IK2206 – Internet Security and Privacy
Chapter 15 – Public Key Infrastructure (PKI)

15.1 Introduction (READ)

- What are the components of a PKI?

  - public key certificates
  - a repository for retrieving certificates
  - a method of revoking certificates
  - a method of evaluating chains of certificates

- Assume Alice and Bob are not using any PKI. When Bob sends a signed message to Alice he also attaches his public key, so that Alice can verify the message. What kind of attack would this approach be subject to?

  Someone could sneak in between and replace the public key. How could the receiver (Alice) check the validity of the received certificate?

- Some related questions not explained in this section: Does your CA need to see your private key? Who should generate your private/public key pair? Could your CA impersonate you?

  No, the CA does only need to see public key certificates of peers. They're signed by the CA and returned to their "owners" (for further distribution). The key pairs are generated by the effective owner. Impersonation is (afaik) not possible.

15.2 Some terminology (READ)

- What is a "trust anchor"? What is an "issuer"? (Sometimes you may hear the term "Root CA" instead of "trust anchor". Still, in some PKI trust models the trust anchor is not necessarily the same as a root CA, as for example in Sect. 15.3.8 "Bottom-Up with Name Constraints")

  Trust anchor: When there is a chain of trust, usually the top entity to be trusted becomes the trust anchor. It can be for example a certification authority (CA). The public key (of the trust anchor) is used to verify digital signatures and the associated data. (http://en.wikipedia.org/wiki/Trust_anchor)

  Issuer: An issuer signs certificates of other entities.

15.3 PKI Trust Models (READ)

- Try to compare the various trust models presented in this section. In particular, how vulnerable are they if a CA (or RA) is compromised?

Monopoly = One	• Single point of failure. • There is no universally trusted organization. • It could get pretty expensive if there is only one company issuing certificates. (Abuse of monopolistic power).
Monopoly + Registration Authorities (RA)	• A single CA can define other authorities (=RA) to check identities.
Monopoly + Delegated CA's	• Delegated CA's can sign certificates the same way as the Root CA does. • A client can see the "chain of trust" if a certificate was signed by a delegated CA. (This is not the case with RA's).
Oligarchy = Many	• Multiple Root CA's. • Selectable which Root CA to trust. • No monopoly (usually cheaper + more efficient)
Anarchy = Any	• Everyone can decide whom to trust. • Usually free of charge. • This principle is used in PGP.

- What other characteristics do you find important when selecting a PKI trust model?

15.3.1 Monopoly Model (READ)

- What are the problems with the monopoly model? (see table above)
- Why would it be hard to find a single universally trusted organization? (see table above)
- Why may it be expensive to get a certificate issued? (see table above)

15.3.2 Monopoly plus Registration Authorities (RAs) (READ)

- What is the difference between a Registration Authority (RA) and a Certificate Authority (CA)?
- Does the information between the RA and CA need to be sent securely? Why? How is the security of the PKI affected if a RA becomes compromised?
- Compare the "Monopoly plus RAs" and the simple "Monopoly" models. What issue(s) is (are) solved by introducing RAs? What problems remain?

15.3.3 Delegated CAs (READ)

- Note that Delegated CAs can be used with many other models (e.g., Monopoly and Oligarchy models)
- What is the difference if a Alice gets her certificate from a delegated CA or a RA? Who is the "issuer" of her certificate? If Bob needs to verify her certificate, which CA certificate will he (likely) use as the "trust anchor"?
- How is the security of the PKI affected if a delegated CA becomes compromised?

15.3.4 Oligarchy (READ)

- Can you see how many CAs your favorite web browser includes as trust anchors?
- What is the impact on the security of the PKI if any of the CAs becomes compromised? Compare the risk with the monopoly model.
- If you use a model with "Oligarchy plus delegated CAs", assess the risk risk if one of your trusted CAs being compromised, as compared to
- the "monopoly" model? As compared to the "oligarchy" model?

15.3.5 Anarchy Model (READ)

- What are the pros and cons of the anarchy model? Economic aspects? Ease of finding a certificate path to the target? How much trust can you put in a certificate chain which is longer than a few hops?
- Using the anarchy model for "authentication" may be beneficial when it comes to "authorization". E.g., you may accept authenticated emails (or e.g., phone calls) from a friend of a friend, but perhaps not from an "authenticated stranger".

15.3.6 Name constraints (READ)

- What is the benefit of using name constraints? Why would the security of a PKI be less affected if some delegated CA was compromised if name constraints were used?

15.3.7 Top-Down with Name Constraints (READ BRIEFLY) (See also section 15.3.6)

- As opposed to what is said in the book, it seems possible to use the Top-Down model also with multiple root CAs (oligarchy).

15.3.8 Bottom-Up with Name Constraints (READ BRIEFLY)

- The Bottom-Up with Name constraints model both a "parent" and a "child" in the hierarchy issues certificates for eachother. What is an uplink certificate? What's a downlink certificate?
- What is a cross-link? What is a cross certificate?

15.3.9 Relative Names (SKIP)

15.3.10 Name Constraints in Certificates (SKIP)

15.3.11 Policies in Certificates (SKIP)

15.4 Revocation (READ)

- What is a CRL?

CRL means Certificate Revocation List and it serves as black list for compromised (but unexpired) certificates.

- Why are both, CRLs and expiration dates, used on certificates? Wouldn't one mechanism be enough?

  Expiration dates were introduced by commercial certification authorities to not let certificates be valid forever. The only reason for expiration dates is money.

  CRLs allow blacklisting a certificate just in time – without having to wait for the expiration date.

15.4.1 Revocation Mechanisms (READ)

- What does a CRL consist of?

  A CRL consists of compromised certificates.

- Why is it important for the verifier to have fresh CRL?

  To not get in contact with compromised/faked certificates.

- What could be the problem of requiring verifiers to download a fresh CRL too often?

  CRLs can grow very quickly. It could cost a verifier much bandwidth and other resources to update CRLs regularly.

15.4.1.1 Delta CRLs (READ)

- What is a delta CRL and what is the purpose of using "delta CRLs"?

  Delta CRLs help reducing the size of CRL updates.

15.4.1.2 First Valid Certificate (READ BRIEFLY)

- An idea of how to keep CRLs manageable.

15.4.2 OLRS Schemes (READ)

- What is an on-line revocation server (OLRS)? Is the OLRS as security sensitive as a CA or KDC?
- What is a "non-revocation certificate"? In what situations would the use of "non-revocation certificates" be desirable?

15.4.3 Good-lists vs. Bad-lists (READ)

15.5 Directories and PKI (READ)

15.5.1 Store Certificates with Subject or Issuer? (SKIP)

15.5.2 Finding Certificate Chains (SKIP)

15.6 PKIX and X.509 (READ)

15.6.1 Names (READ)

- What field(s) in a X.509 certificate could be used to hold a DNS name (e.g., www.somesecurebank.com if the subject is an on-line bank using HTTPS(SSL) for secure communication)?

15.6.2 OIDs (SKIP)

15.6.3 Specification of Time (SKIP)

15.7 X.509 and PKIX Certificates (READ PARTS BRIEFLY, see below)

A certificate should bind a subject's name with its public key. In its simplest form the certificate should contain

- the "public key",
- the "name/id" of the subject,
- the "name/id" of the issuer,
- an "expiration time" (and perhaps "issue time"),
- a "serial number", and
- the "signature" covering the other information.

(READ BRIEFLY through this section to try to find what X.509 fields hold that information; SKIP the rest.)

15.7.1 X.509 and PKIX CRLs (READ BRIEFLY)

- What do you think a CRL should contain? Are you able to find those fields in the provided list?

15.8 Authorization Futures (SKIP)