# UNIT-III

## 1. Cryptographic Hash Functions

### Introduction

A Cryptographic Hash Function is a mathematical algorithm that converts an input message of arbitrary length into a fixed-length output called a **hash value**, **message digest**, or **fingerprint**.

### Characteristics of Hash Functions

1. **Variable Input Size** – Accepts messages of any length.
2. **Fixed Output Size** – Produces a fixed-length digest.
3. **Efficient Computation** – Easy to calculate hash values.
4. **Pre-image Resistance** – Difficult to determine original message from hash.
5. **Second Pre-image Resistance** – Difficult to find another message with the same hash.
6. **Collision Resistance** – Difficult to find two different messages producing the same hash.

### Applications

- Password Storage
- Digital Signatures
- Message Authentication
- Data Integrity Verification
- Blockchain Technology
- Secure Communication Protocols

### Hash Function Process

Message → Hash Algorithm → Message Digest

Example:

Input: Hello

Output: 185f8db32271fe25f561a6fc938b2e264306ec304eda518007d1764826381969

## 2. Message Authentication

### Definition

Message Authentication ensures that a received message:

1. Comes from the claimed sender.
2. Has not been modified during transmission.

### Security Services Provided

- Data Integrity
- Source Authentication
- Non-repudiation (with digital signatures)

**Authentication Techniques**

**1. Message Encryption**

Entire message is encrypted.

**2. Message Authentication Code (MAC)**

A secret key and message are used to generate an authentication tag.

**3. Hash Function Based Authentication**

Hash values verify message integrity.

**Authentication Model**

Sender:

Message + Secret Key

↓

Authentication Function

↓

Authentication Tag

Receiver:

Message + Secret Key

↓

Authentication Function

↓

Compare Tags



**3. Secure Hash Algorithm (SHA-512)**

**Introduction**

SHA-512 belongs to the SHA-2 family developed by the National Institute of Standards and Technology.

**Features**

| Property | Value |
|---|---|
| Output Size | 512 bits |
| Block Size | 1024 bits |
| Word Size | 64 bits |
| Number of Rounds | 80 |
| Security Level | Very High |

**SHA Family**

- SHA-1 (Deprecated)
- SHA-224
- SHA-256
- SHA-384
- SHA-512

**SHA-512 Structure**

**Step 1: Padding**

Append bits so message length becomes:

[
Length \equiv 896 \pmod {1024}
]

**Step 2: Append Length**

Add 128-bit representation of original message length.

**Step 3: Initialize Buffer**

Eight 64-bit registers:

H0 H1 H2 H3 H4 H5 H6 H7

**Step 4: Process Message Blocks**

Message divided into 1024-bit blocks.

**Step 5: Compression Function**

80 rounds of operations.

**Step 6: Generate Final Digest**

Concatenate registers.

Output:

[
512 \text{ bits}
]



**Advantages**

- Strong security
- Resistant to practical collision attacks
- Widely used in TLS, VPNs, and digital certificates

**Applications**

- Digital Signatures
- SSL/TLS
- File Integrity Verification
- Blockchain Systems

**4. Message Authentication Codes (MAC)**

**Definition**

A Message Authentication Code (MAC) is a cryptographic checksum generated using:

- Message
- Secret Key

**Formula**

MAC = F(K,M)

Where:

- K = Secret Key
- M = Message

**Objectives**

1. Data Integrity
2. Data Origin Authentication

**MAC Verification**

Sender:

Message + Key

↓

MAC

Receiver:

Message + Key

↓

Generate MAC

↓

Compare

**5. Authentication Requirements**

A secure authentication system should satisfy:

**1. Message Integrity**

Message must not be altered.

**2. Authentication**

Verify sender identity.

**3. Confidentiality**

Prevent unauthorized access.

**4. Non-Repudiation**

Sender cannot deny transmission.

**5. Replay Protection**

Prevent reuse of old messages.

**Threats**

**Masquerade**

Attacker pretends to be a legitimate user.

**Content Modification**

Message contents are altered.

**Sequence Modification**

Messages reordered or deleted.

**Timing Modification**

Messages delayed or replayed.

**Repudiation**

Sender denies sending message.

**6. HMAC (Hash-based Message Authentication Code)**

**Introduction**

HMAC combines:

- Cryptographic Hash Function
- Secret Key

**Standard**

[
HMAC(K,M)=H[(K^+ \oplus opad) , || , H((K^+ \oplus ipad)||M)]
]

**Architecture**

Message + Secret Key

↓

Inner Hash

↓

Outer Hash

↓

HMAC

**Components**

**K+**

Padded secret key

**ipad**

Inner padding constant

0x36 repeated

**opad**

Outer padding constant

0x5C repeated

**Advantages**

- High Security
- Efficient
- Easy Implementation
- Resistant to Extension Attacks

**Applications**

- TLS
- IPSec
- VPN
- Cloud Security
- API Authentication

**7. CMAC (Cipher-based Message Authentication Code)**

**Introduction**

CMAC is a MAC generated using a block cipher such as:

Advanced Encryption Standard

**Working**

**Step 1**

Generate subkeys K1 and K2.

**Step 2**

Divide message into blocks.

**Step 3**

Encrypt blocks using AES.

**Step 4**

Generate final authentication tag.

**Structure**

Message Blocks

↓

AES Operations

↓

Authentication Tag

**Features**

- Based on Symmetric Encryption
- NIST Standardized
- Strong Authentication
- Suitable for Embedded Systems

**Advantages**

- Strong Security
- Efficient Hardware Implementation
- Compatible with AES

**Applications**

- Smart Cards
- Wireless Networks
- Banking Systems
- IoT Security

**8. Digital Signatures**

**Definition**

A Digital Signature is a cryptographic mechanism used to verify:

- Authenticity
- Integrity
- Non-repudiation

**Signature Process**

**Signing**

Message
↓
Hash Function
↓
Message Digest
↓
Private Key Encryption
↓
Digital Signature

**Verification**

Message
↓
Hash Function
↓
Digest
Signature
↓
Public Key Decryption
↓

Digest -→ Compare Digests

**Services Provided**

1. Authentication
2. Integrity
3. Non-repudiation

**Requirements**

- Signature must depend on message.
- Must use unique signer information.
- Easy to verify.
- Difficult to forge.

**Applications**

- E-Governance . Banking
- Software Distribution
- Digital Certificates
- E-Commerce

**Key Management and Distribution**

**Learning Objectives**

After studying this unit, students will be able to:

- Understand key management concepts.
- Explain symmetric key distribution methods.
- Describe key distribution using asymmetric encryption.
- Understand public key distribution techniques.
- Explain Kerberos authentication protocol.
- Describe X.509 authentication service.
- Understand Public Key Infrastructure (PKI).

**1. Introduction to Key Management**

**Definition**

Key Management is the process of:

- Generating cryptographic keys
- Distributing keys
- Storing keys securely
- Updating keys
- Revoking keys
- Destroying expired keys

A cryptographic system is only as secure as its key management mechanism.

**Key Management Life Cycle**

**Stages**

**1. Key Generation**

Creation of secret/public keys.

**2. Key Distribution**

Secure transfer of keys to users.

**3. Key Storage**

Protection of keys against unauthorized access.

**4. Key Usage**

Using keys for encryption/decryption.

**5. Key Revocation**

Invalidating compromised keys.

**6. Key Destruction**

Permanent removal of expired keys.

**2. Symmetric Key Distribution Using Symmetric Encryption**

**Introduction**

In symmetric cryptography:

- Same key is used for encryption and decryption.
- Both sender and receiver must possess the secret key.

**Problem**

How can two users securely obtain the same secret key?

**Method 1: Physical Delivery**

A trusted person physically delivers the key.

**Advantages**

- Simple
- Highly secure

**Disadvantages**

- Expensive
- Not practical for large networks

**Method 2: Trusted Third Party**

A Key Distribution Center (KDC) generates and distributes session keys.

**Working**

**Step 1**

User A shares master key KA with KDC.

**Step 2**

User B shares master key KB with KDC.

**Step 3**

A requests communication with B.

**Step 4**

KDC generates session key KS.

**Step 5**

KDC sends KS securely to both users.

**Diagram**

A --------> KDC <-------- B

Session Key KS

**Advantages**

- Efficient
- Centralized control
- Suitable for large networks

**Disadvantages**

- Single point of failure
- KDC compromise affects entire system

**3. Symmetric Key Distribution Using Asymmetric Encryption**

**Concept**

Public-key cryptography can be used to distribute symmetric keys securely.

**Process**

**Step 1**

Receiver publishes public key.

**Step 2**

Sender generates session key.

**Step 3**

Sender encrypts session key using receiver's public key.

**Step 4**

Receiver decrypts session key using private key.

**Communication Model**

Session Key

↓

Encrypted with Public Key

↓

Sent to Receiver

↓

Private Key Decryption

↓

Shared Secret Key

**Advantages**

- No prior secret sharing required
- Secure over public networks

**Disadvantages**

- Public-key operations are computationally expensive

**4. Distribution of Public Keys**

**Introduction**

Public-key cryptography requires users to obtain authentic public keys.

**Challenge**

How do we ensure that a public key actually belongs to the claimed user?

**Methods of Public Key Distribution**

**1. Public Announcement**

Users publicly announce their public keys.

**Example**

Publishing on:

- Email
- Websites
- Social media

**Problem**

An attacker may replace the public key.

**Vulnerability**

Man-in-the-Middle Attack

**2. Publicly Available Directory**

A trusted directory maintains public keys.

**Working**

- Users register public keys.
- Directory verifies identity.
- Users retrieve keys when required.

**Advantages**

- Centralized management
- Easier verification

**Disadvantages**

- Directory must be trusted

**3. Public-Key Authority**

A trusted authority provides public keys on request.

**Process**

1. User requests public key.
2. Authority verifies request.
3. Authority sends authenticated key.

**Advantages**

- Strong security
- Prevents key substitution attacks

**Disadvantages**

- Authority must always be online

**4. Public-Key Certificates**

Most widely used method.

**Concept**

A certificate binds:

- User identity
- Public key

Digitally signed by a trusted authority.

**Components**

- User Name
- Public Key
- Expiration Date
- Certificate Number

- Digital Signature

**5. Kerberos Authentication Service**

**Introduction**

Kerberos is a trusted third-party authentication protocol developed at Massachusetts Institute of Technology.

**Purpose**

Provides:

- Authentication
- Single Sign-On (SSO)
- Secure communication

**Kerberos Components**

**1. Client**

User requesting service.

**2. Authentication Server (AS)**

Verifies user identity.

**3. Ticket Granting Server (TGS)**

Issues service tickets.

**4. Service Server**

Provides requested service.

**Kerberos Architecture**

Client
|
v
Authentication Server (AS)
|
v
Ticket Granting Server (TGS)
|
v
Application Server

**Kerberos Operation**

**Step 1: User Login**

Client sends authentication request.

**Step 2: AS Response**

AS sends:

- Ticket Granting Ticket (TGT)
- Session Key

**Step 3: Request Service Ticket**
Client sends TGT to TGS.

**Step 4: TGS Response**
TGS issues service ticket.

**Step 5: Access Service**
Client presents service ticket to server.

**Step 6: Authentication Completed**
Secure communication begins.

**Advantages**

- Mutual Authentication
- Single Sign-On
- Password not transmitted
- Strong security

**Limitations**

- Requires synchronized clocks
- KDC availability is critical

**6. X.509 Authentication Service**

**Introduction**

X.509 is an international standard for public-key certificates developed by the International Telecommunication Union.

Used in:

- SSL/TLS
- HTTPS
- VPNs
- Email Security

**X.509 Certificate**

A digital certificate containing identity information and public key.

**X.509 Certificate Format**

**Fields**

| Field | Description |
|---|---|
| Version | Certificate version |
| Serial Number | Unique identifier |
| Signature Algorithm | Algorithm used |
| Issuer Name | Certificate Authority |
| Subject Name | Certificate Owner |
| Public Key | User's Public Key |
| Validity Period | Expiration Dates |
| Digital Signature | CA Signature |

**Certificate Creation**

**Step 1**

User generates key pair.

**Step 2**

Certificate request sent to CA.

**Step 3**

CA verifies identity.

**Step 4**

CA signs certificate.

**Step 5**

Certificate issued.



**Certificate Verification**

Receiver:

1. Checks CA signature.
2. Checks certificate validity.
3. Verifies certificate owner.
4. Extracts public key.

**Authentication Types in X.509**

**One-Way Authentication**

Only sender authenticated.

**Two-Way Authentication**

Both sender and receiver authenticated.

**Three-Way Authentication**

Mutual authentication without synchronized clocks.

**Advantages**

- Global standard
- Strong authentication
- Supports digital signatures

**7. Public Key Infrastructure (PKI)**

**Definition**

Public Key Infrastructure (PKI) is a framework for:

- Creating
- Managing
- Distributing
- Storing
- Revoking

digital certificates and public keys.

**Objectives of PKI**

- Authentication
- Confidentiality
- Integrity
- Non-Repudiation

**PKI Components**

**1. Certificate Authority (CA)**

Trusted organization issuing certificates.

Examples:

- DigiCert
- GlobalSign
-



**2. Registration Authority (RA)**

Verifies user identity before certificate issuance.

**3. Certificate Repository**

Stores certificates.

**4. Certificate Revocation List (CRL)**

Contains revoked certificates.

**5. End Users**

Certificate owners.

**PKI Architecture**

User

|

v

Registration Authority

|

v

Certificate Authority

|

v

Certificate Repository

**PKI Operations**

**Certificate Enrollment**

User requests certificate.

**Certificate Issuance**

CA verifies and issues certificate.

**Certificate Distribution**

Certificate stored in repository.

**Certificate Validation**

Users verify certificate.

**Certificate Revocation**

Invalid certificates added to CRL.

**Applications of PKI**

**Secure Web Browsing**

HTTPS

**Digital Signatures**

Electronic documents

**Secure Email**

S/MIME

**VPN Authentication**

Remote access security

**E-Commerce**

Secure transactions

**Comparison of Key Distribution Methods**

| Method | Security | Cost | Scalability |
|---|---|---|---|
| Physical Delivery | High | High | Low |
| Trusted KDC | High | Medium | High |
| Public Announcement | Low | Low | High |
| Public-Key Authority | High | Medium | Medium |
| Certificates (X.509) | Very High | Medium | Very High |

**Comparison: Kerberos vs X.509**

| Feature | Kerberos | X.509 |
|---|---|---|
| Authentication Method | Symmetric Key | Public Key |
| Trusted Entity | KDC | CA |
| Certificates | Not Required | Required |
| Single Sign-On | Supported | Not Directly |
| Scalability | Medium | High |
| Internet Usage | Limited | Widely Used |