Public Key Infrastructures

Andreas Hülsing
How to share Keys with PGP

• Attach to mail

• Use Key Server

→ Still need to verify key validity!
PGP Keyserver Synchronization Graph

- http://www.rediris.es/keyserver/graph.html
X.509
Example: Secured Website

[Image of a website interface with various sections and links such as Inloggen, Klant worden, Rabobank Eindhoven-Veldhoven, and more.]
Click on button
Dieses Zertifikat wurde für die folgenden Verwendungen verifiziert:

SSL-Server-Zertifikat

Ausgestellt für
Allgemeiner Name (CN)  www.rabobank.nl
Organisation (O)  Rabobank Nederland
Organisationseinheit (OU)  Name IT bt

Ausgestellt von
Allgemeiner Name (CN)  VeriSign Class 3 Extended Validation SSL CA
Organisation (O)  VeriSign, Inc.
Organisationseinheit (OU)  VeriSign Trust Network

Validität
Ausgestellt am  21.06.2013
Läuft ab am  23.06.2014

Fingerabdrücke
The browser is shipped with trusted authorities.
Built-in object token

Zertifikat-Ansicht: "BuiltIn Object Token: VeriSign Class 3 Public Primary Certification Authority - G5"

**Allgemein**

Dieses Zertifikat wurde für die folgenden Verwendungen verifiziert:

- **SSL-Zertifizierungsstelle**

**Ausgestellt für**

- **Allgemeiner Name (CN)**: VeriSign Class 3 Public Primary Certification Authority - G5
- **Organisation (O)**: VeriSign, Inc.
- **Organisationseinheit (OU)**: VeriSign Trust Network

**Ausgestellt von**

- **Allgemeiner Name (CN)**: VeriSign Class 3 Public Primary Certification Authority - G5
- **Organisation (O)**: VeriSign, Inc.
- **Organisationseinheit (OU)**: VeriSign Trust Network

**Validität**

- **Ausgestellt am**: 08.11.2006
- **Läuft ab am**: 17.07.2036

**Fingerabdrücke**

Hierarchical trust

Certification Authority (CA) issues certificates

trust anchor

Alice

Bob

Carl
Hierarchical trust

- Why does Alice trust in Doris’ key?
Why does Alice trust in Doris’ key?
Hierarchical trust

Emil to Alice

Trust anchor
Public-key in question
Certification path
Intermediate CAs
When does Alice accept the certificate of Fred?
Method 1: Trusted List

- Every participant has a list of trusted CAs.
- Alice trusts TC2 and TC3
- Every user maintains an own list (like in the Web of Trust)
- Used in Web Browsers (preinstalled + user defined)
Trusted List: certification path

- Alice to Fred
Trusted List: Example
Trusted List: Example
Method 2: Common Root

- Every user who trusts TC1, accepts every other end-user certificate.
Common Root: certification path

- Alice to Fred
Two hierarchies

Certificate path

UNIGI-SCA-2005
www.uni-giessen.de

Certificate path

TUD-CA
TUD-SCA
www.tu-darmstadt.de

Certificate status:
This certificate is OK.
Common root
Method 3: Cross-certification

TC₂ issues a CA-certificate for TC₃.
TC₃ issues a CA-certificate for TC₂.

➔ Every user who trusts TC₃, accepts every certificate, that was issued by TC₂ (or a subordinate CA).
➔ Every user who trusts TC₂, accepts every certificate, that was issued by TC₃ (or a subordinate CA).
Cross-certification

- Alice to Fred
Cross-certification: Another possibility

TC₂ issues one CA-certificate to TC₇ and vice versa.
→ Hans accepts the certificate of Emil and vice versa.
→ Emil does not accept the certificate of Fred.
TC₄ issues one CA-certificate to TC₆ and vice versa.

- Alice accepts the certificate of Fred and vice versa.
- Fred does not accept the certificate of Emil.

Cross-certification: Another possibility
**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:** 1R BÅK Test CA 1:PN

**Issued by:** 1R BÅK Test CA 1:PN

**Valid from** 20.03.2006 to 20.03.2011

Certificate status: This certificate is OK.
2R root
1R-to-2R 2R-to-1R
2R as trust anchor
Cross-certification

\[ n^2 - n \text{ cross-certificates} = O(n^2) \]
Method 4: Bridge

**Idea:** Bridge TC has cross-certifications with TC₂ and TC₃.
- Alice accepts all certificates beneath TC₃.
- Fred accepts all certificates beneath TC₂.
Bridge: certification path

- Alice to Fred
- Bridge enforces minimal policy
Bridge Trust Center

• The bridge TC acts as a connector.
• This TC is not subordinate to a third CA.

• Interesting for corporate CAs that:
  
  • want to enable secure communication for their users outside the organisation’s borders.
  
  • do not want to be subordinate to a third CA.
European Bridge-CA

URL: http://www.bridge-ca.org
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

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

Version 3 (1997)
- Extensions
Certificate (ASN.1)

Certificate ::= SEQUENCE {
  tbsCertificate                TBSCertificate,
  signatureAlgorithm            AlgorithmIdentifier,
  signatureValue                BIT STRING
}

• To Be Signed (TBS) Certificate
  This part holds all information; this will be signed.

• Algorithm
  The algorithm that is used for signing the TBS part.

• Signature Value
  The calculated signature.
Certificate (ASN.1)

Certificate:

Data: [ ........ ]

Signature Algorithm: ripemd160WithRSA

00:92:0e:fb:67:80:96:c8:e0:af:2c:6c:21:c5:7c:
TBSCertificate (ASN.1)

TBSCertificate ::= SEQUENCE {
  version [0] EXPLICIT Version DEFAULT v1,
  serialNumber CertificateSerialNumber,
  signature AlgorithmIdentifier,
  issuer Name,
  validity Validity,
  subject Name,
  subjectPublicKeyInfo SubjectPublicKeyInfo,
  issuerUniqueID [1] IMPLICIT Uniquelentifier OPTIONAL,
  -- If present, version MUST be v2 or v3
  subjectUniqueID [2] IMPLICIT Uniquelentifier OPTIONAL,
  -- If present, version MUST be v2 or v3
  extensions [3] EXPLICIT Extensions OPTIONAL
  -- If present, version MUST be v3 }

28-5-2014 PAGE 43
Version

Holds the version of X.509 that the certificate is.

Version ::= INTEGER { v1(0), v2(1), v3(2) }
Serial Number

The serial number of the certificate

CertificateSerialNumber ::= INTEGER

- Positive integer

- Must be unique for the same issuer
  - Two certificates from the same issuer are not allowed to have the same serial number
Signature

Specifies the algorithm that was used to sign the certificate e.g. SHA1withRSA

**AlgorithmIdentifier ::= SEQUENCE {**
algorithm OBJECT IDENTIFIER,
parameters ANY DEFINED BY algorithm OPTIONAL }

algorithm: the algorithm OID (1.2.840.113549.1.1.5)

parameters: any needed parameters (like the elliptic curve to be used – in ECDSA) MUST be the same as the signatureAlgorithm of the certificate
TBSCertificate ::= SEQUENCE {
  version              [0] EXPLICIT Version DEFAULT v1,
  serialNumber         CertificateSerialNumber,
  signature             AlgorithmIdentifier,
  issuer                Name,
  validity              Validity,
  subject               Name,
  subjectPublicKeyInfo  SubjectPublicKeyInfo,

  issuerUniqueID        [1] IMPLICIT UniqueIdentifier OPTIONAL, -- If present, version MUST be v2 or v3
  subjectUniqueID       [2] IMPLICIT UniqueIdentifier OPTIONAL, -- If present, version MUST be v2 or v3
  extensions            [3] Extensions OPTIONAL -- If present, version MUST be v3 --

  signatureAlgorithm
  signatureValue
}
Issuer

Holds the name of the issuer (CA)

Looks like:

CN=RBG CA, OU=FB Informatik, O=TU Darmstadt, C=DE
Validity

Shows the period of time that a certificate can be used

Validity ::= SEQUENCE {
  notBefore Time,
  notAfter  Time }
Subject

Holds the name of the certificate holder

Looks like:

CN=Andreas Huelsing, OU=WIN, O=TUE, C=NL

It is an X.500 DN (distinguished name)

Associated to the public key contained in the certificate

The same DN is not allowed to be given to two different entities
Public Key

Holds the public key of the entity

SubjectPublicKeyInfo ::= SEQUENCE {
    algorithm            AlgorithmIdentifier,
    subjectPublicKey     BIT STRING }

28-5-2014   PAGE 51
X.509 unique identifiers


- Subject Unique ID (world wide unique)
- Issuer Unique ID (world wide unique)

- Version 2 and 3 only
- Identifies an issuer or subject, in case a DN is reused

- UniquelIdentifier ::= BIT STRING
X.509 Extensions

Version 3 (1997)

Extensions

Drawbacks of X.509v1 und X.509v2:

- Predetermined naming structure according to X.500 (e.g. usage of email addresses is not possible).
- No statements about the intended usage of the certified key.
- No statements about the underlying policy (e.g. how was the identity of the certificate owner verified?).

Solution:

⇒ Flexible extension fields.
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 Extensions

Hold additional information

Extensions ::= SEQUENCE SIZE (1..MAX) OF Extension

Extension ::= SEQUENCE {
  extnID OBJECT IDENTIFIER,
  critical BOOLEAN DEFAULT FALSE,
  extnValue OCTET STRING }

28-5-2014 PAGE 55
<table>
<thead>
<tr>
<th></th>
<th>Critical</th>
<th>Non-Critical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Known</td>
<td>valid</td>
<td>valid</td>
</tr>
<tr>
<td>Unknown</td>
<td>invalid</td>
<td>valid</td>
</tr>
</tbody>
</table>
Subject Key Identifier (SKIE)

- Identifies certificates that contain a particular public key.
- MUST be included in all CA certificates (non-critical)
- 160 bit hash of the Public Key (exclude tag, length, number of unused bits)
- Or “0100” followed by 60 lsbits of the hash of the public key

SubjectKeyId::= KeyIdentifier
KeyIdentifier ::= OCTET STRING
Authority Key Identifier (AKIE)

• Identifies the public key that corresponds to the private key that has signed the certificate.
• MUST be included in all certificates (non-critical)
  (unless it is a self-signed certificate)

AuthorityKeyIdentifier ::= SEQUENCE {
  keyIdentifier [0]  KeyIdentifier OPTIONAL,
  authorityCertIssuer [1]  GeneralNames OPTIONAL,
  authorityCertSerialNumber [2]  CertificateSerialNumber OPTIONAL }

KeyIdentifier ::= OCTET STRING
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)
The subject alternative name extension allows additional identities to be bound to the subject of the certificate.

for example:

• Internet electronic mail address
• a DNS name
• an IP address
• uniform resource identifier (URI)
• all possible combinations

Before included, this information MUST be verified since it is bound to a public key.
Subject Alternative Name (2)

SubjectAltName ::= GeneralNames
GeneralNames ::= SEQUENCE SIZE (1..MAX) OF GeneralName
GeneralName ::= CHOICE {
  otherName [0] OtherName,
  rfc822Name [1] IA5String,
  dNSName [2] IA5String,
  x400Address [3] ORAddress,
  directoryName [4] Name,
  ediPartyName [5] EDIPartyName,
  uniformResourceIdentifier [6] IA5String,
  iPAddress [7] OCTET STRING,
  registeredID [8] OBJECT IDENTIFIER }

28-5-2014  PAGE 62
Issuer Alternative Name

Associates Internet style identities with the certificate issuer.

SHOULD NOT be marked critical

IssuerAltName ::= GeneralNames
Subject Directory Attributes

It is used to convey identification attributes (e.g., nationality) of the subject. The extension is defined as a sequence of one or more attributes.

MUST be non-critical

SubjectDirectoryAttributes ::= SEQUENCE SIZE (1..MAX) OF Attribute
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
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
Basic Constraints: ASN.1

Identifies
• whether the subject of the certificate is a CA
• the maximum number of non-self-issued intermediate certificates that may follow this certificate in a valid certification path.

It is marked critical
If pathlength is not present => no limit

BasicConstraints ::= SEQUENCE {
cA BOOLEAN DEFAULT FALSE,
pathLenConstraint INTEGER (0..MAX) OPTIONAL }
Basic Constraints: Example

Root CA

Subordinate CA

Subordinate CA

Alice

c: true
pathLength:

c: true
pathLength:

c: false
Basic Constraints: Example

- Root CA
  - Subordinate CA
    - Subordinate CA
      - Alice

Minimum values:
- ca:true
  - pathLength: 1
- ca:true
  - pathLength: 0
- ca:false
Certification Path Validation
Shell model

root certificate

CA certificate

participant certificate

time

signature time

verification time
Modified or hybrid model

- root certificate
- CA certificate
- participant certificate
- signature time
- verification time
- time
Chain model

- root certificate
- CA certificate
- participant certificate
- signature time
- verification time
- time
Shell model

Certificate 1

Certificate 2

Certificate 3

Signed Document

Time

Sig. valid creation

Signature valid verification

Signature invalid verification
Chain model

Sig. valid creation

Signature valid verification
Chain model: multiple-validation

Certificate 1
Certificate 2
Certificate 3

Document A
Document B
Document C

Signature verification: Document A
Document B
Document C

Time

Certificate 1
Certificate 2
Certificate 3

Document A
Document B
Document C

Signature verification: Document A
Document B
Document C
Algorithms

Certificate 1

Certificate 2

Shell model
Signature valid
Sig. valid creation
Signature invalid

Hybrid model
Signature valid
Sig. valid creation

Chain model
Signature valid