7 Key Management and PKIs

7.1 Key Management

Key Management

- For any use of cryptography, keys must be handled correctly.
- Symmetric keys must be kept secret.
  - They must be kept secret within the environment in which they are used.
  - They must be kept secret if communicated over a network.
- Similarly, private keys must be kept secret.
  - In general, private keys should not be sent over a network.
- Obviously, symmetric and private keys are much more vulnerable when they are being transferred, but secure local storage is also very important.

Key Management

- Public keys are made public and do not need to be kept secret.
  - However, they must be associated with the correct user.
  - This association must be established when a public key is first obtained (e.g., over a network).
  - In addition, once an association has been established, it must be correctly maintained.
  - For example, there is little point establishing an association when a public key is obtained and then storing the key and the user's name unprotected on a computer.
- A Public Key Infrastructure (PKI) is used to securely associate public keys with their owners.

Key Lifetime

- Keys should not be used indefinitely.
- If a key is used for a long time, then:
  - The more likely it is that it will be compromised.
  - The greater the loss if it is compromised.
  - It may become worthwhile for an adversary to expend the effort required to break it.
  - There will be more ciphertext available for a cryptanalyst to mount an attack.

- Therefore, keys should have a planned lifetime and need to be replaced.
  - Periodically, when they expire.
  - Immediately, if they are compromised or if a compromise is suspected.
  - Note that periodic replacement helps to limit the impact of a compromise.

Key Lifecycle

- Keys have a lifecycle. Typically, they are:
  1. Created
  2. Distributed
  3. Replaced
  4. Destroyed
- How long should a key be used before being replaced?
- This depends on:
  - The nature of the key (symmetric versus asymmetric keys).
  - The key length.
  - What the key is used for.
  - How much data is encrypted.
  - The value of the data being protected.

Key Usage

Session Keys

- Session keys are symmetric keys that are used for the duration of a connection or for a single message.
- They are automatically replaced.

Fixed Links

- Fixed links (e.g., leased lines) between computers are sometimes secured by hardware encryption devices based on symmetric encryption.
- The keys used within such devices need to be periodically replaced, e.g., every day. Depending on the technology, this can be a time consuming activity and may only be justified in high-value applications.
- The lifetime of the keys depends on the capacity of the link. A high-speed link requires more frequent key replacement.

Key Usage

File Storage

- Keys used to encrypt files on a computer also need to be periodically replaced.
- However, this can be difficult to achieve. We either have to remember the different keys used at different times or re-encrypt all the files when a key is replaced.

Key-Encryption Keys

- Some protocols implement key distribution by encrypting keys using other (key-encryption) keys.
- For example, session keys can be distributed by using a recipient's public key to encrypt them.
- Alternatively, some protocols use a master key to encrypt keys.
- Typically, key distribution is a relatively infrequent event and the amount of ciphertext available to mount an attack is relatively small.
- Therefore key-encryption keys do not have to be changed so frequently.
- Of course, if a master key is compromised, then so are all the keys that were distributed using the master key.

Key Usage

Asymmetric Key-pairs

- The main uses of asymmetric ciphers are authentication, digital signatures and the exchange of session keys.
- Given these uses, key pairs need to be resistant to attack for very long periods of time. With digital signatures, we may need key pairs to remain secure indefinitely so that signatures can be checked.
- In fact, asymmetric ciphers and key lengths are chosen to make them immune to attack for very long periods of time.
- It is still good practice to periodically replace asymmetric key pairs. However, their lifetime is typically anything from 1 to 5 (or even 10) years.
- These time periods make it essential that we can identify compromised keys.

7.2 Public Key Infrastructures

Public Key Infrastructures

There are a number of issues associated with the use of asymmetric keys.

1. The user of a public key assumes that the key belongs to some other entity (person, server, $\cdots$). This ownership relationship must be valid.
2. In general, a key pair will have some lifetime during which the public component can be used.
3. If non-repudiation services are to be implemented (e.g., if digital signatures are to have a legal status equivalent to a hand-written signature), then there needs to be some 3rd party involvement.
4. In a global environment (e.g., the Internet) there is a potentially huge user population.

Public Key Infrastructures
Key Ownership

- An entity owns a key pair if it controls the private key, i.e., only it can use the private key to perform encryption/decryption.
- If Eve can convince Bob that Eve's public key belongs to Alice, then Eve can (as far as Bob is concerned) sign messages, get access to session keys etc. as if she were Alice.

Key-Pair Lifetime

- A user of a public key needs to know when it can be used.
- Typically, when a key pair is generated it will have a relatively long planned lifetime (1 to 5 years). This information needs to be conveyed to users of a public key.
- If a private key is compromised or the need to use the key pair no longer exists (e.g., an employee leaves a company), then users of the public key need to be informed.

Public Key Infrastructures
Non-repudiation

- A trusted 3rd party could witness digital signatures by adding their own digital signature.
- However, a more practical approach would involve a trusted 3rd party certifying that a public key belonged to some entity and is currently valid. Then, for example, all messages signed by the owner of the key pair would be legally binding.

User Population

- Any infrastructure used to solve the problems of using public keys needs to be scalable to a global environment.

Public Key Infrastructures
These problems can be overcome by using a Public Key Infrastructure (PKI) to:

- Give names to entities.
- Certify public keys, i.e., to securely established an entity's ownership of a public key.
  - This is achieved by issuing certificates that bind names to public keys.
- Revoke certificates if the certified public key has been compromised or is no longer used.

7.3 Certificates and Certification

Certificates and Certification

- Before using a public key, we need at least the following information:
  1. The name of the entity that owns the key.
  2. The dates during which the key is valid.
  3. An indication as to whether or not the private key has been compromised or is no longer in use for some other reason.
- The first two pieces of information (1 and 2) are static and can be assigned when a key pair is generated.
- This can be done by some trusted entity issuing a certificate that certifies the relationship between a public key and its owner.

Certificates and Certification

- A certificate contains (at least):
  - A public key.
  - The name of the owner of the public key (i.e., the subject).
  - The dates during which the public key is valid.
  - The identity of the issuer of the certificate.
- To prevent a certificate from being modified, it is digitally signed by its issuer.
  - If we have an issuer's public key, we can check the integrity of a certificate.
  - Then, if we trust the issuer, we know that the given public key belongs to the subject of the certificate.

Certificates and Certification

- The third piece of information (3) required before a public key is used is more difficult to obtain.
  - By its nature this information cannot be included in a certificate.
  - There are various ways of supplying the information, but in all cases we need to revoke a certificate.
  - The issuer of a certificate needs some way of indicating that the certificate has been revoked and that the certificate should no longer be used as justification for believing that the public key is valid.
  - Note that the public key could still be used if there was another valid certificate!

PKIs and Certificates
All PKIs use some form of certificate. However, PKIs differ in a number of ways:

- The format for certificates and what information in addition to public keys can be certified.
  - Some certificates contain serial numbers.
  - Some contain details of the certification policy in force.
  - Some contain details of the uses to which a public key can be put.
- How entities are named (identified).
- How a user can determine if a certificate has been revoked.
  - Some PKIs maintain lists of revoked certificates.
  - Others allow a user to query a server.
- How trust is handled.
  - This is the most important difference between PKIs.
  - Some PKIs have a centralized view of trust in which TTPs act as Certification Authorities (CA), e.g., `X.509` PKI.
  - Others have a more distributed view in which users trust each other, e.g. the `PGP` PKI.

The `X.509` PKI

- `X.509` is the most widely used technology for implementing PKIs.
  - In fact, when most people talk about PKIs, they are talking about `X.509` PKIs.
- An `X.509` certificate contains the following fields:

Field	Description
Serial Number	The serial number of this certificate. Each CA gives their certificates a unique serial number.
Issuer	The name of the CA that issued of the certificate.
Subject	The name of the owner of the public key being certified.
Validity Dates	`from` and `to` dates defining the period of validity of the certificate.
Public Key	The public key being certified.
...	...
Signature	Issuer's signature of the certificate.

The `X.509` PKI

- We will represent a certificate as $\mathscr{C}[N, I, S, (F, T), K]$ where

  N is the serial number of the certificate.

  I is the name of the issuer.

  S is the name of the subject.

  (F, T) are the validity dates.

  K is the public-key being certified.

Certification Authorities (CAs) and Certificate Paths

- `X.509` certificates are issued by entities known as Certification Authorities (CAs).
- Normally a CA is a trusted party or in some cases, a trusted third party.
  - They can be trusted to bind a subject name to the proper public key.
  - The precise manner in which a CA checks these relationships is not standardized.
  - Some CAs do simple checks, while others require legally notified documents etc.
- When a certificate is obtained its signature must be checked by using the issuing CA's public key.
  - For this to work, we must have secure access to the CA's public key.
  - This requires us to get the CA's certificate.
  - The name of the CA is given by the issuer field of the certificate so, if necessary, the CA's certificate can be obtained from a directory.
- In general, the CA's certificate will have been issued by a higher-level CA and we will need to repeat the validation process.

Certification Authorities (CAs) and Certificate Paths

- A certificate path is a list of certificates $\langle \mathscr{C}_0, \mathscr{C}_1, \cdots \rangle$ such that the signature for $\mathscr{C}_i$ was generated by the private key corresponding to the public key certified by $\mathscr{C}_{i+1}$
- For example:

$$\langle \begin{array}{l} \mathscr{C}[\texttt{23}, \texttt{DCU CA}, \texttt{Geoff Hamilton}, (\texttt{Jan 2015}, \texttt{Dec 2019}), k^{+}_{GH}], \\ \mathscr{C}[\texttt{3452}, \texttt{VeriSign}, \texttt{DCU CA}, (\texttt{Jan 2000}, \texttt{Dec 2020}), k^{+}_{DCU}], \cdots \rangle \end{array}$$

- To be practical, certificate paths must be finite.
- This is achieved by having a Root CA.

  - A root CA issues a special self-signed root certificate that does not need to be checked.

- For example:

$$\langle \quad \begin{array}{l} \{|\mathscr{C}[\texttt{23},\texttt{DCU CA},\texttt{Geoff Hamilton},(\texttt{Jan 2015},\texttt{Dec 2019}),k^+_{GH}]|\}_{k^-_{DCU}} \\ \{|\mathscr{C}[\texttt{3452},\texttt{VeriSign},\texttt{DCU CA},(\texttt{Jan 2014},\texttt{Dec 2018}),k^+_{DCU}]|\}_{k^-_{VS}} \\ \{|\mathscr{C}[\texttt{2735},\texttt{VeriSign},\texttt{VeriSign},(\texttt{Jan 2000},\texttt{Dec 2020}),k^+_{VS}]|\}_{k^-_{VS}} \end{array} \quad \rangle$$

Certification Authorities (CAs) and Certificate Paths

- Consider a certificate path $\mathscr{P} = \langle \mathscr{C}_0, \cdots, \mathscr{C}_k \rangle$ where

$$\mathscr{C}_i = \mathscr{C}[N_i, I_i, S_i, (F_i, T_i), K_i]$$

- We can validate this path and extract the public key for S_0 using the following algorithm:

if $(now < F_k)$ **or** $(now > T_k)$ **then fail** ;
if $revoked(I_k, N_k)$ **then fail** ;
for $i := k-1$ **downto** 0 **do**
begin
  if $I_i \neq S_{i+1}$ **then fail** ;
  if $(now < F_i)$ **or** $(now > T_i)$ **then fail** ;
  if $revoked(I_i, N_i)$ **then fail** ;
  if not $validSig(\mathscr{C}_i, K_{i+1})$ **then fail** ;
end
return K_0 ;

`X.509` **Certificates**

- Root certificates are inherently insecure.
  - Anyone can simply generate a key-pair and issue a root certificate for any CA.
  - Therefore, they need to be protected and distributed in a secure manner.
- There are many `X.509` root CAs.
  - Different users of `X.509` have different CA Hierarchies, i.e., we have different `X.509` PKIs that use the same technology, but are managed and controlled by different organizations.
  - These hierarchies have different root certificates and have different policies.
- In theory, anyone can build a CA hierarchy and set themselves up as a CA.
  - However, in practice, a user will only be prepared to trust known CA hierarchies.

X.509 Certificates

- Each X.509 certificate contains a validity period.
  - Certificates should not be used before or after their validity period.
  - Once a certificate has expired, it is possible to issue a new certificate that binds the same public key to the subject.
- To handle revoked certificates, each CA maintains a Certificate Revocation List (CRL).
  - This is a list of the serial numbers of the certificates issued by the CA that have been revoked.
  - Only the serial numbers of revoked certificates that are otherwise within their validity period need to be held on a CRL.
  - When validating a certificate, a user should check that it does not appear on the issuer's CRL.
  - Typically, CRLs are updated periodically, e.g., every week.
- CRLs are probably the weakest aspect of X.509.
  - CRL can become extremely large.
  - An arguably better approach is to have servers which can be queried about the status of a certificate.