

# Web Application Security QBANK

~ By Rasenkai



1. What is Packet Network Performance, explain three main metrics and its network constraints?

Packet network performance refers to the quality of service offered by a computer network as perceived by the user. There are several metrics used to measure network performance, including:

1. **Throughput**: This is the amount of data that can be transmitted over a network in a given time period, usually measured in bits per second (bps).
2. **Latency**: This is the amount of time it takes for a packet of data to travel from one point on a network to another, usually measured in milliseconds (ms).
3. **Packet loss**: This is the percentage of packets that are lost during transmission, which can be caused by network congestion, hardware failures, or other issues.

Network constraints can affect packet network performance. Some of these constraints include:

- **Bandwidth**: This is the maximum amount of data that can be transmitted over a network in a given time period. If the bandwidth is limited, it can cause network congestion and affect network performance.
- **Network delay**: This is the time it takes for a packet of data to travel from one point on a network to another. High network delay can cause latency and affect network performance.
- **Jitter**: This is the variation in delay between packets of data. High jitter can cause packet loss and affect network performance.

2. Explain Delays in Networks?

In computer networks, delays refer to the time it takes for the processing of a particular packet to occur. There are several types of delays that can occur in a network, including:

- **Transmission Delay**: This is the time taken to transmit a packet from the host to the transmission medium. It is influenced by the bandwidth of the network and the MAC protocol used for shared links. Transmission delay is calculated as the size of the file divided by the data rate of the link.
- **Propagation Delay**: After a packet is transmitted to the transmission medium, it has to travel through the medium to reach its destination. The time taken by the last bit of the packet to reach the destination is called propagation delay. It is determined by the distance between the nodes and the speed of propagation, which is dependent on the medium of communication. Propagation delay is constant for a given distance and transmission speed.
- **Queuing Delay**: If a packet arrives at its destination and the destination is busy, it will not be handled immediately. Instead, it will be placed in a queue and experience queuing delay until it can be processed.

These delays can be affected by various network constraints, such as:

- **Bandwidth**: The maximum amount of data that can be transmitted over a network in a given time period. Increasing the bandwidth can decrease transmission delay.
- **Network Distance**: The length of the network path between the source and destination nodes can affect both propagation delay and transmission delay.
- **Network Congestion**: If the network is experiencing high traffic, it can lead to increased queuing delay as packets wait to be processed.

### 3. Explain Packetization delay and Component processing delay?

Packetization delay and component processing delay are two types of delays that can occur in computer networks.

- **Packetization delay**: This is the time it takes to fill a packet payload with encoded or compressed speech. It is a function of the sample block size required by the vocoder and the number of blocks placed in a single frame. Packetization delay can also be called accumulation delay, as the voice samples accumulate in a buffer before they are released.
- **Component processing delay**: This is the time it takes for a packet to be processed by a node, such as a router or switch. It is affected by the processing capabilities of the node and the complexity of the routing algorithm. Component processing delay includes several subcomponents, such as:
  - **Integrity check time**: The time needed to perform an integrity check on the packet[6].
  - **Packet lookup time**: The time needed to lookup packet information in a local table[6].
  - **Packet transfer time**: The time needed to move the packet from an input link to an output link in a router.

### 4. Explain End-to-End delay?

End-to-end delay is the time taken for a packet to be transmitted across a network from source to destination. It is a common term in IP network monitoring and differs from round-trip time (RTT) in that only the path in one direction from source to destination is measured. End-to-end delay includes the following types of delays:

- **Transmission delay**: The time taken to transmit a packet from the host to the transmission medium. It is influenced by the bandwidth of the network and the MAC protocol used for shared links.
- **Propagation delay**: The time taken for a packet to travel from the source to the destination through the transmission medium. It is determined by the distance between

the nodes and the speed of propagation, which is dependent on the medium of communication.

- **Queuing delay**: The time taken for a packet to wait in a queue before it can be processed. It is affected by network congestion and the processing capabilities of the nodes.

- **Processing delay**: The time taken for a node, such as a router or switch, to process a packet. It includes several subcomponents, such as integrity check time, packet lookup time, and packet transfer time.

Calculating end-to-end delay requires knowing the packet length, transmission rate of the link, and propagation delay. It can be calculated using the formula  $N \cdot L / R$ , where  $N$  is the number of links in series,  $L$  is the packet length, and  $R$  is the transmission rate. Understanding end-to-end delay is important for identifying system performance issues and optimizing network performance.

## 5. Explain Packet loss and it's reasons

Packet loss occurs when one or more packets of data transmitted across a network fail to reach their destination. Packet loss rate is expressed in percentages, calculated as the number of packets lost compared to the total number sent. Packet loss can cause noticeable performance issues for all types of digital communications, such as video buffering, stuttering, or the video not loading. The causes of packet loss include:

- **Network congestion**: This is the most common cause of packet loss. When a network becomes congested with traffic, packets can be dropped.

- **Inadequate signal strength**: If the signal strength at the destination is too weak, packets may not be received.

- **Interference**: Natural or human-made interference can cause packet loss.

- **Excessive system noise**: This can cause errors in data transmission and lead to packet loss.

- **Software corruption**: Bugs or glitches in software can cause packet loss.

- **Overburdened network nodes**: If a network node, such as a router or switch, is overburdened with traffic, it may drop packets.

## 6. Explain Throughput and it's working

Throughput is a measure of how much data can be transferred from one location to another in a given amount of time. It is a practical measure of actual packet delivery rather than theoretical packet delivery. Throughput is typically measured in bits per second (bps), as in megabits per second (Mbps) or gigabits per second (Gbps). Here's how it works:

- Data is transmitted from the source to the destination over a network.

- The amount of data transmitted is measured over a given period of time.

- The data rate is calculated by dividing the amount of data transmitted by the time taken to transmit it.
- The data rate is expressed in bits per second (bps).

Throughput can be affected by various factors, including network congestion, packet loss, and network latency. Measuring throughput is important to ensure that the network is functioning properly and to identify any potential performance issues. Here are some ways to measure throughput:

- Use network monitoring tools to measure the amount of data transmitted over a network in a given time period.
- Use network testing tools to simulate network traffic and measure the amount of data transmitted over a network in a given time period.
- Use network benchmarking tools to compare the performance of different networks.

7. explain how to calculate throughput with an example?

To calculate throughput, follow these steps:

1. Determine the file size (A) in bits, bytes, kilobytes, megabytes, gigabytes, or terabytes.
2. Measure the transfer time (T) in seconds, minutes, or hours.
3. Divide the amount of data by the transfer time to solve for the rate or speed (S) of transfer using the equation  $S = A \div T$ .

Here's an example:

Suppose you want to calculate the throughput for a file transfer of 10 megabytes (MB) that took 5 seconds. First, convert the file size to bits or bytes for consistency. Let's use bytes:

$$A = 10 \text{ MB} \times 8 \text{ bits/byte} = 80 \text{ megabits (Mb)}$$

Next, plug the values into the equation:

$$S = \frac{80 \text{ Mb}}{5 \text{ s}} = 16 \text{ Mbps}$$

The throughput for this file transfer is 16 megabits per second (Mbps).

Note that the actual throughput may be lower than the calculated value due to various factors such as network congestion, packet loss, and network latency.

8. explain propagation delay with a sum

Propagation delay is the amount of time it takes for a signal to travel from the sender to the receiver through a medium. It is caused by the time it takes for the signal to travel through the medium, and it is of primary interest in electronic circuit design and computer networking. The fundamental limit on propagation delay is the speed of light, as nothing can travel faster than light and light has a finite speed. Propagation delay is present in data networks as well, as it takes time for the signals to travel from the point of transmission to the receiver. Propagation delay can also be called transmission delay, flight time, or time of flight for the signal. The formula for propagation delay is distance divided by the propagation speed.

Here's an example of how to calculate propagation delay:

Suppose the distance between the sender and the receiver is 10,000 km, and the propagation speed is  $2.4 \times 10^8$  m/s. First, convert the distance to meters:

$$distance = 10,000 \text{ km} \times 1000 \text{ m/km} = 10,000,000 \text{ m}$$

Next, plug the values into the formula:

$$propagation\backslash delay = \frac{10,000,000 \text{ m}}{2.4 \times 10^8 \text{ m/s}} = 0.0417 \text{ s or } 41.7 \text{ ms}$$

The propagation delay for this signal is 41.7 milliseconds.

Propagation delay is an important factor to consider when designing and optimizing computer networks, as it can affect network performance and user experience.

9. Explain OSI Reference Model, it's various layers and design issues on each layer?

The OSI (Open Systems Interconnection) reference model is a conceptual framework that describes the functions of a communication system. It consists of seven layers, each with a specific set of functions and protocols. The seven layers of the OSI model are:

1. **Physical layer**: This layer is responsible for the physical transmission of data over a communication channel. It defines the electrical, mechanical, and physical characteristics of the transmission medium. The design issues on this layer include signal attenuation, interference, and noise.
2. **Data link layer**: This layer is responsible for the reliable transmission of data between two nodes on the same network. It provides error detection and correction, flow control, and access control. The design issues on this layer include framing, error control, and flow control.
3. **Network layer**: This layer is responsible for the routing of data between different networks. It provides logical addressing and routing services. The design issues on this layer include addressing, routing, and congestion control.

4. **Transport layer**: This layer is responsible for the end-to-end delivery of data between two hosts. It provides reliable data transfer, flow control, and error recovery. The design issues on this layer include connection establishment, flow control, and error control.

5. **Session layer**: This layer is responsible for establishing, managing, and terminating sessions between applications. It provides services such as authentication, authorization, and synchronization. The design issues on this layer include session establishment, synchronization, and recovery.

6. **Presentation layer**: This layer is responsible for the representation and transformation of data between different formats. It provides services such as data compression, encryption, and decryption. The design issues on this layer include data representation, encryption, and compression.

7. **Application layer**: This layer is responsible for providing services to user applications. It includes protocols such as HTTP, FTP, and SMTP. The design issues on this layer include application design, user interface, and data exchange.

#### 10. Explain TCP IP Model

The TCP/IP model is a conceptual framework that describes the functions of a communication system. It is a concise version of the OSI (Open Systems Interconnection) model and consists of four layers:

1. **Network Access Layer**: This layer is responsible for the physical transmission of data between the network and the physical medium. It defines the electrical, mechanical, and physical characteristics of the transmission medium.

2. **Internet Layer**: This layer is responsible for the routing of data between different networks. It provides logical addressing and routing services.

3. **Transport Layer**: This layer is responsible for the end-to-end delivery of data between two hosts. It provides reliable data transfer, flow control, and error recovery.

4. **Application Layer**: This layer is responsible for providing services to user applications. It includes protocols such as HTTP, FTP, and SMTP.

The TCP/IP model is widely used in computer networking and is the basis for the Internet. It is a practical model that is easy to implement and maintain. The TCP/IP model has several advantages over the OSI model, including its simplicity, flexibility, and scalability.

#### 11. What is the difference between the TCP/IP model and the OSI Reference Model?

The TCP/IP model and the OSI Reference Model are two different conceptual frameworks that describe the functions of a communication system. Here are the main differences between the two models:

- **Number of layers**: The OSI model has seven layers, while the TCP/IP model has four layers.

- **Layer functions**: The layers in the two models have different functions. For example, the TCP/IP model combines the physical and data link layers into one layer, while the OSI model separates them into two layers.
- **Header size**: The header size in the TCP/IP model is smaller than that in the OSI model. For example, the smallest size of the OSI header is 5 bytes, while the smallest size of the TCP/IP header is 20 bytes.
- **Development**: The OSI model was developed by the International Organization for Standardization (ISO), while the TCP/IP model was developed by the Advanced Research Projects Agency Network (ARPANET).
- **Implementation**: The OSI model is a theoretical model that is not widely implemented, while the TCP/IP model is a practical model that is widely used in computer networking and is the basis for the Internet.
- **Standardization**: The OSI model helps to standardize router, switch, motherboard, and other hardware, while the TCP/IP model helps to establish a connection between different types of computers.

## 12. Explain Client-server architecture?

Client-server architecture is a computing model in which the server hosts, delivers, and manages most of the resources and services requested by the client. The client-server model is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients. Here are some key features of client-server architecture:

- Clients request services or resources from servers, which provide those services or resources.
- Servers share their resources with clients, while clients do not share their resources.
- Servers are always on and have permanent IP
- Clients and servers communicate over a computer network, usually on separate hardware.
- Servers may have dynamic IP
- The client-server architecture is also referred to as the networking computing model or client-server network.
- Present in Data centers for scaling.
- The client-server model is used in many computer applications, such as email, network printing, and the World Wide Web.
- The client-server architecture typically features multiple users' workstations, PCs, or other devices connected to a central server via an Internet connection or other network.
- The client sends a request for data, and the server accepts and accommodates the request, sending the data packets back to the user who needs them.

## 13. Explain P2P architecture?



Peer-to-peer (P2P) architecture is a distributed system in which each node acts as both a client and a server. This allows the nodes to share the workload or tasks among themselves. Here are some key features of P2P architecture:

- Each node in a P2P network has the same capabilities and responsibilities.
- There is no always on server as each client acts as both client and server.
- self scalability – new peers bring new service capacity, as well as new service demands
- P2P networks are often used in small deployments or situations where security is not a major concern, such as home networks or small businesses.
- P2P architecture is most effective when there are many active peers in the network. This allows new peers to easily find other peers to connect to and ensures that there are enough remaining peers to take up the slack if many peers leave the network.
- In a P2P file-sharing program, the more common a file is, the faster it can be downloaded because many peers are sharing it. To make P2P work more efficiently, the workload is often divided into small pieces that can be reassembled later. This allows many peers to work on the same task.
- P2P architecture is based on the concept of decentralization, which means that there is no central server or authority controlling the network.
- P2P architecture is often used in blockchain technology, where it helps to ensure the security and integrity of the blockchain.
- P2P architecture can be contrasted with the classic client/server architecture, in which some computers are dedicated to serving others.

#### 14. Explain Processes communication in client-server and P2P architectures?

In both client-server and P2P architectures, processes communicate with each other to exchange information. However, there are some differences in how this communication occurs:

Client-Server Architecture:

- In client-server architecture, the client sends a request to the server, and the server responds with the requested data or service.
- The communication between the client and server is typically initiated by the client.
- The server is responsible for managing and providing most of the resources and services requested by the client.
- The client and server communicate over a computer network, usually on separate hardware.
- The client and server interact directly with a transport layer protocol to establish communication and send and receive information.

P2P Architecture:

- In P2P architecture, each node acts as both a client and a server.

- The communication between nodes is typically initiated by either node.
- Each node can request and respond to services or resources.
- P2P architecture is based on the concept of decentralization, which means that there is no central server or authority controlling the network.
- The nodes in a P2P network communicate directly with each other, without the need for a central server.
- P2P architecture is often used in small deployments or situations where security is not a major concern, such as home networks or small businesses.

#### 15. Explain Sockets and their working with respect to processes?

Sockets are a way to enable inter-process communication between programs running on a server or between programs running on separate servers. A socket is one endpoint of a two-way communication link between two programs running on the network. Here's how sockets work with respect to processes:

- A socket is created by a process using the socket system call.
- The socket provides bidirectional communication facility over the network.
- Each socket has a specific address, which is composed of an IP address and a port number.
- The process binds the socket to a specific address using the bind system call.
- The process listens for incoming connections on the socket using the listen system call.
- When a connection request arrives, the process accepts the connection using the accept system call.
- The process communicates with the client over the socket using the send and receive system calls.
- The process closes the socket using the close system call when the communication is complete.

In client-server architecture, the client and server communicate over a computer network, usually on separate hardware. The client sends a request to the server, and the server responds with the requested data or service. In P2P architecture, each node acts as both a client and a server, and the communication between nodes is typically initiated by either node.

#### 16. Explain Addressing processes in TCP IP model?

In the TCP/IP model, addressing processes involves the use of IP addresses and port numbers to identify the source and destination of data packets. Here's how addressing processes work in the TCP/IP model:

- Each device on a network is assigned a unique IP address, which is a 32-bit number that identifies the device's location on the network.
- IP addresses are divided into two parts: the network portion and the host portion. The network portion identifies the network to which the device is connected, while the host portion identifies the specific device on that network.
- Devices communicate with each other using IP addresses and port numbers. A port number is a 16-bit number that identifies a specific process or service running on a device.
- When a device sends data to another device, it includes the destination IP address and port number in the packet header.
- The receiving device uses the IP address and port number to identify the process or service that should receive the data.
- The transport layer protocols, such as TCP and UDP, use port numbers to identify the specific process or service that should receive the data.
- The application layer protocols, such as HTTP and FTP, use well-known port numbers to identify the specific service that should receive the data.

#### 17. Explain Application layer protocols and its need?

Application layer protocols are a set of rules that govern how two or more devices can communicate with each other at the highest level of the OSI model. These protocols define how applications can send and receive data over a network and are responsible for enabling many of the services we use every day, such as web browsing, email, and file transfer[3]. Here are some reasons why application layer protocols are needed:

- Application layer protocols enable software programs to negotiate formatting, procedural, security, synchronization, and other requirements with the network.
- Application layer protocols provide a standardized way for applications to communicate with each other over a network.
- Application layer protocols ensure that data is transmitted in a format that can be understood by the receiving application.
- Application layer protocols provide a way for applications to authenticate and authorize users.
- Application layer protocols enable applications to access data and services on remote servers.

- Application layer protocols provide a way for applications to handle errors and recover from failures.

Some examples of application layer protocols include:

- Hypertext Transfer Protocol (HTTP): Used for web browsing.
- File Transfer Protocol (FTP): Used for file transfer.
- Simple Mail Transfer Protocol (SMTP): Used for email.
- Post Office Protocol (POP): Used for email retrieval.

18. What transport service does an app need in terms of application layer protocols?

An application layer protocol needs the following transport services:

1. **Data integrity**: Ensures that the data is transmitted without errors or corruption.
2. **Timing**: Ensures that the data is transmitted with low delay and in a timely manner.
3. **Throughput**: Ensures that the data is transmitted at a sufficient rate to meet the application's needs.

When developing an application, the developer must choose a transport-layer protocol that provides the necessary services for the application. The choice of protocol depends on the application's requirements for data integrity, timing, and throughput. The most common application layer protocols include HTTP, FTP, SMTP, and Telnet.

## Transport service requirements: common apps

<b>application</b>	<b>data loss</b>	<b>throughput</b>	<b>time sensitive</b>
file transfer	no loss	elastic	no
e-mail	no loss	elastic	no
Web documents	no loss	elastic	no
real-time audio/video	loss-tolerant	audio: 5kbps-1Mbps video: 10kbps-5Mbps	yes, 100's msec
stored audio/video	loss-tolerant	same as above	yes, few secs
interactive games	loss-tolerant	few kbps up	yes, 100's msec
text messaging	no loss	elastic	yes and no

## Internet apps: application, transport protocols

<b>application</b>	<b>application layer protocol</b>	<b>underlying transport protocol</b>
e-mail	SMTP [RFC 2821]	TCP
remote terminal access	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
file transfer	FTP [RFC 959]	TCP
streaming multimedia	HTTP (e.g., YouTube), RTP [RFC 1889]	TCP or UDP
Internet telephony	SIP, RTP, proprietary (e.g., Skype)	TCP or UDP

### 19. Explain webpage?

A webpage is a document that is suitable for the World Wide Web and web browsers. It is a hypertext document that is delivered by a web server to the user and displayed in a

web browser. A webpage can contain text, graphics, hyperlinks, and other multimedia elements. They are referred to as objects, object can be HTML file, JPEG image, Java applet, audio file, etc.

A Uniform Resource Locator (URL) is used to address a webpage on the Internet. A URL is a unique identifier that tells a web browser how and where to retrieve a resource. It consists of multiple parts, including a protocol, domain name, and path, that identify the location of the resource on the web. Here's how a URL can be used to address a webpage:

- The URL contains the name of the protocol needed to access the resource, such as HTTP or HTTPS.
- The URL contains the domain name or IP address of the web server where the resource is located.
- The URL may contain a path that identifies the specific webpage or resource on the server.

For example, the URL for the homepage of the Mozilla Developer Network is <https://developer.mozilla.org/en-US/docs/Web>, where "https" is the protocol, "developer.mozilla.org" is the domain name, and "/en-US/docs/Web" is the path that identifies the specific webpage.

## 20. Explain HTTP?

HTTP (Hypertext Transfer Protocol) is an application layer protocol that is used in the client-server model to transfer data over the web. Here's how HTTP works in the client-server model:

- The client sends an HTTP request to the server, which includes a URL that identifies the resource to be retrieved.
- The server processes the request and sends an HTTP response back to the client, which includes the requested resource.
- HTTP uses TCP (Transmission Control Protocol) as its transport protocol to ensure reliable data transfer. It creates a socket at port 80
- HTTP is a stateless protocol, which means that each request is executed independently, without any knowledge of the requests that were executed before it.
- HTTP is connectionless, which means that the client and server are aware of each other only during a current request. Afterwards, both of them forget about each other.
- HTTP is a text-based protocol that uses headers to provide additional information about the request or response.
- HTTP is extensible, which means that new functionality can be introduced by a simple agreement between a client and a server about a new header's semantics.

## 21. Differentiate between types of HTTP connections?

HTTP connections can be broadly categorized into two types: non-persistent and persistent.

Non-Persistent Connection:

- In a non-persistent connection, a separate TCP connection is established for each request/response pair.
- The connection is closed after each request/response pair is completed.
- The client must establish a new connection for each request, which can result in increased latency and overhead.
- Non-persistent connections are useful for small requests or when the server is not heavily loaded.

Persistent Connection:

- In a persistent connection, a single TCP connection is established between the client and server, and multiple requests and responses can be sent over this connection.
- The connection is not closed after each request/response pair is completed.
- The client can send multiple requests over the same connection, which can result in reduced latency and overhead.
- Persistent connections are useful for large requests or when the server is heavily loaded.

## 22. Explain Non-persistent HTTP connection?

Non-persistent HTTP connection is a type of HTTP connection in which a new TCP connection is established for each request/response pair. Here's how non-persistent HTTP connection works:

- The client sends a request to the server over a new TCP connection.
- The server processes the request and sends a response back to the client over the same TCP connection.
- The connection is closed after the response is received.
- If the client needs to send another request, it must establish a new TCP connection.
- Non-persistent HTTP connections are useful for small requests or when the server is not heavily loaded.
- Non-persistent HTTP connections have a higher overhead than persistent connections because a new TCP connection must be established for each request/response pair.

- Non-persistent HTTP connections are less efficient than persistent connections because the overhead of establishing a new TCP connection can result in increased latency.

23. Describe the response time in non-persistent HTTP connection?

In non-persistent HTTP connections, the response time is determined by making an HTTP GET request to the web page URL and measuring the time it takes for the first object to be returned. The total time for a non-persistent connection is  $2RTT + \text{file transmission time}$ . Here's how the response time is calculated for a webpage with non-persistent HTTP:

- The client sends an HTTP GET request to the web page URL.
- The server responds with the first object.
- The response time is the time it takes for the first object to be returned, which is equal to  $2RTT + \text{file transmission time}$ .
- The first RTT is taken to establish the connection between the server and the client.
- The second RTT is taken to request and return the object.
- After the client receives the object in non-persistent, the connection is immediately closed.

the response time in non-persistent HTTP connections is determined by measuring the time it takes for the first object to be returned. The total time for a non-persistent connection is  $2RTT + \text{file transmission time}$ .

24. Explain Persistent HTTP and its response times

Persistent HTTP connection, also known as HTTP keep-alive or HTTP connection reuse, is a type of HTTP connection that allows multiple HTTP requests and responses to be sent and received over a single TCP connection.

- The client sends an HTTP request to the server over a TCP connection.
- The server processes the request and sends an HTTP response back to the client over the same TCP connection.
- The connection remains open after the response is received, and the client can send additional requests over the same connection.
- The server can send additional responses over the same connection.
- Persistent HTTP connections are useful for large requests or when the server is heavily loaded.



- Persistent HTTP connections have a lower overhead than non-persistent connections because a single TCP connection is used for multiple requests and responses.

- Persistent HTTP connections are more efficient than non-persistent connections because the overhead of establishing a new TCP connection is avoided.

The response time for persistent HTTP connections is faster than non-persistent connections because the overhead of establishing a new TCP connection is avoided. The response time for persistent HTTP connections is the time it takes for the first object to be returned, which is equal to RTT + file transmission time.

## 25. Explain HTTP request message and its method types?

HTTP request message is a type of message that is sent by the client to the server to initiate an action on the server. The HTTP request message consists of three parts: the request line, headers, and body. Here's how the HTTP request message works:

- The request line contains the HTTP method, the URL, and the HTTP version.
- The headers contain additional information about the request, such as the user agent, content type, and content length.
- The body contains optional data that is sent along with the request.

HTTP defines several methods that can be used in the request line of an HTTP request message. These methods indicate the desired action to be performed on the resource identified by the given URL. The most common HTTP methods are:

- GET: Retrieves a representation of the specified resource.
- POST: Submits an entity to the specified resource, often causing a change in state or side effects on the server.
- PUT: Replaces all current representations of the target resource with the request payload.
- DELETE: Deletes the specified resource.
- HEAD: Asks for a response identical to a GET request, but without the response body.
- OPTIONS: Describes the communication options for the target resource.
- CONNECT: Establishes a tunnel to the server identified by the target resource.
- TRACE: Performs a message loop-back test along the path to the target resource.

The HTTP methods that fall under HTTP/1.0 and HTTP/1.1 are:

- HTTP/1.0: GET, POST, HEAD, PUT, DELETE.
- HTTP/1.1: GET, POST, HEAD, PUT, DELETE, CONNECT, OPTIONS, TRACE.

26. Explain what is HTTP response message and describe its structure?

HTTP response message is a type of message that is sent by the server to the client in response to an HTTP request message. The HTTP response message consists of three parts: the status line, headers, and body. Here's how the HTTP response message works:

- The status line contains the HTTP version, status code, and reason phrase.
- The headers contain additional information about the response, such as the content type, content length, and server information.
- The body contains the response data, such as the HTML content of a webpage.

The structure of an HTTP response message is as follows:

- Status line: Contains the HTTP version, status code, and reason phrase.
- Headers: Contains additional information about the response, such as the content type, content length, and server information.
- Body: Contains the response data, such as the HTML content of a webpage.

The HTTP status code is a three-digit number that indicates the status of the response. The most common HTTP status codes are:

- 1xx: Informational.
- 2xx: Success.
- 3xx: Redirection.
- 4xx: Client errors.
- 5xx: Server errors.

# HTTP response message

status line (protocol ▶ status code status phrase)	HTTP/1.1 200 OK\r\n Date: Sun, 26 Sep 2010 20:09:20 GMT\r\n Server: Apache/2.0.52 (CentOS)\r\n Last-Modified: Tue, 30 Oct 2007 17:00:02 GMT\r\n
header lines	ETag: "17dc6-a5c-bf716880"\r\n Accept-Ranges: bytes\r\n Content-Length: 2652\r\n Keep-Alive: timeout=10, max=100\r\n Connection: Keep-Alive\r\n Content-Type: text/html; charset=ISO-8859-1\r\n \r\n
data, e.g., requested HTML file	data data data data data ... ▶

## 27. Explain HTTP response status codes and give some examples?

HTTP response status codes indicate the status of an HTTP response and whether a specific HTTP request has been successfully completed. The status codes are grouped into five classes:

1. Informational responses (100–199): These responses indicate that the request has been received and is being processed.
2. Successful responses (200–299): These responses indicate that the request has been successfully received, understood, and accepted.
3. Redirection messages (300–399): These responses indicate that further action needs to be taken by the client to complete the request.
4. Client error responses (400–499): These responses indicate that the request was invalid or could not be completed.
5. Server error responses (500–599): These responses indicate that the server encountered an error while processing the request.

Some examples of HTTP response status codes are:

- 200 OK: The request has been successfully completed.

- 301 Moved Permanently: The requested resource has been permanently moved to a new URL.

- 400 Bad Request: The request was invalid or could not be understood by the server.

- 404 Not Found: The requested resource could not be found on the server.

- 500 Internal Server Error: The server encountered an error while processing the request.

- 503 Service Unavailable: The server is currently unable to handle the request due to a temporary overload or maintenance.

HTTP/1.0 and HTTP/1.1 define different sets of status codes. The status codes that fall under HTTP/1.0 are:

- 100 Continue

- 200 OK

- 302 Found

- 400 Bad Request

- 401 Unauthorized

- 403 Forbidden

- 404 Not Found

- 500 Internal Server Error

The status codes that fall under HTTP/1.1 are:

- 100 Continue

- 101 Switching Protocols

- 200 OK

- 201 Created

- 202 Accepted

- 204 No Content

- 206 Partial Content

- 301 Moved Permanently

- 302 Found

- 304 Not Modified
- 307 Temporary Redirect
- 400 Bad Request
- 401 Unauthorized
- 403 Forbidden
- 404 Not Found
- 405 Method Not Allowed
- 500 Internal Server Error
- 503 Service Unavailable

28. Explain User-server state cookies, it's components, it's usage and how it maintains state?

HTTP cookies are a mechanism for maintaining state between a client and a server. Cookies are small pieces of data that are sent by the server to the client's web browser and stored on the client's machine. The cookie is then sent back to the server with subsequent requests, allowing the server to maintain state between requests. Here's how cookies work:

- The server sends a Set-Cookie header with the response, which contains the cookie data.
- The browser stores the cookie data on the client's machine.
- The browser sends the cookie data back to the server with subsequent requests.
- The server uses the cookie data to maintain state between requests.

The components of an HTTP cookie are:

- Name: The name of the cookie.
- Value: The value of the cookie.
- Attributes: Zero or more attributes that provide additional information about the cookie, such as the cookie's expiration, domain, and flags.

Cookies are mainly used for three purposes:

- Session management: Cookies are used to maintain session information, such as user login information and shopping cart contents.

- Personalization: Cookies are used to personalize the user's experience, such as by remembering the user's preferences and displaying customized content.

- Tracking: Cookies are used to track the user's browsing activity, such as by recording which pages were visited in the past.

HTTP cookies can maintain state by storing information about the user on the client's machine and sending that information back to the server with subsequent requests. This allows the server to remember information about the user, such as login information and shopping cart contents, between requests.

## 29. Explain Web caching and why it's needed?

Web caching is the process of storing copies of data or files in a temporary location known as a cache. Caching is used to improve the performance of websites by storing commonly accessed content and reducing the number of requests that need to be made to the server. Here's why web caching is needed:

- Web caching can significantly reduce server load, decrease page load times, and enhance the user experience.

- Web caching can help speed up your website by serving content from memory rather than generating it from scratch by the server each time.

- Web caching can reduce the amount of data that needs to be transferred over the network, which can result in faster page load times and reduced bandwidth usage.

- Web caching can help improve the scalability of a website by reducing the load on the server and allowing it to handle more requests.

There are different types of web caching, including:

- Server-side caching: Temporarily stores web files and data on the server for later use, effectively reducing the server's load and latency.

- Client-side caching: Temporarily stores the copy of a webpage in the browser's memory, so the next time a user revisits a website, it won't make a call to the server for the data.

## 30. Explain web caches (proxy server)?

A web cache, also known as a proxy server, is a dedicated network server or service that stores webpages or other internet content locally. Its goal is to satisfy client request without involving origin server, Here's how web caches work:

- When a client requests information through an HTTP message, the request is first sent to the proxy server.

- The proxy server checks if it has a copy of the requested content stored locally in its cache.

- If the content is found in the cache, the proxy server forwards the result directly to the client, reducing the response time and the load on the original server.
- If the content is not found in the cache, the proxy server queries the original server on behalf of the client, stores a copy of the result locally, and then forwards the result back to the client.

Web caches have several advantages, including:

- Improved website performance: By serving frequently accessed content from the cache, web caches can significantly reduce response times and enhance the user experience.
- Reduced server load: Web caches offload the server by serving some of the requests, reducing the server's workload and allowing it to handle more requests.
- Bandwidth savings: By serving content from the cache, web caches can reduce the amount of data that needs to be transferred over the network, resulting in lower bandwidth usage.
- Data logging and security: Proxy servers can store data about user requests, such as IP addresses, for activity analysis and security purposes.

### 31. Explain Conditional GET in proxy server?

Conditional GET is a mechanism used in web caching by proxy servers to verify if the cached content is still valid or if it needs to be updated. Here's how Conditional GET works:

- When a client requests a resource, the proxy server checks if it has a cached copy of the resource.
- If the proxy server has a cached copy of the resource, it sends a conditional GET request to the origin server, asking if the resource has been modified since the last time it was cached.
- If the resource has not been modified, the origin server sends a 304 Not Modified response to the proxy server, indicating that the cached copy is still valid.
- If the resource has been modified, the origin server sends a new version of the resource to the proxy server, which replaces the cached copy.

Conditional GET is useful because it allows the proxy server to serve cached content without having to download the entire resource again, reducing bandwidth usage and improving performance. It also ensures that the cached content is up-to-date and accurate.

### 32. Explain DNS, its services and structure?

DNS (Domain Name System) is a hierarchical and decentralized naming system for computers, services, or any other resources connected to the Internet or a private

network. It associates various information with domain names assigned to each of the associated entities and translates readily memorized domain names to the numerical IP addresses needed for locating and identifying computer services and devices with the underlying network protocols. Here's the structure and services of DNS:

- DNS Hierarchy: The DNS hierarchy is a hierarchical system consisting of five distinct components. At the highest level is the root domain, which is the apex of the domain name hierarchy. Underneath the root domain are the top-level domains, further divided into second-level domains, third-level domains, and so on.

- DNS Name Servers: DNS name servers are servers that store the DNS records for a domain and respond with answers to queries against its database. There are two types of DNS name servers: authoritative name servers and recursive name servers.

- Authoritative Name Servers: Authoritative name servers are servers that store the DNS records for a domain and provide answers to queries about that domain. They are responsible for providing the correct DNS information for a domain.

- Recursive Name Servers: Recursive name servers are servers that take DNS queries from an application, such as a web browser, and either provide the answer to the query if it has it cached or access the next-level server if it doesn't. They are responsible for resolving DNS queries by recursively querying other DNS servers.

- DNS Mapping: DNS mapping is distributed throughout the internet in a hierarchy of authority. Access providers and enterprises, as well as governments, universities, and other organizations, operate DNS servers that communicate with each other to resolve DNS queries.

DNS services include:

- Hostname to IP address translation: DNS is responsible for translating domain names to IP addresses, allowing users to access websites using domain names instead of IP addresses.

- Load balancing: DNS can be used to distribute traffic across multiple servers, improving website performance and availability.

- Email routing: DNS is used to route email messages to the correct mail server.

- Security: DNS can be used to implement security measures, such as DNSSEC, which provides authentication and integrity for DNS data.

### 33. What are DNS root name servers?

DNS root name servers are the highest level of DNS servers in the DNS hierarchy, responsible for managing domain names for the entire internet. They are the first step in resolving domain names into IP addresses and are responsible for answering queries for records in the root zone and returning a list of authoritative name servers for the appropriate top-level domain (TLD). Although any local implementation of DNS can implement its own private root name servers, the term "root name server" is generally used to describe the thirteen well-known root name servers that implement the root



name space domain for the Internet's official global implementation of the Domain Name System. The root name servers are a critical part of the internet infrastructure because they are the first step in resolving human-readable host names into IP addresses that are used in communication between internet hosts.

#### 34. Differentiate between TLD servers and authoritative servers?

TLD Servers:

- TLD (Top-Level Domain) servers are responsible for managing domain names for a specific top-level domain, such as .com, .org, .net, etc.
- TLD servers are the next level down from the root servers in the DNS hierarchy.
- TLD servers are responsible for directing queries to the authoritative name servers for the domain in question.
- TLD servers maintain information for all the domain names that share a common domain extension, such as .com, .net, .org, etc.
- There are multiple TLD servers for each top-level domain, and they are distributed around the world.

Authoritative Servers:

- Authoritative servers are responsible for storing the DNS records for a specific domain name.
- Authoritative servers are the final authority on the DNS information for a domain name.
- Authoritative servers are responsible for providing answers to queries about a specific domain name.
- There can be multiple authoritative servers for a single domain name, and they are typically managed by the domain owner or their DNS provider.
- Authoritative servers are responsible for providing the correct DNS information for a domain.

#### 35. what is Local DNS name server?

A local DNS name server is a DNS server that is used to resolve domain names on a local network. It is typically used to provide name resolution for computers and devices on a local network, such as file servers, printers, and other devices. Here's how a local DNS name server works:

- A local DNS name server is installed on a computer or device on the local network.
- The local DNS name server is configured to resolve domain names for the local network.
- When a client on the local network requests a domain name, the request is sent to the local DNS name server.
- The local DNS name server checks its cache to see if it has a record for the domain name. If it does, it returns the IP address associated with the domain name.
- If the local DNS name server does not have a record for the domain name, it queries other DNS servers on the internet to resolve the domain name.

Local DNS name servers are useful for several reasons, including:

- Improved performance: By resolving domain names locally, local DNS name servers can reduce the time it takes to resolve domain names and improve network performance.
- Custom domain names: Local DNS name servers can be used to create custom domain names for devices on the local network, making it easier to access them.
- Security: Local DNS name servers can be used to block access to malicious websites and protect the local network from security threats.

36. Explain DNS name resolution, its types, each with an example?

DNS name resolution is the process of converting a domain name into an IP address. There are different types of DNS name resolution, each with its own purpose and mechanism. Here are the types of DNS name resolution:

1. Recursive Resolution: In this type of resolution, the client requests the local server to provide either the requested mapping or an error message. A DNS query is generated by the application program to the resolver to fetch the IP address of the domain name. The resolver then queries the root servers, TLD servers, and authoritative servers to resolve the domain name.
2. Iterative Resolution: In this type of resolution, the client requests the local server to provide the IP address of the domain name. The local server queries the root servers, TLD servers, and authoritative servers in a step-by-step manner until it finds the IP address of the domain name.
3. Non-recursive Resolution: In this type of resolution, the client requests the local server to provide the IP address of the domain name, but the local server does not query other servers to resolve the domain name. Instead, it returns the best answer it has in its cache.

Examples of DNS name resolution types:

- Recursive Resolution: When a user types a domain name into a web browser, the browser generates a DNS query to the local DNS server to fetch the IP address of the domain name. The local DNS server then queries the root servers, TLD servers, and authoritative servers to resolve the domain name.
- Iterative Resolution: When a user types a domain name into a command prompt, the command prompt generates a DNS query to the local DNS server to fetch the IP address of the domain name. The local DNS server then queries the root servers, TLD servers, and authoritative servers in a step-by-step manner until it finds the IP address of the domain name.
- Non-recursive Resolution: When a user types a domain name into a web browser, the browser generates a DNS query to the local DNS server to fetch the IP address of the domain name. If the local DNS server has the IP address of the domain name in its cache, it returns the IP address to the browser. Otherwise, it returns the best answer it has in its cache.

37. Explain DNS caching and updating records?

DNS caching is the process of temporarily storing DNS records on a device to reduce the time it takes to resolve domain names into IP addresses. When a device requests a domain name, the DNS resolver checks its cache to see if it has a record for the domain name. If it does, it returns the IP address associated with the domain name. If it doesn't, it queries other DNS servers to resolve the domain name and stores the result in its cache for future use.

Updating DNS records involves changing the information stored in the DNS cache or authoritative servers. Here's how DNS records are updated:

- DNS cache: DNS records in the cache are updated automatically based on their time-to-live (TTL) values. When the TTL of a record expires, the record is deleted from the cache, and the resolver queries the authoritative server for the updated record.
- Authoritative servers: DNS records in authoritative servers are updated manually by the domain owner or their DNS provider. The domain owner or DNS provider can change the DNS records by accessing the DNS management interface provided by the DNS provider.

### 38. Explain types of DNS records?

There are several types of DNS records that provide important information about a hostname or domain. Here are the five major DNS record types:

1. A Record (Address): The A record is the most important DNS record type. It shows the IP address for a specific domain name.
2. AAAA Record (Quad A): The AAAA record is similar to the A record, but it is used to store IPv6 addresses.
3. CNAME Record (Canonical Name): The CNAME record is used to create an alias for a domain name. It maps one domain name to another domain name.
4. ANAME Record: The ANAME record is similar to the CNAME record, but it is used for the root domain.
5. SOA Record (Start of Authority): The SOA record is used to provide information about the DNS zone, such as the primary name server for the zone and the email address of the zone administrator.

Other common DNS record types include:

- NS Record (Name Server): The NS record is used to specify the authoritative name servers for a domain.
- MX Record (Mail Exchange): The MX record is used to specify the mail servers that are responsible for accepting email messages for a domain.
- TXT Record (Text): The TXT record is used to store arbitrary text data in DNS records.

- SRV Record (Service): The SRV record is used to specify the location of servers for specific services, such as SIP and XMPP.

### 39. Explain types of DNS attacks?

DNS attacks are a type of cyber attack that targets the Domain Name System (DNS) infrastructure, which is responsible for translating domain names into IP addresses. There are several types of DNS attacks, including:

1. DNS Tunneling: This technique enables threat actors to compromise network connectivity and gain remote access to a targeted server. It involves using DNS queries to transfer data between a client and a server.
2. DNS Amplification: This type of attack involves sending a small DNS query to a DNS server and receiving a large response. The attacker then uses the large response to flood the victim's network with traffic, causing a Distributed Denial of Service (DDoS) attack.
3. DNS Flood Attack: This type of attack involves overwhelming a DNS server with a large number of requests, causing it to crash or become unresponsive.
4. DNS Spoofing: This type of attack involves redirecting a user to a fraudulent website by modifying the DNS records of a domain name.
5. NXDOMAIN Attack/ DDOS: This type of attack involves sending a large number of queries for non-existent domain names to a DNS server, causing it to become overloaded and unresponsive.

### 40. Explain TCP

# TCP: Overview

point-to-point:

one sender, one receiver

reliable, in-order *byte stream*:

no “message boundaries”

pipelined:

TCP congestion and flow  
control set window size

- full duplex data:
  - bi-directional data flow in same connection
  - MSS: maximum segment size
- connection-oriented:
  - handshaking (exchange of control msgs) inits sender, receiver state before data exchange
- flow controlled:
  - sender will not overwhelm receiver

## 41. Explain TCP Segment structure?

The TCP segment structure is the format used by the TCP protocol to transmit data between two devices. The TCP segment structure consists of a header and a data section. The header contains control information that is used to manage the transmission of data, while the data section contains the actual data being transmitted. Here's a breakdown of the TCP segment structure:

- Header: The TCP header contains 10 mandatory fields and an optional extension field. The fields in the header include:

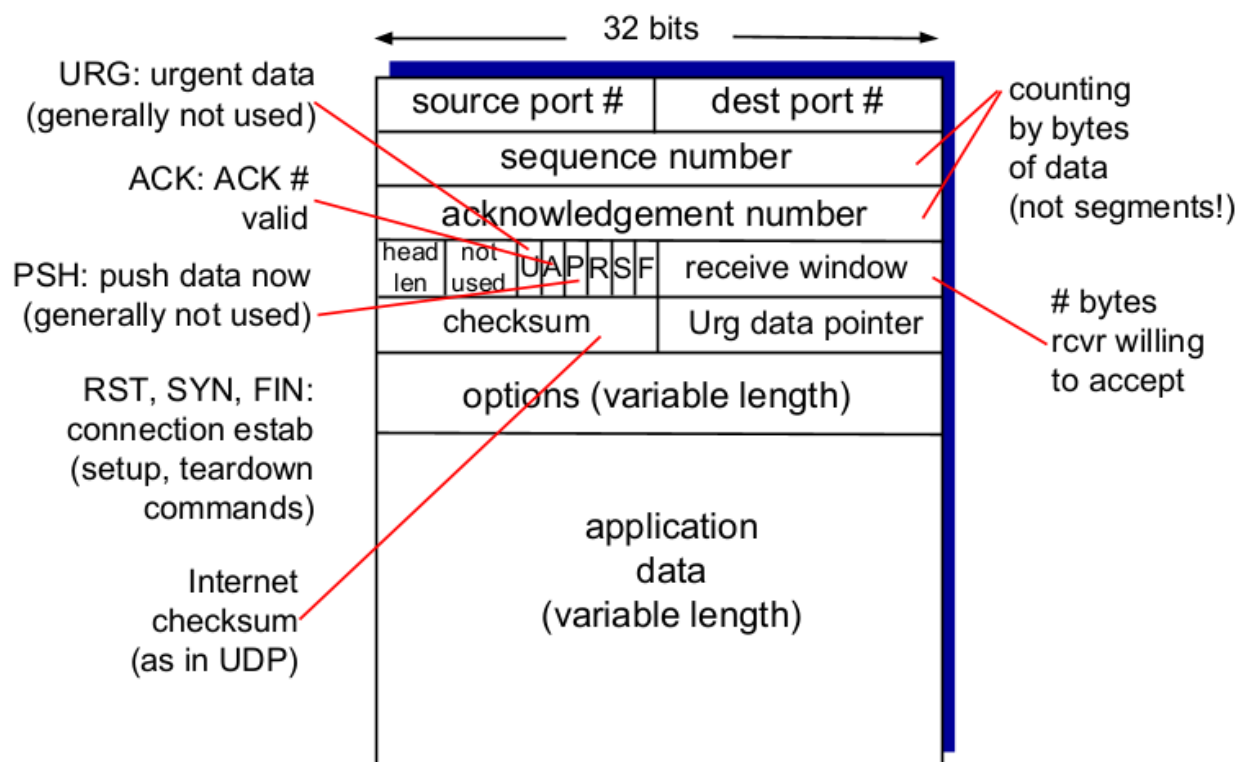
- Source Port: The 16-bit port number of the process that originated the TCP segment on the source device.

- Destination Port: The 16-bit port number of the process that is the intended recipient of the TCP segment on the destination device.

- Sequence Number: The 32-bit sequence number of the first byte of data in the TCP segment.

- Acknowledgment Number: The 32-bit acknowledgment number of the next byte of data that the sender of the TCP segment expects to receive.
- Data Offset: The 4-bit field that specifies the length of the TCP header in 32-bit words.
- Reserved: A 6-bit field that is reserved for future use.
- Control Bits: A 6-bit field that contains control information, such as whether the TCP segment is a SYN, ACK, or FIN segment.
- Window Size: The 16-bit size of the receive window, which is used for flow control.
- Checksum: A 16-bit checksum that is used to verify the integrity of the TCP segment.
- Urgent Pointer: A 16-bit field that is used to indicate the end of urgent data in the TCP segment.
- Data: The data section of the TCP segment contains the actual data being transmitted. The size of the data section is determined by the MSS (Maximum Segment Size) field in the TCP header.

## TCP segment structure



42. Explain TCP seq. numbers and ACKs?

TCP sequence numbers and ACKs are important components of the TCP protocol that enable reliable and ordered data transfer between two devices. Here's an overview of TCP sequence numbers and ACKs based on the search results:

- TCP sequence numbers: TCP sequence numbers are byte-order numbers that indicate how much data has been sent for the session. The sequence number is sent by the TCP client and is used to keep track of every byte sent during the connection. The sequence number is always valid and is the byte number of the first byte of data in the TCP segment sent. The sequence number is coordinated with the ACK number and is a key value during the TCP handshake, TCP close, and while data is transferred between the client and server.

- TCP ACKs: TCP ACKs are sent by the TCP server and indicate that it has received cumulated data and is ready for the next segment. The ACK number is the next byte of data that the sender of the ACK expects to receive. The ACK number is only valid when the ACK flag is set to one. The ACK number is coordinated with the sequence number and is a key value during the TCP handshake, TCP close, and while data is transferred between the client and server.

- Relationship between TCP sequence numbers and ACKs: The TCP seq and ACK numbers are coordinated with one another and are key values during the TCP handshake, TCP close, and while data is transferred between the client and server. The ACK number contains the next sequence number the sender of the ACK expects to receive, thus acknowledging all data up to the ACK minus one. The TCP client sends the sequence number, indicating how much data has been sent for the session, while the TCP server sends the ACK number, indicating that it has received cumulated data and is ready for the next segment.

#### 43. What is TCP reliable data transfer?

TCP reliable data transfer is a mechanism that ensures that data is transmitted reliably and without errors between two devices over an IP network. TCP provides reliable data transfer by implementing several features, including:

- Acknowledgments (ACKs): TCP uses ACKs to ensure that data is received by the receiver. When the receiver receives a segment, it sends an ACK back to the sender to confirm that the data has been received.

- Sequence numbers: TCP uses sequence numbers to ensure that data is transmitted in the correct order. Each segment is assigned a unique sequence number, and the receiver uses these sequence numbers to reassemble the data in the correct order.

- Retransmission: TCP uses retransmission to ensure that lost or corrupted segments are retransmitted. If the sender does not receive an ACK for a segment within a certain time period, it retransmits the segment.

- Flow control: TCP uses flow control to ensure that data is transmitted at a rate that the receiver can handle. The receiver sends a window size to the sender, which indicates how much data it can receive at a time.

- Congestion control: TCP uses congestion control to ensure that the network is not overloaded with traffic. If the sender detects congestion on the network, it reduces the rate at which it sends data.

#### 44. Explain TCP sender events?

TCP sender events are the various actions and triggers that occur on the sender's side during a TCP connection. These events are responsible for the reliable and efficient transmission of data. Here are some examples of TCP sender events based on the search results:

- Data received from the application: When the sender receives data from the application layer, it creates a TCP segment with a sequence number and sends it to the receiver[1].
- Sequence number generation: The sender assigns a unique sequence number to each TCP segment it sends. The sequence number indicates the position of the first byte of data in the segment within the overall data stream.
- Retransmission timeout (RTO): If the sender does not receive an acknowledgment (ACK) for a segment within a certain time period, it retransmits the segment. The RTO is a timer that helps the sender determine when to retransmit a segment[5].
- Congestion control: The sender adjusts its sending rate based on the network conditions to avoid congestion. This is achieved through various algorithms, such as additive increase/multiplicative decrease (AIMD), slow start, and congestion window (CWND).
- Flow control: The sender's data transmission is controlled by the receive window advertised by the receiver. The sender can only send data up to the size of the receive window before it must wait for an ACK and receive window update from the receiver.

#### 45. How is Connection Management handled in TCP?

TCP connection management is the process of establishing, maintaining, and terminating a TCP connection between two devices. The connection management process is handled by the TCP protocol and involves several events that occur on the sender's side. Here's an overview of how connection management is handled in TCP:

- Connection establishment: The sender initiates the connection establishment process by sending a SYN (synchronize) segment to the receiver. The receiver responds with a SYN-ACK (synchronize-acknowledge) segment, and the sender sends an ACK (acknowledge) segment to complete the three-way handshake.
- Data transfer: Once the connection is established, data can be transmitted between the sender and receiver. The sender sends data segments to the receiver, and the receiver sends ACK segments back to the sender to confirm that the data has been received.
- Flow control: The sender adjusts its sending rate based on the receiver's advertised window size, which indicates how much data the receiver can receive at a time.



- Congestion control: The sender adjusts its sending rate based on the network conditions to avoid congestion. This is achieved through various algorithms, such as additive increase/multiplicative decrease (AIMD), slow start, and congestion window (CWND).

- Connection termination: When the data transfer is complete, the sender initiates the connection termination process by sending a FIN (finish) segment to the receiver. The receiver responds with an ACK segment, and then sends its own FIN segment to the sender. The sender responds with an ACK segment to complete the four-way handshake.

#### 46. explain TCP 3-way handshake?

The TCP 3-way handshake is a process used by the TCP protocol to establish a reliable connection between two devices over an IP network. Here's how the TCP 3-way handshake works:

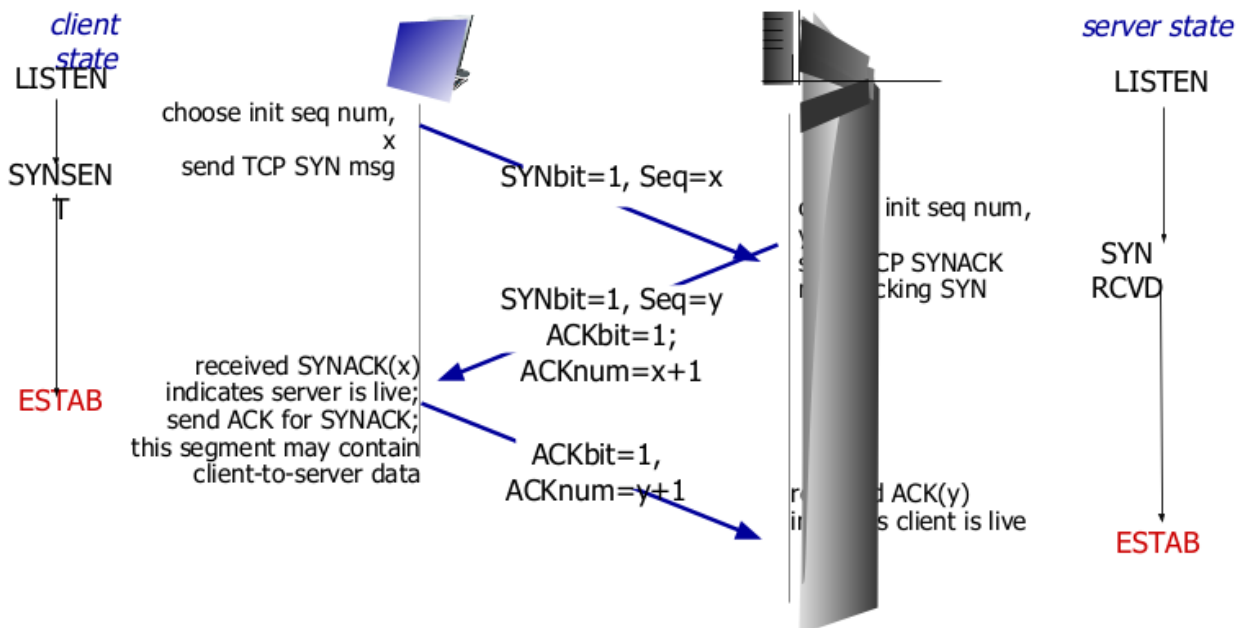
- Step 1 (SYN): The client sends a SYN (synchronize) segment to the server to initiate the connection. The SYN segment contains a sequence number that the server will use to acknowledge the connection request.

- Step 2 (SYN + ACK): The server responds to the client's request with a SYN-ACK (synchronize-acknowledge) segment. The SYN-ACK segment contains the server's own sequence number and an acknowledgment number that confirms the client's sequence number.

- Step 3 (ACK): The client sends an ACK (acknowledge) segment to the server to confirm that it has received the SYN-ACK segment. The ACK segment contains the client's own sequence number and an acknowledgment number that confirms the server's sequence number.

Once the 3-way handshake is complete, data can be transmitted between the sender and receiver. The sender sends data segments to the receiver, and the receiver sends ACK segments back to the sender to confirm that the data has been received. The TCP 3-way handshake is also used to terminate the connection between the sender and receiver.

# TCP 3-way handshake



## 47. How does TCP close a connection?

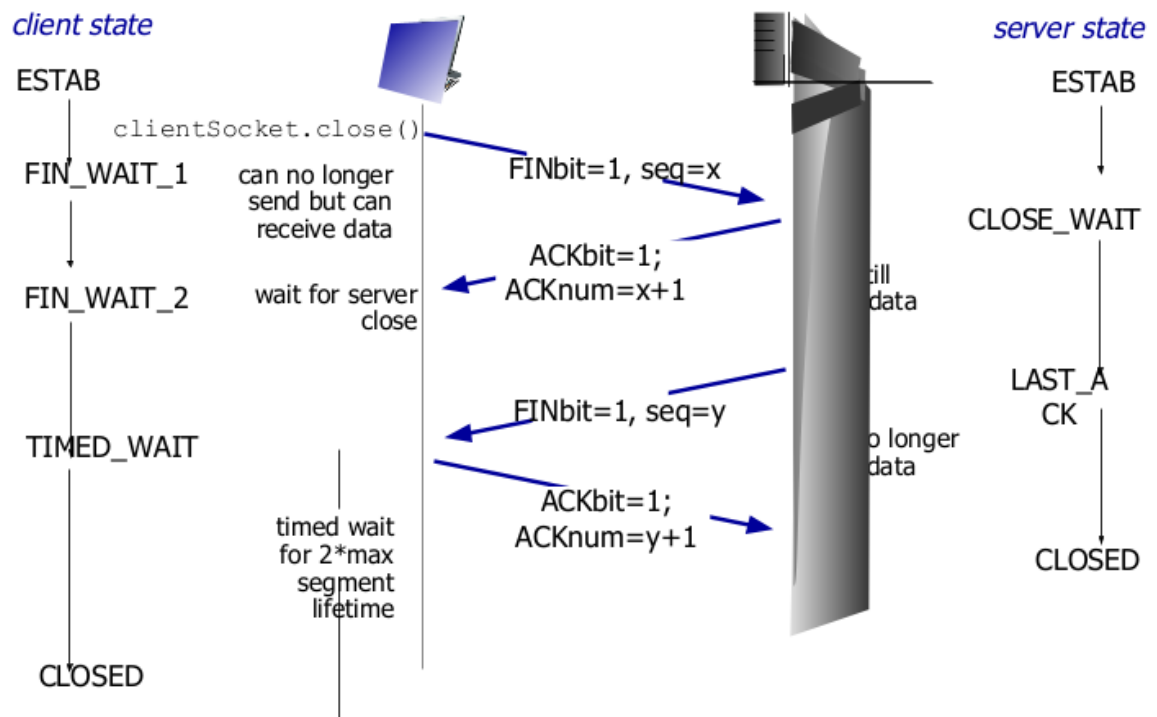
TCP uses a four-way handshake to close a connection, allowing each host to release its own side of the connection individually. The steps involved in TCP connection termination are as follows:

- **Step 1 (FIN from the client):** The client initiates the connection termination process by sending a FIN (finish) segment to the server. This segment indicates that the client has finished sending data and wants to close the connection.
- **Step 2 (ACK from the server):** Upon receiving the FIN segment, the server sends an ACK (acknowledge) segment back to the client to confirm the receipt of the FIN segment.
- **Step 3 (FIN from the server):** After sending the ACK segment, the server also decides to close the connection and sends its own FIN segment to the client.
- **Step 4 (ACK from the client):** The client responds to the server's FIN segment by sending an ACK segment, confirming the receipt of the server's FIN segment.

Once the four-way handshake is complete, the connection is closed, and both the client and server can no longer send or receive data on that connection. The TCP connection termination process ensures that all data has been transmitted and received before

closing the connection, providing a reliable and orderly way to end the communication between the sender and receiver.

## TCP: closing a connection



### 48. Explain Encoding and Type of Encoding Techniques?

Encoding is the process of converting data from one form to another form for the secured transmission of data. Encoding is used to transform the data so that data can be supported and used by different systems. Encoding works similarly to converting temperature from centigrade to Fahrenheit, as it just gets converted in another form, but the original value always remains the same. Encoding is used in mainly two fields: computing and communication. There are different types of encoding techniques used in computing and communication, including:

1. **Character Encoding:** Character encoding encodes characters into bytes. It informs the computers how to interpret the zero and ones into real characters, numbers, and symbols. The computer understands only binary data; hence it is required to convert these characters into numeric values.
2. **HTML Encoding:** HTML encoding is used to encode special characters in HTML code. It is used to prevent the browser from interpreting certain characters as HTML code.
3. **URL Encoding:** URL encoding is used to encode special characters in a URL. It is used to prevent the browser from interpreting certain characters as part of the URL.

4. UNICODE Encoding: UNICODE encoding is used to represent characters from all languages in a single character set. It is used to support multilingual text.

5. Base64 Encoding: Base64 encoding is used to encode binary data into ASCII text format. It is used to transmit binary data over channels that only support text data.

6. ASCII Encoding: ASCII encoding is used to represent characters in the English language using 7 bits.

#### 49. Explain URL Encoding?

URL encoding is a process of converting special characters in a URL into a format that can be transmitted over the internet. URLs can only be sent over the internet using the ASCII character set. If a URL contains characters outside the ASCII set, the URL has to be converted. URL encoding replaces non-ASCII characters with a "%" followed by hexadecimal digits. URLs cannot contain spaces. URL encoding normally replaces a space with a plus (+) sign, or %20. Here are some key points about URL encoding:

- URL encoding is also called percent encoding since it uses percent sign (%) as an escape character.
- URL encoding converts reserved, unsafe, and non-ASCII characters in URLs to a format that is universally accepted and understood by all web browsers and servers.
- URL encoding is used to prevent the browser from interpreting certain characters as part of the URL.
- URL encoding is used to encode special characters in a URL. It is used to prevent the browser from interpreting certain characters as HTML code.
- URL encoding is used to support multilingual text.
- URL encoding is used to transmit binary data over channels that only support text data.
- URL encoding replaces non-ASCII characters with a "%" followed by hexadecimal digits.
- URLs cannot contain spaces. URL encoding normally replaces a space with a plus (+) sign, or %20.

#### 50. Explain UTF Encoding?

Unicode encoding is a character encoding system that is used to represent characters from all languages in a single character set. Unicode encoding is used to support multilingual text and eliminate the need for multiple character sets. The Unicode standard defines Unicode Transformation Format (UTF) to encode the code points. There are different types of Unicode encoding techniques, including UTF-8, UTF-16, and UTF-32. Here's an overview of the types of UTF:

1. UTF-8: UTF-8 is the most commonly used Unicode encoding technique. It is a variable-length encoding system that uses one to four bytes to represent a character. UTF-8 is backward compatible with ASCII, which means that the first 128 characters are mapped to the same values as ASCII.
2. UTF-16: UTF-16 is an extension of the UCS-2 Unicode encoding technique. It uses two bytes to represent 65,536 characters and supports four bytes for additional characters up to one million. UTF-16 is used mainly in Windows operating systems.
3. UTF-32: UTF-32 is a fixed-length encoding system that uses four bytes to represent a character. UTF-32 is used mainly in Unix and Linux operating systems.

#### 51. Explain Base64 Encoding?

Base64 encoding is a binary-to-text encoding scheme that represents binary data in an ASCII string format. It is used to transmit binary data over channels that only support text data. Base64 encoding takes any form of data and transforms it into a long string of plain text. Here are some key points about Base64 encoding:

- Base64 encoding is used to encode binary data as printable text.
- Base64 encoding is a way of taking binary data and turning it into text so that it's more easily transmitted in things like e-mail and HTML.
- Base64 encoding is designed to carry data stored in binary format across channels that only reliably support text content.
- Base64 encoding is particularly prevalent on the World Wide Web where one of its uses is the ability to embed image files or other binary assets inside textual assets such as HTML and CSS files.
- Base64 encoding can be helpful when fairly lengthy identifying information is used in an HTTP environment.
- Base64 encoding uses 64 characters in its "alphabet", which are A-Z, a-z, 0-9, +, and /.
- Base64 encoding is often used to represent binary data in a text format that can be easily transmitted over networks or stored in a text file.
- Base64 encoding increases the size of the data by approximately 33%.

There are different types of Base64 encoding techniques, including:

1. Standard Base64: Standard Base64 encoding uses the 64 characters in its "alphabet" to represent binary data.
2. URL-safe Base64: URL-safe Base64 encoding replaces the "+" and "/" characters with "-" and "\_" characters, respectively, to make it safe for use in URLs.
3. MIME Base64: MIME Base64 encoding is used to encode binary data in email messages and other MIME (Multipurpose Internet Mail Extensions) content.

## 52. Explain ASCII Encoding?

ASCII (American Standard Code for Information Interchange) is a character encoding standard for electronic communication. It is a 7-bit character set containing 128 characters, including upper and lower case English letters, digits from 0 to 9, and some special characters.

ASCII encoding is based on character encoding used for telegraph data. The American National Standards Institute first published it as a standard for computing in 1963.

ASCII encoding is used to represent text in computers, telecommunications equipment, and other devices. ASCII codes represent text in computers, telecommunications equipment, and other devices. ASCII encoding is limited to 128 code points, of which only 95 are printable characters, which severely limited its scope. ASCII encoding is used in many applications, including email, text editors, and programming languages.

## Solved papers

1. TTL stands for **Time To Leave**
2. What layer provides services to user - **Application Layer**
3. Which protocol uses both TCP and UDP? - **DNS**
4. which layer 4 protocol used for telnet? - **TCP**
5. DoS abbreviated as - **Denial of Service**
6. ARP stands for **Address Resolution Protocol**.

## 7. How TCP/UDP Protocol works?

TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) are two different methods for transferring data via the internet. They enable servers and devices to communicate so you can send emails, watch Netflix, play games, and browse web pages. Here's how TCP and UDP protocols work:

### TCP Protocol:

- TCP is a connection-oriented protocol that provides reliable, ordered, and error-checked delivery of a stream of octets (bytes) between applications running on hosts communicating via an IP network.

- TCP creates a secure communication line to ensure the reliable transmission of all data. Once a message is sent, the receipt is verified to make sure all the data was transferred.

- TCP establishes a connection before transmitting data.

- TCP sequences data (sends in a specific order).

- TCP can retransmit lost packets.

- TCP uses handshakes like SYN, ACK, SYN-ACK.

- TCP reads data as byte streams and transmits messages to segment boundaries.

- TCP does not support broadcasting.

#### UDP Protocol:

- UDP is a connectionless protocol that does not establish a connection when sending data. It sends data without confirming receipt or checking for errors. That means some or all of the data may be lost during transmission.
- UDP does not sequence or arrange data.
- UDP has no lost packet retransmission capability.
- UDP has no overhead for opening a connection, maintaining a connection, or terminating a connection.
- UDP supports broadcasting.

#### 8. What is fingerprinting web applications?

Web application fingerprinting is the process of identifying the components and technologies used by a web application or framework. Fingerprinting can be performed by attackers or security testers to gather information about the target application, which can be used to formulate a targeted attack plan. Here are some key points about web application fingerprinting:

- Web application fingerprinting involves identifying the version of the web server and the web application framework used by the target application.
- Fingerprinting can be performed using various tools and techniques, including analyzing HTTP headers, cookies, and directory structures.
- Web application fingerprinting can help attackers or security testers to identify vulnerabilities and weaknesses in the target application.
- Fingerprinting can also help security testers to reduce the effort required during the testing process.
- Web application fingerprinting is not intended to be damaging, but certain activities, such as network scans, can sometimes cause disruptions to vulnerable applications inadvertently.
- There are several tools available for web application fingerprinting, including WhatWeb, Wappalyzer, and BlindElephant.

#### 9. What is subdomains enumeration?

Subdomain enumeration is the process of identifying all subdomains for a given domain. It is an essential part of the reconnaissance phase in the cyber kill chain. Subdomain enumeration can be performed using various tools and techniques, including analyzing HTTP headers, cookies, and directory structures. Subdomain enumeration can help attackers or security testers to identify vulnerabilities and weaknesses in the target application. It can also help security testers to reduce the effort required during the

testing process. Subdomain enumeration can be used for a variety of purposes, such as identifying potential targets for an attack, inventorying owned domains, or identifying which domains are being used for which purposes. There are several tools available for subdomain enumeration, including WhatWeb, Wappalyzer, DNSRecon, Amass, Sublist3r, and Findomain.

#### 10. Explain about OWASP Zed attack proxy

The Open Web Application Security Project (OWASP) Zed Attack Proxy (ZAP) is a free and open-source web application security testing tool. It is a web proxy that sits between the user's browser and the web application under test. It can intercept and modify all HTTP traffic, which allows it to be used to perform a variety of security tests, such as:

- \* **Vulnerability scanning**: ZAP can scan web applications for known vulnerabilities, such as SQL injection, cross-site scripting, and command injection.
- \* **Fuzzing**: ZAP can generate random or specific inputs to web applications to test for vulnerabilities that may not be detected by vulnerability scanners.
- \* **Session hijacking**: ZAP can intercept and steal users' session cookies, which allows it to take over their sessions.
- \* **Man-in-the-middle attacks**: ZAP can modify the contents of HTTP traffic, which allows it to perform man-in-the-middle attacks.
- Passive Scanner**: ZAP can detect vulnerabilities without directly contacting the target machine.
- Active Scanner**: ZAP can detect vulnerabilities by directly contacting the target machine, which makes it easier to detect by the administrator.
- Proxy Server**: ZAP can be used as a proxy server, allowing users to manipulate all traffic passing through it, including HTTPS traffic.
- Port Identification**: ZAP can identify open ports on a target machine[1].
- Directory Searching**: ZAP can search for directories and files on a target machine[1].
- Brute Force Attack**: ZAP can perform brute force attacks to test the strength of passwords and other credentials[1].
- Web Crawler**: ZAP can crawl through web pages to find links and other important details.

ZAP is a powerful tool that can be used to find and exploit a wide range of web application vulnerabilities. It is used by security professionals all over the world to test the security of web applications.

Here are some of the benefits of using OWASP Zed Attack Proxy:

- \* It is free and open-source, which makes it accessible to everyone.
- \* It is easy to use and does not require any programming skills.
- \* It is very powerful and can be used to find and exploit a wide range of web application vulnerabilities.
- \* It is regularly updated with new features and bug fixes.



## 11. Compare authentication to authorization

Authentication and authorization are two distinct processes used in information security to protect systems and data. While they are often used interchangeably, they serve different purposes and work together to ensure the security of an application or system.

### **\*\*Authentication\*\*:**

- Verifies the identity of a user or entity
- Determines whether users are who they claim to be
- Involves challenging the user to validate credentials (e.g., passwords, security questions, or biometric factors)
- Usually done before authorization
- Transmits information through an ID Token
- Popular techniques include password-based authentication, passwordless authentication, two-factor authentication (2FA), multi-factor authentication (MFA), single sign-on (SSO), and social authentication.

### **\*\*Authorization\*\*:**

- Determines what specific applications, files, and data a user has access to
- Verifies whether access is allowed through policies and rules
- Usually done after successful authentication
- Transmits information through an Access Token
- Popular techniques include role-based access controls (RBAC), JSON web token (JWT) authorization, Security Assertion Markup Language (SAML) authorization, OpenID authorization, and OAuth 2.0 authorization

To illustrate the difference between authentication and authorization, consider the example of airport security. When you go through security at an airport, you show your ID to authenticate your identity. Then, when you arrive at the gate, you present your boarding pass to the flight attendant for authorization to board the plane and access the designated seat. Similarly, in the digital world, authentication verifies the user's identity, while authorization grants the user permission to access different levels of information and perform specific functions, depending on the established rules.

## 12. write a note about XSS

**\*\*Cross-site scripting (XSS)\*\*** is a type of web security vulnerability that allows an attacker to inject malicious code into a web page viewed by another user. The malicious code can then be executed by the victim's browser, giving the attacker control over the victim's session.

XSS attacks are one of the most common web security vulnerabilities, and they can be used to steal sensitive information, such as cookies and session tokens, redirect users to malicious websites, or even take over victims' accounts.

There are three main types of XSS attacks:

\* \*\*Reflected XSS:\*\* This type of XSS attack occurs when an attacker injects malicious code into a web request, and the malicious code is then reflected back to the victim in the response. For example, an attacker might inject malicious code into a search form, and the malicious code would then be displayed in the search results.

\* \*\*Stored XSS:\*\* This type of XSS attack occurs when an attacker injects malicious code into a web page that is stored on the server. The malicious code is then executed whenever the page is loaded by a victim. For example, an attacker might inject malicious code into a comment on a blog post, and the malicious code would then be executed whenever the blog post is viewed by a victim.

\* \*\*DOM-based XSS:\*\* This type of XSS attack occurs when an attacker injects malicious code into the Document Object Model (DOM) of a web page. The malicious code is then executed whenever the DOM is manipulated by the browser. For example, an attacker might inject malicious code into a JavaScript variable, and the malicious code would then be executed whenever the variable is used.

XSS attacks can be mitigated by following a few simple best practices:

\* \*\*Escape all user input:\*\* Before displaying any user input in a web page, it is important to escape it to prevent malicious code from being executed. This can be done using a variety of different escaping techniques, such as HTML escaping and JavaScript escaping.

\* \*\*Use a content security policy (CSP):\*\* A CSP is a security policy that can be used to control what scripts and styles can be loaded on a web page. By using a CSP, you can prevent attackers from injecting malicious code into your web pages.

\* \*\*Keep your software up to date:\*\* Software vendors regularly release security patches to fix known vulnerabilities. It is important to keep your software up to date to protect your web applications from XSS attacks.

### 13. what is burp sequencer

Burp Sequencer is a tool in Burp Suite that is used to analyze the quality of randomness in a sample of tokens. Tokens can be anything from session IDs to password reset tokens to CSRF tokens. Burp Sequencer runs a variety of randomness tests on the tokens and then compiles the results to give you an indication of the quality of randomness in the sample.

Burp Sequencer is a useful tool for testing the security of web applications. By testing the randomness of tokens, you can identify weaknesses that could be exploited by attackers. For example, if an attacker is able to predict the next session ID, they could hijack a user's session.

Here are some of the benefits of using Burp Sequencer:

\* It can help you to identify weaknesses in the randomness of tokens, which could be exploited by attackers.

\* It is easy to use and does not require any programming skills.

\* It is integrated with other Burp Suite tools, such as the Intruder and Repeater tools.

To use Burp Sequencer, you first need to capture a sample of tokens. This can be done by using the Proxy or Spider tools in Burp Suite. Once you have captured a sample of tokens, you can send them to Burp Sequencer by right-clicking on them and selecting "Send to Sequencer".

Once the tokens have been sent to Burp Sequencer, you can start the analysis by clicking the "Analyze now" button. Burp Sequencer will then run a variety of randomness tests on the tokens and compile the results.

The results of the analysis are displayed in the Burp Sequencer window. The results include information on the distribution of characters in the tokens, the number of unique tokens in the sample, and the results of the randomness tests.

If the results of the analysis indicate that the tokens are not random, you should investigate further to identify the cause of the weakness. You may need to change the way that the tokens are generated or stored.

#### 14. Explain steps for secure source code review

To perform a secure source code review, you can follow these steps:

1. **\*\*Prepare for the review.\*\*** This includes understanding the codebase, identifying the security requirements, and developing a review checklist.
2. **\*\*Review the code.\*\*** This includes looking for common security vulnerabilities, such as SQL injection, cross-site scripting, and insecure coding practices.
3. **\*\*Document the findings.\*\*** This includes identifying the severity of each vulnerability and providing recommendations for remediation.
4. **\*\*Remediate the vulnerabilities.\*\*** The development team should fix the vulnerabilities that were identified during the review.
5. **\*\*Verify the fixes.\*\*** Once the vulnerabilities have been fixed, the security team should verify that they have been fixed correctly.

Here are some additional tips for performing a secure source code review:

- \* Use a variety of review techniques. This includes manual code review, automated static analysis, and dynamic analysis.
- \* Involve multiple reviewers. This will help to ensure that different perspectives are considered and that no vulnerabilities are missed.
- \* Focus on the most critical code. This includes code that handles sensitive data, performs authentication and authorization, and interacts with external systems.
- \* Use a consistent review process. This will help to ensure that all code is reviewed thoroughly and that vulnerabilities are identified and remediated in a timely manner.

Here is a sample secure code review checklist:

- \* **\*\*Authentication and authorization:\*\***
  - \* Are users authenticated and authorized before accessing sensitive data?
  - \* Are permissions assigned correctly?
  - \* Is session management secure?

- \* \*\*Input validation:\*\*
  - \* Is all user input validated to prevent malicious code from being executed?
  - \* Are sensitive data fields protected from unauthorized access?
- \* \*\*Error handling:\*\*
  - \* Are errors handled securely?
  - \* Are sensitive data errors logged?
  - \* Are users prevented from exploiting errors?
- \* \*\*Security configuration:\*\*
  - \* Are security settings configured correctly?
  - \* Are security updates applied regularly?
- \* \*\*Cryptography:\*\*
  - \* Is sensitive data encrypted?
  - \* Are cryptographic algorithms used correctly?
- \* \*\*Logging:\*\*
  - \* Are security events logged?
  - \* Are logs reviewed regularly for suspicious activity?

By following these steps and tips, you can perform secure source code reviews that will help to identify and fix security vulnerabilities before they can be exploited by attackers.

## 15. Explain VAPT lifecycle

### 1. Initial Preparation:

the team decides the scope and goals of vulnerability testing  
Identifying protected assets and equipment and mapping out all endpoints.

Determining the business value of each asset and the impact if it is attacked.  
Identifying access controls and other security requirements of each system.

Determining if systems hold sensitive data, and how sensitive data is transferred between systems.

Recording a baseline of services, processes, and open ports on protected assets.

Determining operating systems and software deployed on assets.

### 2. Vulnerability Assessment Testing:

In this stage, the team runs automated vulnerability scans on target devices and environments. If necessary, they use manual tools to investigate the security posture of a system.

In order to automate this stage and make it more efficient, teams will typically rely on one or more vulnerability databases, vendor security advisories, and threat Intelligence feeds.

A single test can take anywhere from a minute to several hours, depending on the size of the target system and the type of scan.

### 3. Prioritize Vulnerabilities:

At this stage, the team removes false positives from vulnerability scanning results and prioritizes vulnerabilities according to several factors. These can include:

- Severity score provided by a vulnerability database
- The business impact if a vulnerability is exploited
- Sensitive data that might be at risk
- The ease of exploiting the vulnerability
- How long the vulnerability has been in place
- The ability to perform lateral movement from this system to other sensitive systems
- The availability of a patch and the effort needed to deploy it

#### 4. Create a Vulnerability Assessment Report:

At this stage, the team creates a unified report showing vulnerabilities found in all protected assets, with a plan for remediating them.

For medium to high risk vulnerabilities, the report should provide information about the vulnerability, when it was discovered, which systems it affects, the potential damage if attackers exploit it, and the plan and effort required to remediate it.

Where possible, the team should also provide a proof of concept (PoC) demonstrating how each critical vulnerability could be exploited.

#### 5. Continuous Vulnerability Assessment:

Vulnerability scans provide a point-in-time snapshot of vulnerabilities that exist in an organization's digital infrastructure.

However, new deployments, configuration changes, newly discovered vulnerabilities, and other factors can result in new vulnerabilities.

Because vulnerabilities are not static, vulnerability management should also be a continuous process.

Software development teams should incorporate automated vulnerability assessment into their continuous integration and deployment (CI/CD) pipeline.

This allows vulnerabilities to be identified and fixed as early as possible in the software development lifecycle (SDLC), eliminating the need to develop and release patches for vulnerable code.

However, because this process cannot catch all vulnerabilities, and many vulnerabilities occur in legacy or third-party systems, it must be complemented by continuous vulnerability scans of production systems.

#### 16. write steps about this website for SQL injection : <https://esjindex.org/search.php?id=1>

1. Open the website <https://esjindex.org/search.php?id=1> in your browser.

2. In the search box, enter a single quote (') and click the search button.
3. The website will return an error message that includes a SQL query. This indicates that the website is vulnerable to SQL injection.
4. Modify the query by adding a comment symbol (--), followed by a space, and then the word "password" to the end of the query. The modified query should look like this:  
...  
`?id=1' -- password`  
...
5. Press Enter to execute the modified query.
6. The website will now display the password field, which was previously hidden.
7. You can now enter any value in the password field and click the search button to see the results.

## 17. Describe click jacking

Clickjacking is a type of web attack in which an attacker tricks a user into clicking on something different from what the user perceives they are clicking on. This can be done by overlaying a hidden layer on top of a legitimate web page, or by using HTML elements such as iframes or transparent images to create the illusion of a clickable element.

When a user clicks on the hidden layer or HTML element, they are actually clicking on the underlying element, which could be a malicious button or link. This can lead to a variety of negative consequences, such as:

- \* Downloading malware
- \* Visiting malicious websites
- \* Providing sensitive information, such as passwords or credit card numbers
- \* Authorizing fraudulent transactions
- \* Installing unwanted software

Clickjacking attacks can be very difficult to detect, as the user may not even be aware that they have been attacked. However, there are a few things that users can do to protect themselves, such as:

- \* Being careful about what links they click on, especially in emails and messages from unknown senders
- \* Using a browser that supports clickjacking protection features, such as X-Frame-Options and Content Security Policy
- \* Keeping their software up to date, including their browser and operating system

Website owners can also help to protect their users from clickjacking attacks by implementing clickjacking protection measures, such as:

- \* Setting the X-Frame-Options header to prevent their website from being embedded in iframes
- \* Using Content Security Policy to restrict the types of resources that can be loaded on their website
- \* Using CAPTCHAs or other challenges to verify that users are human before allowing them to perform sensitive actions

## 18. Explain OWASP top ten attack

The OWASP Top 10 is a list of the most critical web application security risks, as identified by the Open Web Application Security Project (OWASP). The list is updated annually based on input from security experts around the world.

The 2023 OWASP Top 10 is as follows:

1. **Injection:** Injection attacks occur when an attacker is able to inject malicious code into a web application, which is then executed by the application. This can allow the attacker to take control of the application or steal data from it.
2. **Broken Authentication:** Broken authentication attacks occur when a web application does not properly authenticate its users. This can allow attackers to log in to the application as other users or even create new user accounts without authorization.
3. **Sensitive Data Exposure:** Sensitive data exposure attacks occur when a web application exposes sensitive data, such as passwords or credit card numbers, to unauthorized users. This can happen due to vulnerabilities in the application's code or configuration.
4. **XML External Entities (XXE):** XXE attacks occur when a web application does not properly validate XML input. This can allow attackers to inject malicious code into the XML, which is then executed by the application.
5. **Broken Access Control:** Broken access control attacks occur when a web application does not properly restrict access to its resources. This can allow attackers to access resources that they should not be able to access, or to perform actions that they should not be able to perform.
6. **Security Misconfigurations:** Security misconfigurations attacks occur when a web application is not properly configured. This can leave the application vulnerable to attack, even if the application's code is secure.
7. **Cross-Site Scripting (XSS):** XSS attacks occur when a web application does not properly sanitize user input. This can allow attackers to inject malicious code into the application, which is then executed by other users' browsers.
8. **Insecure Direct Object References:** Insecure direct object references attacks occur when a web application allows users to access resources directly without proper authentication or authorization. This can allow attackers to access resources that they should not be able to access, or to perform actions that they should not be able to perform.
9. **Using Components with Known Vulnerabilities:** Using components with known vulnerabilities attacks occur when a web application uses components, such as libraries or plugins, that have known vulnerabilities. This can leave the application vulnerable to attack, even if the application's own code is secure.
10. **Insufficient Logging & Monitoring:** Insufficient logging and monitoring attacks occur when a web application does not properly log or monitor its activity. This can make it difficult to detect and respond to attacks.

By understanding and addressing the OWASP Top 10, organizations can significantly reduce their risk of being attacked.

- **Injection:** Use a web application firewall (WAF) to filter out malicious input. Also, use prepared statements to sanitize user input before using it in database queries.

- **Broken Authentication:** Use strong authentication mechanisms, such as two-factor authentication. Also, implement password lockout policies to prevent attackers from brute-forcing user passwords.
- **Sensitive Data Exposure:** Encrypt sensitive data at rest and in transit. Also, implement access control restrictions to prevent unauthorized users from accessing sensitive data.
- **XML External Entities (XXE):** Disable XXE processing in your web application. Also, use a WAF to filter out malicious XML input.
- **Broken Access Control:** Implement role-based access control (RBAC) to restrict user access to resources based on their role in the application. Also, test your access control rules regularly to ensure that they are working as expected.
- **Security Misconfigurations:** Follow security best practices when configuring your web application. Also, use a WAF to protect your application from common attack vectors.
- **Cross-Site Scripting (XSS):** Sanitize all user input before displaying it in the application. Also, use a WAF to filter out malicious XSS attacks.
- **Insecure Direct Object References:** Implement access control restrictions to prevent unauthorized users from accessing resources directly. Also, use a WAF to filter out malicious requests.
- **Using Components with Known Vulnerabilities:** Keep all of your web application components, such as libraries and plugins, up to date. Also, use a vulnerability scanner to scan your application for known vulnerabilities.
- **Insufficient Logging & Monitoring:** Implement logging and monitoring for your web application. Also, review your logs regularly to detect and respond to attacks.

## 19. write note on fuzzing with burp intruder

Fuzzing with Burp Intruder is a powerful way to test web applications for vulnerabilities. Burp Intruder allows you to send a large number of requests to a web application with different payloads in order to see how the application responds. This can help you to identify vulnerabilities such as SQL injection, XSS, and LFI.

To fuzz with Burp Intruder, you will first need to intercept a request to the web application that you want to fuzz. You can do this by enabling the proxy in Burp Suite and then visiting the web application in your browser.

Once you have intercepted a request, you can send it to Burp Intruder by right-clicking on the request and selecting "Send to Intruder". In Burp Intruder, you will need to configure the attack. This includes specifying the payload positions, the payload type, and the payload list.

The payload positions are the parts of the request that you want to fuzz. You can specify the payload positions by clicking on the "Positions" tab in Burp Intruder and then selecting the parts of the request that you want to fuzz.

The payload type is the type of data that you want to use to fuzz the request. Burp Intruder provides a variety of predefined payload types, such as common SQL injection strings and XSS payloads. You can also create your own custom payload types.



The payload list is the list of payloads that you want to use to fuzz the request. Burp Intruder provides a variety of predefined payload lists, such as the Common Crawling List and the Web Scarab List. You can also import your own custom payload lists.

Once you have configured the attack, you can start the attack by clicking on the "Start Attack" button. Burp Intruder will then send a large number of requests to the web application with different payloads. You can monitor the progress of the attack in the "Results" tab.

After the attack has finished, you can review the results to identify any vulnerabilities. Burp Intruder will highlight any responses that contain errors or other suspicious behavior. You can then investigate these responses further to see if they indicate a vulnerability.

Fuzzing with Burp Intruder is a powerful way to test web applications for vulnerabilities. However, it is important to note that fuzzing can be slow and resource-intensive. It is also important to be careful when fuzzing live web applications, as you could accidentally cause a service outage or data loss.

## 20. What is CVE

A list of publicly disclosed information security vulnerabilities and exposures

CVE was launched in 1999 by the MITRE corporation to identify and categorize vulnerabilities in software and firmware.

CVE provides a free dictionary for organizations to improve their cyber security.

MITRE is a nonprofit that operates federally funded research and development centers in the United States.

The goal of CVE is to make it easier to share information about known vulnerabilities across organizations.

CVE does this by creating a standardized identifier for a given vulnerability or exposure. CVE identifiers or CVE names allow security professionals to access information about specific cyber threats across multiple information sources using the same common name.

CVE allows organizations to set a baseline for evaluating the coverage of their security tools. CVE's common identifiers allow organizations to see what each tool covers and how appropriate they are for your organization.

CVE means security advisories that can for vulnerabilities and check for threats can use CVE information to search for known attack signatures to identify particular vulnerability exploits as part of any digital forensics process.

Look for security tools with CVE compatibility rather than proprietary vulnerability assessments, it's a great way to reduce your organization's cyber security risk.

CVE isn't a vulnerability database. CVE is designed to allow vulnerability databases and other tools to be linked together. It also facilitates comparisons between security tools and services.

The latest version of the CVE list can always be found on [cve.mitre.org](https://cve.mitre.org). While the CVE list is free, it can be hard to know which vulnerabilities affect your organization without additional tools. This is why many organizations now use tools that monitor for changes in the CVE list that affect them.

## 21. What are CNAs?

CVE Numbering Authorities (CNAs) are organizations that identify and distribute CVE id numbers to researchers and vendors for inclusion in public announcements of new vulnerabilities. CNAs include software vendors, open source projects, coordination centers, bug bounty service providers and research groups.

CNAs are a federated systems that helps identify vulnerabilities and assigns them an ID without directly involving MITRE which is the primary CNA.

There are currently 104 CNAs in 18 countries including many household names like Microsoft, Adobe, Apple, Cisco, Google, Hewlett Packard Enterprise, Huawei, IBM, Intel, Mozilla, Oracle, Red Hat, Siemens, Symantec, VMWare, Atlassian, Autodesk, Cloudflare, Elastic, GitHub, Kubernetes, Netflix and Salesforce.

MITRE serves as the primary CNA while root CNAs cover a certain area or niche.

In many cases, a root CNA is a major company like Apple who posts vulnerabilities about its own products. In other cases, the root CNA may be focused on open source vulnerabilities.

## 22. What is the Common Vulnerability Scoring System (CVSS)?

The Common Vulnerability Scoring System (CVSS) is a free and open industry standard for assessing the severity of computer system security vulnerabilities. CVSS attempts to assign severity scores to vulnerabilities, allowing responders to prioritize responses and resources according to threat.

CVSS scores are based on three metric groups: Base, Temporal, and Environmental. The Base metrics produce a score ranging from 0 to 10, which can then be modified by scoring the Temporal and Environmental metrics. The Base metrics measure the intrinsic properties of a vulnerability, including its exploitability, impact, and scope. The Temporal metrics measure the characteristics of a vulnerability that change over time, such as the availability of exploits and the prevalence of affected systems. The Environmental metrics measure the characteristics of a specific environment, such as the presence of mitigating controls and the value of affected assets.

CVSS scores are used by a variety of organizations, including security vendors, government agencies, and enterprises, to prioritize vulnerability remediation and assess risk. CVSS scores can also be used to compare the severity of vulnerabilities across different products and platforms.

Here is an example of how CVSS scores are used:

\* A security researcher discovers a new vulnerability in a popular web application. The researcher assigns the vulnerability a CVSS score of 9.3, indicating that it is a critical vulnerability.

\* A vulnerability management team at a large enterprise uses the CVSS score to prioritize remediation of the vulnerability. The team decides to patch the vulnerability immediately, as it poses a high risk to the organization.

\* A government agency uses the CVSS score to issue a security alert to the public. The alert warns users of the vulnerability and recommends that they apply the patch as soon as possible.

CVSS is a valuable tool for assessing the severity of security vulnerabilities and prioritizing remediation efforts. By using CVSS, organizations can reduce their risk of being exploited by attackers.

### 23. What is the CVE Board?

The CVE Board is comprised of cyber security organizations including security tool vendors, academia, research institutions, government departments and agencies, security experts and end-users of vulnerability information.

The CVE Board provides critical input regarding data sources, product coverage, coverage goals, operating structure and strategic direction of the CVE program.

All CVE Board discussions can be found via their email discussion archives and meeting archives. The CVE Board Charter is also publicly accessible.

### 24. What is information gathering and types of information gathering

Information gathering is the process of collecting data and facts from various sources to answer a question, solve a problem, or make a decision. It's a crucial skill in many fields, from academic research and journalism to business and everyday life.

#### **\*\*Primary Sources:\*\***

\* **\*\*Direct observation:\*\*** This involves personally observing and recording data firsthand. For example, a scientist might observe the behavior of animals in their natural habitat, or a journalist might interview people at a protest.

\* **\*\*Experiments:\*\*** These are controlled tests designed to gather data under specific conditions. For example, a psychologist might conduct an experiment to study the effects of stress on memory.

\* **\*\*Surveys:\*\*** These involve collecting data from a group of people, usually through questionnaires. For example, a marketing company might conduct a survey to learn about consumer preferences.

#### **\*\*Secondary Sources:\*\***

\* \*\*Books and articles:\*\* These provide information that has already been gathered and analyzed by others. They can be a good starting point for research, but it's important to be critical of their sources and biases.

\* \*\*Databases:\*\* These are collections of data that can be searched electronically. They can be a valuable resource for finding specific information, but it's important to make sure the database is reliable and up-to-date.

\* \*\*Websites:\*\* The internet is a vast source of information, but it's important to be careful about the quality of the information you find. Not all websites are created equal, and some may contain inaccurate or misleading information.

## 25. Difference between cve and cwe

**\*\*CVE (Common Vulnerabilities and Exposures):\*\***

\* Identifies **\*\*specific instances\*\*** of vulnerabilities in software, hardware, or firmware.

\* Each CVE has a unique identifier, like **\*\*CVE-2023-42477\*\*** for a critical vulnerability in the Apache Log4j logging library.

\* Think of it as a **\*\*fingerprint\*\*** for a specific security flaw.

**\*\*CWE (Common Weakness Enumeration):\*\***

\* Catalogs **\*\*common types of software weaknesses\*\*** that can lead to vulnerabilities.

\* Each CWE has a unique identifier, like **\*\*CWE-79\*\*** for Cross-Site Scripting (XSS).


\* Think of it as a **\*\*category\*\*** that groups similar vulnerabilities.

\* Helps developers and security professionals understand and prevent vulnerabilities before they occur.

- CWE is about the types of vulnerabilities.
- CVE is about specific instances of vulnerabilities.

Here's a table summarizing the key differences:

Feature	CWE	CVE
Focus	Types of vulnerabilities	Specific instances of vulnerabilities
Purpose	Help developers understand and prevent vulnerabilities	Help security professionals track and prioritize vulnerabilities
Identifier	CWE-ID	CVE-ID
Example	CWE-79: Cross-site scripting (XSS)	CVE-2023-22722: Critical XSS vulnerability in popular web browser

 Export to Sheets

## 26. explain privilege escalation and its type

In the world of computers, privilege escalation is a cyberattack technique where an attacker with low-level access tries to gain unauthorized access to higher levels of privileges within a system or network. This could be like a regular user gaining administrator rights on a computer or a low-level employee accessing confidential company data.

There are two main types of privilege escalation:

### **\*\*1. Vertical Privilege Escalation:\*\***

**\* \*\*Aim:\*\*** Climb the castle ladder, reaching the highest level of privileges (e.g., administrator).

**\* \*\*Method:\*\*** Exploiting vulnerabilities in the system or tricking existing users with higher privileges. For example, an attacker might find a bug in a program that allows them to bypass security checks or convince an administrator to run malicious code.

[Image of Vertical Privilege Escalation]

## **\*\*2. Horizontal Privilege Escalation:\*\***

\* **Aim:** Move laterally within the same level, gaining access to other accounts with similar privileges (e.g., other users on the same computer).

\* **Method:** Exploiting weaknesses in shared resources or compromising one account to gain access to others. For example, an attacker might steal a user's login credentials or exploit a vulnerability in a file server to access other users' files.

## 27. define : 1. vulnerability 2. threat 3. attack 4. Shellcode

### **\*\*1. Vulnerability\*\***

\* **Definition:** A weakness or flaw in a system, software, or process that can be exploited by a threat actor to gain unauthorized access, disrupt operations, or steal data.

### **\*\*2. Threat\*\***

\* **Definition:** A potential danger to a system or organization, often caused by the exploitation of a vulnerability. Threats can be intentional (e.g., cyberattacks) or unintentional (e.g., natural disasters, human error).

### **\*\*3. Attack\*\***

\* **Definition:** An attempt to exploit a vulnerability to damage, disrupt, or steal data from a system or organization. Attacks can be carried out through various means, such as malware, phishing, or social engineering.

### **\*\*4. Shellcode\*\***

\* **Definition:** A small piece of code, often written in assembly language, that can be executed by an attacker to perform malicious actions on a compromised system. Shellcode is often used to establish a backdoor, escalate privileges, or install malware.

## 28. explain broken authentication and authorization

### **\*\*Broken Authentication:\*\***

- **What it is:** Weaknesses in authentication mechanisms that allow attackers to compromise user accounts and impersonate legitimate users.

- **Common causes:**

- Weak or easily guessed passwords
- Reusing passwords across multiple sites
- Failure to implement multi-factor authentication
- Poor password storage practices (e.g., storing passwords in plain text)
- Vulnerabilities in password reset mechanisms
- **Example attacks:**
  - Brute force attacks
  - Credential stuffing
  - Password spraying
  - Session hijacking
  - Cross-site request forgery (CSRF)

**Broken Authorization:**

- **What it is:** Flaws in access control mechanisms that allow attackers to access unauthorized resources or perform unauthorized actions.

- **Common causes:**

- Incorrect access control configuration
- Insecure direct object references (IDOR)
- Failure to enforce authorization checks properly
- Bugs in authorization logic

- **Example attacks:**

- Path traversal attacks (e.g., accessing files outside the web root)
- Forced browsing attacks (e.g., accessing hidden pages or resources)
- Privilege escalation attacks (e.g., gaining administrator privileges)

### **\*\*Key Differences:\*\***

- **\*\*Authentication\*\*** focuses on verifying a user's identity.
- **\*\*Authorization\*\*** focuses on controlling what a user can do within a system.

### **\*\*Consequences:\*\***

- **\*\*Data breaches:\*\*** Attackers can steal sensitive data like personal information or financial records.
- **\*\*Financial losses:\*\*** Businesses can incur costs due to fraud, system downtime, and reputational damage.
- **\*\*Regulatory fines:\*\*** Organizations can face penalties for non-compliance with data privacy laws.

## 29. explain nmap and it's scanning techniques

Nmap, short for Network Mapper, is a powerful open-source tool used for network discovery and security audits. It's popular among network administrators and security professionals for its comprehensive scans and extensive features. Let's dive deeper into Nmap and its scanning techniques:

### **\*\*What Nmap does:\*\***

- \* **\*\*Discovers hosts:\*\*** Identifies devices connected to a network, including PCs, servers, and network appliances.
- \* **\*\*Detects open ports:\*\*** Scans for open ports on discovered hosts, revealing potential services running on those ports.
- \* **\*\*Identifies operating systems and services:\*\*** Fingerprints operating systems and services running on open ports, providing useful information about the target system.
- \* **\*\*Performs vulnerability scans:\*\*** Identifies potential vulnerabilities in detected services and software versions.

### **\*\*Scanning Techniques:\*\***

Nmap offers a variety of scanning techniques, each with its own advantages and disadvantages. Here are some common ones:



\* \*\*SYN Scan (default):\*\* Sends a SYN packet to each port, mimicking a normal TCP connection establishment. If the port is open, the target returns a SYN-ACK packet, confirming its openness. This is a fast and reliable technique, but it can trigger IDS/IPS alerts.

\* \*\*UDP Scan:\*\* Sends UDP packets to different ports and analyzes the responses. Useful for detecting open UDP ports, but harder to determine actual services running.

\* \*\*Connect Scan:\*\* Completes a full TCP connection for each port to determine its state. Slower than SYN scan, but stealthier as it doesn't trigger most IDS/IPS alarms.

\* \*\*Xmas Scan:\*\* Sends a specific combination of flags in the TCP packet, helping to identify certain operating systems based on their response.

\* \*\*Script Scanning:\*\* Utilizes Nmap NSE (Nmap Scripting Engine) scripts to perform advanced scans for specific vulnerabilities or services.

### 30. Explain using components with known vulnerabilities with example

**\*\*What it means:\*\***

- Incorporating software components (libraries, frameworks, plugins) into your application or system that have known security flaws.
- These flaws are publicly documented, often with assigned CVE (Common Vulnerabilities and Exposures) identifiers.
- Attackers actively seek out systems using vulnerable components to exploit them.

**\*\*Why it's dangerous:\*\***

- **\*\*Easy targets:\*\*** Attackers can leverage known exploits to break into systems, steal data, launch attacks, or cause damage.
- **\*\*Widespread impact:\*\*** A single vulnerable component can expose the entire system and potentially other connected systems.
- **\*\*Hard to detect:\*\*** You might not realize you're using vulnerable components until it's too late.

## **\*\*Real-world examples:\*\***

1. **\*\*Equifax breach (2017):\*\*** Attackers exploited a known vulnerability in Apache Struts, a widely used web application framework, to steal sensitive data of over 147 million people.
2. **\*\*Panama Papers leak (2016):\*\*** Attackers used a vulnerable version of the Drupal CMS plugin to expose confidential documents from a Panamanian law firm.
3. **\*\*British Airways breach (2018):\*\*** Attackers injected malicious code into a vulnerable third-party script on the company's website, redirecting customers to a phony domain to steal payment card data.
4. **\*\*Mossack Fonseca breach (2015):\*\*** Attackers exploited a vulnerability in WordPress plugins to expose sensitive documents from a Panamanian law firm.
5. **\*\*Heartbleed bug (2014):\*\*** A critical flaw in OpenSSL, a popular encryption library, allowed attackers to steal sensitive data from millions of websites and devices.

## **\*\*Prevention and mitigation:\*\***

- **\*\*Stay informed:\*\*** Regularly monitor vulnerability databases (like NVD) and vendor security advisories.
- **\*\*Update components promptly:\*\*** Apply patches and updates as soon as they're available.
- **\*\*Use software composition analysis (SCA) tools:\*\*** Scan your codebase to identify vulnerable components.
- **\*\*Remove unused components:\*\*** Reduce attack surface by eliminating unnecessary components.
- **\*\*Consider alternatives:\*\*** Explore more secure options if a component has a history of vulnerabilities.
- **\*\*Implement defense-in-depth:\*\*** Use firewalls, intrusion detection systems, and other security measures to protect your systems even if a vulnerability is exploited.

### 31. explain tcp header

**\*\*TCP (Transmission Control Protocol)** is a fundamental protocol in computer networking, responsible for establishing and managing connections between devices, ensuring data integrity, and handling errors. **\*\*** The TCP header is attached to each TCP segment (data unit), providing essential information for reliable communication.

**\*\*Key Fields in the TCP Header:\*\***

\* **\*\*Source Port (16 bits):\*\*** Identifies the port number of the sending application, like a web browser or email client.

\* **\*\*Destination Port (16 bits):\*\*** Identifies the port number of the receiving application, ensuring data reaches the intended recipient.

\* **\*\*Sequence Number (32 bits):\*\*** Tracks the order of data segments, ensuring they're reassembled correctly at the destination.

\* **\*\*Acknowledgment Number (32 bits):\*\*** Acknowledges the successful receipt of previous segments, allowing for efficient retransmission of lost segments.

\* **\*\*Data Offset (4 bits):\*\*** Specifies the length of the TCP header in 32-bit words, indicating where the actual data payload begins.

\* **\*\*Reserved (6 bits):\*\*** Set to zero for future use.

\* **\*\*Flags (6 bits):\*\*** Control various aspects of the TCP connection, including:

- SYN (Synchronize): Initiates a connection.
- ACK (Acknowledgment): Acknowledges a segment.
- FIN (Finish): Terminates a connection.
- RST (Reset): Aborts a connection.
- PSH (Push): Requests immediate data delivery to the application.
- URG (Urgent): Indicates urgent data that should be prioritized.

\* **\*\*Window Size (16 bits):\*\*** Specifies the amount of data the receiver can currently accept, helping regulate data flow and prevent congestion.

\* **\*\*Checksum (16 bits):\*\*** Detects errors in the TCP header and data payload, ensuring data integrity.

\* **\*\*Urgent Pointer (16 bits):\*\*** Points to urgent data within the segment if the URG flag is set.

\* **Options (variable length):** Provides additional features or configuration options, if needed.

\* **Padding (variable length):** Ensures the TCP header ends on a 32-bit boundary for efficiency.

#### **TCP Header Size:**

\* The minimum TCP header size is 20 bytes.

\* Optional fields, like Options and Padding, can increase the header size up to 60 bytes.

#### **Importance of TCP Header:**

\* **Reliable Data Delivery:** Ensures all data segments reach their destination in the correct order, even in the presence of network errors or congestion.

\* **Flow Control:** Manages the rate of data transmission to prevent overwhelming the receiver and ensure efficient communication.

\* **Congestion Control:** Adjusts data transmission rates to avoid network congestion and maintain overall network performance.

\* **Error Detection and Handling:** Detects and retransmits lost or corrupted segments, ensuring data integrity.

### 32. explain file inclusion vulnerability

- A security flaw in web applications that allows attackers to trick the application into including and executing malicious files on the server.
- It occurs when user-supplied input is used to specify the path to a file that the application then includes in its execution flow.
- Attackers can leverage this to inject malicious code, steal sensitive data, or even take control of the server.

#### **\*\*Types of File Inclusion:\*\***

##### 1. **\*\*Local File Inclusion (LFI):\*\***

- Allows attackers to read and execute files located on the server itself.
- Attacker might try to access sensitive files like configuration files, password lists, or source code.

##### 2. **\*\*Remote File Inclusion (RFI):\*\***

- More dangerous as it allows attackers to include and execute files from external servers they control.
- Attackers can execute malicious code, install backdoors, or exploit other vulnerabilities on the server.

#### **\*\*How Attackers Exploit File Inclusion:\*\***

1. **\*\*Identify Vulnerable Application:\*\*** Attackers scan for applications with file inclusion flaws using automated tools or manual techniques.
2. **\*\*Craft Malicious Input:\*\*** They carefully construct a request that includes a path to a malicious file they want to execute.
3. **\*\*Manipulate Application:\*\*** They inject this input into vulnerable parts of the application, such as form fields or URL parameters.
4. **\*\*Application Includes Malicious File:\*\*** If successful, the application includes and executes the malicious file as part of its normal processing.
5. **\*\*Attacker Gains Control:\*\*** The malicious code executes, granting the attacker access to the server, sensitive data, or the ability to perform unauthorized actions.

#### **\*\*Prevention and Mitigation:\*\***

- **Validate Input Rigorously:** Sanitize and validate all user input to ensure it doesn't contain malicious file paths or characters.
- **Use Whitelisting:** Restrict file inclusion to a specific list of allowed files or directories.
- **Disable Dynamic File Includes:** If possible, avoid using dynamic file inclusion features altogether.
- **Keep Software Updated:** Regularly apply patches and updates to address known vulnerabilities in web applications and frameworks.
- **Implement Security Measures:** Use firewalls, intrusion detection systems, and other security tools to detect and block attacks.

### 33. what is cms and why cms security is important

A **Content Management System (CMS)** is a software platform that allows users to create, manage, and publish content on the web without needing extensive technical knowledge. Think of it as a user-friendly interface for building and maintaining websites or web applications. Popular CMS examples include WordPress, Drupal, Joomla!, and Magento.

**CMS Security** is crucial because, as these platforms become more and more popular, they also become attractive targets for cyberattacks. Hackers see CMS vulnerabilities as entry points to potentially access sensitive data, disrupt operations, or even spread malware.

Here's why CMS security is so important:

**1. Vulnerability Surface:** CMS platforms, with their plugins, themes, and extensions, create a larger attack surface compared to static websites. Attackers have more entry points to exploit vulnerabilities.

**2. Sensitive Data Storage:** CMS often stores sensitive user information, financial data, and website content. Breaches can lead to serious privacy violations and financial losses.

**3. Widespread Usage:** The popularity of CMS makes them attractive targets for mass attacks, impacting numerous websites simultaneously.

**\*\*4. Evolving Threats:\*\*** Attackers constantly develop new techniques to exploit vulnerabilities, requiring continuous security updates and vigilance.

**\*\*5. Potential Reputational Damage:\*\*** Security breaches can severely damage a website's reputation and business, leading to loss of trust and customers.

Here are some ways to ensure CMS security:

**\* \*\*Regularly update your CMS, plugins, and themes:\*\*** Apply security patches as soon as they become available to address known vulnerabilities.

**\* \*\*Use strong passwords and multi-factor authentication:\*\*** Implement secure password policies and enable multi-factor authentication for administrator accounts.

**\* \*\*Choose secure plugins and themes:\*\*** Only install plugins and themes from reputable sources and keep them updated.

**\* \*\*Minimize third-party integrations:\*\*** Limit the number of third-party extensions and scripts you use, as they can introduce additional vulnerabilities.

**\* \*\*Regularly back up your website:\*\*** Maintain regular backups of your website content and database to minimize damage in case of a breach.

**\* \*\*Implement security protocols:\*\*** Use firewalls, intrusion detection systems, and web application firewalls to protect your website from attacks.

**\* \*\*Train your staff:\*\*** Educate your team about cybersecurity best practices to minimize human error and phishing risks.

34. describe terminology: 1. STRIDE 2. DREAD

## 1. STRIDE:

STRIDE is a mnemonic used in threat modeling to identify and categorize potential security threats to a system. Each letter stands for a specific type of threat:

- **Spoofing:** Impersonating another user or system to gain unauthorized access.
- **Tampering:** Modifying data or processes without authorization.
- **Repudiation:** Denying involvement in an action or transaction.
- **Information Disclosure:** Unauthorized access to confidential information.
- **Denial-of-Service:** Preventing legitimate users from accessing resources.
- **Elevation of Privilege:** Gaining higher levels of access within a system.
- **Destruction:** Maliciously deleting or corrupting data or resources.

By considering these categories, STRIDE helps security professionals identify and prioritize potential threats, allowing them to focus on mitigating the most critical risks.

## 2. DREAD:

DREAD is another mnemonic used in risk assessment to evaluate the severity and likelihood of a identified threat. Each letter stands for a specific factor:

- **Damage:** The potential impact of the threat occurring, measured in terms of financial loss, reputation damage, data loss, etc.
- **Reproducibility:** The ease with which the threat can be exploited.
- **Exploitability:** The technical skills and resources required to carry out the threat.
- **Affected Users:** The number of users or systems potentially impacted by the threat.
- **Discoverability:** The ease with which the threat can be detected by security measures.

By considering these factors, DREAD helps security professionals prioritize vulnerabilities and allocate resources for mitigation efforts.

By combining STRIDE and DREAD, security professionals can gain a comprehensive understanding of potential threats, their severity, and their likelihood of success. This information helps them make informed decisions about how to best protect their systems and data.

### 35. how tcp/udp protocol works

#### TCP (Transmission Control Protocol):

- **Connection-Oriented:** Establishes a reliable connection between two devices before data transmission.
- **Guarantees Data Delivery:** Ensures all data is delivered in the correct order, without errors or loss.
- **Suitable for:**
  - Web browsing
  - Email
  - File transfers
  - Remote access
- 

#### Key Features:

- **Handshaking:** 3-way handshake (SYN, SYN-ACK, ACK) to initiate and establish a connection.
- **Sequence Numbers:** Tracks each data segment to ensure correct ordering.
- **Acknowledgments:** Receiver sends ACKs to confirm successful receipt of segments.
- **Retransmission:** Sender retransmits lost or corrupted segments.
- **Flow Control:** Manages data transmission rates to prevent overwhelming the receiver.
- **Congestion Control:** Adjusts transmission rates to avoid network congestion.



## UDP (User Datagram Protocol):

- **Connectionless:** Sends data without establishing a connection beforehand.
- **Faster but Unreliable:** Doesn't guarantee delivery or order of data packets.
- **Suitable for:**
  - Real-time applications (streaming, gaming, VoIP)
  - DNS lookups
  - Network management tasks
- 

## Key Features:

- **No Handshaking:** No connection establishment, data is sent immediately.
- **No Retransmission:** Lost or corrupted packets are not automatically resent.
- **Lower Overhead:** Less processing overhead, making it faster than TCP.

## Key Differences:

- **Reliability:** TCP guarantees delivery, UDP doesn't.
- **Speed:** UDP is generally faster due to less overhead.
- **Ordering:** TCP ensures data arrives in order, UDP doesn't.
- **Congestion Control:** TCP has built-in congestion control, UDP doesn't.

## Choosing the Right Protocol:

- **TCP:** When reliable data delivery is crucial (e.g., file transfer, web browsing).
- **UDP:** When speed is more important than reliability, and some data loss is acceptable (e.g., live streaming, gaming).

## 36. how to find virtual hosts in web application

### \*\*1. Analyzing DNS Records:\*\*

- **\*\*Check Public DNS:\*\*** Use online tools like `dig` or `nslookup` to query public DNS records for the target domain. Look for additional A records or CNAME records that point to different IP addresses, hinting at virtual hosts.

### \*\*2. Inspecting Server Configuration Files:\*\*

- **\*\*Access Server Files:\*\*** If you have access to the web server's configuration files (e.g., Apache's `httpd.conf` or Nginx's `nginx.conf`), search for virtual host directives. These typically contain the domain name and document root for each virtual host.

### **\*\*3. Using Network Scanning Tools:\*\***

- **\*\*Nmap:\*\*** Run a scan with the `--script=http-vhosts` script to identify virtual hosts based on HTTP headers and responses.

### **\*\*4. Examining HTTP Headers:\*\***

- **\*\*Browser Developer Tools:\*\*** Inspect HTTP headers in your browser's developer tools (Network tab) for the `Host` header, which typically reveals the virtual host name.

### **\*\*5. Brute Force Guessing:\*\***

- **\*\*Wordlist:\*\*** If other methods fail, use a tool like `Gobuster` or `Dirb` to brute force potential virtual host names based on a wordlist.

## 37. what is log monitoring

Log monitoring is the process of **continuously analyzing log files generated by systems, applications, and devices**. It's like having a detective on your team, constantly sifting through log entries to uncover potential issues, track activity, and ensure smooth operation.

Think of it as a **treasure trove of information**: Every time your system logs an event, like a user logging in, an application error occurring, or a security alert being triggered, it's like leaving a clue in the log file. Log monitoring helps you identify and interpret these clues, uncovering valuable insights about your system's health and performance.

### **Here's why log monitoring is crucial:**

- **Early Detection of Issues:** It helps you identify and troubleshoot problems early on, before they escalate into bigger issues. Think of it like catching a small fire before it turns into a full-blown inferno.
- **Security Monitoring:** It keeps an eye out for suspicious activity and potential security threats, helping you stay ahead of attackers and protect your system. Imagine having a security guard constantly scanning your system for intruders.
- **Performance Insights:** It provides valuable data about your system's performance, allowing you to optimize resource usage and identify bottlenecks. It's like having a performance coach analyzing your system's workout routine to improve its efficiency.
- **Compliance Requirements:** Many regulations require organizations to monitor and log specific activities, and log monitoring helps you meet these compliance requirements. Think of it like having a recordkeeper documenting all your system's actions to ensure legal and regulatory compliance.

### **How does log monitoring work?**

1. **Log Collection:** Logs are collected from various sources, like system files, application logs, and network devices.
2. **Parsing and Aggregation:** Logs are parsed into a structured format and aggregated from different sources into a central repository.
3. **Analysis and Alerting:** Logs are analyzed using tools and rules to identify anomalies, errors, and security threats. Alerts are triggered when specific criteria are met.
4. **Visualization and Reporting:** Logs are visualized on dashboards and reports to provide insights into system health, performance, and activity.

**Here are some of the tools used for log monitoring:**

- **Open-source:** ELK Stack (Elasticsearch, Logstash, Kibana), Graylog, Prometheus
- **Commercial:** Splunk, Sumo Logic, Datadog

38. what is false positive

A **false positive** is an error in a test or classification system where something is identified as positive when it's actually negative. In simpler terms, it's like an alarm going off when there's no actual fire.

Think of it like this:

- **True positive:** You have a fever and the test correctly identifies it (alarm correctly goes off).
- **True negative:** You don't have a fever and the test correctly identifies it (alarm stays silent).
- **False positive:** You don't have a fever but the test mistakenly says you do (false alarm).
- **False negative:** You have a fever but the test mistakenly says you don't (alarm doesn't go off when there's fire).

**Why are false positives a problem?**

- **They waste time and resources:** False alarms can lead to unnecessary investigations, wasted effort, and anxiety.
- **They can damage trust:** If false positives happen too often, people may lose trust in the test or system.
- **They can have serious consequences:** In some cases, false positives can lead to real-world harm, like financial loss or even physical injury.

**Examples of false positives:**

- A security scanner mistakenly identifies a harmless file as malware.
- A spam filter mistakenly flags a legitimate email as spam.
- A medical test incorrectly diagnoses a disease when the patient is actually healthy.

**How to minimize false positives:**

- **Use reliable testing methods:** Choose tests with high accuracy and low false positive rates.

- **Fine-tune the test parameters:** Adjust the sensitivity and specificity of the test to reduce false positives without missing true positives.
- **Consider additional context:** Don't rely solely on the test result, but also consider other factors and information before making a decision.

### 39. explain web authentication mechanisms and fuzzing

#### **\*\*Web Authentication Mechanisms:\*\***

- **\*\*Purpose:\*\*** Verify the identity of users attempting to access web applications or resources, ensuring authorized access and protecting sensitive data.
- **\*\*Common Methods:\*\***
  - **\*\*Basic Authentication:\*\*** Simplest method, involves sending username and password in plain text (not secure for sensitive data).
  - **\*\*Digest Authentication:\*\*** More secure, uses cryptographic hashes of credentials (avoids plain text transmission).
  - **\*\*Form-Based Authentication:\*\*** Uses HTML forms to collect credentials, often integrated with session management.
  - **\*\*Token-Based Authentication:\*\*** Relies on tokens (like JSON Web Tokens) to represent user identity, often used in APIs and single-sign-on.
  - **\*\*Multi-Factor Authentication:\*\*** Adds extra layers of security, such as one-time passwords or biometric verification.

#### **\*\*Fuzzing:\*\***

- **\*\*Purpose:\*\*** A security testing technique that involves injecting unexpected or malformed data into an application to uncover vulnerabilities like crashes, buffer overflows, or authentication bypasses.
- **\*\*Web Application Fuzzing:\*\*** Focuses on testing web application inputs like URLs, form fields, cookies, and headers.
- **\*\*Authentication Fuzzing:\*\*** Specifically targets authentication mechanisms to try to bypass them or expose weaknesses.
- **\*\*Tools:\*\*** Burp Suite, ZAP, Peach Fuzzer, AFL

## **\*\*Fuzzing and Authentication Security:\*\***

- **\*\*Identifying Vulnerabilities:\*\*** Fuzzing can reveal authentication vulnerabilities like:
  - Weak password policies
  - SQL injection attacks
  - Cross-site scripting (XSS) flaws
  - Session management issues
  - Broken authentication and session management (OWASP Top 10)
- **\*\*Testing for Robustness:\*\*** Fuzzing helps ensure authentication mechanisms can handle unexpected inputs without crashing or compromising security.
- **\*\*Improving Security:\*\*** Developers can use fuzzing results to patch vulnerabilities and strengthen authentication mechanisms.

## **\*\*Best Practices:\*\***

- **\*\*Combine authentication methods:\*\*** Use multiple authentication factors for stronger protection.
- **\*\*Enforce strong password policies:\*\*** Require complex passwords and regular updates.
- **\*\*Implement secure session management:\*\*** Protect session tokens and prevent hijacking.
- **\*\*Regularly test authentication mechanisms:\*\*** Use fuzzing and other security testing techniques to identify and address vulnerabilities.
- **\*\*Stay updated:\*\*** Patch known vulnerabilities promptly.

40. describe leaky APIs and insecure data storage

## **Leaky APIs:**

**Definition:** APIs (Application Programming Interfaces) that unintentionally expose sensitive data or functionality to unauthorized users, putting data and systems at risk.

## **Common Causes:**

- **Overly permissive access controls:** Granting excessive permissions to users or third-party applications.

- **Insufficient input validation:** Failing to validate and sanitize user-provided data, allowing attackers to inject malicious code.
- **Lack of authentication or authorization:** Not requiring proper authentication or authorization for sensitive API endpoints.
- **Poor error handling:** Revealing sensitive information in error messages or stack traces.

#### Consequences:

- **Data breaches:** Exposure of sensitive data, leading to financial loss, reputation damage, and regulatory fines.
- **Unauthorized access:** Attackers gaining control of systems or manipulating data.
- **Abuse of functionality:** Attackers exploiting APIs for unintended purposes, such as spamming or denial-of-service attacks.

#### Insecure Data Storage:

**Definition:** Storing sensitive data without adequate security measures, making it vulnerable to theft or unauthorized access.

#### Visual Analogy:

#### Common Causes:

- **Unencrypted data:** Storing data in plain text, making it readable if compromised.
- **Weak encryption:** Using outdated or poorly implemented encryption algorithms.
- **Vulnerable storage locations:** Using cloud storage without proper security controls or storing data on unpatched or misconfigured servers.
- **Poor access controls:** Failing to restrict access to sensitive data, allowing unauthorized users to view or modify it.

#### Consequences:

- **Data breaches:** Exposure of sensitive data to unauthorized parties.
- **Data theft:** Attackers stealing sensitive data for financial gain or espionage.
- **Identity theft:** Attackers using stolen data to impersonate individuals and commit fraud.

#### Best Practices to Prevent Leaky APIs and Insecure Data Storage:

- **Implement strong authentication and authorization:** Require users and applications to have appropriate permissions to access data and functionality.
- **Validate and sanitize inputs:** Prevent malicious code injection and protect against attacks like SQL injection and cross-site scripting (XSS).
- **Encrypt sensitive data:** Protect data at rest and in transit using strong encryption algorithms.
- **Regularly review and update security measures:** Stay informed about latest threats and vulnerabilities, and promptly apply patches and updates.
- **Conduct security audits and penetration testing:** Identify and address potential weaknesses before attackers can exploit them.

#### 41. How REST(API) works and describe error response (status code)

**\*\*REST (Representational State Transfer) APIs\*\*** are a popular way to build web services that allow applications to communicate and exchange data over the internet. They follow a set of architectural principles that make them flexible, scalable, and easy to use.

##### **\*\*Key Principles of REST APIs:\*\***

- **\*\*Client-Server Architecture:\*\*** Clients (e.g., web browsers, mobile apps) make requests to servers, which process the requests and send responses.
- **\*\*Statelessness:\*\*** Each request is independent, containing all necessary information for the server to process it. The server doesn't need to store any client-specific state between requests.
- **\*\*Cacheability:\*\*** Responses can be cached by clients or intermediaries to improve performance and reduce network traffic.
- **\*\*Layered System:\*\*** Intermediary layers can be added between clients and servers for security, load balancing, or other purposes.
- **\*\*Uniform Interface:\*\*** REST APIs use a consistent set of operations (GET, POST, PUT, DELETE) to interact with resources.

##### **\*\*How REST APIs Work:\*\***

1. **\*\*Client sends a request:\*\*** The client uses an HTTP method (GET, POST, PUT, DELETE) to specify the desired operation and a URI (Uniform Resource Identifier) to identify the resource.
2. **\*\*Server processes the request:\*\*** The server receives the request, identifies the resource, performs the requested operation, and generates a response.
3. **\*\*Server sends a response:\*\*** The server sends an HTTP response back to the client, containing the status code, response headers, and optional response body (e.g., JSON or XML data).

##### **\*\*Error Response Status Codes:\*\***

REST APIs use HTTP status codes to indicate the outcome of a request. Here are some common error status codes:

- **\*\*400 Bad Request:\*\*** The request was malformed or couldn't be understood by the server.

- **\*\*401 Unauthorized:\*\*** The request requires authentication, but the client didn't provide valid credentials.
- **\*\*403 Forbidden:\*\*** The client is authenticated, but lacks permission to access the requested resource.
- **\*\*404 Not Found:\*\*** The requested resource couldn't be found on the server.
- **\*\*500 Internal Server Error:\*\*** An error occurred while processing the request on the server.

42. describe terminology: secure source code review details

**Secure Source Code Review (SSCR)** is a systematic process of examining source code to identify and address potential security vulnerabilities, weaknesses, or coding errors that could be exploited by attackers. It's a crucial step in ensuring software security and protecting sensitive data.

#### **Key Terminology:**

- **Source Code:** The human-readable instructions that make up a software program.
- **Vulnerabilities:** Weaknesses or flaws in the code that could be exploited to compromise security.
- **Threat Modeling:** A process of identifying potential threats and attack vectors to prioritize review efforts.
- **Static Analysis:** Automated tools that analyze code without executing it to detect potential vulnerabilities.
- **Manual Review:** Human experts examining code to identify more complex or context-dependent issues.
- **Reviewers:** Individuals with expertise in secure coding practices and common vulnerabilities.
- **Findings:** Issues identified during the review, categorized by severity and potential impact.
- **Remediation:** The process of fixing or mitigating vulnerabilities to reduce risk.

#### **Process Details:**

1. **Planning:**
  - Define scope and objectives.
  - Select tools and methodologies.
  - Assign reviewers with appropriate expertise.
2. **Threat Modeling:**
  - Identify potential threats and attack surfaces.
  - Prioritize code areas for review.



### 3. **Static Analysis:**

- Employ automated tools to scan code for known vulnerabilities.
- Generate reports for initial analysis.

### 4. **Manual Review:**

- Examine code line-by-line for deeper analysis.
- Focus on areas flagged by static analysis or threat modeling.
- Identify logical flaws, design issues, and potential attack vectors.

### 5. **Reporting Findings:**

- Document vulnerabilities with severity ratings and evidence.
- Provide recommendations for remediation.

### 6. **Remediation:**

- Developers fix or mitigate vulnerabilities according to recommendations.
- Re-review code to ensure fixes are effective.

### 7. **\*\*Tracking and \*\***

- Track progress and ensure remediation completion.
- Conduct regular reviews as code evolves.

## **Benefits of SSCR:**

- Early identification and remediation of vulnerabilities.
- Reduced risk of security breaches.
- Improved code quality and maintainability.
- Compliance with security standards and regulations.

### 43. what is dockers containers security

Docker container security involves protecting your applications and data running within Docker containers from unauthorized access, data breaches, and other security threats. It requires a layered approach encompassing the entire container lifecycle, from build to runtime to orchestration.

**\*\*Base Images.\*\***

\* \*\*Use trusted sources:\*\* Download base images from reliable repositories like Docker Hub or your own private registry.

\* \*\*Minimize dependencies:\*\* Limit unnecessary libraries and applications in your base image to reduce attack surface.

\* \*\*Scan for vulnerabilities:\*\* Regularly scan base images for known vulnerabilities and patch them promptly.

#### \*\*Dockerfile:\*\*

\* \*\*Minimize sensitive data:\*\* Avoid storing secrets or credentials directly in the Dockerfile. Use environment variables or external configuration files instead.

\* \*\*Use secure practices:\*\* Follow secure coding practices within the Dockerfile instructions, such as setting appropriate user permissions and disabling unnecessary services.

#### \*\*Container Runtime:\*\*

\* \*\*Control user privileges:\*\* Run containers with non-root users to reduce potential damage in case of a breach.

\* \*\*Network isolation:\*\* Implement network security policies to restrict container communication and prevent unauthorized access.

\* \*\*Resource limits:\*\* Allocate resources like CPU and memory to control container behavior and prevent resource exhaustion.

#### \*\*Container Orchestration:\*\*

\* \*\*Secure configuration:\*\* Secure your container orchestration platform (like Kubernetes) with access controls, secrets management, and vulnerability scanning.

\* \*\*Monitor for anomalies:\*\* Utilize monitoring tools to detect suspicious container activity and potential security incidents.

\* \*\*Rolling updates:\*\* Implement rolling updates to minimize downtime and rollback failures during container deployments.

#### \*\*Additional Considerations:\*\*

\* \*\*Vulnerability management:\*\* Implement a vulnerability management program to scan containers and images for vulnerabilities and patch them promptly.

\* \*\*Security best practices:\*\* Train developers and IT staff on secure coding practices and general security hygiene.

\* \*\*Incident response:\*\* Develop a plan for responding to security incidents involving Docker containers.

**\*\*By implementing these best practices, you can significantly improve the security of your Docker containers and protect your applications and data from unauthorized access and security threats.\*\***

Remember, Docker container security is an ongoing process, not a one-time event. Continuously monitor your containers, update your configurations and dependencies, and stay informed about the latest security vulnerabilities to maintain a secure containerized environment.

#### 44. sql injection vulnerable code describe

**\*\*SQL Injection (SQLi) is a technique where attackers exploit vulnerabilities in applications that construct SQL queries directly from user input.\*\*** It allows them to inject malicious SQL code into the queries, enabling them to:

- **\*\*Exfiltrate sensitive data:\*\*** Retrieve confidential information from the database, such as user credentials, credit card numbers, or personal details.

- **\*\*Modify database content:\*\*** Alter, delete, or insert new data into the database, potentially corrupting sensitive information or disrupting operations.

- **\*\*Execute arbitrary commands:\*\*** In some cases, take control of the database server or underlying operating system, leading to severe security breaches.

**\*\*Vulnerable Code Example (PHP):\*\***

```
```php
```

```
$username = $_GET['username'];
```

```
$password = $_GET['password'];
```

```
$query = "SELECT * FROM users WHERE username='$username' AND  
password='$password'";
```

```
$result = mysqli_query($conn, $query);
```

```
...
```

#### **\*\*How the Attack Works:\*\***

1. The attacker enters a malicious payload into the username or password field, such as `` OR 1=1 --`.
2. This payload is concatenated into the SQL query, resulting in:

```
```sql
```

```
SELECT * FROM users WHERE username=" OR 1=1 --' AND password='...'`
```

```
...
```

3. The `--` comments out the rest of the original query, effectively bypassing the password check.
4. The `OR 1=1` condition is always true, so the query returns all user records from the table.

#### **\*\*Prevention Best Practices:\*\***

- **\*\*Prepared Statements:\*\*** Use prepared statements with parameterized queries, which separate SQL code from user input, preventing injection attacks.
- **\*\*Input Validation:\*\*** Sanitize and validate all user input to ensure it conforms to expected formats and doesn't contain malicious characters.
- **\*\*Error Handling:\*\*** Avoid revealing sensitive information in error messages, as attackers can use them to gather database details.
- **\*\*Least Privilege:\*\*** Grant database users only the minimum permissions necessary for their tasks.
- **\*\*Regular Updates:\*\*** Keep application frameworks, libraries, and database software up-to-date with security patches.
- **\*\*Penetration Testing:\*\*** Conduct regular penetration testing to identify and address vulnerabilities before attackers can exploit them.