Security (CS4028)

Lecture 9. Certificates and PKI

Chunyan Mu

chunyan.mu@abdn.ac.uk

The School of Natural and Computing Sciences

Lots of slides of this lecture taken from Matthew Collinson's

Schedule

	Week	Lecture 1	Lecture 2	Tutorial
	1	Intro to course & security	Intro to Crypto	-
	2	Symmetric Crypto	Hash	Math for crypto
	3	Asymmetric Crypto-1	Asymmetric Crypto-2	Symmetric Crypto
	4	Signatures	Zero Knowledge Proof	Asymmetric Crypto
⇒	5	Certificates	Authentication	Signature & certificates
	6	Access Control	AC models	Authentication
	7	Information flow	Information flow	Access control
	8	Security management	Protocols	Information flow & management
	9	Network security	Network security	Protocols and communications
	10	Advanced topics	Advanced topics	Network, web, malware & mgt
	11	Revision		

Outline

Binding keys to identities and certificates

Public key infrastructure (PKI)

X.509 and PKIX

Summary

Outline

Binding keys to identities and certificates

Public key infrastructure (PKI)

X.509 and PKIX

Summary

Binding keys to identities and certificates

Attack on public key and binding

- How can we have assurance that a public key, purported to come from some entity, really does belong to that identity?
- We need to bind the *identity* to the *public key*.

Binding keys to identities and certificates

Example: Attack on public key and binding

- I may publish my public key on my webpage in order for others to send me an encrypted message in email.
- An attacker who can access and alter my webpage could change the public key to their own, intercept the emails (or get them from my email server), and then decrypt the content.
- Similarly, if a public key is used for verification of a signature purporting to be by some entity A, how do we really know it is A's and not some other?

Binding keys to identities and certificates

Possible binding solutions

- If the principal (entity) needs to be physically present, then you can have them carry the key around on a tamper-resistant smart card with protection by a password/PIN etc.
- The original proposal was in fact to have bindings listed in a heavily write-protected file that could easily be accessed for reading.
    - This was difficult at the time it was proposed (1976).
- Use a (public key) certificate that contains a unique name for the principal, and the public key.
    - Have such certificates transmitted or consulted as necessary.

Binding keys to identities and certificates

Certificates (Main idea)

- A (public key) certificate is a signed message containing
  - the entity's identity
  - the value of its public key
  - other stuff, e.g., expiry date
- Analogy: can loosely think of them as letter of introduction for the entity, signed by someone.

Binding keys to identities and certificates

Certificates (Main idea - more detail)

- Notation: in abstract terms, certificates have the form:

$$\mathrm{Sig}_{\mathrm{CA}}(A, v_A)$$

where

  - $\mathrm{Sig}_{\mathrm{CA}}$ is a signature by the signer,
  - A is the entity named,
  - and v_A is the public key of A.

Certificate:
Key: V_A
Identity: A
Signed by: CA

$\mathrm{Sig}_{\mathrm{CA}}(A, v_A)$

Binding keys to identities and certificates

Certificates and proof of identity

- Mere possession of a valid certificate does <u>not</u> identify the entity in possession as the entity named in the certificate.
- Entity A could hold a certificate containing B's name and public key.
- Challenge-response protocols are often used for entity authentication instead.
  - B might be issued a challenge, which requires a signature with its private key.
  - Assuming that B has kept the private key private, this proves the identity of B.

Binding keys to identities and certificates

Certificates and key use and trust

- Suppose that there is a certificate, c, containing principal's name A, and public key v_A corresponding to private key s_A.
- The certificate will be signed by some other entity CA using some private key s_{CA}.
- Entity B might trust CA's key, in the sense that it accepts that it belongs to CA.
  - It will therefore accept that anything signed with s_{CA} really was intended to be signed by CA.
  - In particular, it accepts c was intended to be signed by CA.
- Furthermore, B might also trust that CA will only sign a certificate that contains a correct binding, in this case,
  - B will accept v_A belongs to A, i.e., B will trust A's key.
  - CA is a certification authority (CA) for B.

Binding keys to identities and certificates

Certificates and key use and trust

- If A and B both trust the same CA, T, to sign certificates binding their identities and public keys correctly, then they can use T as a key certification centre.
- They use this as follows:
  - A and B both have their own public-private key pairs
  - The public keys are put in separate certificates signed by T
    - Both A and B are then confident that each other's public keys do indeed belong to them
  - A encrypts the session key with B's public key
    - So A confident that only B can decrypt
  - A signs the result with its own private key
    - So B confident that message came from A
  - Encrypted session key and signature can be sent.
- Note that T does not get (symmetric) session keys in clear.

Binding keys to identities and certificates

Certificate Authorities

- A certification authority (CA) is there to provide digitally signed certificates that bind public key values to unique identities of entities.
- CA is a form of *trusted third party (TTP)*.
- This means that the CA's own public key must be trusted!
  - Have replaced one trust problem with another.
  - If attacker can get CA's private key, then B's can also be subverted.
  - Since CA's sign for many entities this makes them juicy targets.
  - There have been a number of successful attacks.

Binding keys to identities and certificates

Certifying certificates

- But how do we know that the certificate authority's key really belongs to it? We just have the same problem all over again.
- We could have it certified.
- We end up with a chains of certificates and keys that we trust. These have to terminate somewhere (i.e., A_0 below):

$$\begin{gathered}
\mathrm{Sig}_{A_0}(A_1, v_{A_1}) \\
\mathrm{Sig}_{A_1}(A_2, v_{A_2}) \\
\dots \\
\mathrm{Sig}_{A_{N-2}}(A_{N-1}, v_{A_{N-1}}) \\
\mathrm{Sig}_{A_{N-1}}(A_N, v_{A_N})
\end{gathered}$$

Binding keys to identities and certificates

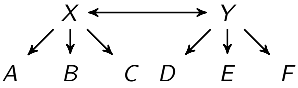
Certifying certificates

- How do we decide to arrange larger systems of certificates and chains.
- Two basic ways (currently):
    - Centralised and hierarchical (e.g., PKIX / X.509)
    - Decentralised and ad-hoc structure: no central CA
        - This kind of thing arises with Pretty Good Privacy (PGP) and its relatives.
- Different kinds of "trust" end up baked into the system:
    - in a centralised system: trust is institutional (CAs and the hierarchical structure)
    - in a decentralised system: trust is personal, based on the relationships between individuals

Binding keys to identities and certificates

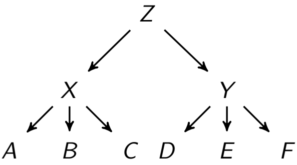
Arranging CAs

- Suppose: A has a certificate from CA X, D has a certificate from CA Y, A wants assurance about authenticity of D's public key.
- Then A needs a copy of Y's public key.
- The system needs one of:
  - Cross certification: X and Y issue certificates to each other.
  - Certification hierarchy: X and Y both sit under another parent CA, say Z, that issues certificates to its children.



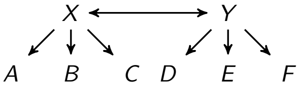
(1) Cross certificate

$M \rightarrow N$ means that M issues a certificate to N



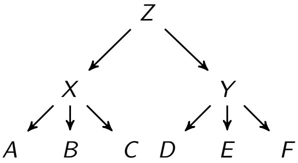
(2) Certificate hierarchy

Binding keys to identities and certificates



(1) Cross certificate

$M \to N$ means that M issues a certificate to N



(2) Certificate hierarchy

Using the CA arrangements

- Suppose that B needs to trust E's public key.
- Then B needs a copy of Y's public key.
- In situation (1):
  - B needs to verify cert. of Y issued by X, and
  - B needs to verify cert. of E issued by Y.
- In situation (2):
  - B needs to verify cert. of Y issued by Z, and
  - B needs to verify cert. of E issued by Y.

Outline

Binding keys to identities and certificates

Public key infrastructure (PKI)

X.509 and PKIX

Summary

Public key infrastructure (PKI)

PKI

A PKI is a system consisting of hardware, software, people, polices and procedures needed to create, manage, store, distribute, revoke and support use of public key certificates (PKCs).

- They are widely used to underpin e-commerce.
- They are currently critical, and sometimes argued to be essential, for:
  - Secure email
  - Secure web-server access
  - Secure virtual private networks
  - Other secure communications applications.

Public Key Infrastructure (PKI)

PKI in Use

- The Federal Public Key Infrastructure FPKI authority
- NASA PKI
- State of Illinois PKI
- University of Wisconsin PKI
- Secure e-mail
- Virtual private network
- Wireless
- Web servers (SSL/TLS)
- Network authentication

Public key infrastructure (PKI)

Need for PKI

- Identity verification of central importance.
- Cross verification requires CAs to use compatible technologies.
- When should a CA be trusted?
- CAs need to publish policy and practice statements containing clear information about their security

Public key infrastructure (PKI)

Three main players in a PKI system

- Certificate owner: applies for certificate
- CA: issues certificate
  - This binds owner identity to owner public key value
- User ("relying party"): uses the certificate for verification.

Public key infrastructure (PKI)

Validation Authority (VA)

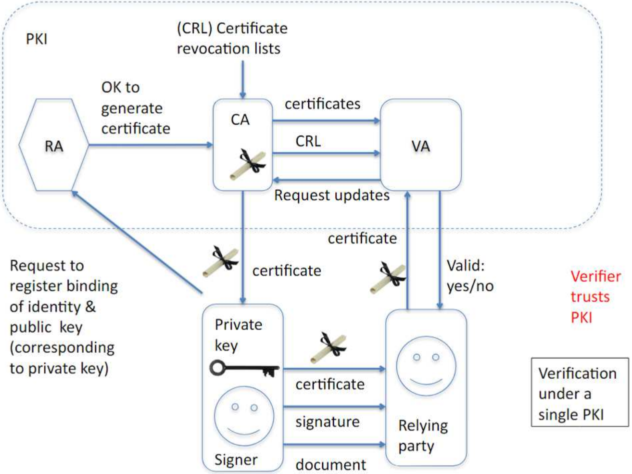
- For a user, verifying the authenticity of a public key may involve verifying the signatures in a long chain of certificates.
- Possibly too expensive/time-consuming for users.
- Instead the task can be delegated to an entity to call a validation authority (VA).
    - so the user relies on the VA.

Public key infrastructure (PKI)

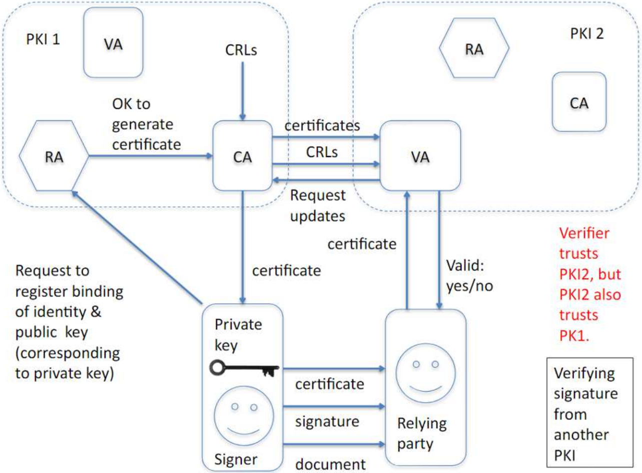
Registration Authority (RA)

- Sometimes the process of issuing certs is split up.
- Putative owner, O, applies to a Registration authority (RA).
  - Verifies (somehow) the O's identity.
  - How can O prove their identity?
  - Typically,
    - Might be sufficient that the O can authenticate to some system (e.g., with password): Something they know.
    - Might be that the O is required to produce another certificate!: Something they have.
- The RA and CA ensure that the key is bound to just that individual. Enables non-repudiation.

Public key infrastructure (PKI)


PKI
(CRL) Certificate revocation lists
OK to generate certificate
RA
CA
certificates
CRL
VA
Request updates
certificate
Request to register binding of identity & public key (corresponding to private key)
certificate
Valid: yes/no
Verifier trusts PKI
Private key
certificate
signature
document
Signer
Relying party
Verification under a single PKI

Public key infrastructure (PKI)



Verifier trusts PKI2, but PKI2 also trusts PK1.

Verifying signature from another PKI

Outline

Binding keys to identities and certificates

Public key infrastructure (PKI)

X.509 and PKIX

Summary

X.509

- X.509 (v3) is the most common PKI standard.
- The X.509 Directory is a hierarchical tree structure able to contain names of devices, people, roles, etc. and corresponding attributes.
  - Access control often tied to the identities in the directory.
- X.509 is a later addition to this family of standards. Applications of X.509 are different than those originally envisaged within X.500. Applies to a broader access control situations (many situations involving cryptographic authentication and communication, including SSL/TLS).
- X.509 can also be used for one-way authentication or mutual authentication of identities. We will discuss an application of this when we do SSL/TLS.

X.509

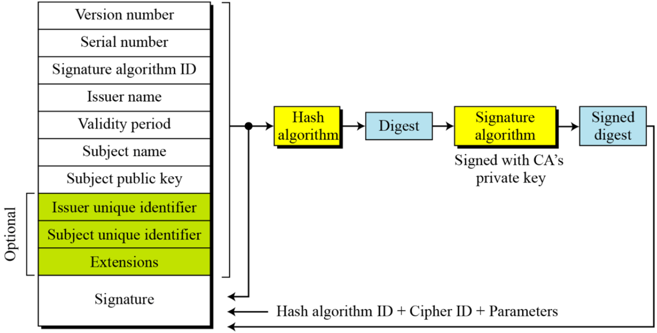
X.509 Format

A way to unify certificate formats, including the following fields:

- Version number
- Serial number: assigned to each certificate
- Signature algorithm ID
- Issuer name: identifies the CA that issued the certificate
- Validity period
- Subject name: the entity that owns the public key
- Subject public key: value of the public key of the certificate
- Signature

X.509

X.509 Format



X.509

Certificate Renewal

- Each certificate has a period of validity
- If there is no problem with the certificate, the CA issues a new one before the old one expires
- In some cases, a certificate must be revoked before its expiration

Delta Revocation

- To make revocation more efficient, the delta Certificate Revocation List (delta CRL) has been introduced.

X.509 PKI (PKIX)

Basic Components

- Provider side:
  - Certificate Authority (CA)
  - Registration Authority (RA)
  - Certification Distribution System: certs, Certification Revocation List (CRL)
- Consumer side:
  - PKI enabled applications, e.g., Browsers

X.509 PKI

Certificate Authority (CA)

Trusted authority for certifying individuals and creating an electronic document. The basic tasks include:

- Key generation
- Digital certification generation
- Certificate issuance and distribution
- Revocation
- Key backup and recovery system
- Cross certification

X.509 PKI

Registration Authority (RA)

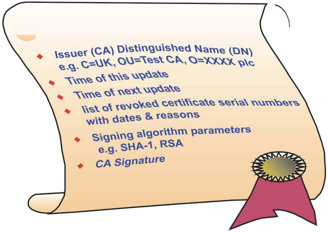
Trusted authority for accepting requests for digital certificates and performing necessary authority steps of registering and authenticating people. The basic tasks include:

- Registration of certificate information
- Face-to-face registration
- Remote registration
- Automatic registration
- Revocation
- Cross certification

X.509 PKI

Certificate Distribution System

- Digital certificates
- Certification revocation lists (CRLs)
- Special purpose databases
- Lightweight Directory Access Protocol (LDAP) directories



- Issuer (CA) Distinguished Name (DN) e.g. C=UK, OU=Test CA, O=XXXX plc
- Time of this update
- Time of next update
- list of revoked certificate serial numbers with dates & reasons
- Signing algorithm parameters e.g. SHA-1, RSA
- *CA Signature*

X.509 PKI Trust and Legal Aspects

Trust Approaches

- certificate hierarchies
- cross-certification

Legal Aspects

- Certificate policy: a document that sets out the rights, duties and obligations of each party in a public key infrastructure
- Certificate policy statement: a document which may have legal effect in limited circumstances

X.509 PKI Trust and Legal Aspects

The PKI trust model defines rules that specify how a user can verify a certificate received from a CA.

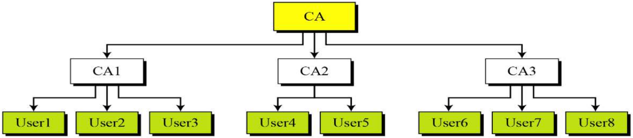
PKI Trust Model

- e.g., the PKI hierarchical trust model defines hierarchical rules that specify how a user can verify a certificate received from a CA
- PKI uses the following notation to denote the certificate issued and signed by certification authority X for entity Y:
  - $X << Y >>$

X.509 PKI

Example 1: certificate validation

User1 knows only the public key of the root CA. Show how can User1 obtain a verified copy of User3's public key.



X.509 PKI

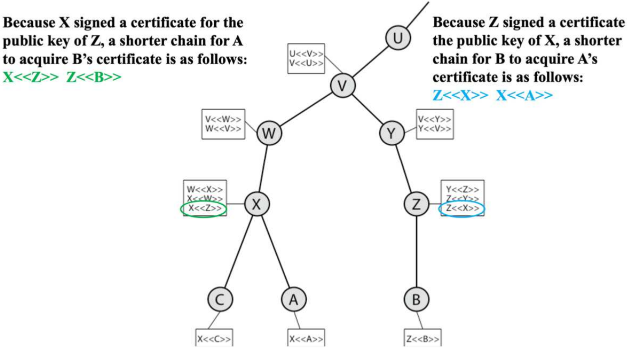
Example1: Solution

1. User1 validates $CA << CA1 >>$ using the public key of CA.
2. User1 extracts the public key of $CA1$ from $CA << CA1 >>$.
3. User1 validates $CA1 << User3 >>$ using the public key of $CA1$.

X.509 PKI

Example 2: hierarchy-certificate validation

User A only knows the public key of X and B only knows the public key of Z.



Because X signed a certificate for the public key of Z, a shorter chain for A to acquire B's certificate is as follows:
X<<Z>> Z<<B>>

Because Z signed a certificate the public key of X, a shorter chain for B to acquire A's certificate is as follows:
Z<<X>> X<<A>>

Image taken from:

`https://www.brainkart.com/article/X-509-Certificates_8470/`

X.509 PKI

Example2: Solution

1. A acquires B's certificate using the chain:

   $X << W >> W << V >> V << Y >> Y << Z >> Z << B >>$

2. B acquires A's certificate using the chain:

   $Z << Y >> Y << V >> V << W >> W << X >> X << A >>$

X.509 PKI

Hierarchy-Certificate Validation Path

- Web browsers, s.a. Firefox and Chrome, include a set of certificates from independent roots without a single, high-level authority to certify each root.
- you can check the CA roots in the browsers.

X.509 PKI

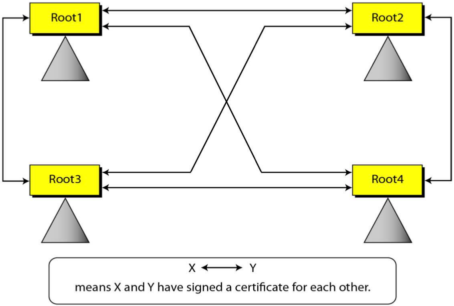
The Mesh Model

- The hierarchical model with a single root is not suitable for a very large community.
- The mesh model connects several roots together
- Roots are connected by the mesh but each root has its own hierarchical structure
- Certificates in the mesh are cross-certificates, i.e., each root certifies each other root

X.509 PKI

The Mesh Model

- For a fully connected mesh with four roots, we need $4 \times 3 = 12$ certificates.
- Each double-arrow represents 2 certificates.

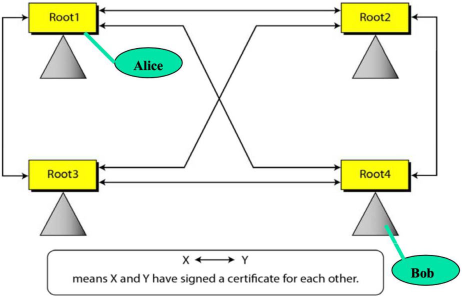


X.509 PKI

Example: the Mesh Model

- Alice is under the authority Root1
- Bob is under the authority Root4

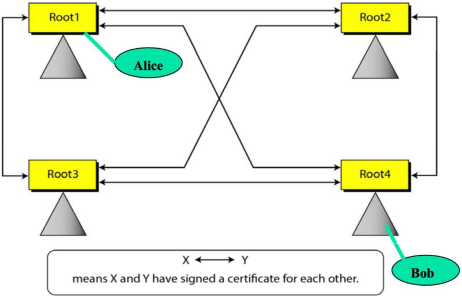
Show how Alice can obtain Bob's verified public key.



X.509 PKI

Solution: the Mesh Model

- Alice looks at the directory of Root1 to find Root1 $<<$ Root4 $>>$, using the public key of Root4.
- Alice can follow the chain of certificates from Root4 to Bob.
- Alice can then extract and verify Bob's public key.



Outline

Binding keys to identities and certificates

Public key infrastructure (PKI)

X.509 and PKIX

Summary

Summary

This week

- Public key distribution
- X.509
- Public key infrastructure (PKI)

Next lecture

- Authentication