CTMSCS S1 P3: Web Application Security

Unit 1. Introduction to Web Technology and Information Gathering

Prepared By – Dr. Jay B. Teraiya

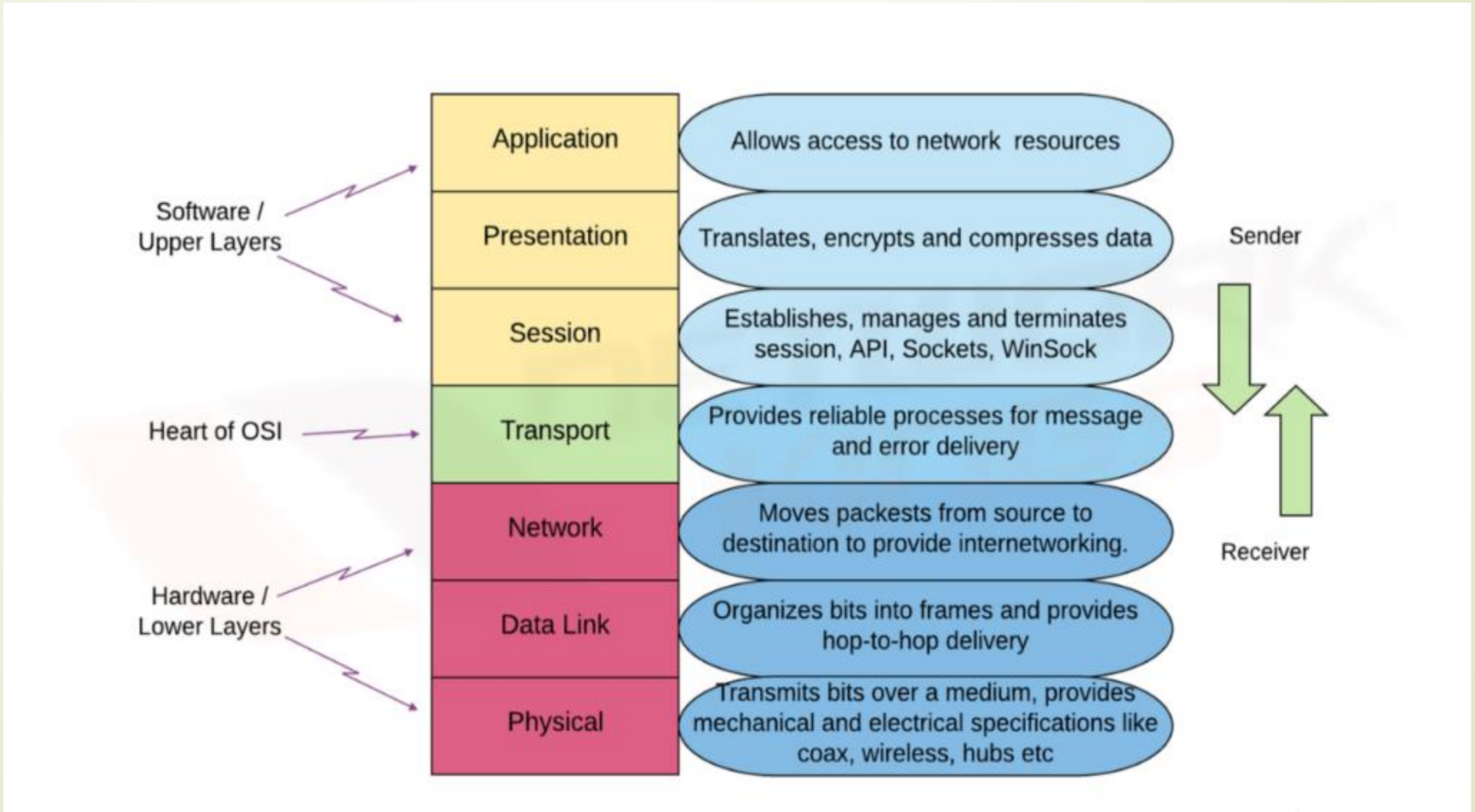
Supported By – Dr. Digvijaysinh Rathod

Agenda

- TCP (Transmission Control Protocol)
- HTTP / S Protocol Basics
- HTTP headers
- Session
- HTTP Cookie
- HTTP Encoding
- Fingerprinting the web server
- Subdomains enumeration
- Finding virtual hosts
- Security Misconfiguration

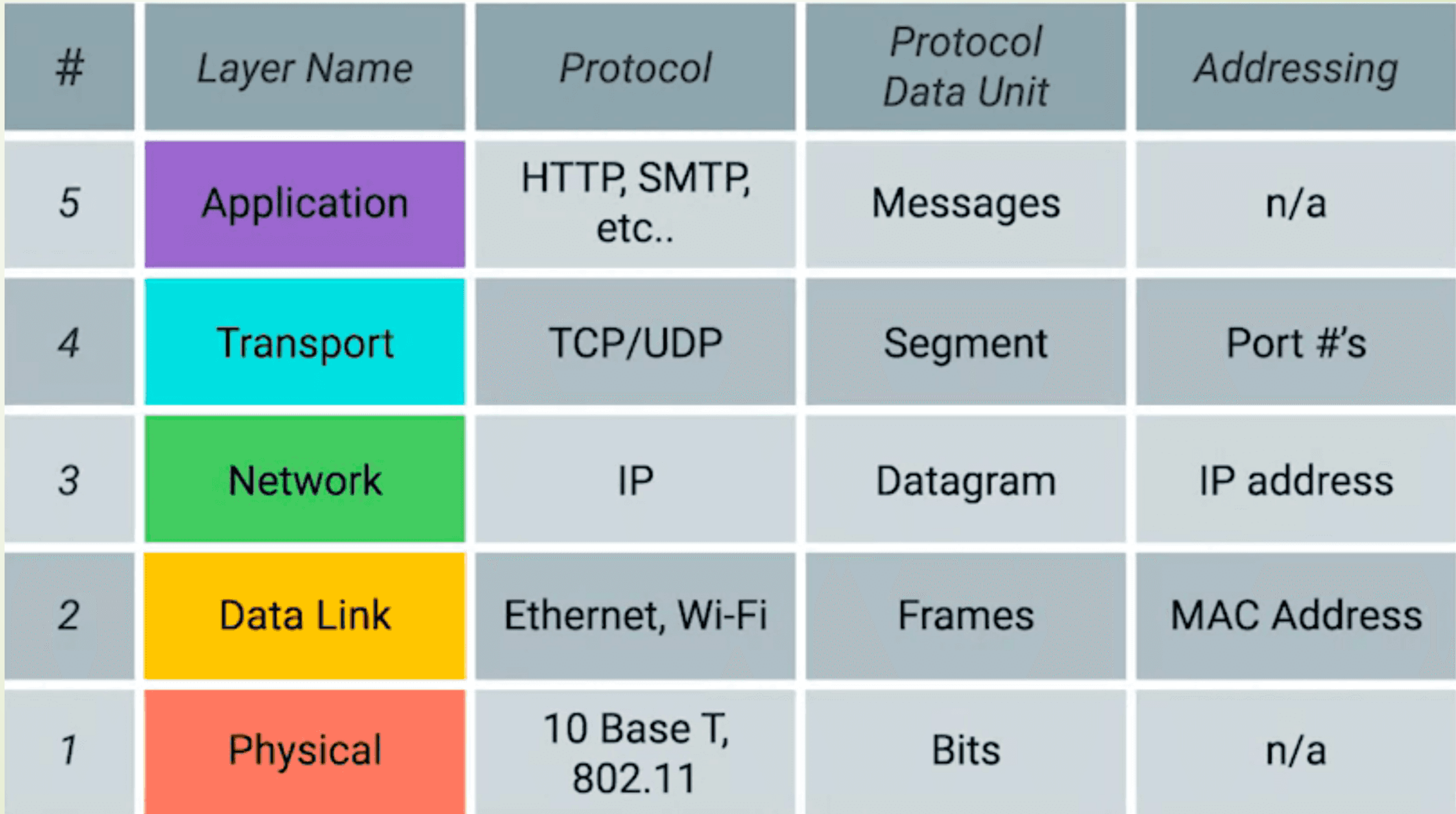
TCP (Transmission Control Protocol)

- OSI Model (Open Systems Interconnection)


Software / Upper Layers
Heart of OSI
Hardware / Lower Layers
Application
Presentation
Session
Transport
Network
Data Link
Physical
Allows access to network resources
Translates, encrypts and compresses data
Establishes, manages and terminates session, API, Sockets, WinSock
Provides reliable processes for message and error delivery
Moves packests from source to destination to provide internetworking.
Organizes bits into frames and provides hop-to-hop delivery
Transmits bits over a medium, provides mechanical and electrical specifications like coax, wireless, hubs etc
Sender
Receiver

TCP (Transmission Control Protocol)

- TCP IP Model



#	Layer Name	Protocol	Protocol Data Unit	Addressing
5	Application	HTTP, SMTP, etc..	Messages	n/a
4	Transport	TCP/UDP	Segment	Port #'s
3	Network	IP	Datagram	IP address
2	Data Link	Ethernet, Wi-Fi	Frames	MAC Address
1	Physical	10 Base T, 802.11	Bits	n/a

TCP (Transmission Control Protocol)

- 3 – Way Handshaking

  - This could also be seen in how a TCP connection is established. Before getting into the details, let us look at some basics.

  - TCP stands for **Transmission Control Protocol**, which indicates that it does something to control the transmission of data reliably.

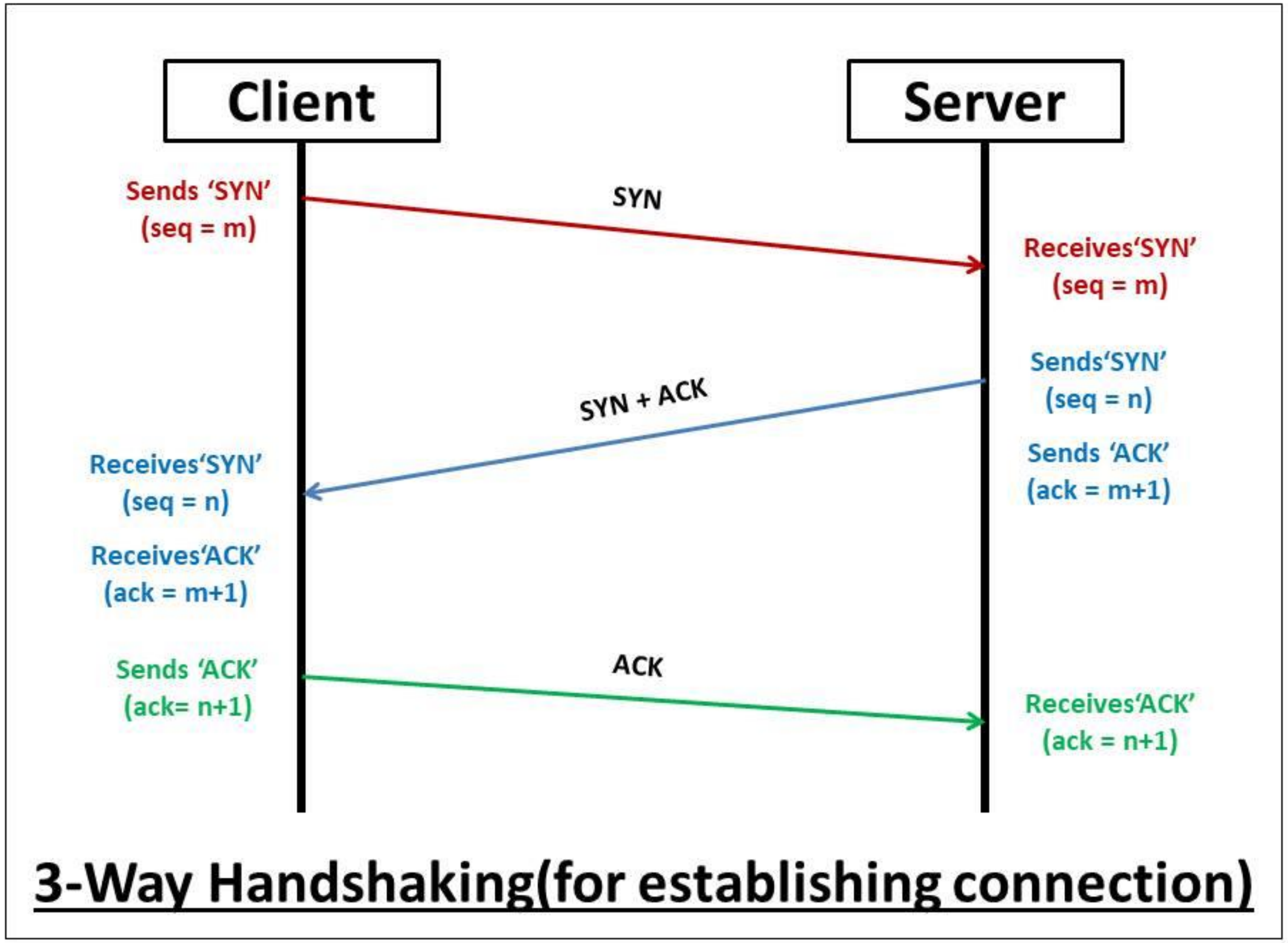
  - The process of communication between devices over the internet happens according to the current TCP/IP suite model(stripped-out version of OSI reference model).

TCP (Transmission Control Protocol)

- 3 – Way Handshaking
  - The Application layer is a top pile of TCP/IP models from which network-referenced applications like web browsers on the client side establish a connection with the server. From the application layer, the information is transferred to the transport layer, where our topic comes into the picture.
  - The two important protocols of this layer are TCP and UDP(User Datagram Protocol), of which TCP is prevalent(since it provides reliability for the connection established).
  - However, you can find an application for UDP to query the DNS server to get the binary equivalent of the domain name used for the website.

TCP (Transmission Control Protocol)

- 3 – Way Handshaking



TCP (Transmission Control Protocol)

- Connection-Oriented Transport:
  - **Step 1 (SYN):** In the first step, the client wants to establish a connection with a server, so it sends a segment with **SYN(Synchronize Sequence Number)**, which informs the server that the client is likely to start communication and with what sequence number it starts segments with.
  - **Step 2 (SYN + ACK):** The server responds to the client request with SYN-ACK signal bits set. Acknowledgment (ACK) signifies the response of the segment it received, and SYN signifies the sequence number with which it is likely to start the segments.
  - **Step 3 (ACK):** In the final part, the client acknowledges the response of the server, and they both establish a reliable connection with which they will start the actual data transfer

TCP (Transmission Control Protocol)

- Reliable Delivery Service:

  - Features of the interface between application programs and the TCP/IP reliable delivery service include:

  - **Stream Orientation -** data is considered as a bit stream divided into bytes.

  - **Buffered Transfer -** transport mechanisms buffer application data until it can fill a reasonably large datagram, using PUSH for immediate transfer.

TCP (Transmission Control Protocol)

- Providing Reliability:

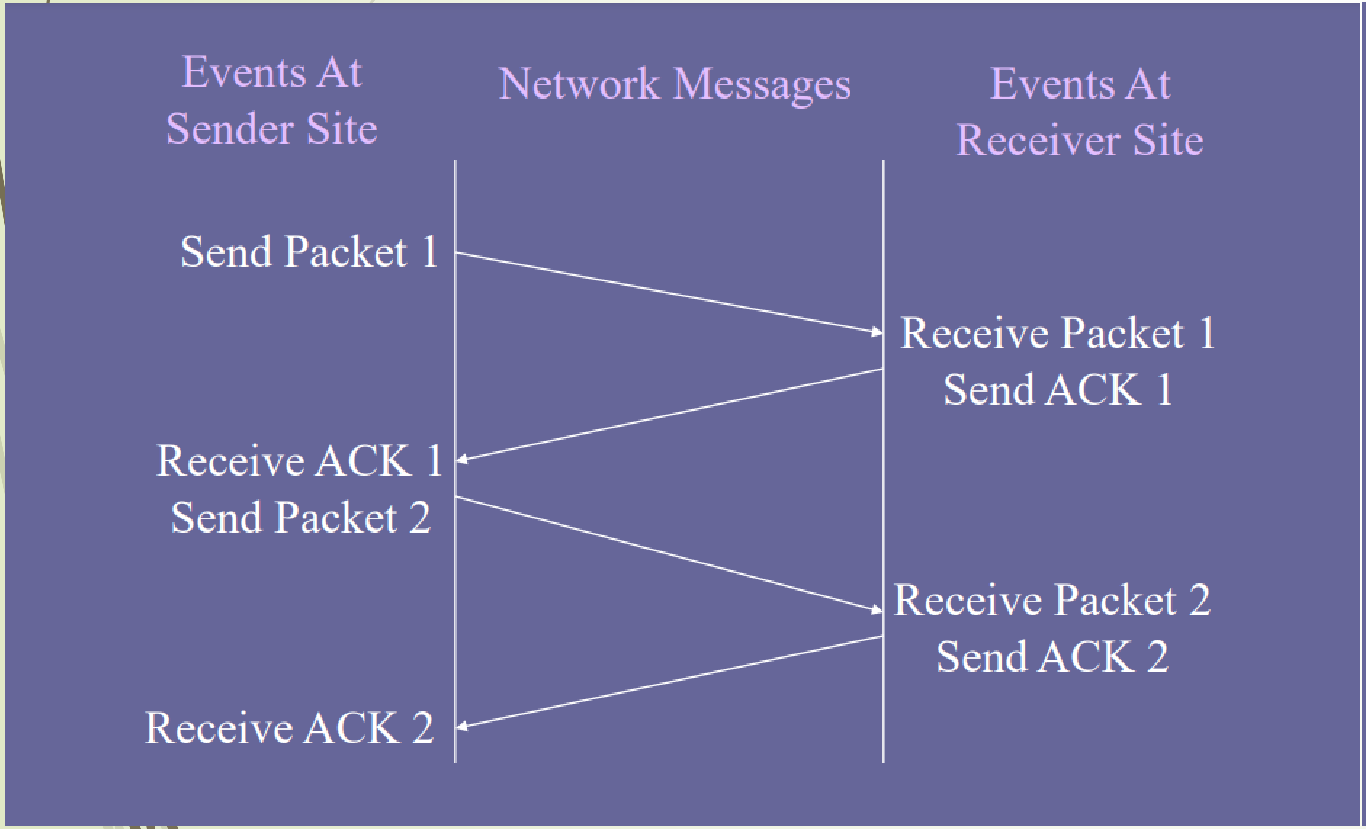
  - Sending positive acknowledgments (ACKs) with retransmission is a fundamental technique for reliable transfer.

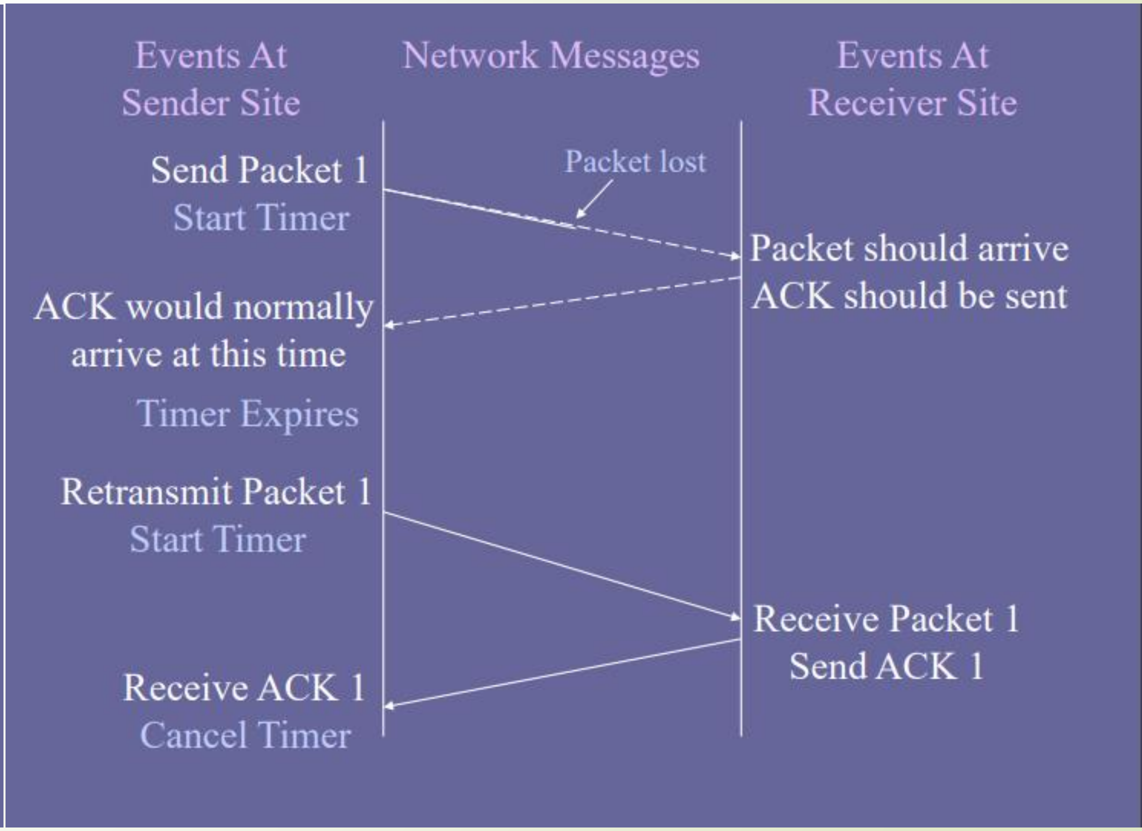
  - **A timer is started** during each transmission, and if it expires, the message is then **retransmitted**.

  - A combination of several timers are used to provide a reliable delivery system.

TCP (Transmission Control Protocol)

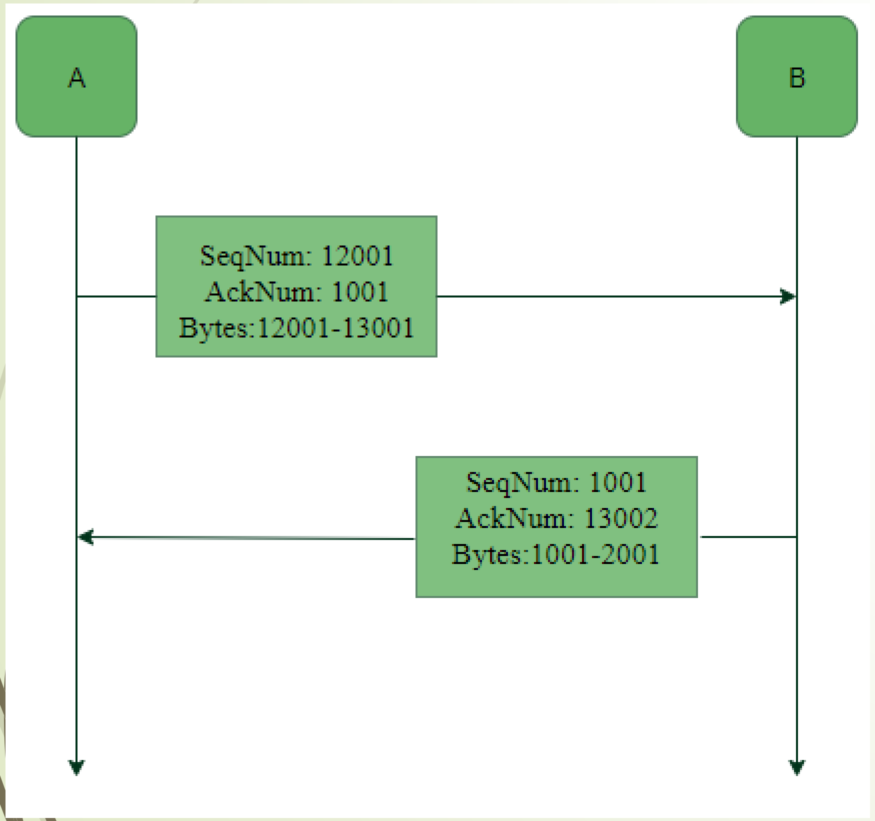
- Providing Reliability:

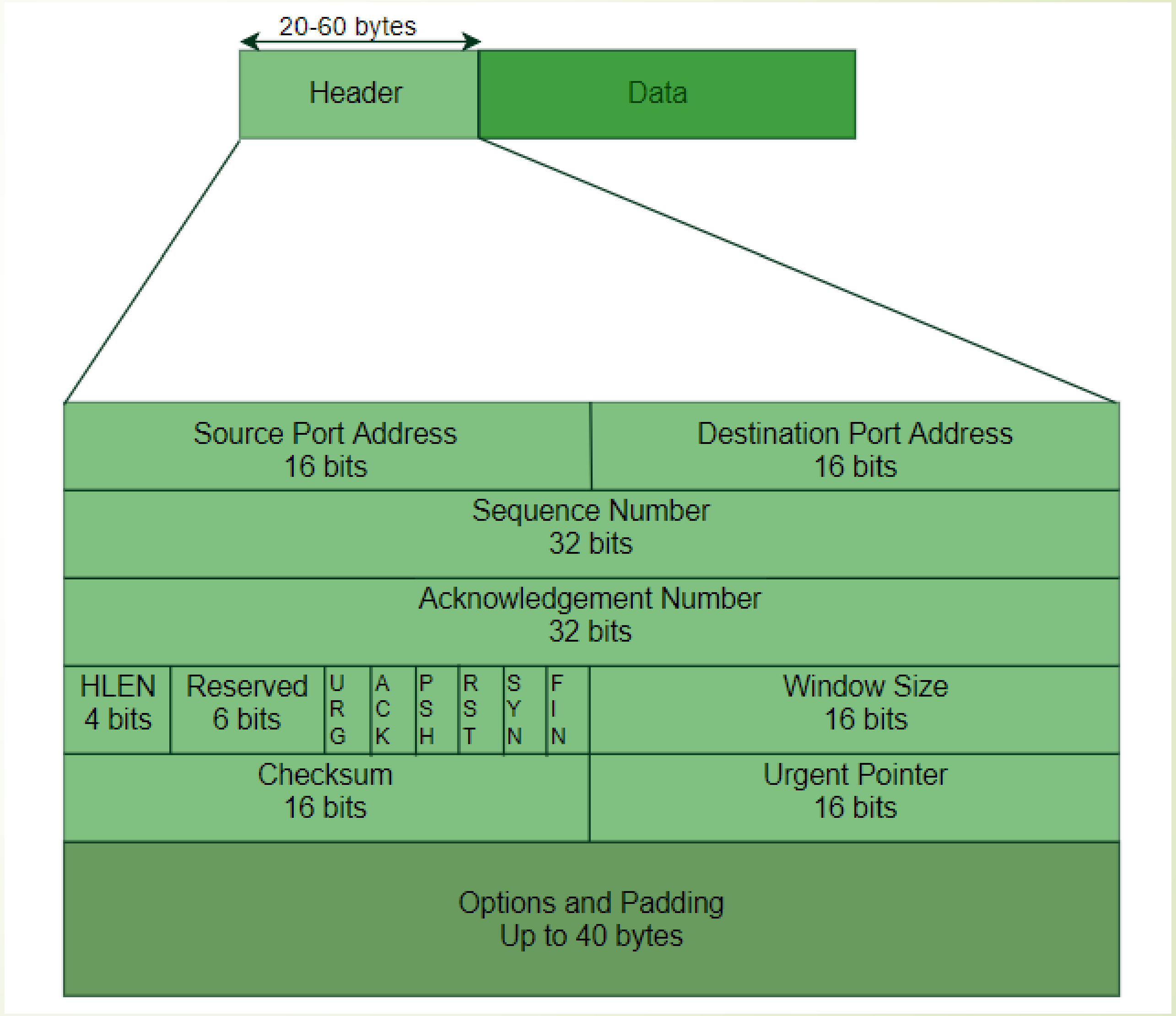




TCP (Transmission Control Protocol)

- Providing Reliability:





TCP (Transmission Control Protocol)

- Providing Reliability:

TCP Segment Header Fields

Code bits - identify the contents of the segment:

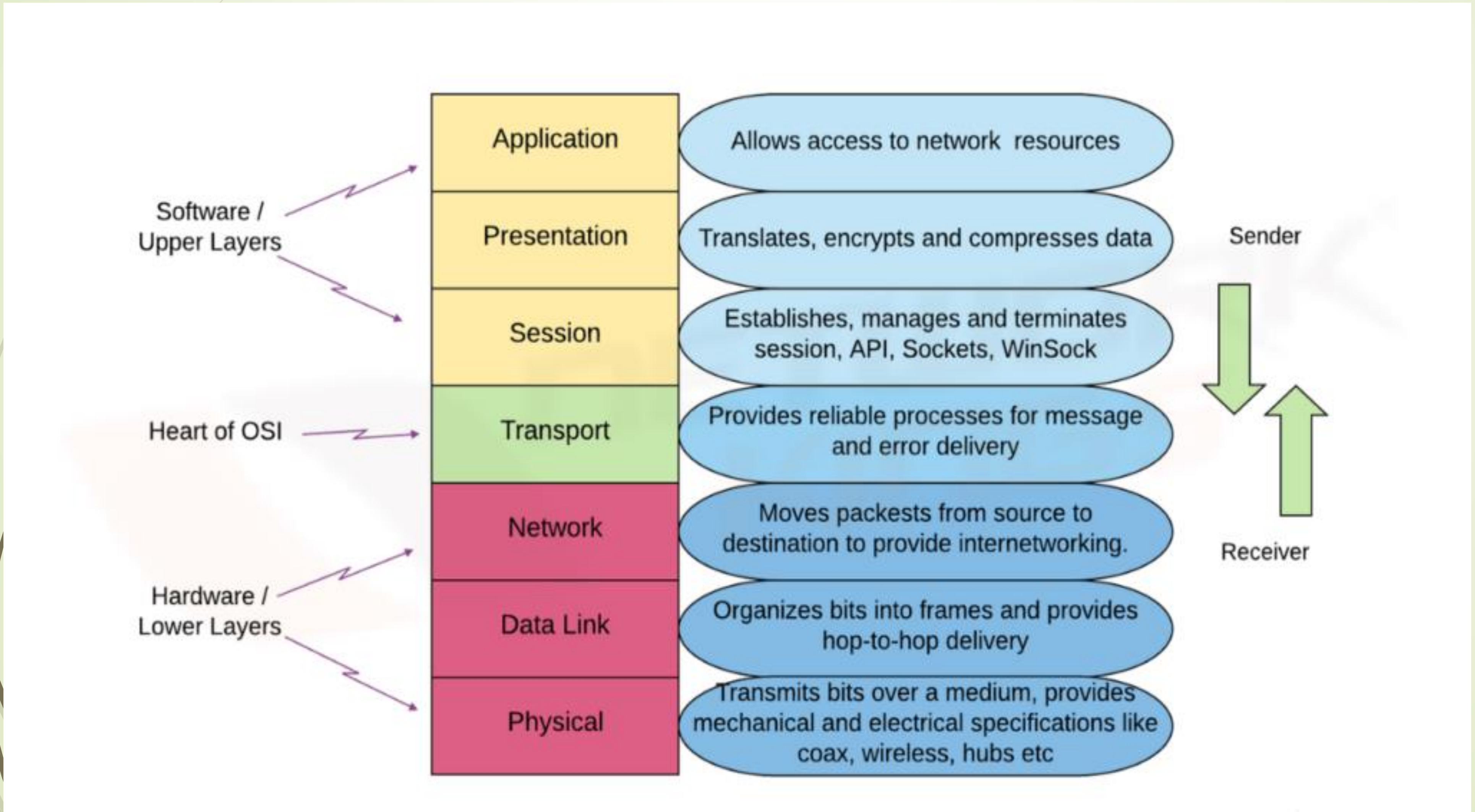
Bit (left to right)	**Meaning if bit is set to 1**
URG	Urgent pointer field is valid
ACK	Acknowledgment field is valid
PSH	This segment requests a push
RST	Reset the connection
SYN	Synchronize sequence numbers
FIN	Sender has reached end of its byte stream

Window - how much data the sender is willing to accept (flow control)

Urgent pointer - specifies the position in the segment where urgent data ends

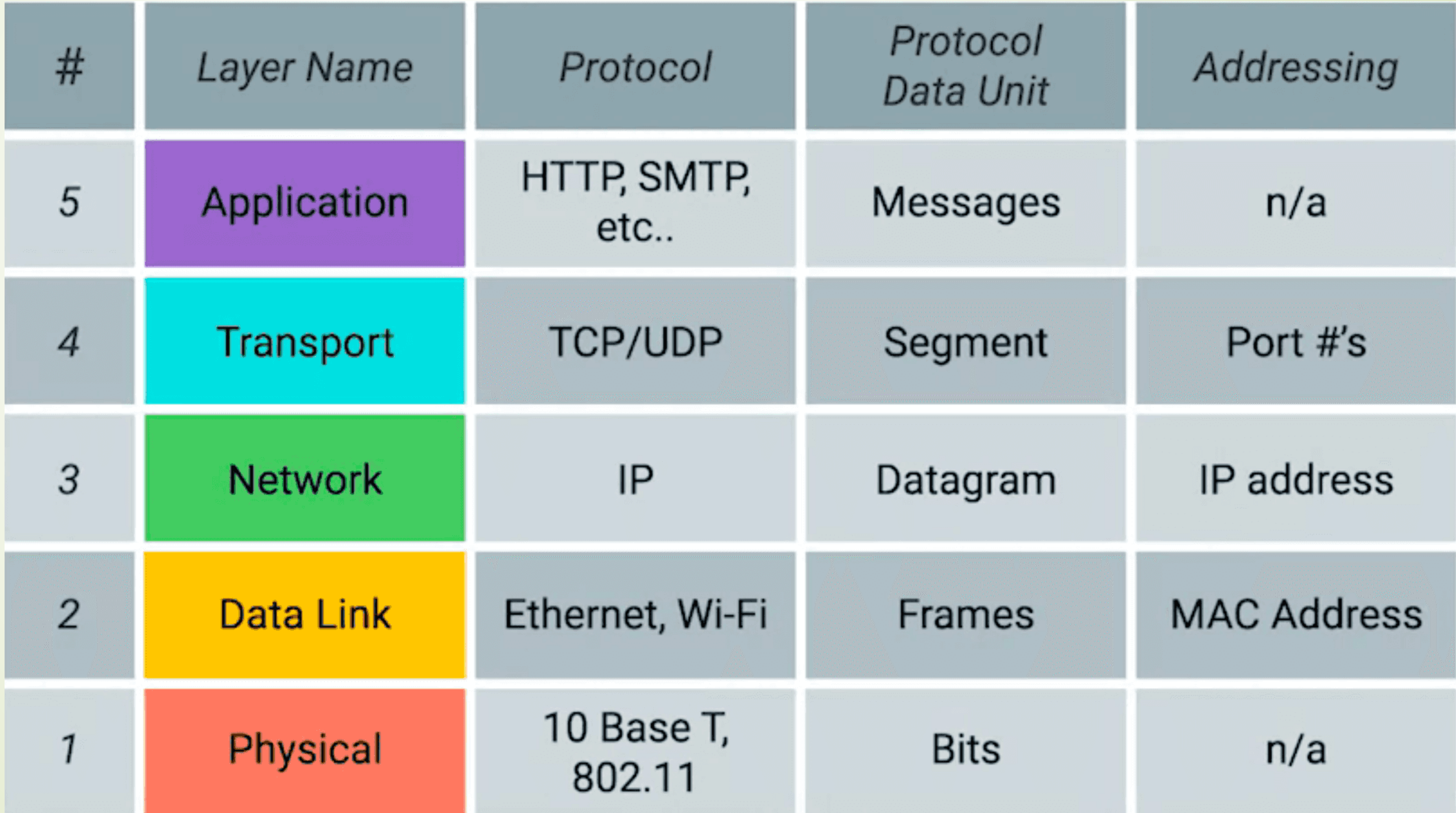
HTTP / S Protocol Basics

- HTTP (Hyper Text Transfer Protocol)


Software / Upper Layers
Heart of OSI
Hardware / Lower Layers
Application
Presentation
Session
Transport
Network
Data Link
Physical
Allows access to network resources
Translates, encrypts and compresses data
Establishes, manages and terminates session, API, Sockets, WinSock
Provides reliable processes for message and error delivery
Moves packests from source to destination to provide internetworking.
Organizes bits into frames and provides hop-to-hop delivery
Transmits bits over a medium, provides mechanical and electrical specifications like coax, wireless, hubs etc
Sender
Receiver

HTTP / S Protocol Basics

- TCP IP Model



#	Layer Name	Protocol	Protocol Data Unit	Addressing
5	Application	HTTP, SMTP, etc..	Messages	n/a
4	Transport	TCP/UDP	Segment	Port #'s
3	Network	IP	Datagram	IP address
2	Data Link	Ethernet, Wi-Fi	Frames	MAC Address
1	Physical	10 Base T, 802.11	Bits	n/a

HTTP / S Protocol Basics

- HTTP (Hyper Text Transfer Protocol)

  - The Hyper Text Transfer Protocol is a **Stateless Protocol**, TCP / IP-based protocol for communicating on the WWW.
  - HTTP defines the precise manner in which Web clients communicate with Web Servers.
  - HTTP Version 1.0 is the most commonly used, but many browsers and web servers now support the HTTP 1.1 version because it is standardized.
  - The HTTP follows a simple Request–Response paradigm (i.e. architecture, standard).

HTTP / S Protocol Basics

- HTTP (Hyper Text Transfer Protocol)

  - A conversation between a Client (Web Browser) and a Server (Web Server) goes something like this:
    1. The client opens a connection (TCP/IP) to the Server.
    2. The client requests the Server.
    3. The server responds to the request.
    4. The connection is closed.

HTTP / S Protocol Basics

- Understanding HTTP Working:

  - At this stage there is not much HTTP-specific activity takes place because TCP/IP connection is established at the Transport Layer of the Protocol Stack.
  - The Client's request of http://www.awl.com/index.html goes in the form

    GET / index.html HTTP/1.0
  - This request can be broken into 3 parts:
    a) the request method i.e. GET,
    b) the source name i.e. /index.html, and
    c) the protocol, i.e., HTTP/1.0.

HTTP / S Protocol Basics

- Understanding HTTP Working:

  - GET is an HTTP method that requests the server to send a file. /index.html is a relative path to the file being requested. HTTP/1.0 is the name and version of the Protocol implemented by the Client.

  - In addition to the GET method or command, the Browser may send other information about itself to the Server.

HTTP / S Protocol Basics

- Understanding HTTP Working:

  - The Server responds to the request with the Status Code, Various Header fields, and, if possible, the contents of the requested file.
    - Protocol: HTTP/1.0 200 OK
    - Server: Netscape-Enterprise/2.01
    - Content-Type: text/html
    - Content-Length: 87
    - HTML Code of the File…
  - The TCP/IP connection is closed either by the Server, Client, or both.

HTTP / S Protocol Basics

- Features of HTTP

  - **Connection-Oriented and Connectionless Protocol:**
  - **HTTP is a Connectionless Protocol.** Due to HTTP's Connectionless nature, very few Server resources are required to service a large number of Clients.
  - At the same time, the connectionless nature also has drawbacks. Those sites that receive requests in millions have strong performance overhead.

  - **Stateless Protocol:**
  - HTTP is a stateless protocol. By definition, a protocol is said to be stateless if it has no memory of prior connections and cannot distinguish one client's request from another.

HTTP / S Protocol Basics

- Features of HTTP

  - In contrast, in **stateful protocols like FTP**, the connection is not opened and closed with every request. For example, after the login, the stateful protocols maintain the user’s credentials throughout the session. Due to its stateless nature, **there is no inherent method in HTTP for tracking a client’s traversal of a site.**

  - The stateless nature of HTTP is both a **strength and a weakness**. Its **strength is** that it keeps the protocol straightforward. It can handle multiple clients because it doesn’t have to remember them all and thus requires fewer resources. **Its weakness** is that some special mechanism is needed to track the user’s traversal.

HTTP / S Protocol Basics

- Status Codes: HTTP

Code Range	Category	Description
1xx	Informational	Only from HTTP 1.1
2xx	Successful	Request is successfully received, understood, and accepted.
3xx	Redirection	Server requesting the Web Client to redirect to another URL.
4xx	Client Error	Request is improperly formatted or cannot be fulfilled.
5xx	Server Error	A valid request was received but the server cannot fulfill it.

HTTP / S Protocol Basics

- GET Method:

  - It is the most common method to request the resource from the server. It contains no body content, a GET request comprises only a method statement and various request header fields.

  - GET / login.html?username=autin&password=servlets HTTP/1.0
  - User–Agent: Mozilla/4.02 [en] (Win95; I)
  - Accept: image/gif, image/jpeg; image/pjpeg, */*

HTTP / S Protocol Basics

- GET Method:

  - A drawback of using a GET method is that (like in login transactions) information entered by the user is appended to the URL and **displayed in plain text** by the browser.
  - Another disadvantage is that **a limited amount** of data can be passed as part of the URL in a GET request.
  - **HTTP Servers store GET parameters in the system environment variables that CGI programs and other out-of-process applications can access. The number of system variables differs from OS to OS.**

HTTP / S Protocol Basics

- POST Method:

  - POST is an HTTP method commonly used to pass user input to the server. The POST method differs from the GET method in that all parameter information is stored in the body of the request rather than in the URL portion of the method statement.
  - This has **2 merits**; first, the information submitted by the user is not visible in the URL. Second, there is no limit to the amount of information that can be passed when stored in the request's body. This is because the name/value pairs passed in a POST request are accessed via the **client's input stream rather than the server's environment variables (like GET parameters).**

HTTP / S Protocol Basics

- POST Method:

  - POST / login.html HTTP/1.0
  - User–Agent: Mozilla/4.02 [en] (Win95; I)
  - Accept: image/gif, image/jpeg; image/pjpeg, */*
  - Content–Length: 34
  - Username=auting&password=servlets

HTTP headers

- **Origin**

  - The Origin request header indicates the origin (scheme, hostname, and port) that caused the request.

  - **The Origin HTTP Header** is a response HTTP header that indicates the security contexts that initiate an HTTP request without indicating the path information.

  - **The browser adds the Origin header, which the user can not control.**

  - **Origin: <scheme> "://" <hostname> ":" <port>**

HTTP headers

- **Origin Directives:**
  - **<scheme>:** This is usually the HTTP or HTTPS protocol used.
  - **<hostname>:** This is the IP or domain name of the server.
  - **<port>:** This is an optional directive that tells the Transmission Control Protocol port number of the server. The default port is implied if it is not specified.
  - **Examples:**
    - Origin: null
      - This means that there is no origin for the service requested.
    - Origin: https://www.nfsu.ac.in

HTTP headers

- **Referer:**

  - The HTTP Referer header is a request-type header **that identifies the address of the previous web page,** which is linked to the current web page or resource being requested.

  - Using this header **increases the risk of privacy** and **security** breaches on a website. Still, it allows **websites and web servers to identify where the traffic is coming from.**

  - The browsers can not send the Referer if the resource is a local file or data.

HTTP headers

- **Referer:**

  - Syntax:

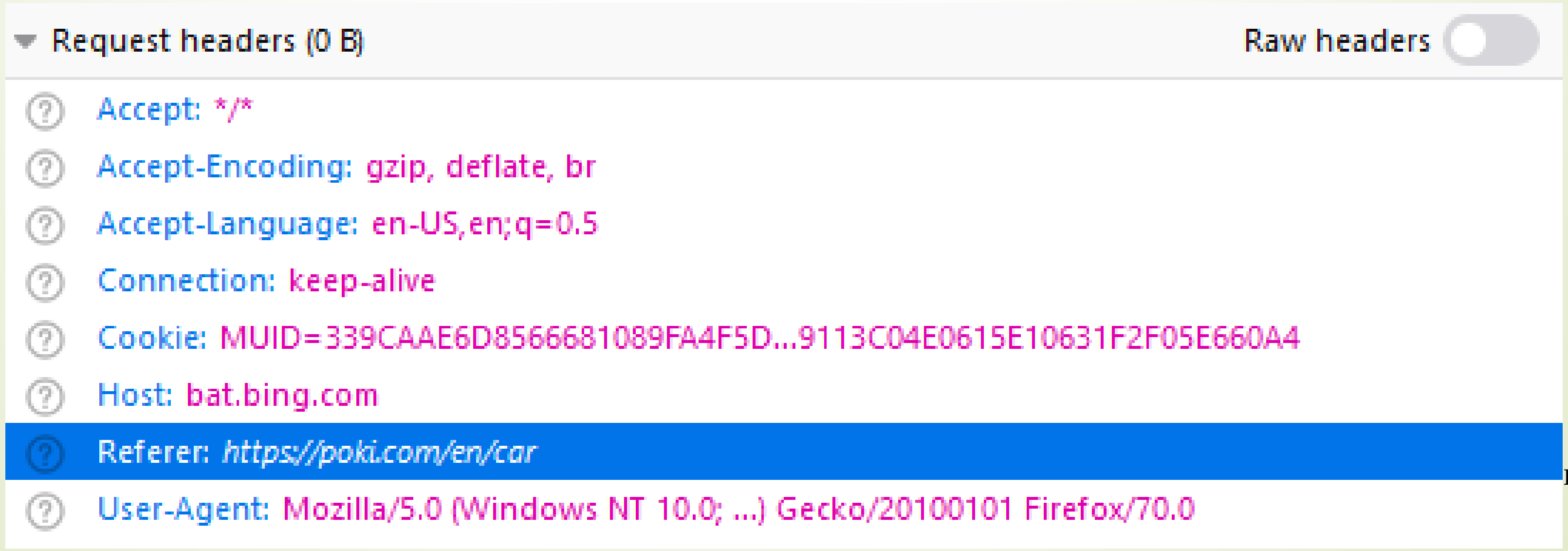
    Referer: <url>

  - **Directives**: The HTTP Referer header accepts a single directive as mentioned above and described below:

  - **<url>:** This directive is the address(partial or full) of the previous World Wide Web page followed by a link to the currently requested page.

HTTP headers

- **Referer:**
  - The below examples illustrate the HTTP Referer header:
  - Examples: In this example, google.com is the address of the previous web page.
  - Referer: https://www.google.com/
  - To check the Referer in action, go to Inspect Element -> Network. Check the request header for Referer like below. The referer header is highlighted.



```
Accept: */*
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.5
Connection: keep-alive
Cookie: MUID=339CAAE6D8566681089FA4F5D...9113C04E0615E10631F2F05E660A4
Host: bat.bing.com
Referer: https://poki.com/en/car
User-Agent: Mozilla/5.0 (Windows NT 10.0; ...) Gecko/20100101 Firefox/70.0
```

HTTP headers

- **Supported Browsers:**
  - The browsers are compatible with HTTP header Referer are listed below:
    - Google Chrome
    - Internet Explorer
    - Microsoft Edge
    - Firefox
    - Opera
    - Safari

HTTP headers

- **Origin Vs Referer**

- The Origin header is similar to the Referer header but does not disclose the path and may be null.

- It provides the "security context" for the origin request, except when the origin information is sensitive or unnecessary.

HTTP headers

- **Host**
- The HTTP Host represents the domain name of the server.
- It may also represent the Transmission Control Protocol (TCP) port number that the server uses.
- Defining the port number is optional, the default value is considered. For example, when no port number is specified, **"80" is assigned as the port number for an HTTP URL**.
- **The HTTP Host header is a request-type header. The host header field must be sent in all HTTP/1.1 request messages.**
- **If a request message has no header field or more than one header field, a 400 Bad.**

HTTP headers

- **Host**
- The request is sent.
- Syntax :
- Host: <host>:<port>
- **Directives**: The HTTP header Host accepts two directives mentioned above and described below:

<host>: This directive represents the domain name of the server.

<port>: This directive is an optional one. It represents the TCP port number in which the server is working.

- Example: http://www.nfsu.ac.in Host: nfsu

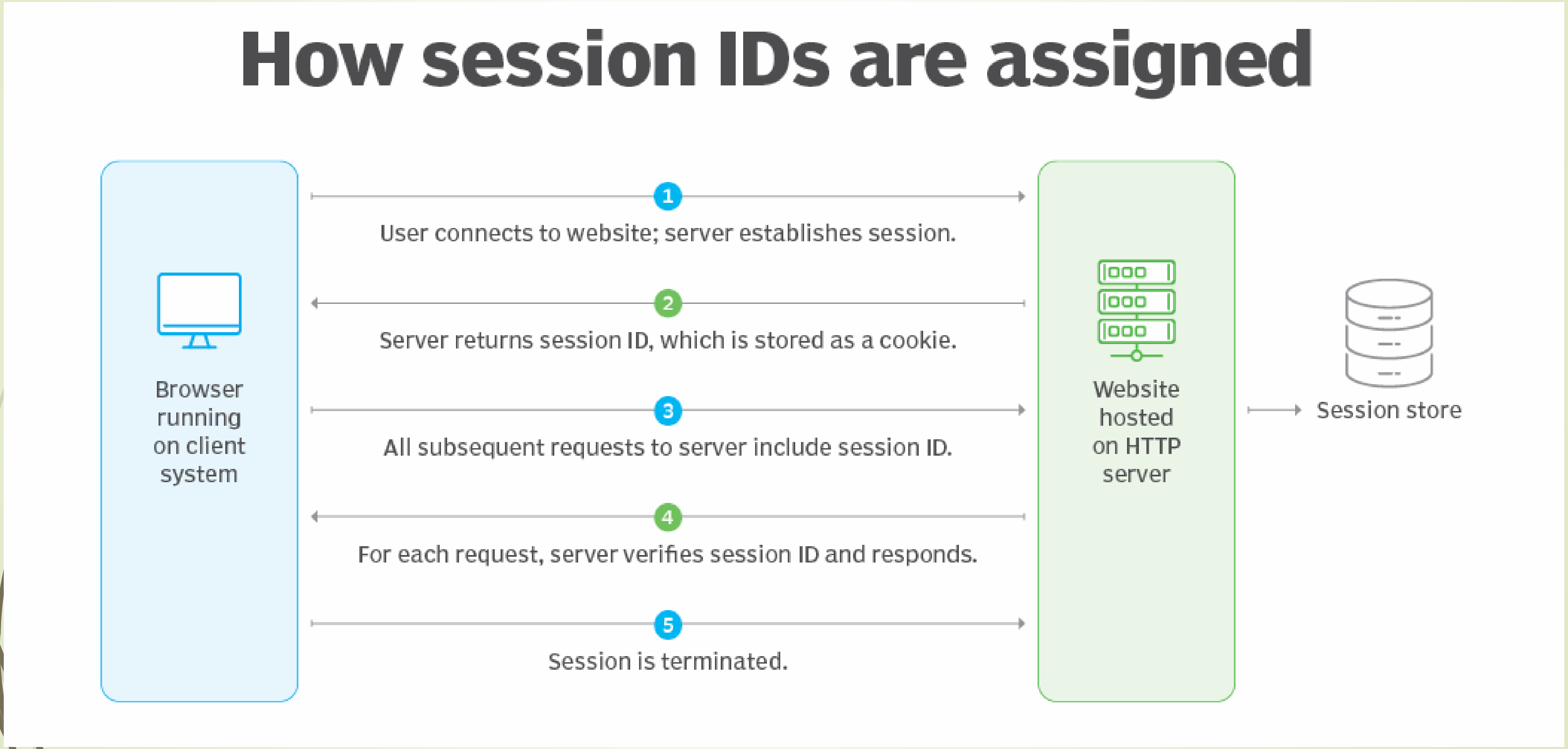
Session

- HTTP is a “stateless” protocol.
- This means there is no “built-in” standard to keep track of interrelated requests. **Each request is treated as independent.**
- These web applications are **very advanced** and usually handle **complex operations** that take **more than one pair of requests/responses to complete**, this requires something to track the present state of operation.
- These apps also present **tailored content** to each user.
- This requires **identifying a user** across multiple requests.
- **So, something must add an “enhancement” to HTTP.**

Session

- Some solutions are the results of **clever thinking by web programmers**.
- HTTP uses a client-server architecture and TCP as its transmission protocol, and multiple requests can be sent over just one TCP connection.
- However, these are also considered independent by both the client and server.

Session


How session IDs are assigned
Browser running on client system
1
User connects to website; server establishes session.
2
Server returns session ID, which is stored as a cookie.
3
All subsequent requests to server include session ID.
4
For each request, server verifies session ID and responds.
5
Session is terminated.
Website hosted on HTTP server
Session store

Session

- **HttpSession**
- It provides a way to identify a user across more than one page, request or visit a website, and store information about that user.
- An HTTP session is a mechanism for maintaining **stateful** communication between **a web server and a client over the stateless HTTP protocol.**
- In a stateless protocol like **HTTP, each request from a client to the server is independent,** meaning the server has no inherent knowledge of previous requests.
- However, **HTTP sessions enable the server to associate multiple requests from the same client as part of a logical session**, allowing for the retention of user-specific data across requests.

Session

- **Key Components of an HTTP Session:**

- **1. Session Identifier:** A unique identifier assigned to each session, typically as a session ID or token. This identifier is used to associate subsequent requests with the correct session data.
- **2. Session Data:** Information specific to a particular session, such as user authentication credentials, preferences, or items in a shopping cart.
- **3. Session Management:** Mechanisms for creating, maintaining, and destroying sessions. This includes techniques for session initialization, expiration, and storage.

Session

- **Implementation of HTTP Sessions:**
- HTTP sessions can be implemented using various techniques, including:
- **1. Cookies:** One of the most common methods, where a unique session identifier is stored in a cookie on the client side. Subsequent requests from the client include this cookie, allowing the server to identify the session.
- **2. URL Rewriting:** Session identifiers are appended to URLs as query parameters, enabling the server to track sessions.
- **3. Hidden Form Fields:** Session identifiers can be embedded within HTML forms as hidden fields, allowing for session tracking during form submissions.
- **4. Server-Side Storage:** Session data can be stored on the server side, either in memory, in a database, or through other persistent storage mechanisms.

Session

- **Security Considerations:**

- While HTTP sessions facilitate stateful communication, they also introduce security considerations:

- **Session Hijacking:** Attackers may attempt to steal session identifiers to impersonate legitimate users. Implementing secure session management techniques, such as **HTTPS and secure cookies**, helps mitigate this risk.

- **Session Expiry:** Properly managing session expiration helps minimize the window of opportunity for attackers to exploit inactive sessions.

Session

- **Security Considerations:**

- **Session Fixation:** Attackers may attempt to fixate a user's session identifier, allowing them to hijack the session once the user authenticates. Techniques like session regeneration after authentication can help prevent this.

- **Cross-Site Scripting (XSS):** XSS vulnerabilities can be exploited to steal session identifiers or manipulate session data. Sanitizing user input and implementing proper output encoding can mitigate this risk.

HTTP Cookie

- HTTP cookies are text-based files web servers create and store locally on a user's device.
- They consist of key-value pairs and are primarily used to store user browsing activity and preferences information.
- When a user visits a website, the server sends one or more cookies along with the HTTP response, which are then stored by the browser.
- Subsequent requests to the same website include these cookies, allowing the server to recognize the user and retrieve relevant information.

HTTP Cookie

- **Uses of HTTP Cookies:**

- **Session Management:** Cookies are commonly employed to maintain the session state between the client and server. A session cookie containing a unique identifier allows the server to associate multiple requests from the same user as part of a session.

- **Personalization:** Websites use cookies to remember user preferences, such as language settings, theme choices, or personalized content recommendations.

HTTP Cookie

- **Uses of HTTP Cookies:**

- **Tracking and Analytics:** Cookies are utilized by advertisers and website owners to track user behavior, monitor site performance, and gather analytics data for targeted advertising and website optimization.

- **Authentication:** Cookies play a vital role in user authentication by storing authentication tokens or session identifiers, allowing users to remain logged in across multiple visits to a website.

HTTP Cookie

- **Security Considerations:**
- **1. Cross-Site Scripting (XSS):** Malicious scripts injected into web pages can access and manipulate cookies, potentially leading to session hijacking or data theft.
- **2. Cross-Site Request Forgery (CSRF):** Attackers may exploit cookies to perform unauthorized actions on behalf of the user by tricking them into making unintended requests.
- **3. Session Hijacking:** Insecure transmission of cookies over unencrypted connections (HTTP) can expose them to interception, allowing attackers to hijack user sessions.
- **4. Privacy Concerns:** Cookies can be used to track users across websites, raising privacy concerns. Regulations like GDPR(General Data Protection Regulation) and CCPA (California Consumer Privacy Act) impose restrictions on cookie usage and require transparent consent mechanisms.

HTTP Encoding

- URL encoding converts characters into a format that can be transmitted over the Internet.
- URLs can only be sent over the Internet using the ASCII character set.
- Since URLs often contain characters outside the ASCII set, the URL has to be converted into a valid ASCII format.
- URL encoding replaces unsafe ASCII characters with a "%" followed by two hexadecimal digits.
- URLs cannot contain spaces. URL encoding normally replaces a space with a plus (+) sign or with %20.

HTTP Encoding

- Complete Tutorial: **https://nooblinux.com/how-to-use-netcat/**

- Netcat is one of the most versatile network tools for system administrators – the Swiss Army knife of Networking.

- This tool can create any connections over TCP or UDP protocol, making it an excellent debugging tool. It helps the user investigate connections directly by connecting to them.

Fingerprinting the web server

- **Web server fingerprinting** identifies the type and version of the web server a target is running on.

- While web server fingerprinting is often encapsulated in **automated testing tools**, researchers need to understand the fundamentals of how these tools attempt to identify software and why this is useful.

- Accurately discovering the type of web server an application runs on can enable security testers to determine **if the application is vulnerable** to attack.

- In particular, servers running **older software versions without up-to-date security patches** can be susceptible to known version-specific exploits.

Fingerprinting the web server

- Techniques used for web server fingerprinting include **banner grabbing, eliciting responses to malformed requests, and using automated tools** to perform more robust scans that use a combination of tactics.

- The fundamental premise on which all these techniques operate is the same.

- They all strive to elicit some **response from the web server**, which can then be compared to a database of known responses and behaviors and thus matched to a known server type.

Fingerprinting the web server

- **Telnet nfsu.ac.in 80**
  - GET HTTP/1.0
  - Telnet is a protocol that allows you to connect to remote computers (called hosts) over a TCP/IP network (such as the Internet). Using telnet client software on your computer, you can connect to a telnet server (that is, the remote host).
- **Netcat** is one of the most versatile networking tools for system administrators –the Swiss army knife of Networking.
- This tool can create any connections over TCP or UDP protocol, making it an excellent debugging tool. It helps the user investigate connections directly by connecting to them.

Fingerprinting the web server

- **HTTPrint**

- httprint is a **web server fingerprinting tool**.
- It relies on web server characteristics to accurately identify web servers, even though they may have been complicated by changing the server banner strings or by plug-ins such as mod_security or servermask.
- httprint can also be used to detect **web-enabled devices** that do not have a server banner string, such as wireless access points, routers, switches, cable modems, etc.
- httprint uses text signature strings, making it very easy to add signatures to the signature database.

Fingerprinting the web server

- **HTTPrint**
- httprint -h <host> -s signatures.txt
- httprint –h www.net-square.com –s usr/share/httpring/signatures.txt
  - ***if error comes that signature file** or **corrupted signature file** download the signature file from https://net-square.com/httprint.html
  - Download : from Screenshots and reports : Download - signatures.txt
  - And replace downloaded signatures.txt the same at /usr/share/httprint/
  - And try again

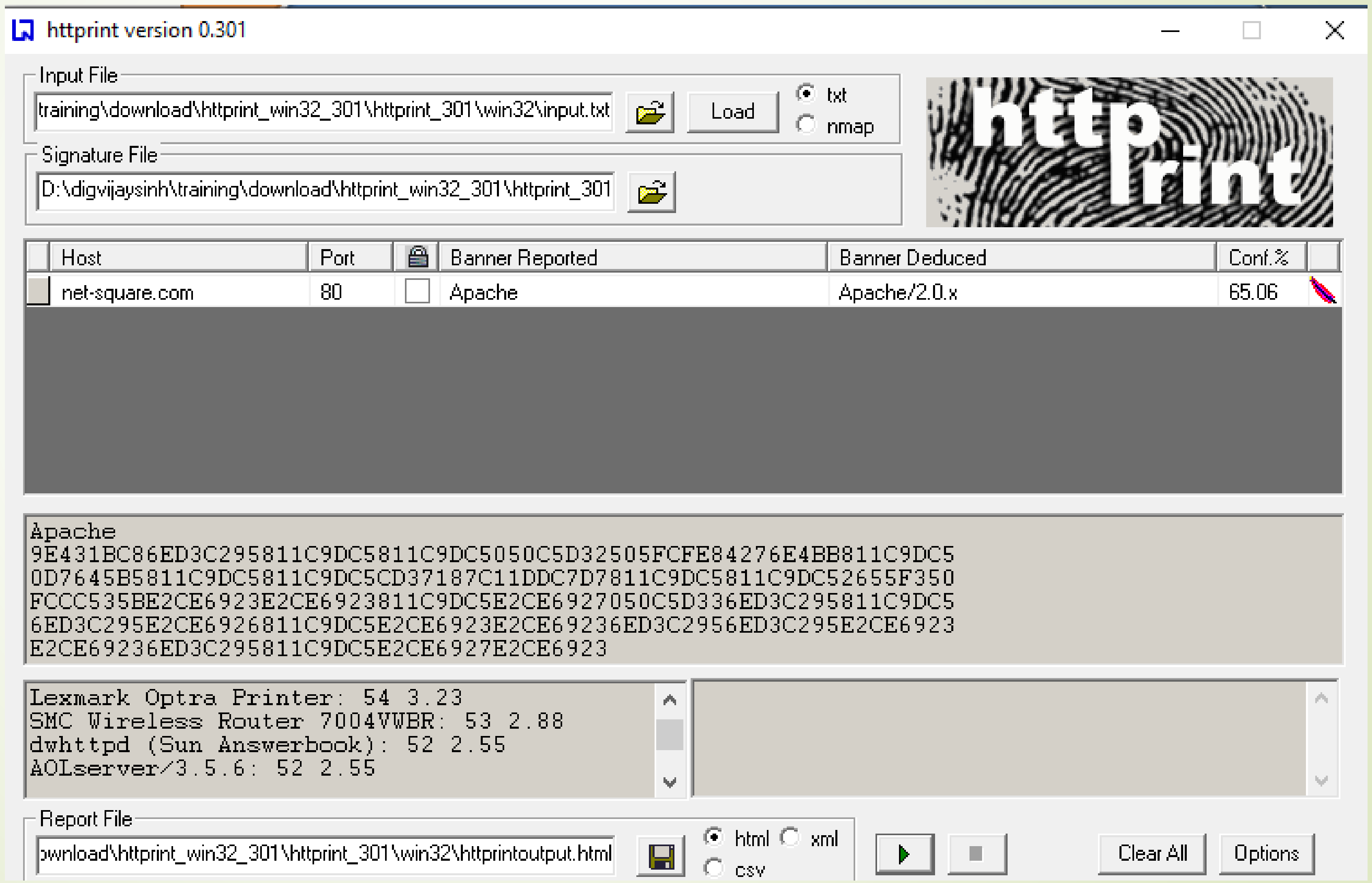
Fingerprinting the web server

- **HTTPrint**

- For Windows and GUI Download the HTTprint GUI version from **https://net-square.com/httprint.html** and download the Win32 GUI and cmd line version
- Unzip and start GUI using **httprint_gui.exe**
- It has an **input.txt** file available in the **httprint_win32_301\httprint_301\win32**
- Provide the name of the sites for which you would like to perform fingerprinting

Fingerprinting the web server

- **HTTPrint**



httprint version 0.301

Input File: training\download\httprint_win32_301\httprint_301\win32\input.txt [Load] [✓ txt] nmap

Signature File: D:\digvijaysinh\training\download\httprint_win32_301\httprint_301

Host	Port	🔒	Banner Reported	Banner Deduced	Conf.%
net-square.com	80	☐	Apache	Apache/2.0.x	65.06

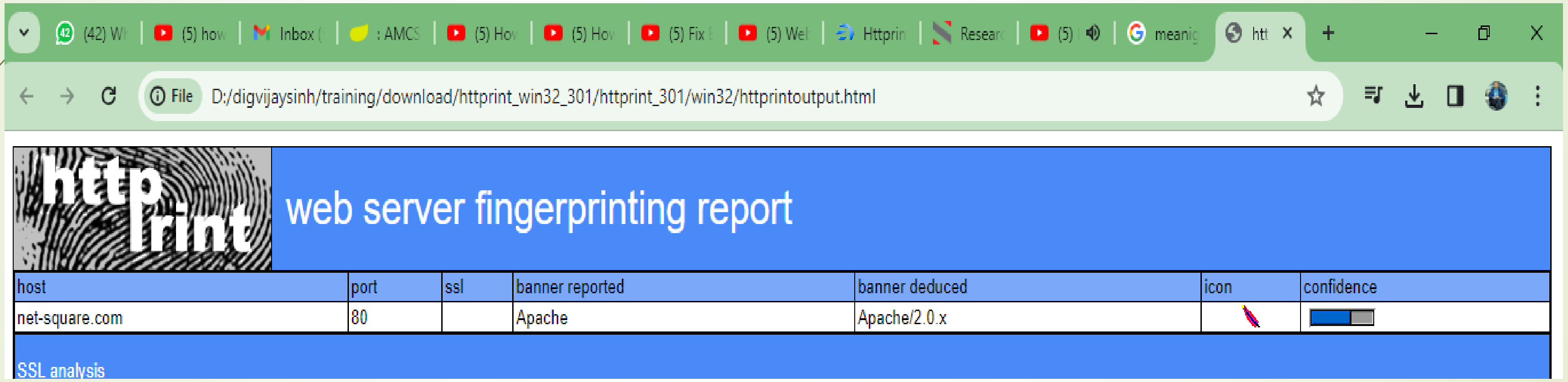
```
Apache
9E431BC86ED3C295811C9DC5811C9DC5050C5D32505FCFE84276E4BB811C9DC5
0D7645B5811C9DC5811C9DC5CD37187C11DDC7D7811C9DC5811C9DC52655F350
FCCC535BE2CE6923E2CE6923811C9DC5E2CE6927050C5D336ED3C295811C9DC5
6ED3C295E2CE6926811C9DC5E2CE6923E2CE69236ED3C2956ED3C295E2CE6923
E2CE69236ED3C295811C9DC5E2CE6927E2CE6923
```

```
Lexmark Optra Printer: 54 3.23
SMC Wireless Router 7004VWBR: 53 2.88
dwhttpd (Sun Answerbook): 52 2.55
AOLserver/3.5.6: 52 2.55
```

Report File: ownload\httprint_win32_301\httprint_301\win32\httprintoutput.html [✓ html] xml csv

Fingerprinting the web server

- **HTTPrint**



D:/digvijaysinh/training/download/httprint_win32_301/httprint_301/win32/httprintoutput.html

web server fingerprinting report

host	port	ssl	banner reported	banner deduced	icon	confidence
net-square.com	80		Apache	Apache/2.0.x		

SSL analysis

Subdomains enumeration

- Subdomain enumeration is the process of **identifying all subdomains for a given domain.**

- This can be useful for various purposes, such as **identifying potential targets for an attack** or organizational purposes.

- It also helps to **broaden the attack surface** and **find hidden applications and forgotten subdomains**.

Subdomains enumeration

- **Importance of Subdomain:**
    - There are several reasons why you might want to enumerate all subdomains for a given domain:
- **To identify potential targets for an attack:**
    - By enumerating all subdomains, you may find less well-protected subdomains than the root domain or the target organization, making them more vulnerable to attack.
- **To gain insights into the organization:**
    - Subdomain enumeration can give insights into how an organization is structured, what services it offers, and so on.
    - This information can be valuable when performing reconnaissance for a penetration test or security assessment.

Subdomains enumeration

- **To find misconfigured DNS entries:**
  - Organizations may sometimes have misconfigured DNS entries that reveal sensitive information, such as internal IP addresses.
- **Why sub-domain enumeration?**
  - Sub-domain enumeration can reveal a lot of domains/sub-domains that are in the scope of a security assessment, which, in turn, increases the chances of finding vulnerabilities.
  - Finding applications running on hidden, forgotten sub-domains may uncover critical vulnerabilities.
  - Frequently, the same vulnerabilities tend to be present across different domains/applications of the same organization

Subdomains enumeration

- **enumeration techniques**

- Search engines like **Google and Bing** support various advanced search operators to refine search queries. These operators are often referred to as **“Google dorks”.**
- We can use the **“site:”** operator in **Google search** to find all the sub-domains that Google has seen for a domain. Google also supports an additional **minus** operator to exclude sub-domains we are not interested in, such as **“site:*.wikimedia.org -www -store -jobs -uk”.**
- **Bing search engine** supports some advanced search operators as well. Like Google, **Bing also supports a “site:”** operator that you might want to check for additional results apart from the Google search.

Subdomains enumeration

- **enumeration techniques**

- **VirusTotal** runs its own passive DNS replication service, built by storing DNS resolutions that are performed when users visit URLs that have been submitted. To retrieve the information of a domain, you have to put the **domain name in the search bar.**
- **DNSdumpster** is another interesting tool that can find a potentially large number of sub-domains for a given domain.
- **The OWASP Amass tool** suite obtains subdomain names by scraping data sources, recursive brute forcing, crawling web archives, permuting/altering names, and reverse DNS sweeping.

Finding virtual hosts

- The term **Virtual Host refers** to the practice of **running more than one website** (such as company1.example.com and company2.example.com) on a **single machine.**

- Virtual hosts can be "**IP-based,**" meaning that you have a **different IP address for every website**, or "**name-based,**" meaning that you have **multiple names running on each IP address**. The fact that they are running on the **same physical server** is not apparent to the end user.

- Virtual hosts allow more than one website to be on one system or web server.

Finding virtual hosts

- The servers are differentiated by their hostname. Visitors to the Web site are routed by hostname or IP address to the correct virtual host.

- Virtual hosting allows companies to share one server so that each has its own domain name. For example, **www.company1.com and www.company2.com can be hosted on the same server.**

- **"Virtual hosting is a method for hosting multiple domain names (with separate handling of each name) on a single server (or pool of servers).** This allows one server to **share its resources**, such as **memory and processor** cycles, without requiring all services provided to use the same hostname."

Finding virtual hosts

- There are three variations of virtual hosts on HTTP Server:

- **1. IP address-based virtual host**
  - The IP address-based virtual host requires one IP address per Web site (hostname).
  - This approach works well but requires a dedicated IP address for every virtual host. For more information on virtual hosts, refer to the <VirtualHost> directive.

- Apache Example - https://httpd.apache.org/docs/2.4/vhosts/examples.html

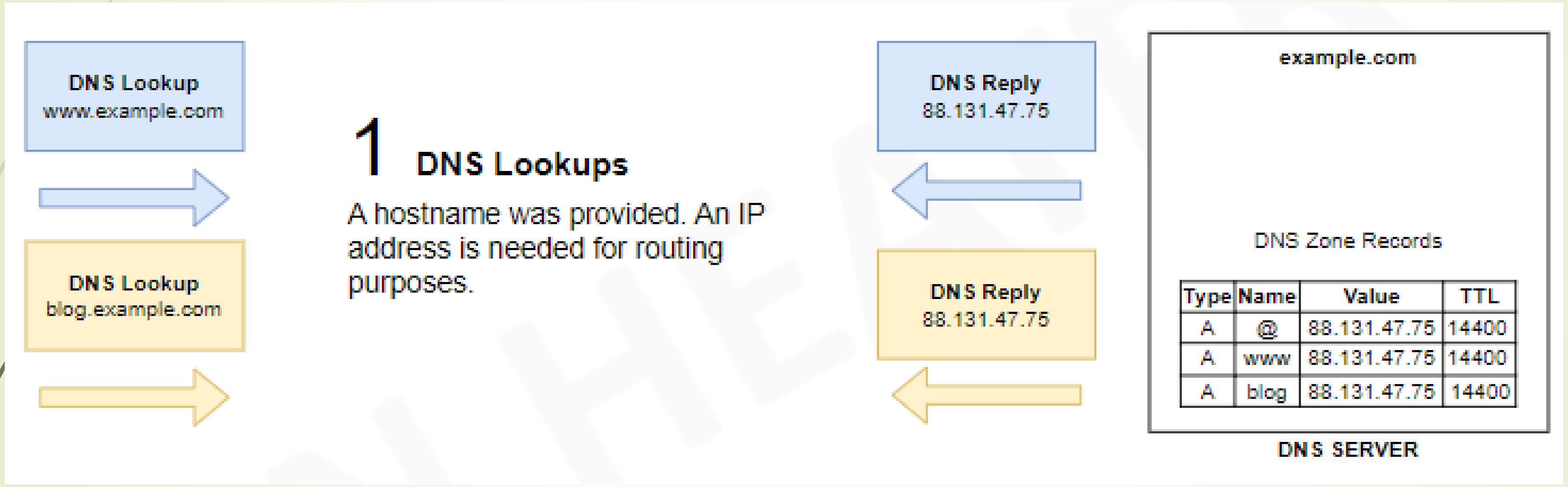
Finding virtual hosts

- **2. Name-based virtual host**

  - The name-based virtual host allows one IP address to host more than one Web site (hostname).
  - This approach allows practically **unlimited servers, ease of configuration and use, and requires no additional hardware or software**.
  - The main **disadvantage** to this approach is that the client **must support HTTP 1.1** (or HTTP 1.0 with 1.1 extensions), which includes the hostname information inside the HTTP document requests.
  - The latest versions of most browsers support HTTP 1.1 (or HTTP 1.0 with 1.1 extensions), but there are still old browsers that only support HTTP 1.0. For more information on virtual hosts, refer to the <VirtualHost> directive.

Finding virtual hosts

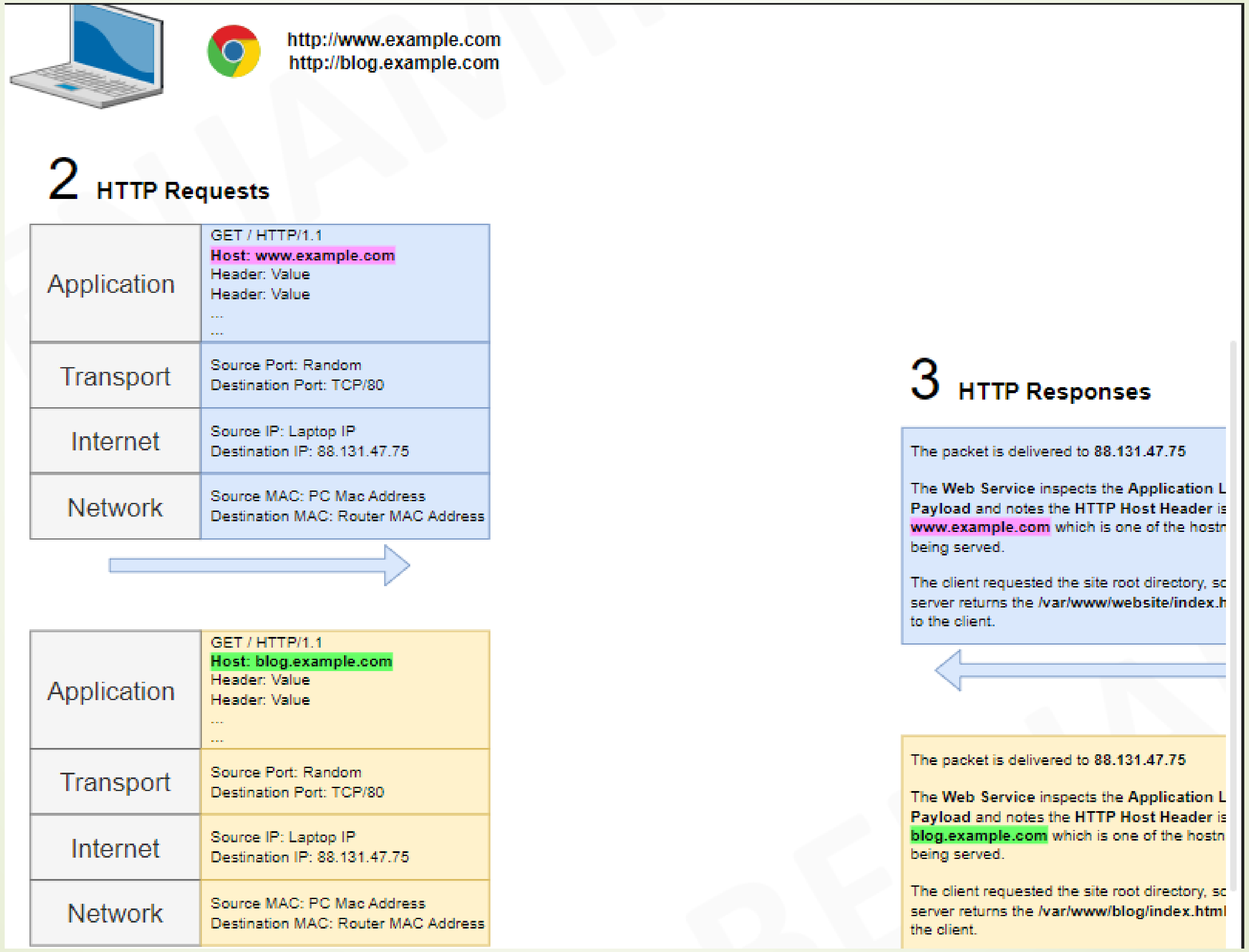
- **2. Name-based virtual host**



Type	Name	Value	TTL
A	@	88.131.47.75	14400
A	www	88.131.47.75	14400
A	blog	88.131.47.75	14400

- https://notes.benheater.com/books/web/page/virtualhost-enumeration#bkmrk-virtualhosts-example

Finding virtual hosts



http://www.example.com
http://blog.example.com

2 **HTTP Requests**

Application	GET / HTTP/1.1 **Host: www.example.com** Header: Value Header: Value ... ...
Transport	Source Port: Random Destination Port: TCP/80
Internet	Source IP: Laptop IP Destination IP: 88.131.47.75
Network	Source MAC: PC Mac Address Destination MAC: Router MAC Address

Application	GET / HTTP/1.1 **Host: blog.example.com** Header: Value Header: Value ... ...
Transport	Source Port: Random Destination Port: TCP/80
Internet	Source IP: Laptop IP Destination IP: 88.131.47.75
Network	Source MAC: PC Mac Address Destination MAC: Router MAC Address

3 **HTTP Responses**

The packet is delivered to **88.131.47.75**

The **Web Service** inspects the **Application L** **Payload** and notes the **HTTP Host Header** is **www.example.com** which is one of the hostn being served.

The client requested the site root directory, sc server returns the **/var/www/website/index.h** to the client.

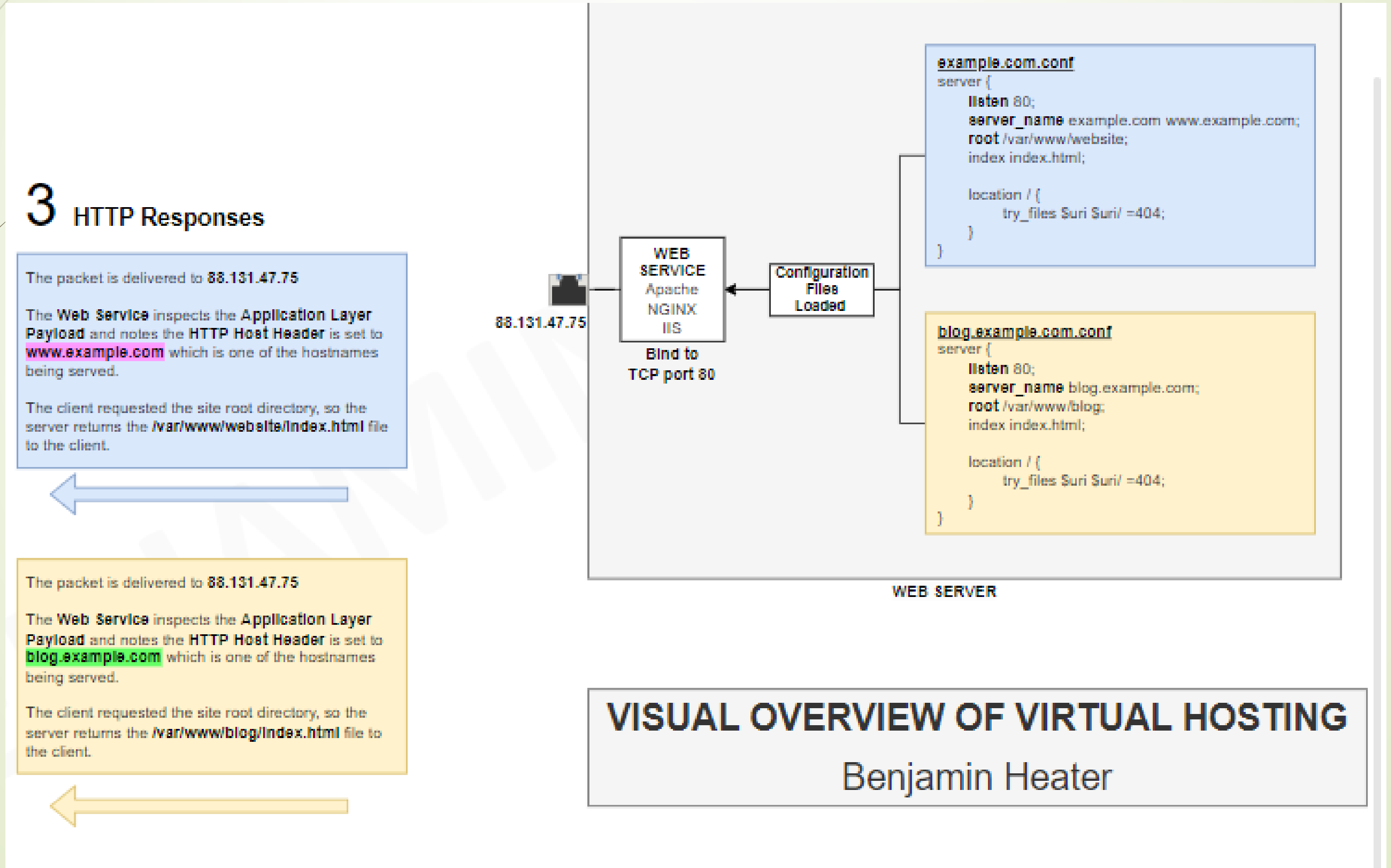
The packet is delivered to **88.131.47.75**

The **Web Service** inspects the **Application L** **Payload** and notes the **HTTP Host Header** is **blog.example.com** which is one of the hostn being served.

The client requested the site root directory, sc server returns the **/var/www/blog/index.html** the client.

Finding virtual hosts

- **2. Name-based virtual host**



The packet is delivered to **88.131.47.75**

The **Web Service** inspects the **Application Layer Payload** and notes the **HTTP Host Header** is set to **www.example.com** which is one of the hostnames being served.

The client requested the site root directory, so the server returns the **/var/www/website/index.html** file to the client.

The packet is delivered to **88.131.47.75**

The **Web Service** inspects the **Application Layer Payload** and notes the **HTTP Host Header** is set to **blog.example.com** which is one of the hostnames being served.

The client requested the site root directory, so the server returns the **/var/www/blog/index.html** file to the client.

example.com.conf
```
server {
    listen 80;
    server_name example.com www.example.com;
    root /var/www/website;
    index index.html;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

blog.example.com.conf
```
server {
    listen 80;
    server_name blog.example.com;
    root /var/www/blog;
    index index.html;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

VISUAL OVERVIEW OF VIRTUAL HOSTING

Benjamin Heater

Finding virtual hosts

- **3. Dynamic virtual host**

  - The dynamic virtual host allows you to dynamically add Websites (host names) by adding content directories.
  - This approach is based on automatically inserting the IP address and the Host: header's contents into the file's pathname that is used to satisfy the request.
  - **The advantages of a dynamic virtual host are:**
    - A smaller configuration file so the server starts faster and uses less memory.
    - Adding virtual hosts does not require the configuration to be changed or the server to be restarted.
  - **The disadvantage** of a dynamic virtual host is that you cannot have a different log file for each virtual host.

Finding virtual hosts

- **Gobuster**

  - Gobuster is a tool used to brute force URLs (directories and files) from websites, DNS subdomains, Virtual Hostnames, and open Amazon S3 buckets. It can be particularly useful during CTF challenges that require you to brute force web server data and during pentest engagements.
  - Installation on Linux (Kali)
  - Sudo apt-get install gobuster
  - gobuster [mode] -u [target ip] -w [wordlist]
  - Gobuster can run in multiple scanning modes, when writing these are: dir, dns and vhost.
  - It is recommended to download the wordlist from GitHub for the Vhost, DNS, and Dir.

Security Misconfiguration

- A security misconfiguration can occur when security settings are either
  - (1) Not implemented,
  - (2) Deployed with errors.
- These errors create security gaps that expose the application and its data to a cyber attack or possible breach.
- These errors can happen at any level of the application stack, including:
  - Web or application servers or Databases Server
  - Network services
  - Custom code or Development platforms and frameworks
  - Storage
  - Virtual machines or Cloud containers

Security Misconfiguration

- What Can Security Misconfigurations Lead To?
  - Unpatched flaws
  - Unused pages and unnecessary service
  - Inadequate access controls
  - Unprotected files and directories
  - Poor coding practices and using vulnerable XML files
  - Disabled antivirus
  - Inadequate hardware management

Security Misconfiguration

- How a Server Misconfiguration Can Create Vulnerabilities
  - Exposes sensitive data
  - Prompts directory traversal attacks
  - Increases cyberattacks on mobile applications
  - Creates remote attacks
  - Provides unauthorized access to the organization

Security Misconfiguration

- How Organizations Can Prevent Security Configuration Vulnerabilities
  - Pay attention to alerts
  - Regularly patch all devices and software
  - Strengthen remote access controls
  - Provide cybersecurity training and awareness to all users
  - Follow secure coding practices

Security Misconfiguration

- Some other ways to avoid security misconfiguration errors are:

  - Regularly monitor web application security and vulnerabilities
  - Define and monitor non-default security settings for apps and programs
  - Remove unused applications, programs, and features
  - Change all default accounts, usernames, and passwords
  - Develop an application architecture with secure separation of elements
  - Encrypt data-at-rest and data-in-transit

- Reference: https://reciprocity.com/blog/security-misconfigurations-how-to-avoid-them/

QUESTIONS?
COMMENTS?
CONCERNS?