Secure Socket Layer (SSL/TLS)

What is SSL/TLS ?

“The primary goal of the TLS protocol is to provide privacy and data integrity between two communicating applications”

In practice, used to protect information transmitted between browsers and Web servers

- Why is secure communication essential in today's digital age?
- Data breaches, identity theft, and cyberattacks are rampant threats.
- SSL/TLS helps protect against these threats by encrypting data and ensuring it's safe during transmission.

History of the Protocol

SSL 1.0

- Internal Netscape design, early 1994?
- Lost in the mists of time

SSL 2.0

- Published by Netscape, November 1994
- Several problems (next slide)

SSL 3.0

- Designed by Netscape and Paul Kocher, November 1996

TLS 1.0

- Internet standard based on SSL 3.0, January 1999
- Not interoperable with SSL 3.0

TLS 1.3

TLS 1.3 is the latest version of the TLS protocol. TLS, which is used by HTTPS and other network protocols for encryption, is the modern version of SSL. TLS 1.3 dropped support for older, less secure cryptographic features, and it sped up TLS handshakes, among other improvements.

For context, the Internet Engineering Task Force (IETF) published TLS 1.3 in August 2018. TLS 1.2, the version it replaced, was standardized a decade previous, in 2008.

SSL/TLS Basics

- SSL/TLS stands for Secure Sockets Layer and Transport Layer Security.
- These protocols establish a secure connection between two parties over the internet.
- SSL was the predecessor to TLS, but TLS is more commonly used today.
- TLS 1.2 and TLS 1.3 are the latest versions, with TLS 1.3 being the most secure.

How SSL/TLS Works

- SSL/TLS uses a handshake process to establish a secure connection.
- During this handshake, the client and server agree on encryption parameters.
- Once established, data is encrypted and decrypted using these parameters.

Key Components of SSL/TLS

- SSL/TLS relies on certificates, public keys, and private keys.
- Certificates are issued by Certificate Authorities (CAs) and contain public keys.
- Private keys are kept secret and are used to decrypt data encrypted with the corresponding public key.

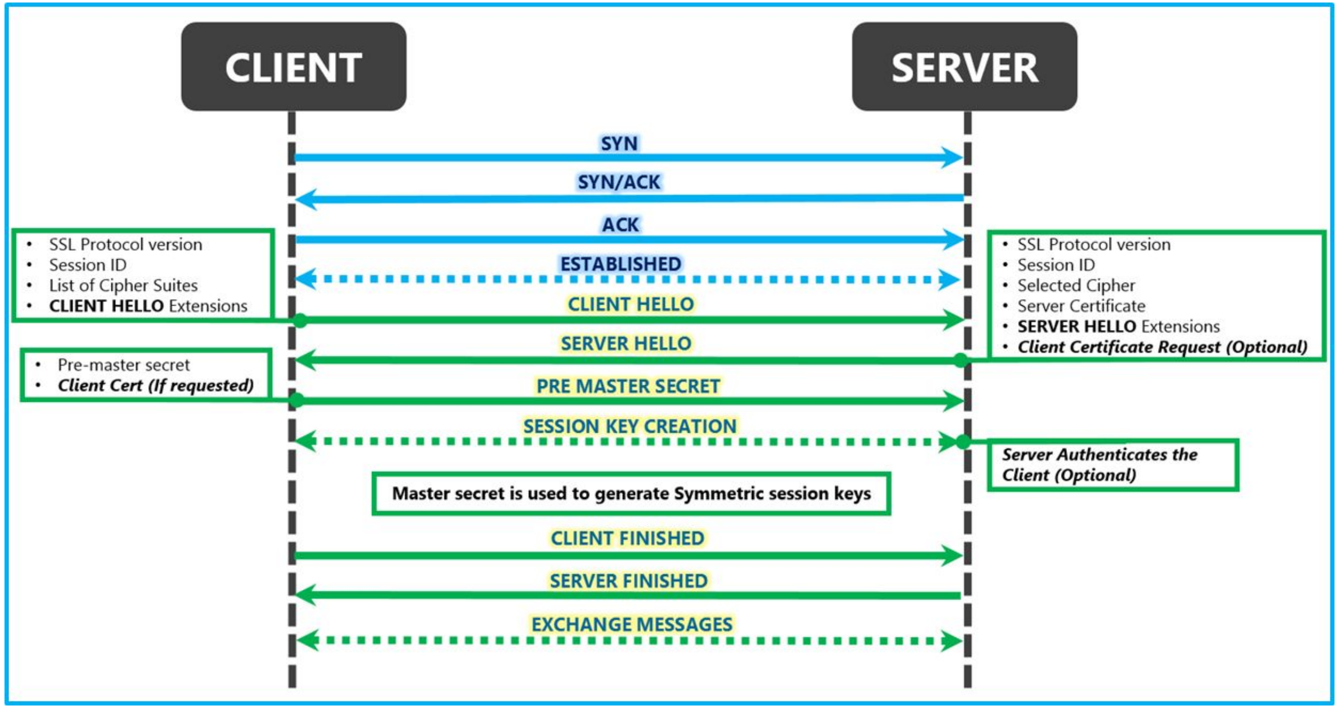
The Role of Certificate Authorities (CAs)

- CAs verify the authenticity of entities' public keys through digital certificates.
- These certificates create a trust chain, ensuring the security of SSL/TLS connections.
- Popular CAs include DigiCert, Let's Encrypt, and GlobalSign.

Secure Browsing and Email etc

- SSL/TLS secure web browsing by encrypting data between your browser and the web server.
- Look for the padlock icon in your browser's address bar to verify a secure connection.
- SSL/TLS is used beyond web browsing, including securing email (SMTP/POP/IMAP), VPNs, and more.

SSL/TLS Handshake using WireShark…?



Key steps of Handshake

Step 1: Client Hello

- The SSL/TLS handshake begins when the client (e.g., a web browser) sends a "ClientHello" message to the server.
- This message contains information such as the highest TLS version supported by the client, a random number (ClientRandom), and a list of supported cryptographic algorithms.

Key steps of Handshake …

Step 2: Server Hello

- The server receives the ClientHello message and responds with a "ServerHello" message.
- The ServerHello includes the TLS version to be used for the session, a random number (ServerRandom), and the selected cryptographic algorithm from the client's list.

Key steps of Handshake …

Step 3: Server Certificate

- If the server is configured with an SSL/TLS certificate (usually issued by a Certificate Authority), it sends its certificate to the client.
- This certificate contains the server's public key and is used to establish the server's authenticity.

Key steps of Handshake …

Step 4: Server Key Exchange (optional)

- In some cases, the server may send a "Server Key Exchange" message. This is used when certain key exchange methods, like Diffie-Hellman, are chosen.
- The server's public parameters for the key exchange are included in this step.

Key steps of Handshake …

Step 5: Client Key Exchange

- The client generates a pre-master secret, encrypts it using the server's public key from the server's certificate (or other key exchange method), and sends it to the server.
- This step ensures that only the server, with its private key, can decrypt the pre-master secret.

Key steps of Handshake …

Step 6: Finished

- Both the client and server have all the necessary information to compute a shared session key.
- They exchange "Finished" messages, which are encrypted and hashed using the negotiated encryption algorithms.
- These messages serve as a confirmation that both parties have the correct session keys.

Key steps of Handshake …

Step 7: Secure Data Transfer

- With the shared session key now established, the client and server switch to the encrypted communication mode.
- All data transmitted between them is encrypted and decrypted using the session key.
- The secure connection is established, and data can flow securely.

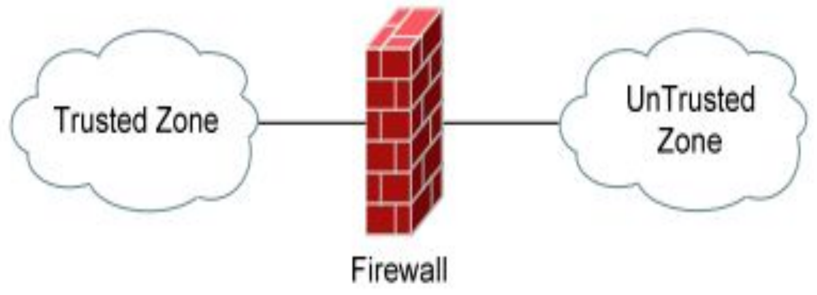
Key steps of Handshake …

This SSL/TLS handshake process ensures that the communication between the client and server is encrypted, secure, and authenticated. It also prevents eavesdropping and Man-in-the-Middle attacks, making it a fundamental part of internet security. The strength of encryption and the specific algorithms used can vary based on the negotiated parameters during the handshake.

Can we decrypt TLS traffic in wireshark?

Please go through SSL Lab Notes.

Introduction to Firewalls

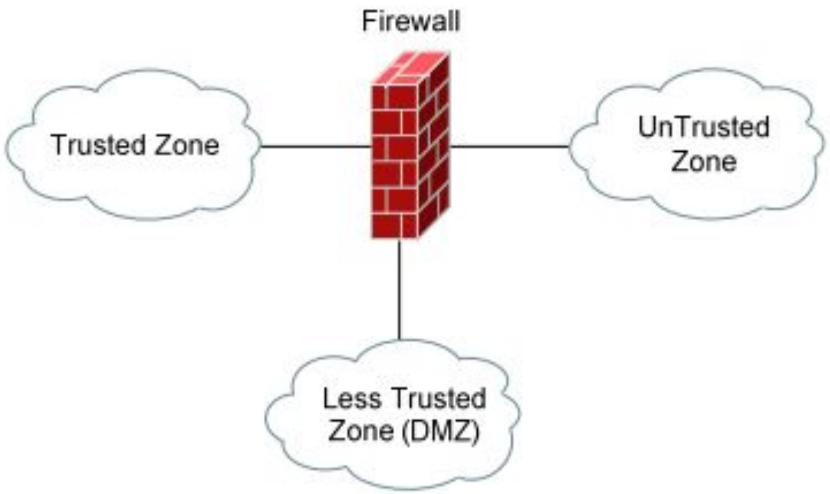


Traditionally, a firewall is defined as any device (or software) used to filter flow of traffic. Firewalls are typically implemented on the network perimeter, and function by defining trusted and untrusted zones. Most firewalls will permit traffic from the trusted zone to the untrusted zone, without any explicit configuration. However, traffic from the untrusted zone to the trusted zone must be explicitly permitted.

- **A firewall controls incoming and outgoing network traffic.**
- **It acts like a security gate between trusted and untrusted networks.**
- **Used to allow, block, or log network packets based on defined rules**

Demilitarized (less trusted) Zones

As briefly described earlier, a DMZ is essentially a less trusted zone that sits between the trusted zone (generally the LAN) and the untrusted zone (generally the Internet). Devices that provide services to the untrusted world are generally placed in the DMZ, to provide separation from the trusted network.



IpTables

iptables is a command-line tool used in Linux to configure the firewall rules that control incoming and outgoing network traffic. It uses a set of tables and chains to decide what to do with packets — whether to accept, drop, or forward them.

- **iptables = Linux built-in firewall management tool.**
- **Works with netfilter (kernel-level packet filter)**
- **Defines rules in chains:**

  INPUT → incoming traffic

  OUTPUT → outgoing traffic

  FORWARD → packets passing through the system

```
$sudo iptables -L
```

Command Structure and Actions

```
sudo iptables -A <chain> -p <protocol> --dport <port> -j <action>
```

```
sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT (Allow SSH Traffic)
```

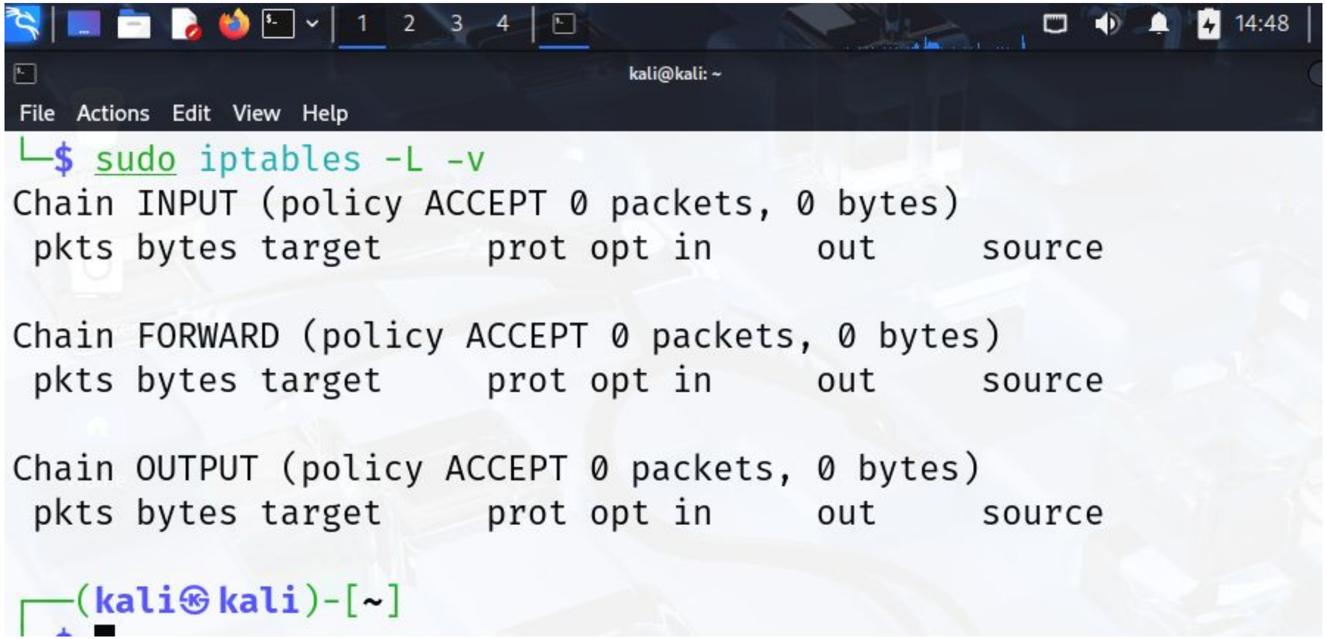
ACCEPT means Allow packet

DROP means Silently discard

REJECT means Block and notify sender

LOG means Record packet info in system logs

Activity-1 View Existing Firewall rules



```
└─$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source

┌──(kali㉿kali)-[~]
```

Iptables Rules and Commands

Block Traffic to a Website `sudo iptables -A OUTPUT -d <IP> -j DROP`

Accept all traffic from an IP address `sudo iptables -A INPUT -s [IP-address] -j ACCEPT`

Drop traffic from an IP address `sudo iptables -A INPUT -s [IP-address] -j DROP`

Delete a Rule:

Use the -F option to clear all iptables firewall rules. To delete a specific rule, list all rules:

```
sudo iptables -L --line-numbers

sudo iptables -D INPUT [number]
```

Activity 2: Objective: Permit ICMP (ping) but block other traffic

Commands:

```
sudo iptables -P INPUT DROP
sudo iptables -A INPUT -p icmp -j ACCEPT
```

Reset:

```
sudo iptables -F
sudo iptables -P INPUT ACCEPT
```

Activity 3: Firewall Evasion

1- Configure iptables to block a target website (uet.edu.pk/example.com)
2- Attempt to access the site without changing defender rules
3- Observe Wireshark for Blocked IP beforehand after , is there a SYN packet?