Week 11 revision notes

IFB240 Week 11 - Trust and Public Key Infrastructure (PKI)

Overview

This week focuses on **Public Key Infrastructure (PKI)**, the different trust models used for **public keys**, and how these models help ensure the integrity and authenticity of key ownership. We also discuss the **browser trust model** and real-world attacks involving certificate spoofing.

1. Key Management in Cryptography

Why is Key Management Important?

- **Key management** is crucial because the security of cryptographic systems hinges on the secrecy and integrity of the keys.
- If key management processes fail, it can lead to:
  - **Unauthorized access**: If private keys are compromised, sensitive data can be exposed.
  - **Data Loss**: Loss of keys can render encrypted data **irrecoverable**.
  - **Spoofing and Man-in-the-Middle Attacks**: Improper handling of public keys can allow attackers to impersonate legitimate users.

Keys Requiring Protection

- **Symmetric Ciphers**:
  - **Shared Secret Key**: Needs protection against disclosure and unauthorized access, as it's used for both encryption and decryption.
- **Asymmetric Ciphers**:
  - **Private Key**: Must remain confidential as it is used for decryption and signing.
  - **Public Key**: Needs integrity assurance, ensuring that the key hasn't been altered or replaced.

Techniques for Key Protection

- **Physical Security**: Store keys in secure hardware modules (e.g., **HSM**).
- **Key Encryption**: Encrypt keys with a master key to add an extra layer of security.
- **Digital Certificates**: Use certificates to ensure **authenticity** and **integrity** of public keys through certification authorities.

2. Public Key Trust Models

Challenges with Asymmetric Cryptography

- **Integrity of Public Keys**: Are you certain that the public key you have received is the correct one?
- **Trustworthiness**: Does the public key belong to the entity that you think it does, or has it been **spoofed** by an attacker?

Types of Trust Models

1. **User-Centric Trust Model**:
    - Users maintain a **key ring** containing public keys they trust.
    - Users are responsible for deciding which keys are trusted.
    - **Example**: **Pretty Good Privacy (PGP)**, where users sign each other's keys in a **web of trust**.
    - **Advantages**:
        - **Simple and Free**: Effective for small groups with mutual trust.
        - **User-Driven**: No centralized control.
    - **Disadvantages**:
        - **Reliance on Human Judgement**: Not suitable for **trust-sensitive** environments like finance.
        - **Scalability Issues**: Difficult to manage in larger settings.
2. **Trusted Authority Model (Certificate Authority - CA)**:
    - **Certificate Authorities** perform identity checks and issue digital certificates endorsing public keys.
    - **X.509 Certificates** are used to provide key information, including the identity of the owner, key validity period, and the **CA's digital signature**.
    - **Example**: **DigiCert** issuing a certificate for QUT’s Canvas.
    - **Advantages**:
        - Centralized and **structured trust**.
        - Suitable for large-scale deployment.
    - **Disadvantages**:
        - Requires trust in the **CA** itself.
        - Vulnerable to **single-point-of-failure** and potential compromise of a CA.
3. **Browser Trust Model**:
    - **Web browsers** come pre-installed with trusted CA certificates.
    - When accessing secure websites (e.g., QUT Canvas), browsers use these certificates to verify the authenticity of the server.
    - **Limitations**:
        - **User Control**: Users can import or remove certificates, which can compromise security if they don't fully understand the implications.

- Vulnerable to **man-in-the-middle (MITM)** attacks, especially when users accept unverified certificates without scrutiny.

---

3. Public Key Infrastructure (PKI)

What is PKI?

- **PKI** is a framework comprising:
  1. **Policies**: Define rules for certificate issuance and management.
  2. **Products**: Hardware/software to generate, store, and manage keys and certificates.
  3. **Procedures**: Manage the lifecycle of keys.

Digital Certificates

- **What is a Digital Certificate?**
  - A digital document issued by a **Certificate Authority (CA)** that binds a **public key** to the **identity** of the entity.
  - Follows the **X.509 standard** and contains key details like:
    - **Subject**: Entity that owns the public key.
    - **Issuer**: CA that issued the certificate.
    - **Validity Period**: Dates between which the certificate is valid.
    - **Digital Signature**: The CA's digital signature for the certificate.

Example: How to Use Alice's Digital Certificate

1. **Alice Generates a Key Pair**:
   - **Private Key**: Kept secret.
   - **Public Key**: Sent to the CA with identity information.
2. **CA Issues a Certificate**:
   - After verifying Alice's identity, the CA issues a **digital certificate** binding Alice's identity with her public key.
3. **Bob Encrypts a Message for Alice**:
   - Bob uses Alice's **public key** from her certificate to encrypt the message.
   - Only Alice can decrypt the message using her **private key**.

Certification Path and Trust

- **Certification Path**: The chain of trust between CAs to establish the validity of a public key.
  - A CA's **certificate** is often signed by another **higher-level CA**, creating a trust chain.
- **Root CAs**: Trust begins with the **root CA**, whose certificate is often pre-installed in browsers or devices.

Real-World Issues and Fraudulent Certificates

- **Fraudulent Certificates**:
  - Certificates can be **spoofed** to impersonate trusted entities.
  - **Example**: In **2016**, a DNS redirect moved Brazilian bank traffic to a spoofed site using free **Let's Encrypt** certificates. Users thought they were securely communicating with their bank but were actually interacting with attackers.
- **Revocation Methods**:
  - **Certificate Revocation Lists (CRLs)**: A list of certificates revoked before their expiry.
  - **Online Certificate Status Protocol (OCSP)**: A real-time check to verify if a certificate is valid.

---

4. Browser Trust Model

How Browsers Establish Trust

- **Pre-Installed Root Certificates**:
  - Browsers come with root certificates from trusted CAs (e.g., **DigiCert**).
  - When accessing a website, the server's certificate is verified against these trusted root certificates.

Limitations and Vulnerabilities

- **User Control Over Certificates**:
  - Users can manually add or remove certificates.
  - Attackers can exploit this by getting users to accept self-signed or fraudulent certificates.
- **Example of MITM Attack**:
  1. **Phishing Email**: A user receives a phishing email that directs them to a fake banking website.
  2. **Fake Certificate**: The fake site uses a self-signed certificate.
  3. **User Acceptance**: If the user manually accepts the certificate, they believe they are securely communicating with their bank.
  4. **Attacker Access**: The attacker receives the data (e.g., login credentials), which can be used later to access the legitimate bank account.

Real-World Incident: Brazilian Bank Attack (2016)

- Attackers used **DNS spoofing** to redirect traffic to a fake website.
- They obtained **Let's Encrypt** certificates, which appeared valid to users, leading them to believe their connection was secure.

---

5. Summary

Key Takeaways

- **Trust Models** are essential for managing public keys in asymmetric cryptography.
  - **User-Centric Model**: Users decide who to trust, suitable for small-scale systems.
  - **Trusted Authority Model**: CAs issue certificates, suitable for larger networks.
  - **Browser Trust Model**: Browsers come with a pre-installed set of trusted certificates.
- **PKI**:
  - Provides the infrastructure for **issuing**, **verifying**, and **revoking** digital certificates.
  - Involves **Certification Authorities (CAs)** that play a key role in binding public keys to the identities of their owners.
- **Digital Certificates**:
  - Provide **authentication**, **integrity**, and **non-repudiation**.
  - Users must trust the **CA** and ensure that the certification path is verifiable.
- **Security Implications**:
  - Improper key management or user acceptance of unverified certificates can lead to severe security breaches like **MITM** or **spoofing** attacks.

Practical Tips

- Always verify the **certification path** when presented with a new digital certificate.
- Be cautious about accepting **self-signed certificates** or those from **unknown CAs**.
- Use **OCSP** to check the validity of certificates before trusting them, especially in financial transactions.

These notes provide a comprehensive overview of **Trust and Public Key Infrastructure**, detailing how asymmetric cryptography ensures secure communication through **certificates** and **trusted models**. They highlight both the strengths and vulnerabilities in current trust practices, ensuring a thorough understanding for effective use in cybersecurity.