Unit III: Cryptography and Key Management

1. Basics of Cryptography

Cryptography is the science and art of securing communication and data in the presence of adversaries. It involves techniques for transforming information into an unreadable format, thereby ensuring confidentiality, integrity, authenticity, and non-repudiation.

Elaboration with Core Concepts and Terminology:

1. **Plaintext:** The original, readable message or data that is to be protected.
   - *Example:* "Transfer $1000 to Account 12345"
2. **Ciphertext:** The scrambled, unreadable version of the plaintext produced through encryption.
   - *Example (conceptual):* "Xmzftr$1000uzBdppunt2345"
3. **Encryption (Enciphering):** The process of converting plaintext into ciphertext using an algorithm and a key.
4. **Decryption (Deciphering):** The reverse process of converting ciphertext back into its original plaintext using an algorithm and a key.
5. **Cryptographic Algorithm (Cipher):** The mathematical function or set of rules used for encryption and decryption. It is assumed to be public knowledge.
   - *Examples:* DES, AES, RSA.
6. **Key:** A secret value (a string of bits) that is used by the cryptographic algorithm to control the encryption and decryption process. The security of the system rests on the **secrecy of the key**, not the algorithm.

Objectives of Cryptography (Beyond Confidentiality):

- **Confidentiality:** Ensures that information is accessible only to those authorized to have access (via encryption).
- **Data Integrity:** Ensures that data has not been altered in an unauthorized manner (via Hash Functions).
- **Authentication:** Verifies the identity of the sender or the origin of the data (via Digital Signatures).
- **Non-Repudiation:** Prevents an entity from denying having sent a message or performed an action (via Digital Signatures).

Conclusion: Cryptography is the foundational pillar of modern information security. It provides the tools to enforce the core principles of the CIA triad in a digital environment, transforming trust from a physical concept into a mathematically verifiable one.

---

2. Symmetric Cryptography (DES, Triple DES, AES)

Symmetric Cryptography, also known as **Secret-Key Cryptography**, uses a **single, shared secret key** for both encryption and decryption. The same key must be known and kept secret by both the sender and the receiver.

Core Characteristic:

- **Speed:** Very fast, suitable for encrypting large volumes of data.
- **Key Distribution Problem:** The major challenge is securely distributing the shared key to all parties without it being intercepted.

Elaboration on Key Algorithms:

1. DES (Data Encryption Standard):

- **History:** Developed by IBM in the 1970s and adopted as a U.S. federal standard.
- **Key Size:** 56-bit effective key length (64-bit with parity).
- **Mechanism:** A block cipher that encrypts data in 64-bit blocks using a complex series of permutations and substitutions (Feistel network).
- **Status: BROKEN.** With modern computing power (brute-force attacks), a 56-bit key can be cracked in a matter of hours. **It is considered obsolete and insecure.**

2. Triple DES (3DES):

- **History:** A stop-gap solution to overcome the weaknesses of DES without designing a completely new algorithm.
- **Mechanism:** It applies the DES algorithm three times to each data block.
    - **Encryption:** Ciphertext = E**K1**(D**K2**(E**K1**(Plaintext)))
    - **Decryption:** Plaintext = D**K1**(E**K2**(D**K1**(Ciphertext)))
- **Key Size:** Effectively 112 or 168 bits (using two or three independent keys).
- **Status:** Now considered deprecated and being phased out due to its relative slowness and vulnerability to certain attacks. Not recommended for new systems.

3. AES (Advanced Encryption Standard):

- **History:** Selected by NIST in 2001 after a public competition to replace DES. The winning algorithm was originally called Rijndael.
- **Key Sizes:** 128, 192, or 256 bits. (AES-256 is considered extremely secure, even against nation-states).
- **Mechanism:** A block cipher that encrypts data in 128-bit blocks. It uses substitution-permutation networks (SPN).
- **Status:** The **global standard** for symmetric encryption. It is efficient, highly secure, and widely used in various applications (Wi-Fi encryption, file encryption, VPNs, TLS/SSL).

Example Scenario:
Alice and Bob want to exchange encrypted messages. They first meet in person and agree on a secret AES-256 key. Now, when Alice sends a message, she encrypts it with this key. Bob receives the ciphertext and decrypts it using the same key. If Eve intercepts the ciphertext, she cannot read it without the secret key.

---

3. Asymmetric Cryptography: Public and Private Keys, RSA

Detailed Notes:

Asymmetric Cryptography, also known as **Public-Key Cryptography**, uses a pair of mathematically related keys: a **Public Key** and a **Private Key**.

Core Characteristic:

- **Key Pair:** What one key encrypts, only the other key in the pair can decrypt.
- **Public Key:** Can be freely distributed to anyone. It is used for encryption and verifying signatures.
- **Private Key:** Must be kept secret by the owner. It is used for decryption and creating signatures.
- **Solves Key Distribution:** Eliminates the need to share a secret key beforehand.
- **Speed:** Computationally intensive and much slower than symmetric encryption. Not suitable for bulk data encryption.

The Two Primary Functions:

1. **Confidentiality (Encryption):**

   - Anyone can encrypt a message with the **recipient's public key**.
   - Only the recipient, holding the corresponding **private key**, can decrypt it.
   - *Example:* Bob sends a secret message to Alice by encrypting it with Alice's public key. Only Alice's private key can decrypt it.
2. **Digital Signatures (Authentication/Non-Repudiation):**
   - The sender can create a signature for a message using their **own private key**.
   - Anyone can verify that signature using the **sender's public key**.
   - *Example:* Alice signs a document with her private key. Bob can verify the signature using Alice's public key, proving the document came from Alice and hasn't been tampered with.

RSA Algorithm (Rivest-Shamir-Adleman):

- **Basis:** Its security relies on the practical difficulty of factoring the product of two large prime numbers (the "factoring problem").
- **Key Generation:**
  1. Select two large, random prime numbers, `p` and `q`.
  2. Compute `n = p * q`. (`n` is the modulus, part of both public and private keys).
  3. Compute Euler's totient function: `ϕ(n) = (p-1)(q-1)`.
  4. Choose an integer `e` (public exponent) such that `1 < e < ϕ(n)` and `e` is coprime to `ϕ(n)`.
  5. Determine `d` (private exponent) such that `d * e ≡ 1 mod ϕ(n)`.
- **Keys:**
  - **Public Key:** The pair `(e, n)`.
  - **Private Key:** The pair `(d, n)`.
- **Encryption:** `Ciphertext = Plaintext^e mod n`
- **Decryption:** `Plaintext = Ciphertext^d mod n`

Conclusion: Asymmetric cryptography is a revolutionary concept that enables secure communication over insecure channels and forms the basis for digital trust on the internet.

---

4. Hash Functions

A cryptographic hash function is a one-way mathematical algorithm that takes an input (or 'message') of any length and returns a fixed-size string of bytes, typically a 'digest' or 'hash value'. The process is irreversible.

Core Properties:

1. **Deterministic:** The same input will always produce the same hash.
2. **Fast Computation:** The hash value should be quick to compute for any given input.
3. **Pre-image Resistance (One-Way):** It should be computationally infeasible to reverse the function and find the original input from its hash value.
4. **Small Change, Big Difference (Avalanche Effect):** A tiny change in the input (even one bit) should produce a drastically different hash value.
5. **Collision Resistance:** It should be extremely difficult to find two different inputs that produce the same hash output.

Common Hash Algorithms:

- **MD5 (Message-Digest 5):** Produces a 128-bit hash. **Considered broken and insecure** due to widespread collision vulnerabilities.
- **SHA-1 (Secure Hash Algorithm 1):** Produces a 160-bit hash. **Also considered insecure** and deprecated.
- **SHA-2 Family:** Includes SHA-256 and SHA-512. Currently **secure and widely used** (in Bitcoin, TLS certificates).
- **SHA-3 (Keccak):** The latest standard, based on a different structure, providing another secure option.

Applications of Hash Functions:

- **Password Storage:** Websites store the hash of your password, not the password itself. During login, they hash your input and compare it to the stored hash.
- **Data Integrity Verification:** To verify a downloaded file, you can compare its hash with the one provided by the publisher. If they match, the file is intact.
- **Digital Signatures:** Hash functions are used to create a compact digest of a message, which is then signed (see next topic).
- **Blockchain:** Used extensively to link blocks and maintain the integrity of the chain.

Example:

- **Input (Message):** `"Hello World"`

- **SHA-256 Hash:** `a591a6d40bf420404a011733cfb7b190d62c65bf0bcda32b57b277d9ad9f146e`
- **Input (Message):** `"hello world"` (changed 'H' to 'h')
- **SHA-256 Hash:** `b94d27b9934d3e08a52e52d7da7dabfac484efe37a5380ee9088f7ace2efcde9`
  (Notice the completely different output for a minor change)

5. Digital Signatures

A digital signature is a cryptographic technique that provides authentication, data integrity, and non-repudiation. It is the digital equivalent of a handwritten signature or a sealed stamp, but with far more security.

How it Works (Combining Hash Functions & Asymmetric Cryptography):

Signing a Message (by the Sender, Alice):

1. **Hash:** The original message is passed through a cryptographic hash function (like SHA-256) to produce a fixed-size message digest.
2. **Encrypt:** This message digest is then encrypted using the **sender's (Alice's) private key**. This encrypted hash is the **digital signature**.
3. **Send:** The original message and the digital signature are sent to the receiver (Bob).

Verifying the Signature (by the Receiver, Bob):

1. **Decrypt:** Bob uses the **sender's (Alice's) public key** to decrypt the digital signature. This reveals the original message digest that Alice computed.
2. **Re-compute Hash:** Bob independently takes the received message and passes it through the same hash function (SHA-256) to compute a new message digest.
3. **Compare:** Bob compares the newly computed digest with the one he obtained by decrypting the signature.
   - If they **match exactly**, it proves:
     - **Authentication:** The message was signed by Alice (only she has her private key).
     - **Integrity:** The message was not altered in transit (the hash would be different).
     - **Non-Repudiation:** Alice cannot deny having sent the message, as no one else could have created a signature that verifies with her public key.

Conclusion: Digital signatures are a cornerstone of secure digital transactions, enabling trust in e-commerce, digital contracts, and software distribution.

6. PKI (Public Key Infrastructure)

A Public Key Infrastructure (PKI) is a comprehensive framework of policies, roles, hardware, software, and procedures used to create, manage, distribute, use, store, and revoke digital certificates. It solves the fundamental problem in asymmetric cryptography: **"How can you be sure that a public key truly belongs to the person or entity it claims to belong to?"**

Core Components of PKI:

1. **Certificate Authority (CA):** A trusted third-party entity that issues and manages digital certificates. It is the root of trust. (e.g., DigiCert, Let's Encrypt, IdenTrust).
2. **Registration Authority (RA):** Acts as the verifier for the CA. It receives certificate requests, authenticates the identity of the applicant, and then forwards the request to the CA.
3. **Digital Certificate:** An electronic "passport" or "ID card" that binds a public key to an identity (a person, server, or company). It is digitally signed by the CA.
   - The most common standard is **X.509**.
4. **Certificate Repository (CR):** A publicly accessible database that stores and distributes certificates and Certificate Revocation Lists (CRLs).
5. **Certificate Revocation List (CRL):** A list of certificates that have been revoked before their expiration date (e.g., because the private key was compromised).

The PKI Process in Action (Securing a Website with HTTPS):

1. **Application:** A company (e.g., `www.example.com`) generates a key pair and submits a Certificate Signing Request (CSR) containing its public key and identity to a CA.
2. **Validation:** The CA (through its RA) verifies that the company actually owns the `example.com` domain.
3. **Issuance:** The CA creates an X.509 certificate for `example.com`, which includes the company's public key and identity. The CA then **signs this certificate with its own private key**.
4. **Installation:** The company installs this certificate on its web server.
5. **Trust Verification (by User's Browser):**

- When you connect to `https://www.example.com`, the server sends its certificate to your browser.
- Your browser comes pre-installed with public keys of major CAs (a "Trust Store").
- The browser uses the CA's public key to verify the CA's digital signature on the certificate.
- If the signature is valid, the browser trusts that the public key in the certificate genuinely belongs to `example.com`. A secure TLS connection is then established.

Conclusion: PKI is the invisible trust fabric of the internet. It enables secure web browsing (HTTPS), secure email, digital signatures for documents, and code signing. Without PKI, we would have no reliable way to verify identities online, making e-commerce and secure communication impossible.