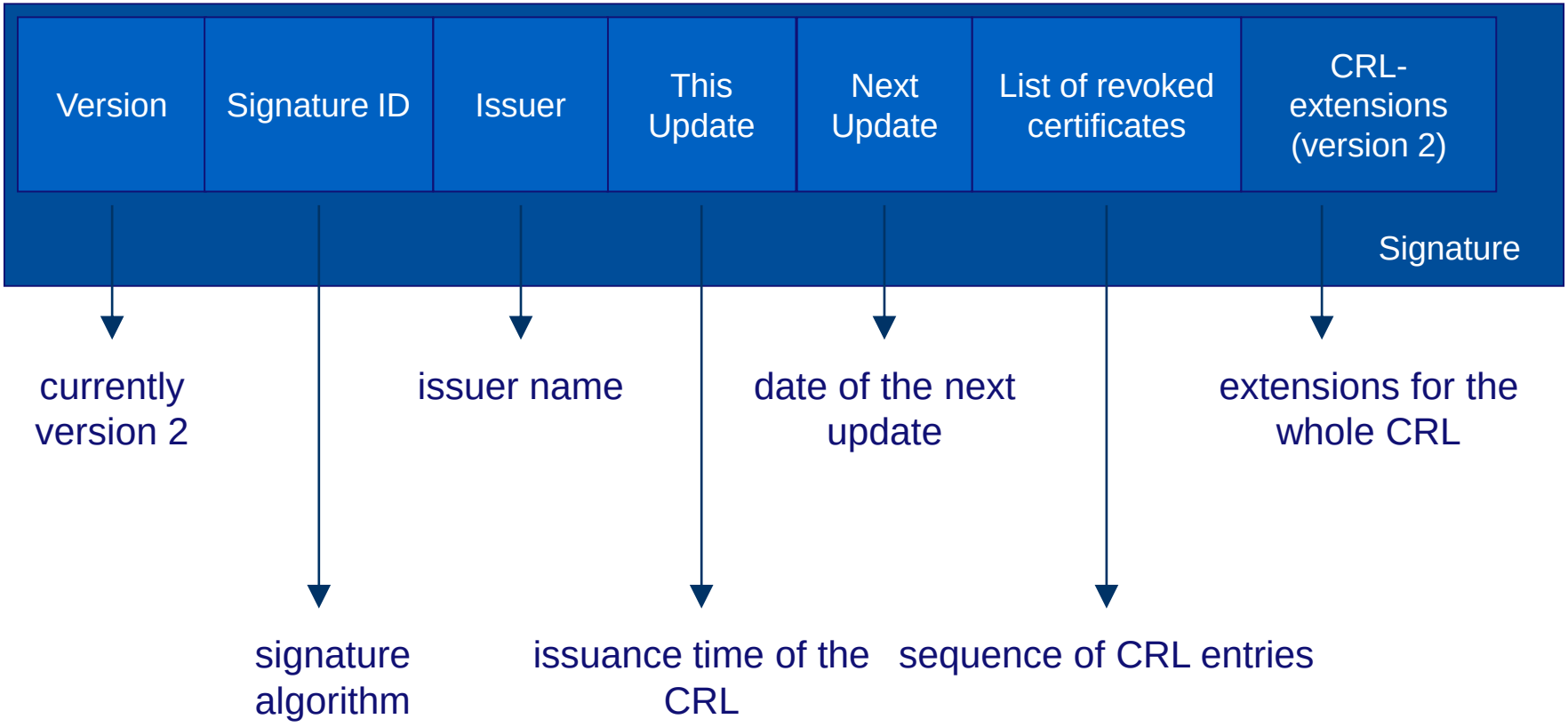
CRLs (Certificate Revocation Lists)

Certificate Revocation List (CRL)

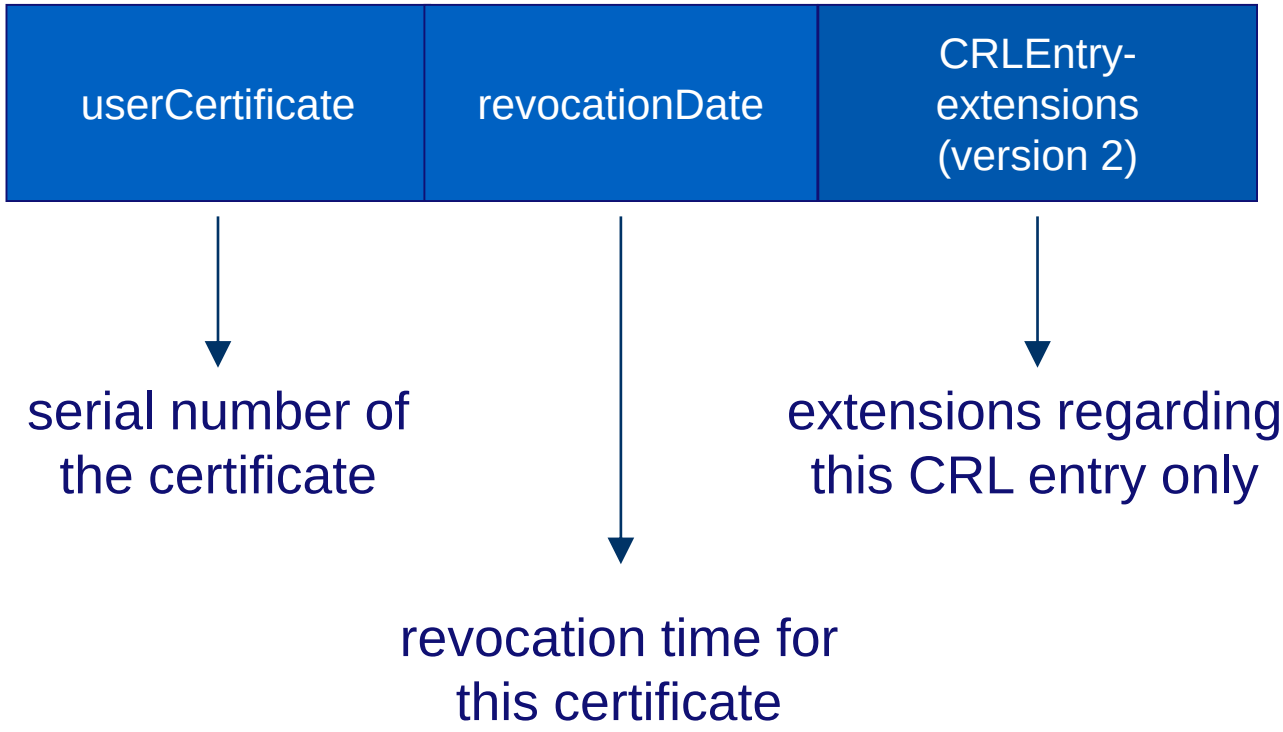
- **Signed list of revoked certificates**
- **“Blacklist”, i.e. no positive information about the validity of a certificate**
- **Standard mechanism (e.g. X.509)**
- **Wide-spread mechanism**

27	SSE-08\IUSR_SS...	-----BEGIN CERTI...	725b89d800000000001b	7/28/2005 10:54 PM
28	SSE-08\IUSR_SS...	-----BEGIN CERTI...	735a887800000000001c	7/29/2005 3:33 AM
29	SSE-08\IUSR_SS...	-----BEGIN CERTI...	148511c700000000001d	8/3/2005 11:30 PM
30	SSE-08\IUSR_SS...	-----BEGIN CERTI...	14a7170100000000001e	8/4/2005 12:07 AM
31	SSE-08\IUSR_SS...	-----BEGIN CERTI...	14fc45b500000000001f	8/4/2005 1:40 AM
32	SSE-08\IUSR_SS...	-----BEGIN CERTI...	486ce80b000000000020	8/17/2005 3:58 AM
33	SSE-08\IUSR_SS...	-----BEGIN CERTI...	4ca4a3aa000000000021	8/17/2005 11:37 PM
47	SSE-08\IUSR_SS...	-----BEGIN CERTI...	1aa55c8e00000000002f	9/1/2005 11:36 PM
63	SSE-08\IUSR_SS...	-----BEGIN CERTI...	3f0845dd00000000003f	9/9/2005 1:11 AM
66	SSE-08\IUSR_SS...	-----BEGIN CERTI...	3f619b7e000000000042	9/9/2005 2:48 AM
82	SSE-08\IUSR_SS...	-----BEGIN CERTI...	6313c463000000000052	9/16/2005 1:09 AM

Structure of a CRL (X.509)



CRLEntry (X.509)



CRL Extensions

- **Affect the CRL as a whole**

or maybe

- **Each single CRL entry (all of them)**

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*

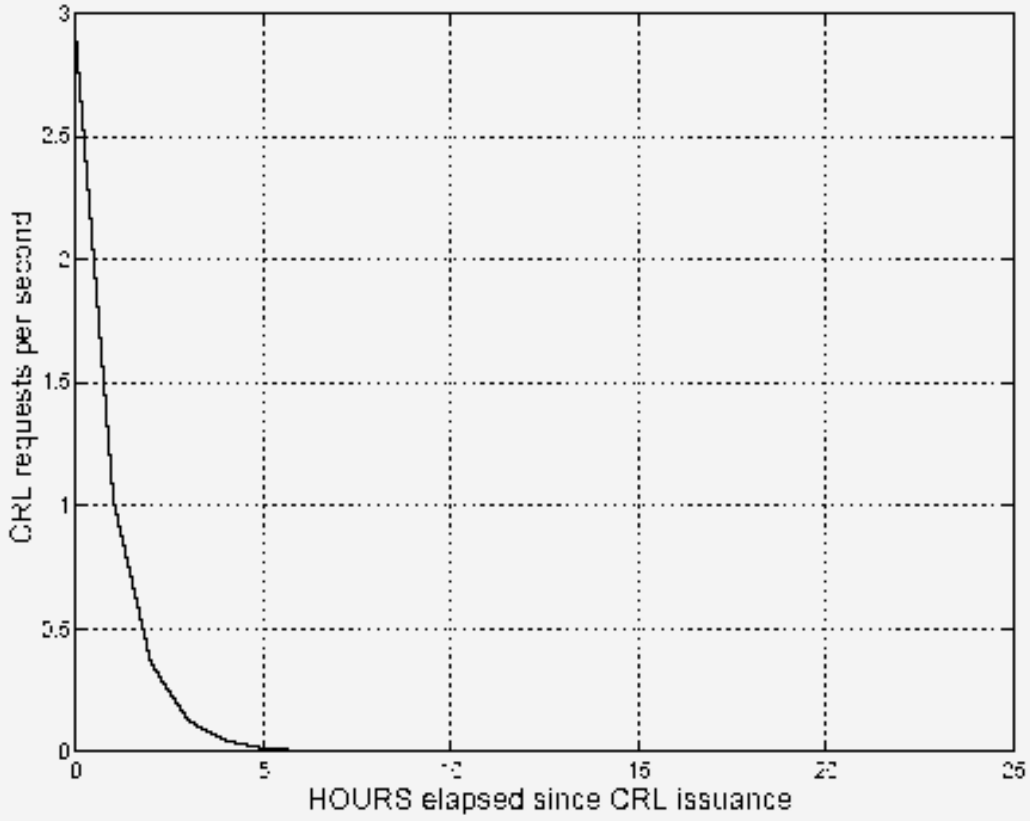


Figure 1: Request Rate following CRL issuance

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 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

Modifications

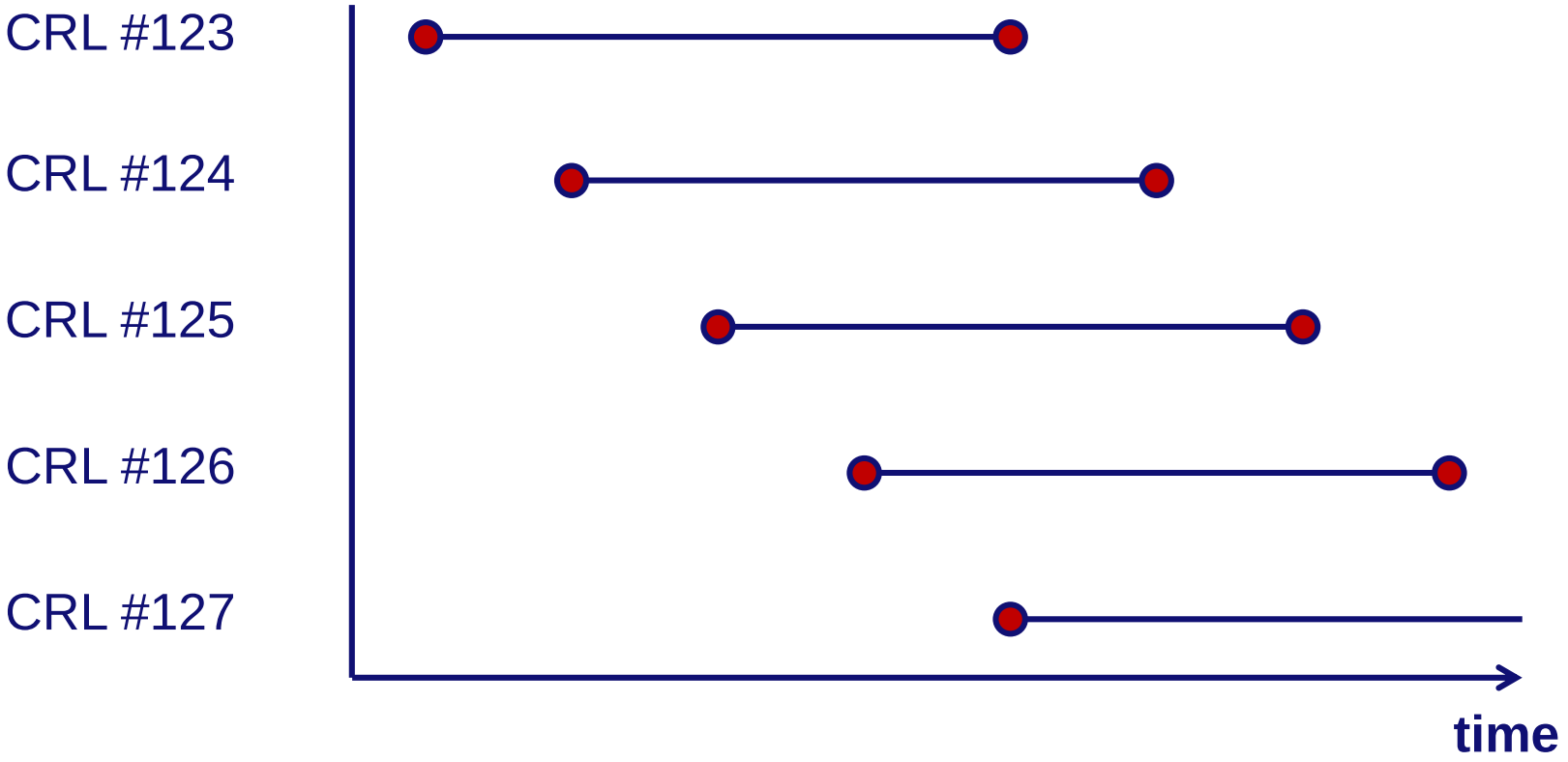
- **Lean CRLs**
  - **Expired certificates are removed from the CRL**
  - **Expired certificates cannot be checked anymore**

- **(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

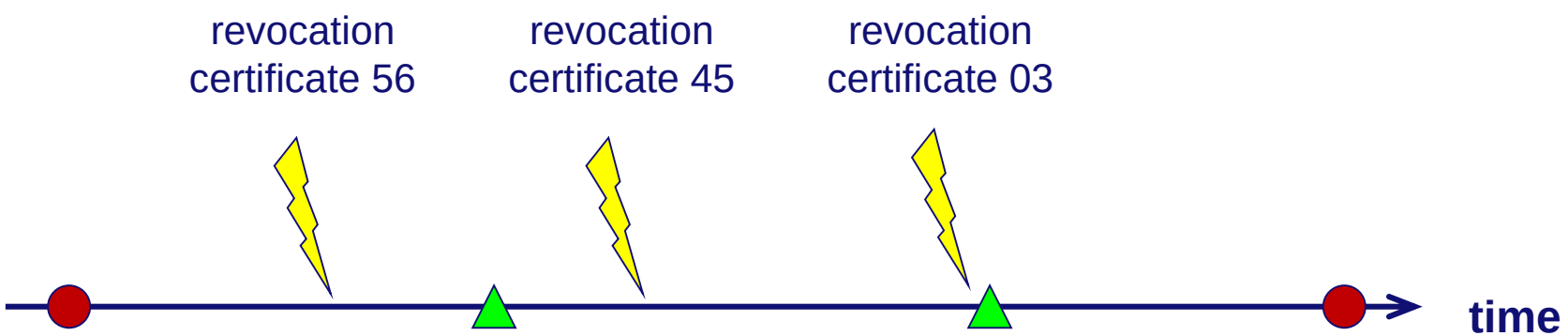
Over-Issued CRLs



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:	Δ-CRL	Δ-CRL	CRL#17:
02	56	03	02
23		45	03
34		56	23
			34
			45
			56

BaseCRL

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)

CRL Extension: Freshest CRL (a.k.a. Delta CRL Distribution Point)

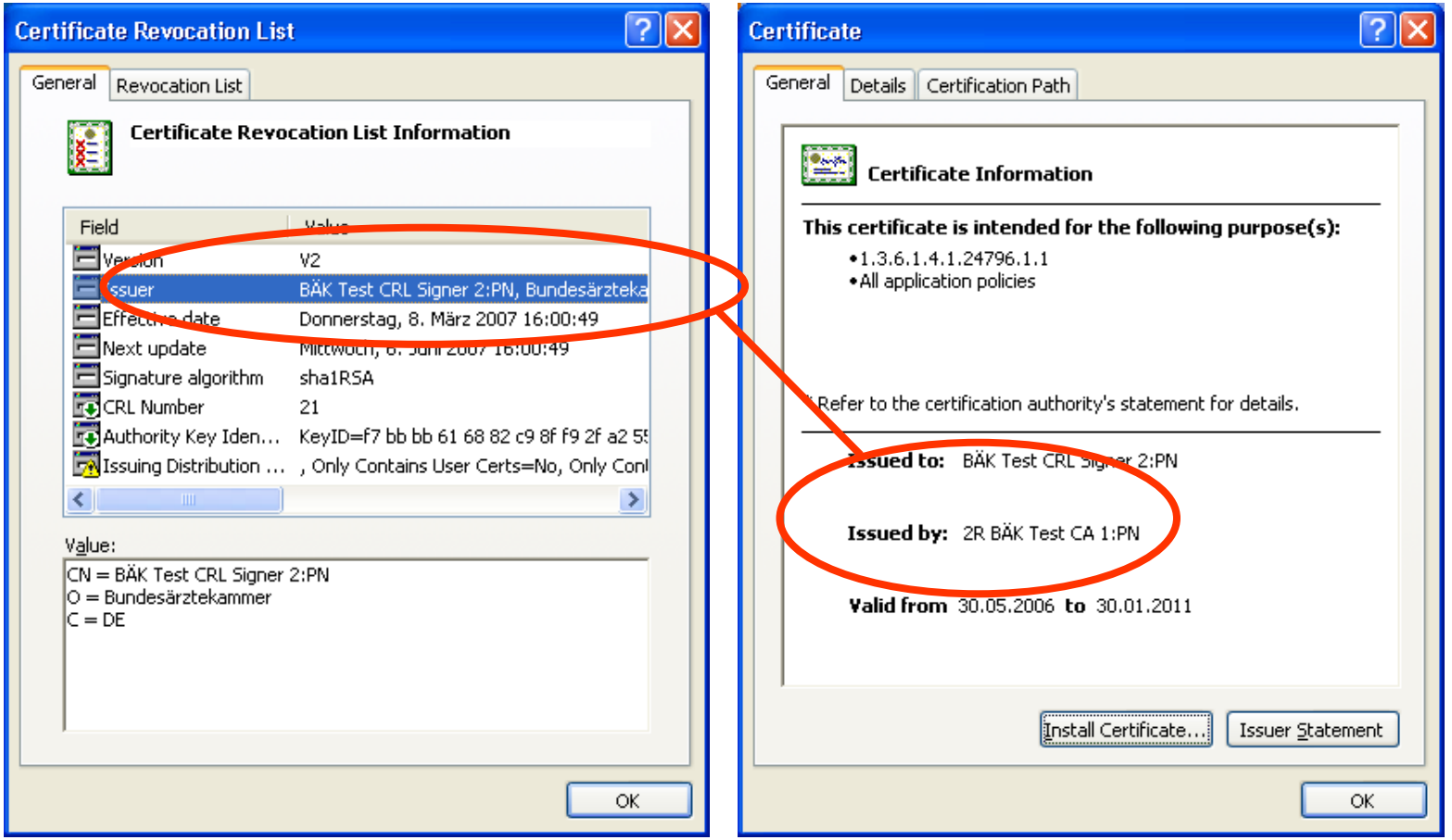
The freshest CRL extension **identifies how delta CRL** information for this complete CRL **is obtained**. The extension MUST be **non-critical**. This extension **MUST NOT appear in delta CRLs**.

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.**

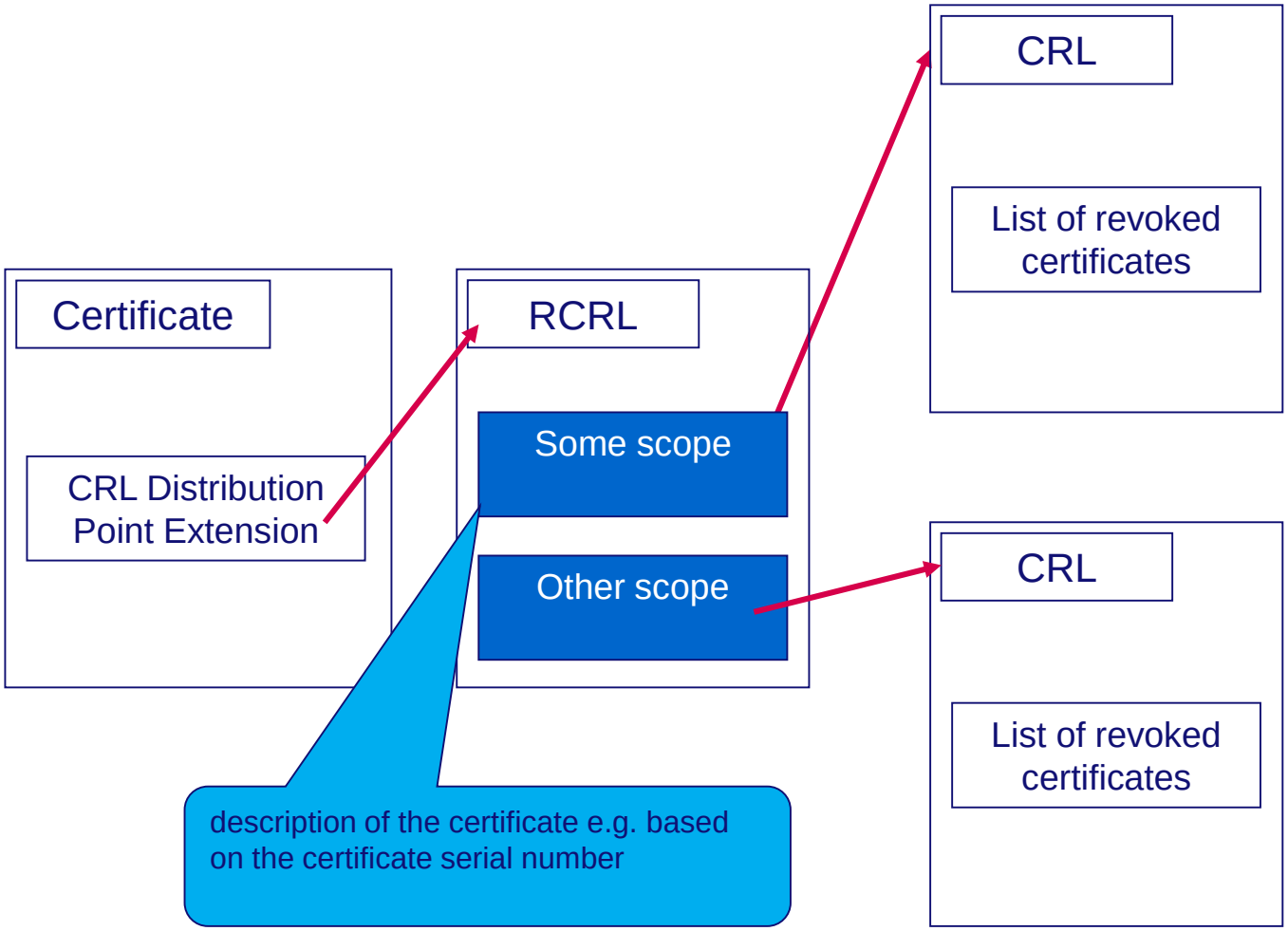
Indirect CRL Example



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**

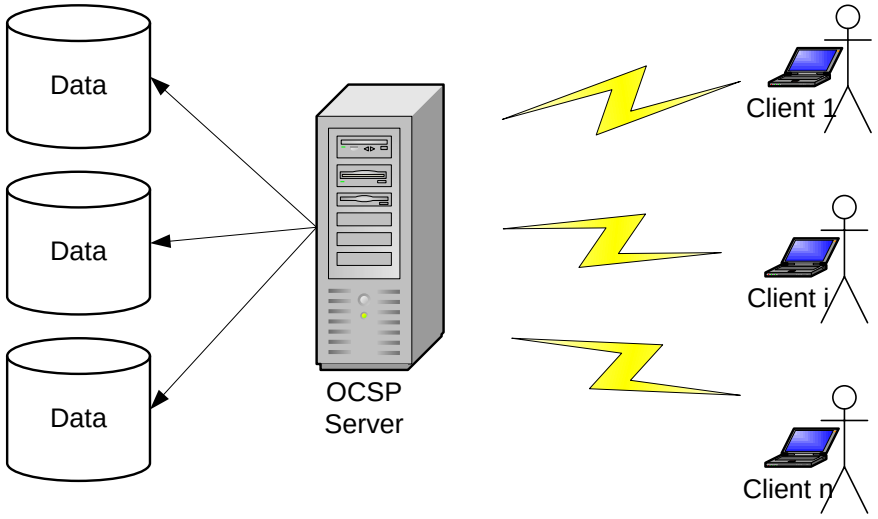
Redirect CRLs



OCSP
(Online Certificate Status Protocol)

Online Certificate Status Protocol (OCSP)

- **RFC 2560** http://www.ietf.org/rfc/rfc2560.txt
- **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 responds (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                  TBSRequest,
    optionalSignature   [0] EXPLICIT  Signature OPTIONAL }

TBSRequest ::= SEQUENCE {
    version             [0] EXPLICIT  Version DEFAULT v1,
    requestorName       [1] EXPLICIT  GeneralName OPTIONAL,
    requestList                       SEQUENCE OF Request,
    requestExtensions   [2] EXPLICIT  Extensions OPTIONAL }
```

OCSP Request – ASN.1 (cont.)

```
Request ::= SEQUENCE {
    reqCert                    CertID,
    singleRequestExtensions    [0] EXPLICIT    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:
C3CF75EAC011534513FE97656300695302 96B964

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 - Example

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: C3CF75EAC011534513FE976563006953029 6B964
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.

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

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.

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,
    crlTime     [2]   EXPLICIT   GeneralizedTime OPTIONAL }
```

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**

OCSP Stapling

- **Transport Layer Security (TLS) Extensions: Extension Definitions - RFC 6066: https://tools.ietf.org/html/rfc6066**
- **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**

Novomodo

Drawbacks of CRLs and OCSP

CRL

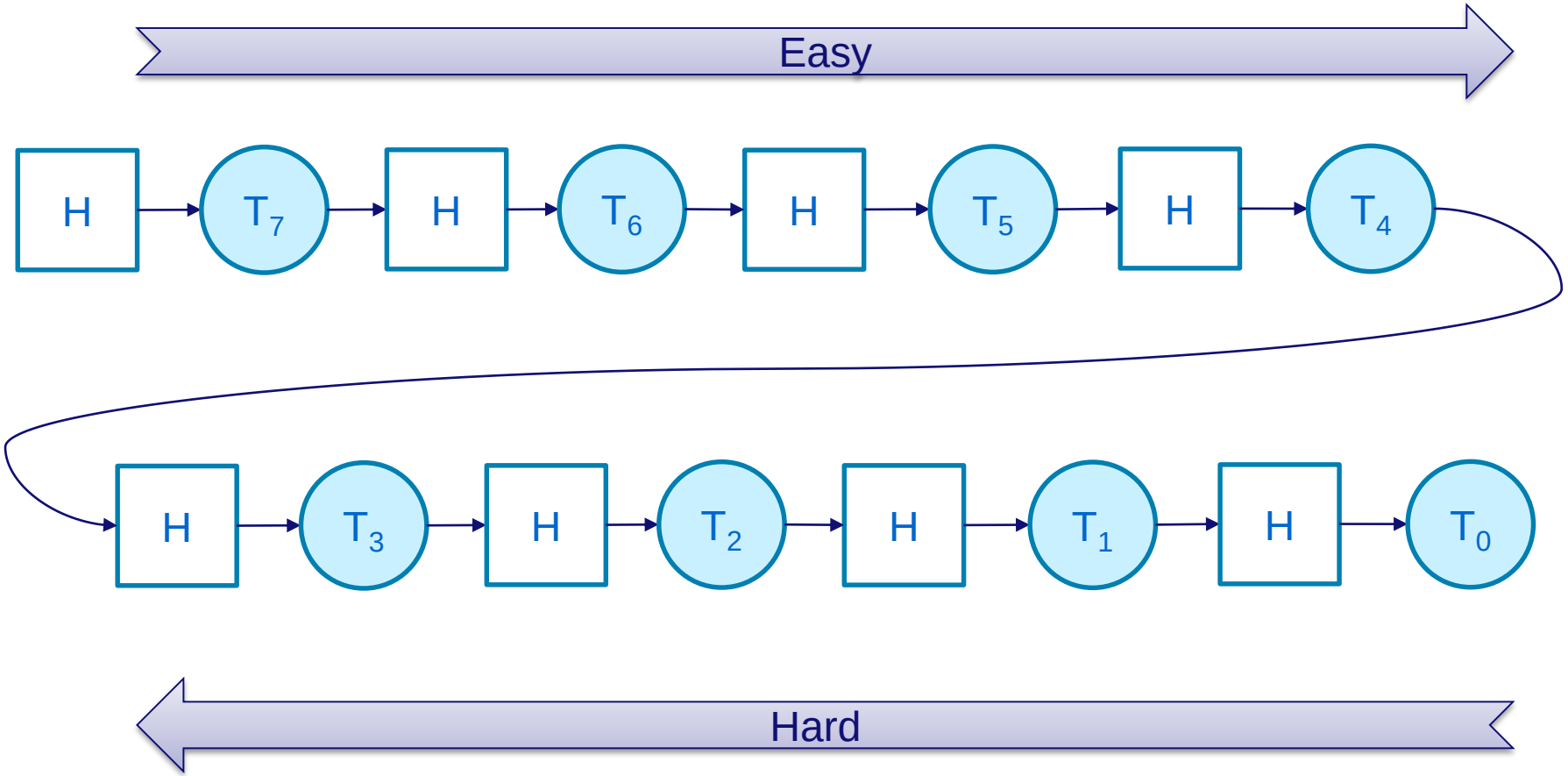
- **Traffic peaks at nextUpdate**
- **Hard to keep up-to-date w/ reasonable performance**
- **CRL issuer revocation**

OCSP

- **Bad scaling (one request per website)**
- **Bandwidth (everytime at least one signature)**
- **Computation (everytime one signature)**
- **OCSP issuer revocation**

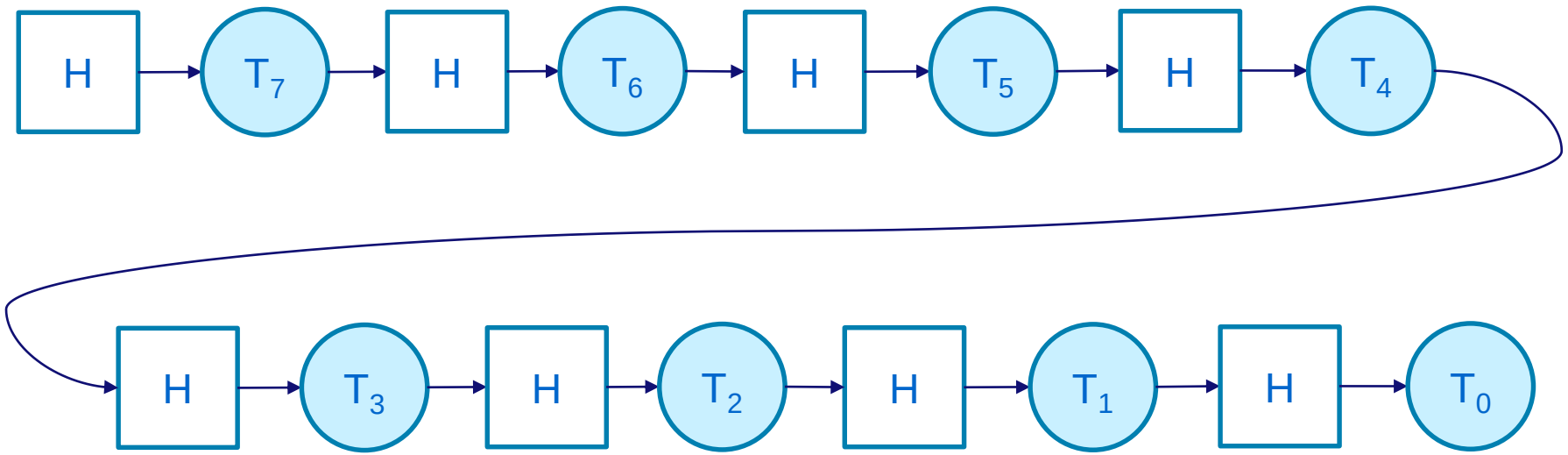
Novomodo [Micali´97]

- **Uses hash-chains**



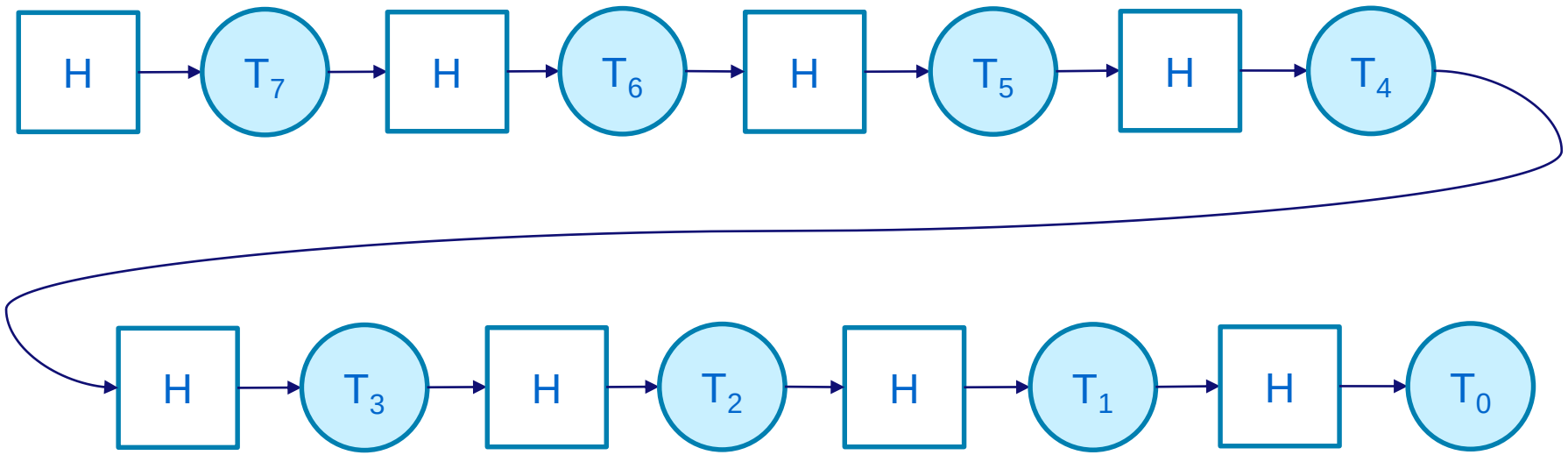
Novomodo, cont'd

- Uses hash-chains
- Given T_0, easy to verify T_x is x-th predecessor
- Given T_y, hard to compute T_x, x > y



Novomodo, basic concept

- Fix time periods (e.g. 356 days * 1 / day)
- One token (T_x) per time period
- Place T_0 in certificate
- At time period issued+x publish T_x



Novomodo, certificate generation

Assume 1 time period / day for one year (356 TPs)

When generating a cert, CA does

- **Generate two random n bit values T, R**
- **Compute $R_0 = H(R)$**
- **Compute $T_0 = H^{356}(T)$**
- **Put R_0, T_0 into certificate**
- **Keep T, R secret**

- **T_0 = validity target**
- **R_0 = revocation target**

Novomodo, status check

On day issued+i CA publishes token:

- **R, if cert was revoked,**
- **T_i, if cert is still valid**

For verification on day issued+i a user

- **obtains token X,**
- **if $H(X) = R_0$, user rejects cert (revoked)**
- **if $H^i(X) = T_0$, user accepts cert (valid)**
- **in any other case, user rejects cert (unknown)**

Security

- **Revocation forgery**
  => Breaks one-wayness of H
- **Valid-Token-Forger**
  => Breaks „one-wayness on iterates“ of H

Efficiency of Novomodo

- **Bandwidth:**
  - **Cert: Only two add. hash values.**
  - **Status information: Only one hash value.**
- **Computation with t time periods:**
  - **CA: t+1 hashes for setup; ≤ t for update.**
    - **Speed-ups possible using add. storage. (S:log t, C:log t/2)**
  - **User: ≤ t hashes for update.**
    - **Speed-up storing last verified token**
- **Servers can distribute validity token for own cert at connection establishment (like OCSP Stapling)**

QuasiModo & Co

Several extentions exist

Main concept:
Use Merkle tree

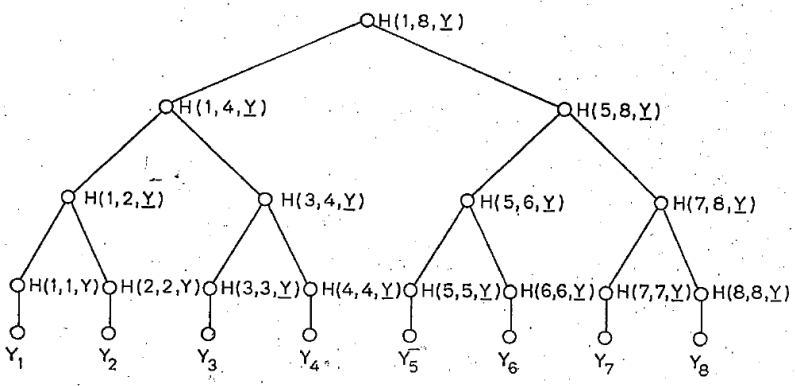


FIG 1
AN AUTHENTICATION TREE WITH N = 8.

PAGE 41B

The WebPKI

The WebPKI

- ~1.500 trusted CAs in MS & Mozilla trust stores
- ~ 650 organizations

(https://www.eff.org/observatory)

Everyone of those can create a valid certificate for any URL

DigiNotar

- **“DigiNotar was a Dutch certificate authority owned by VASCO Data Security International. On September 3, 2011, after it had become clear that a security breach had resulted in the fraudulent issuing of certificates, the Dutch government took over operational management of DigiNotar's systems. That same month, the company was declared bankrupt.” -- Wikipedia**

The DigiNotar hack

- Main target: 300,000 Iranian Gmail users
- > 500 fraudulent certs issued, including one wildcard cert for google (CN=*.google.com).
- Used for MitM attacks
- Recognized by chromium because of certificate pinning. (Google never used DigiNotar as CA)

Comodo (Report of incident on 15-MAR-2011)

9 certificates were issued as follows:

- **Domain: mail.google.com [NOT seen live on the internet]**
  Serial: 047ECBE9FCA55F7BD09EAE36E10CAE1E
- **Domain: www.google.com [NOT seen live on the internet]**
  Serial: 00F5C86AF36162F13A64F54F6DC9587C06
- **Domain: login.yahoo.com [Seen live on the internet]**
  Serial: 00D7558FDAF5F1105BB213282B707729A3
- **Domain: login.yahoo.com [NOT seen live on the internet]**
  Serial: 392A434F0E07DF1F8AA305DE34E0C229
- **Domain: login.yahoo.com [NOT seen live on the internet]**
  Serial: 3E75CED46B693021218830AE86A82A71
- **Domain: login.skype.com [NOT seen live on the internet]**
  Serial: 00E9028B9578E415DC1A710A2B88154447
- **Domain: addons.mozilla.org [NOT seen live on the internet]**
  Serial: 009239D5348F40D1695A745470E1F23F43
- **Domain: login.live.com [NOT seen live on the internet]**
  Serial: 00B0B7133ED096F9B56FAE91C874BD3AC0
- **Domain: global trustee [NOT seen live on the internet]**
  Serial: 00D8F35F4EB7872B2DAB0692E315382FB0

Verisign

Trustwave

THE H SECURITY

Security

In association with heise

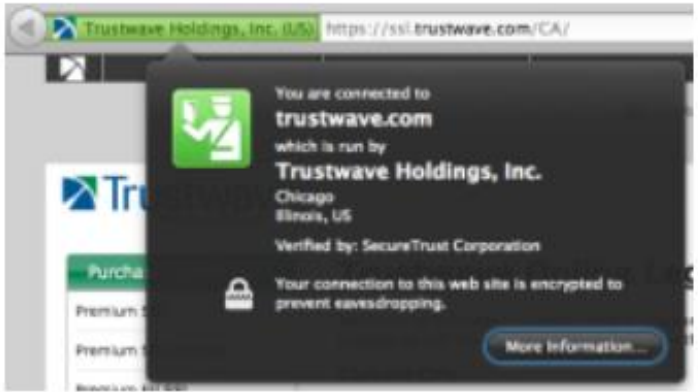
Last 7 days News Archive Features

07 February 2012, 19:02

« previous | next »

Trustwave issued a man-in-the-middle certificate

Certificate authority Trustwave issued a certificate to a company allowing it to issue valid certificates for any server. This enabled the company to listen in on encrypted traffic sent and received by its staff using services such as Google and Hotmail. Trustwave has since revoked the CA certificate and vowed to refrain from issuing such certificates in future.



versiteit
chnology

Trustwave

THE H SECURITY

Security

In association with heise

Last 7 days News Archive Features

07 February 2012, 19:02

« previous | next »

Trustwav

Since April 27, 2012, Mozilla removes CAs that issue MitM certificates / sub-CA certs that are used for MitM from its trust store

Certificate authority Trustwave issued a certificate to a company allowing it to issue valid certificates for any server. This enabled the company to listen in on encrypted traffic sent and received by its staff using services such as Google and Hotmail. Trustwave has since revoked the CA certificate and vowed to refrain from issuing such certificates in future.

versiteit
chnology

And there are more

http://lmgtfy.com/?q="certificate authority breach"

Honest Achmed

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

Solutions?

- None in wide-spread use
- Certificate Pinning:
  - „White-listing“ of public keys
  - Implemented in FF & Chrome for few popular websites (google, twitter, FB, TOR, mozilla, dropbox)
- Certificate Transparency (next slides)
- Convergence
  - Compares view of many users
  - Soft pinning

Certificate Transparency

Goals:

- **Make it impossible (or at least very difficult) for a CA to issue a SSL certificate for a domain without the certificate being visible to the owner of that domain.**
- **Provide an open auditing and monitoring system that lets any domain owner or CA determine whether certificates have been mistakenly or maliciously issued.**
- **Protect users (as much as possible) from being duped by certificates that were mistakenly or maliciously issued.**

Certificate Transparency

- **Mainly auditing framework**

Means:

- **Certificate Logs: Append-only logs (web service)**
- **Monitors: Automated checking of logs for suspicious certs (dedicated servers)**
- **Auditors: Make sure that certs appear in logs (e.g. part of monitors or browsers)**

- **Relies on CAs and domain owners filling and checking logs**