INTRODUCTION TO SSL

22Z208- AKSHARAA P

INTRODUCTION TO SSL

Definition:

- SSL (Secure Sockets Layer) is a **cryptographic protocol** designed to provide **secure communication** over a **computer network**, especially the Internet.
- Developed by **Netscape Communications** in the mid-1990s.
- Later replaced by **Transport Layer Security (TLS)**, but the term “SSL” is still widely used.
- **Purpose:** Protect data confidentiality and integrity between client and server.

BASIC OPERATION

Authentication:

Verify the identity of the communicating parties (using digital certificates).

Data Encryption:

Encrypt data to ensure privacy during transmission.

Data Integrity:

Ensure that the data is not altered during transfer.

Confidentiality:

Prevent eavesdropping by third parties.

REAL LIFE EXAMPLES

E-commerce Websites: Protecting payment and personal information.
Example: Amazon, Flipkart, eBay

Online Banking: Securing login and transactions.
Example: HDFC Bank, PayPal.

Email Services: Encrypting communication (Gmail, Outlook).

Social Media Platforms: Protecting user data (Facebook, Instagram).

Government Portals: Secure citizen data exchange.

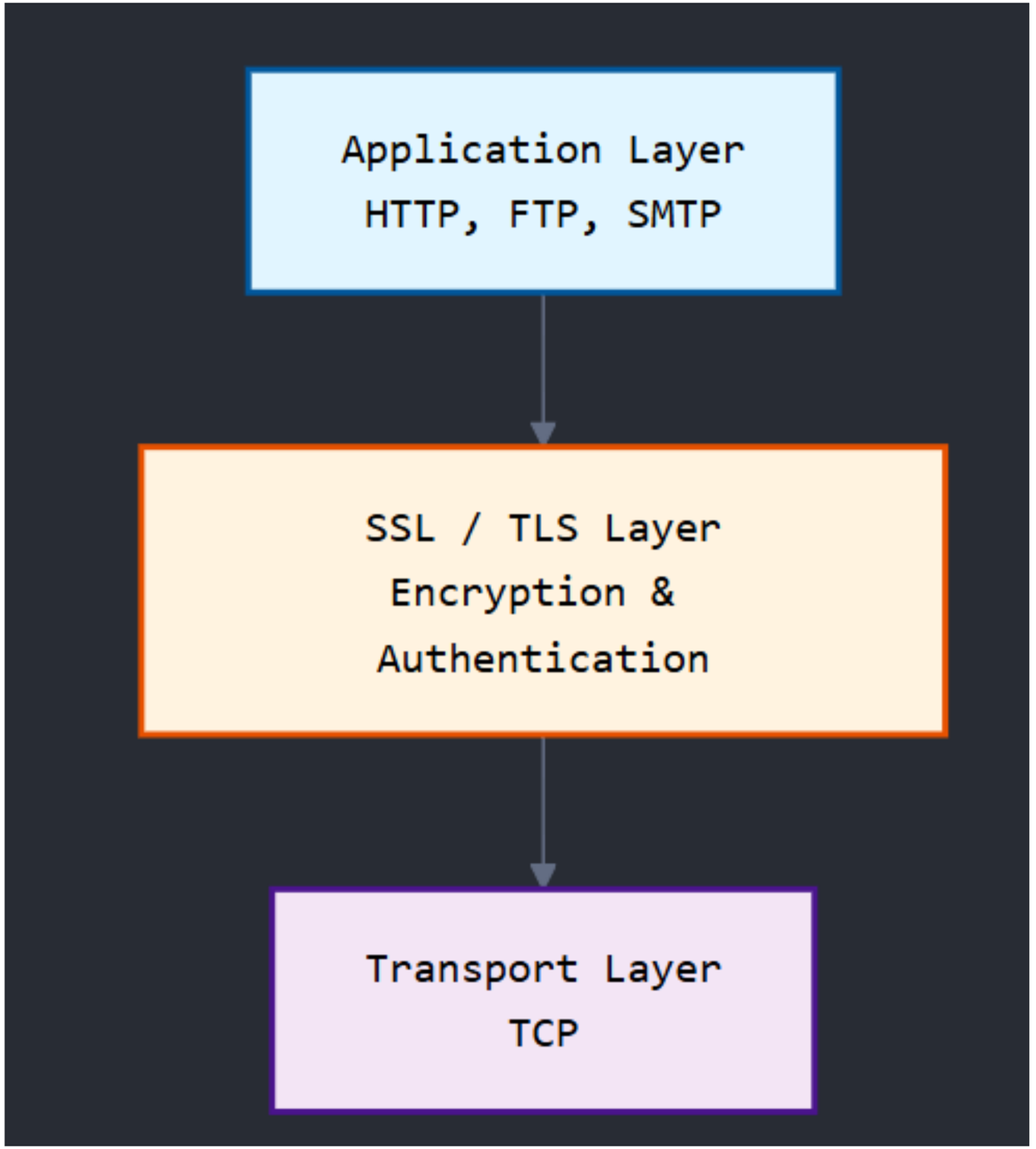
SSL ARCHITECTURE AND COMPONENTS

22Z224 - HARSHINI S P

OVERVIEW OF SSL ARCHITECTURE

- The **Secure Sockets Layer (SSL)** architecture is designed to provide a secure communication channel between a **client** (usually a web browser) and a **server** (like a website).

- The SSL layer ensures that data transmitted between client and server remains:
  - Confidential (encrypted)
  - Authenticated (verified identity)
  - Tamper-proof (data integrity)


Application Layer
HTTP, FTP, SMTP
SSL / TLS Layer
Encryption &
Authentication
Transport Layer
TCP

SSL PROTOCOL STACK

The SSL protocol suite consists of two main sub-protocols, each performing a specific role:

SSL RECORD PROTOCOL

Role:

- Provides confidentiality and integrity for application data.
- Divides the data into manageable blocks (records).
- Applies compression, MAC (Message Authentication Code), and encryption before transmission.

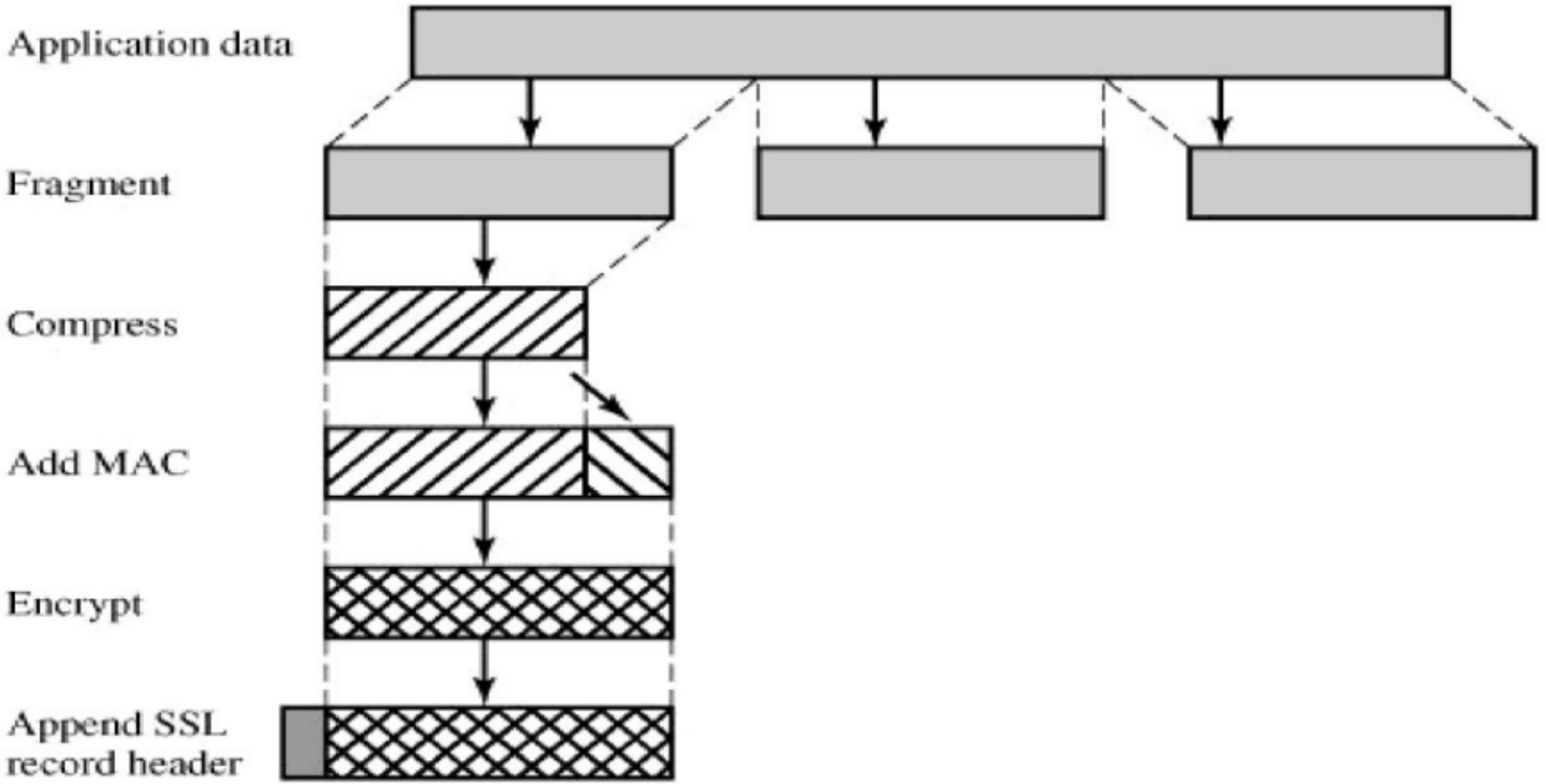
SSL RECORD PROTOCOL

Main Steps:

- **Fragmentation –** Breaks large messages into smaller chunks.
- **Compression –** reduces data size.
- **MAC generation –** Adds a message authentication code to ensure integrity.
- **Encryption –** Uses symmetric encryption to secure data.
- **Transmission –** Sends the encrypted data over TCP.

At receiver's end:

The process is reversed — decryption → MAC verification → decompression → reassembly.


Application data
Fragment
Compress
Add MAC
Encrypt
Append SSL
record header

SSL HANDSHAKE PROTOCOL

Role:

This protocol establishes a secure session before any data is transmitted.

Functions:

- Authenticates client and server identities.
- Negotiates encryption algorithms and keys.
- Establishes shared session keys for encryption.

KEY COMPONENTS OF SSL ARCHITECTURE

a) SSL Certificates

- Digital documents that verify a website's identity and ownership.
- Issued by Certificate Authorities (CAs).
- Contain **public key, organization name**, and **validity period.**
- The server presents this certificate during the handshake.

b) Public Key Infrastructure (PKI)

- The framework enabling secure key and certificate management.
- Components of PKI:
  - **Certificate Authority (CA):** Issues and signs certificates.
  - **Registration Authority (RA):** Verifies identity before certificate issuance.
  - **Certificate Repository:** Stores issued certificates and revocation lists.

KEY COMPONENTS OF SSL ARCHITECTURE

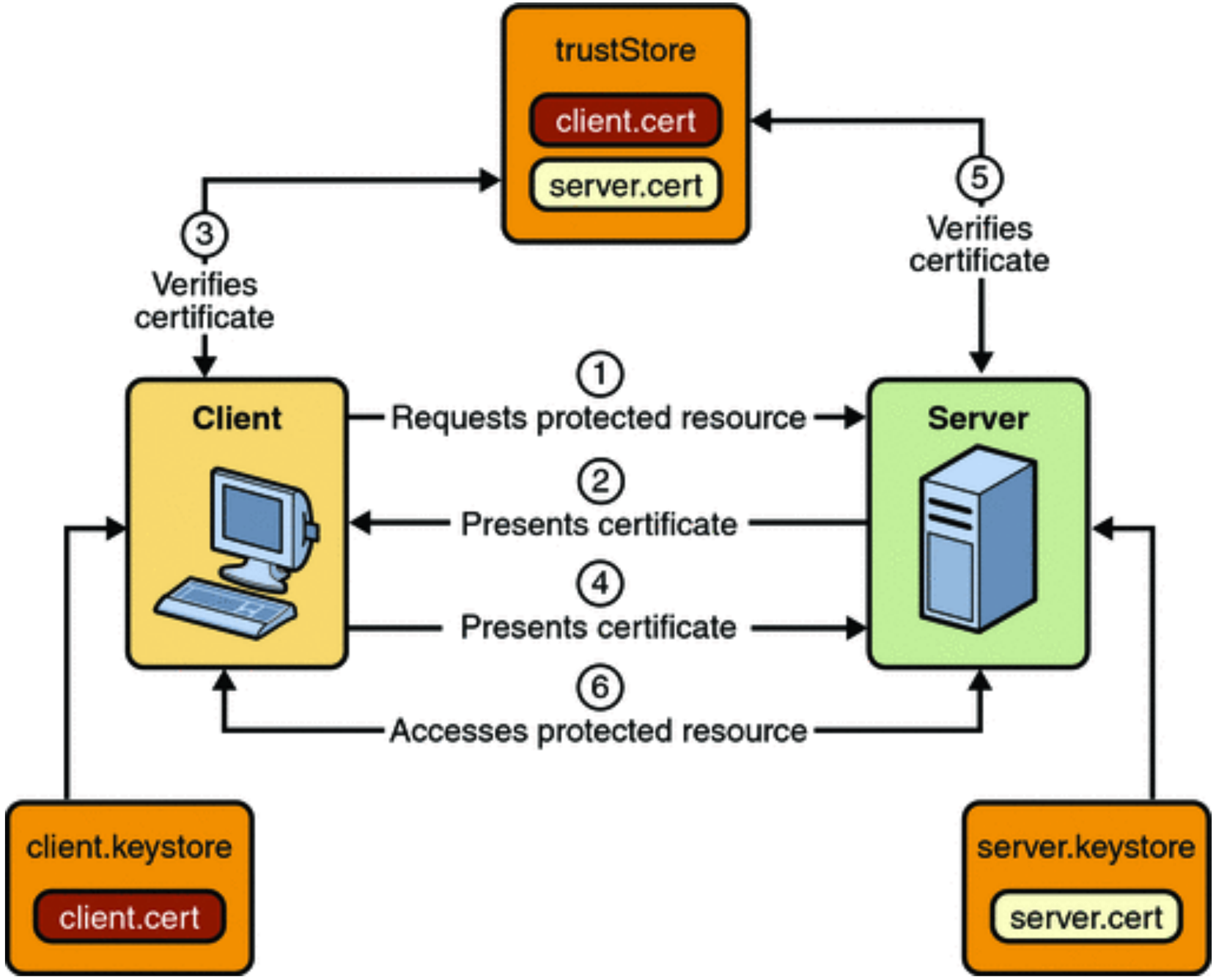
c) Digital Signatures

- Used to verify authenticity and integrity of transmitted data or certificates.
- A CA signs certificates using its private key, and browsers verify using the CA's public key.
- Ensures that:
  - The certificate wasn't tampered with.
  - It came from a legitimate authority.

SSL CLIENT–SERVER MODEL

Communication Flow:

1. Client (Browser) initiates a connection to the Server (Website) via https://.
2. Server sends its SSL certificate.
3. Client verifies certificate authenticity.
4. Once verified, both agree on encryption algorithms and generate session keys.
5. Encrypted communication begins.


trustStore
client.cert
server.cert
3
Verifies certificate
5
Verifies certificate
Client
1
Requests protected resource
Server
2
Presents certificate
4
Presents certificate
6
Accesses protected resource
client.keystore
client.cert
server.keystore
server.cert

SSL HANDSHAKE PROCESS

22Z269 - THEECHANAA RA

What is the SSL HANDSHAKE?

- The SSL Handshake is the process that occurs before data transmission.
- It allows the client and server to:
  - Verify each other’s identity (Authentication)
  - Agree on encryption algorithms and keys
  - Establish a secure communication channel
- Happens automatically when you connect to a secure site (e.g., https://)

Phases of SSL HANDSHAKE

01 **Client Hello**

Client initiates connection, sends supported SSL/TLS versions, cipher suites, and a random value.

02 **Server Hello**

Server selects protocol and cipher, sends its own random value and session ID.

03 **Certificate Exchange**

Server sends its digital certificate to authenticate itself to the client.

04 **Server Key Exchange**

Server provides additional key-exchange parameters and signs them.

05 **Server Hello Done**

Indicates the server has completed its initial setup messages.

06 **Client Key Exchange**

Client sends pre-master secret (encrypted with server’s public key) or its DH public value.

07 **Session Key Generation**

Both sides derive the master secret and session keys using random values and the pre-master secret.

08 **Change Cipher Spec**

Client (and then server) notify that subsequent messages will be encrypted using the session keys.

09 **Finished Messages**

Both sides exchange encrypted “Finished” messages verifying the handshake integrity.

10 **Secure Communication Begins**

All future data is symmetrically encrypted using the shared session keys.

Step	Client	Direction	Message	Direction	Server
1			Client Hello	>	
2		<	Server Hello		
3		<	Certificate		
4		<	Server Key Exchange		
5		<	Server Hello Done		
6			Client Key Exchange	>	
7			Change Cipher Spec	>	
8			Finished	>	
9		<	Change Cipher Spec		
10		<	Finished		

Role of Encryption in SSL HANDSHAKE

01 **Asymmetric encryption:** used for authentication and secure key exchange (RSA, DHE, ECDHE).

02 **Symmetric encryption:** used after handshake for fast, secure data transfer (AES, 3DES).

03 **Hashing/MAC:** ensures message integrity (SHA, MD5).

CRYPTOGRAPHIC ALGORITHMS USED IN SSL

22Z252 - Rithaniyaa B

TYPES OF CRYPTOGRAPHIC ALGORITHMS IN SSL

01 Asymmetric Encryption

02 Symmetric Encryption

03 Hashing Algorithms

Each type performs a different function in protecting communication.

Asymmetric Encryption

- Also called public-key cryptography.
- Used during the SSL handshake to securely exchange keys and verify identity.
- Works with a pair of keys: public key and private key.
- Examples:
  - RSA (Rivest–Shamir–Adleman)
  - DSA (Digital Signature Algorithm)
  - ECDSA (Elliptic Curve DSA)

Symmetric Encryption

- Used after the handshake phase.
- The same key is used for both encryption and decryption.
- It is fast and efficient for large data transfer.

Examples:

- AES (Advanced Encryption Standard)
- 3DES (Triple Data Encryption Standard)

Hashing Algorithms

- Used to ensure that the message is not changed during transfer.
- Converts data into a fixed-size hash value.
- Even a small change in data produces a different hash.

Examples:

- SHA (Secure Hash Algorithm)
- MD5 (Message Digest 5)

How SSL Uses Hybrid Encryption

- SSL combines asymmetric and symmetric cryptography to balance security and performance.
- Handshake phase (Asymmetric encryption):
  - Client and server exchange keys securely using RSA or ECDSA.
  - A shared session key is generated.
- Data transfer phase (Symmetric encryption):
  - The session key is used with AES (or 3DES) to encrypt data efficiently.

Why hybrid?

- Asymmetric encryption is secure but slow.
- Symmetric encryption is fast but needs a secure key exchange

Key Exchange and Session Keys

- Key exchange ensures the session key is shared safely between client and server.
- Session key is a temporary key used only for one SSL session.
- After the session ends, the key is discarded.

Advantages:

- Protects each communication session separately.
- Prevents attackers from using old keys.
- Provides forward secrecy when new keys are used each time.

SSL CERTIFICATES AND AUTHENTICATION

22Z201 - Abinav P

WHAT ARE SSL CERTIFICATES ?

- SSL (Secure Sockets Layer) / TLS ensures secure communication between browser and server.
- Certificates are digital documents proving website identity and enabling data encryption.
- They contain:
  - Domain name
  - Organization details
  - Issuer (CA)
  - Validity period
  - Public key

TYPES OF SSL CERTIFICATES

- Types
  - DV (Domain Validation)
    - Basic
    - Verifies only domain ownership (for blogs/personal sites).
  - OV (Organization Validation)
    - Moderate
    - Confirms both domain and business details.
  - EV (Extended Validation)
    - Highest
    - Thorough verification; shows organization name in browser bar.

ROLE OF CERTIFICATE AUTHORITIES (CA) & CHAIN OF TRUST

- CA = Trusted third party that issues SSL certificates after verification.
- Ensures only legitimate entities receive valid certificates.
- Chain of Trust:
  - Root Certificate Authority (trusted globally)
  - Intermediate CA
  - End-Entity Certificate (your website)

HOW BROWSERS VALIDATE CERTIFICATES

- Browser checks:
  - Certificate validity & expiry
  - Issuer (CA) is trusted
  - Domain matches
  - Certificate not revoked
- If valid → Padlock icon (🔒) and HTTPS appear.
- If invalid → Warning like “Your connection is not private.

ATTACKS, LIMITATIONS & TRANSITION TO TLS

22Z251 - RAMAPRIYA S

COMMON ATTACKS ON SSL

01 **POODLE** (Padding Oracle On Downgraded Legacy Encryption) – exploits SSL 3.0’s CBC mode weakness.

02 **BEAST** (Browser Exploit Against SSL/TLS) – targets vulnerabilities in TLS 1.0 block cipher.

03 **Heartbleed** – vulnerability in OpenSSL’s heartbeat extension exposing private data.

04 **Man-in-the-Middle (MITM) –** intercepts communication between client and server.

LIMITATIONS OF SSL

01 Weak encryption standards (40-bit & 56-bit keys in early SSL).

02 Poor handling of certificate revocation.

03 Lack of forward secrecy — compromising one key exposes past sessions.

04 Outdated cryptographic algorithms (MD5, RC4).

05 Vulnerable to protocol downgrade attacks.

WHY TLS REPLACED SSL

01 TLS = “Transport Layer Security” — the successor to SSL 3.0.

02 Introduced stronger cryptographic algorithms and improved handshake.

03 Enforced message authentication and better key generation.

04 Prevents protocol downgrade attacks.

05 Backward compatible but more secure.

KEY IMPROVEMENTS IN TLS

01 Stronger encryption suites (AES, ChaCha20).

02 Perfect Forward Secrecy (via Diffie–Hellman).

03 Improved certificate validation and handshake performance.

04 HMAC-based integrity verification.

05 TLS 1.3: faster handshake (1-RTT), deprecated insecure algorithms.

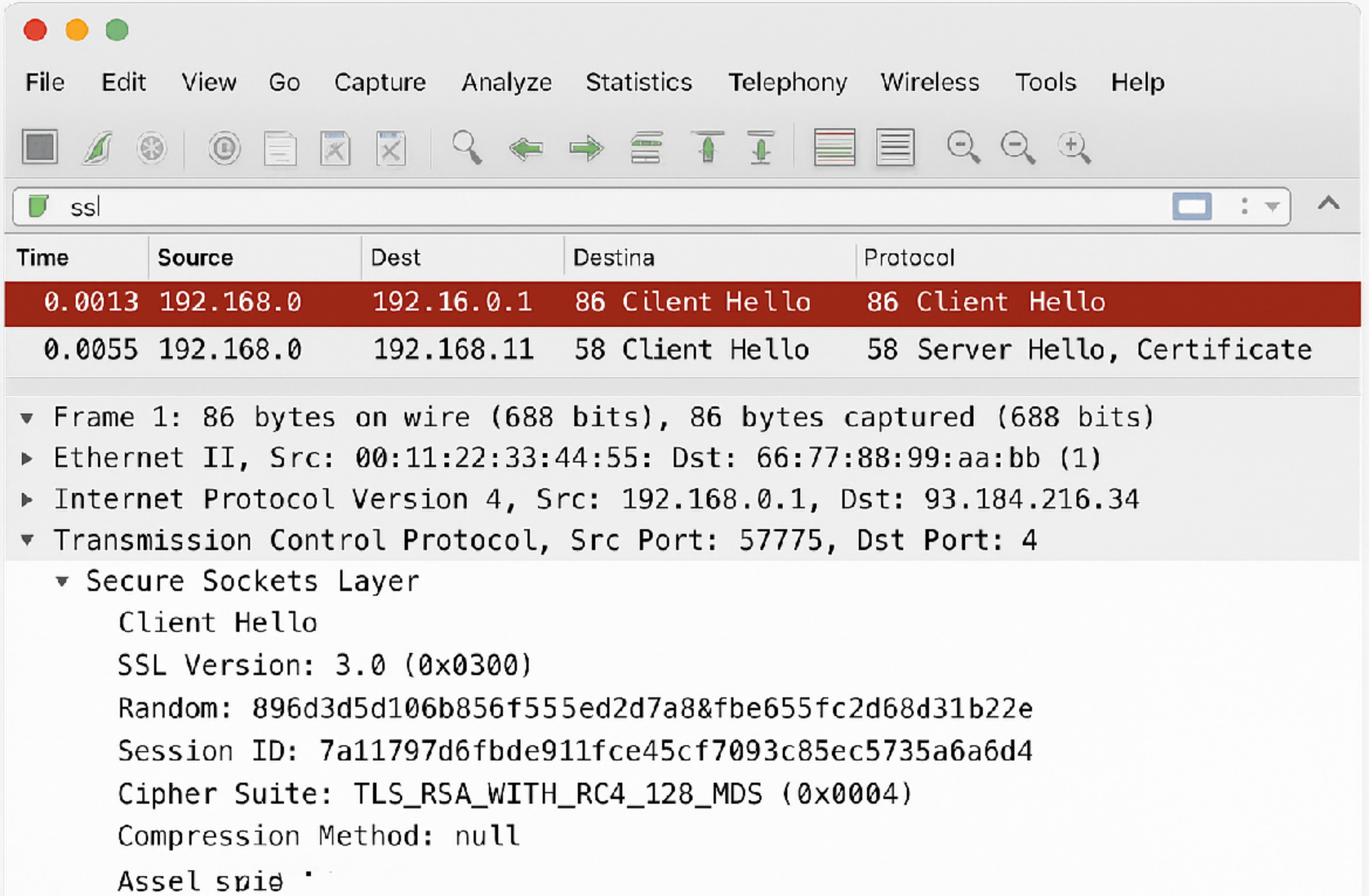
REAL-WORLD IMPACT

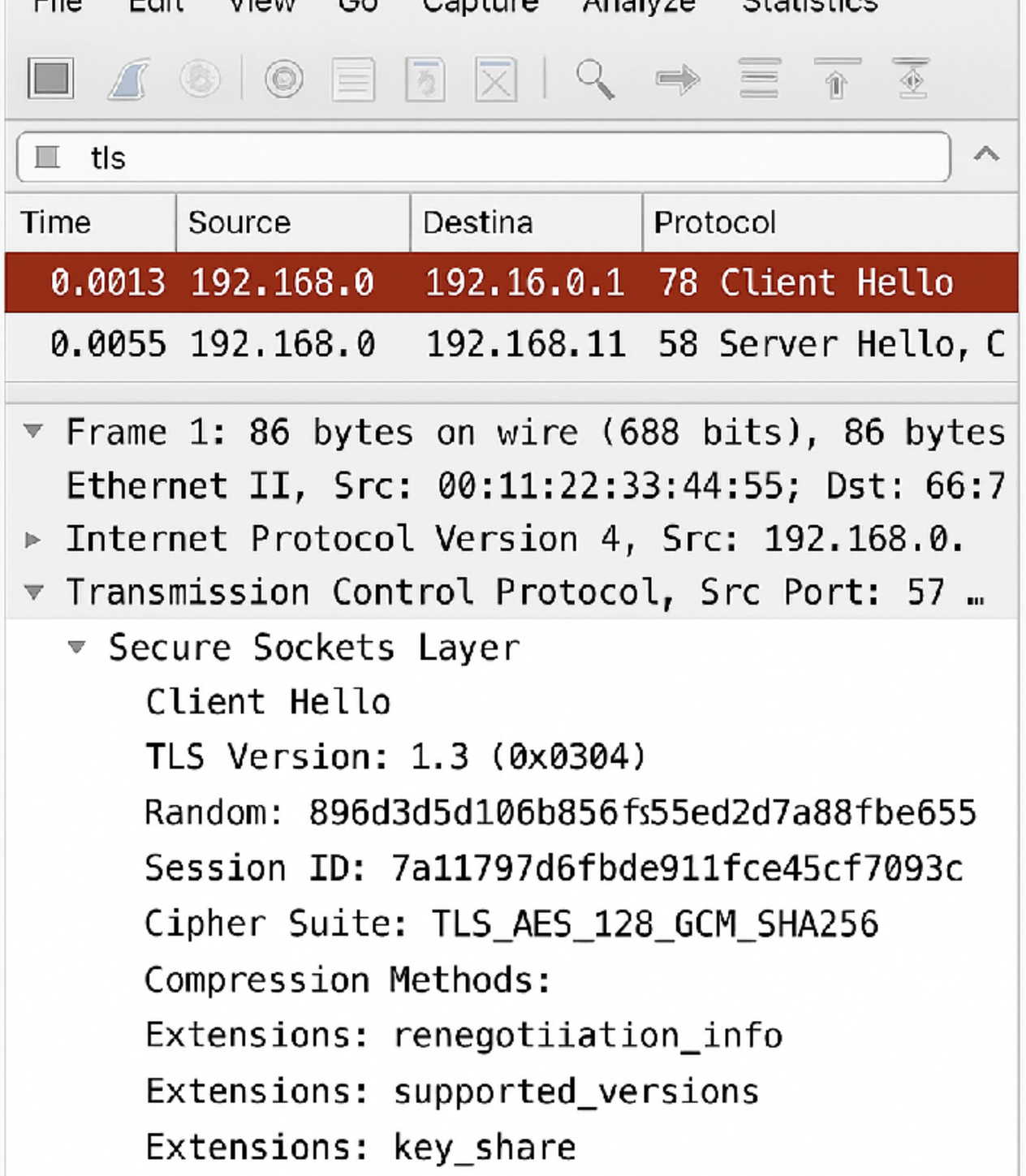
01 Modern browsers disable SSL 2.0 & SSL 3.0 by default.

02 HTTPS today = TLS, not SSL.

03 Secure online banking, e-commerce, and APIs all rely on TLS.

04 Continuous updates protect against evolving cyber threats.


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ssl
Time Source Dest Destina Protocol
0.0013 192.168.0 192.16.0.1 86 Cilent Hello 86 Client Hello
0.0055 192.168.0 192.168.11 58 Client Hello 58 Server Hello, Certificate
Frame 1: 86 bytes on wire (688 bits), 86 bytes captured (688 bits)
Ethernet II, Src: 00:11:22:33:44:55: Dst: 66:77:88:99:aa:bb (1)
Internet Protocol Version 4, Src: 192.168.0.1, Dst: 93.184.216.34
Transmission Control Protocol, Src Port: 57775, Dst Port: 4
Secure Sockets Layer
Client Hello
SSL Version: 3.0 (0x0300)
Random: 896d3d5d106b856f555ed2d7a8&fbe655fc2d68d31b22e
Session ID: 7a11797d6fbde911fce45cf7093c85ec5735a6a6d4
Cipher Suite: TLS_RSA_WITH_RC4_128_MDS (0x0004)
Compression Method: null


File Edit View Go Capture Analyze Statistics
tls
Time Source Destina Protocol
0.0013 192.168.0 192.16.0.1 78 Client Hello
0.0055 192.168.0 192.168.11 58 Server Hello, C
Frame 1: 86 bytes on wire (688 bits), 86 bytes
Ethernet II, Src: 00:11:22:33:44:55; Dst: 66:7
Internet Protocol Version 4, Src: 192.168.0.
Transmission Control Protocol, Src Port: 57 …
Secure Sockets Layer
Client Hello
TLS Version: 1.3 (0x0304)
Random: 896d3d5d106b856fs55ed2d7a88fbe655
Session ID: 7a11797d6fbde911fce45cf7093c
Cipher Suite: TLS_AES_128_GCM_SHA256
Compression Methods:
Extensions: renegotiation_info
Extensions: supported_versions
Extensions: key_share

THANK YOU