Networking Concepts

Learned:

- Tracking MAC addresses is important to see which devices are communicating on a local network. Each frame contains a source MAC and destination MAC Address, showing who is sending and receiving data.

- TCP often carries sensitive data because it’s reliable and ordered, while UDP is used for less crucial information such as video streaming or voice calls.

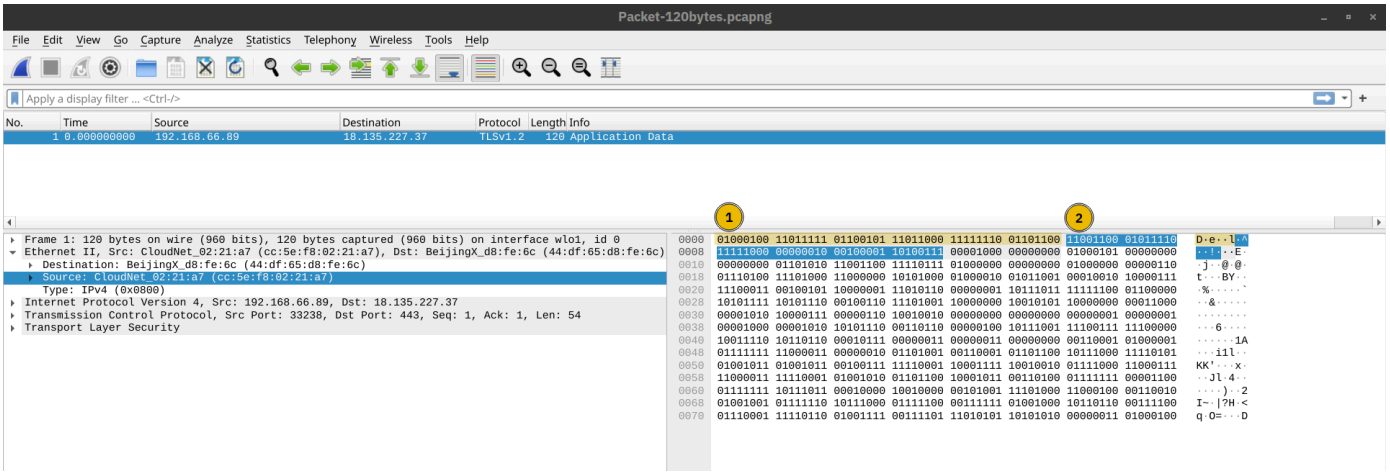
- A subnet mask is identified by its number, `255.255.255.0`, which shows how the IP address is divided between the network and host.

- Public IP addresses are used for communication over the Internet between devices, while private IP addresses are used only within local networks and are not reachable from the Internet.

- Data is sent across networks in layers, with each layer adding its own headers so it reaches the correct destination reliably and can be tracked or analyzed.

We expect to see two MAC addresses in each frame in real network communication over Ethernet or WiFi. The packet in the screenshot below shows:

- The destination data-link address (MAC address) highlighted in yellow
- The source data link address (MAC address) is highlighted in blue
- The remaining bits show the data being sent



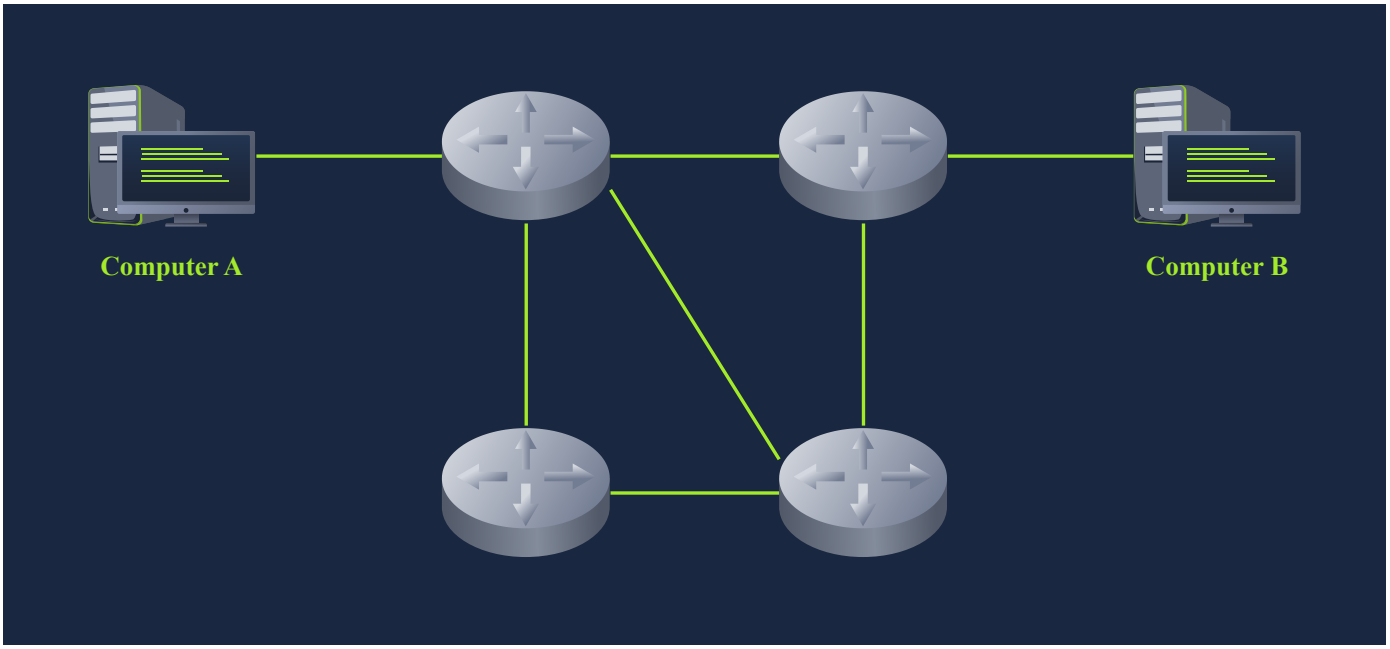
Layer 3: Network Layer

The data link layer focuses on sending data between two nodes on the same network segment. The network layer, i.e., layer 3, is concerned with sending data between different networks. In more technical

terms, the network layer handles logical addressing and routing, i.e., finding a path to transfer the network packets between the diverse networks.

In the data link layer, we gave an example of one company office with ten computers, where the data link layer is responsible for providing a connection between them. Let's say that this company has multiple offices distributed across various cities, countries, or even continents. The network layer is responsible for connecting the different offices together.

The network below shows that computers A and B are connected, although on different networks. You can also notice two paths connecting the two computers; the network layer will route the network packets through the path it deems better.



Examples of the network layer include Internet Protocol (IP), Internet Control Message Protocol (ICMP), and Virtual Private Network (VPN) protocols such as IPSec and SSL/TLS VPN.

Layer 4: Transport Layer

Layer 4, the transport layer, enables end-to-end communication between running applications on different hosts. Your web browser is connected to the TryHackMe web server over the transport layer, which can support various functions like flow control, segmentation, and error correction.

Examples of layer 4 are Transmission Control Protocol (TCP) and User Datagram Protocol (UDP).

Layer 5: Session Layer

The session layer is responsible for establishing, maintaining, and synchronising communication between applications running on different hosts. Establishing a session means initiating communication between applications and negotiating the necessary parameters for the session. Data synchronisation ensures that data is transmitted in the correct order and provides mechanisms for recovery in case of transmission failures.

Examples of the session layer are Network File System (NFS) and Remote Procedure Call (RPC).

Layer 6: Presentation Layer

The presentation layer ensures the data is delivered in a form the application layer can understand. Layer 6 handles data encoding, compression, and encryption. An example of encoding is character encoding, such as ASCII or Unicode.

Various standards are used at the presentation layer. Consider the scenario where we want to send an image via email. First, we use JPEG, GIF, and PNG to save our images; furthermore, although hidden from the user by the email client, we use MIME (Multipurpose Internet Mail Extensions) to attach the file to our email. MIME encodes a binary file using 7-bit ASCII characters.

Layer 7: Application Layer

The application layer provides network services directly to end-user applications. Your web browser would use the HTTP protocol to request a file, submit a form, or upload a file.

The application layer is the top layer, and you might have encountered many of its protocols as you use different applications. Examples of Layer 7 protocols are HTTP, FTP, DNS, POP3, SMTP, and IMAP.

Layer Number	Layer Name	Main Function	Example Protocols and Standards
Layer 7	Application layer	Providing services and interfaces to applications	HTTP, FTP, DNS, POP3, SMTP, IMAP
Layer 6	Presentation layer	Data encoding, encryption, and compression	Unicode, MIME, JPEG, PNG, MPEG
Layer 5	Session layer	Establishing, maintaining, and synchronising sessions	NFS, RPC
Layer 4	Transport layer	End-to-end communication and data segmentation	UDP, TCP
Layer 3	Network layer	Logical addressing and routing between networks	IP, ICMP, IPSec
Layer 2	Data link layer	Reliable data transfer between adjacent nodes	Ethernet (802.3), WiFi (802.11)
Layer 1	Physical layer	Physical data transmission media	Electrical, optical, and wireless signals

TCP/IP Model

The table below shows how the TCP/IP model layers map to the ISO/OSI model layers.

Layer Number	ISO OSI Model	TCP/IP Model (RFC 1122)	Protocols
7	Application Layer	Application Layer	HTTP, HTTPS, FTP, POP3, SMTP, IMAP, Telnet, SSH,
6	Presentation Layer		
5	Session Layer		
4	Transport Layer	Transport Layer	TCP, UDP
3	Network Layer	Internet Layer	IP, ICMP, IPSec
2	Data Link Layer	Link Layer	Ethernet 802.3, WiFi 802.11
1	Physical Layer		

IP Addresses and Subnets

Looking Up Your Network Configuration

You can look up your IP address on the MS Windows command line using the command `ipconfig`. On Linux and UNIX-based systems, you can issue the command `ifconfig` or `ip address show`, which can be typed as `ip a s`. In the terminal window below, we show `ifconfig`.

Terminal

```
user@TryHackMe$ ifconfig
[...]
wlo1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.66.89  netmask 255.255.255.0  broadcast 192.168.66.255
        inet6 fe80::73e1:ca5e:3f93:b1b3  prefixlen 64  scopeid 0x20<link>
        ether cc:5e:f8:02:21:a7  txqueuelen 1000  (Ethernet)
        RX packets 19684680  bytes 18865072842 (17.5 GiB)
        RX errors 0  dropped 364  overruns 0  frame 0
        TX packets 14439678  bytes 8773200951 (8.1 GiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The terminal output above indicates the following:

- The host (laptop) IP address is `192.168.66.89`
- The subnet mask is `255.255.255.0`

- The broadcast address is `192.168.66.255`

Let's use `ip a s` to compare how the network card IP address is presented.

Terminal

```
user@TryHackMe$ ip a s
[...]
4: wlo1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group
default qlen 1000
    link/ether cc:5e:f8:02:21:a7 brd ff:ff:ff:ff:ff:ff
    altname wlp3s0
    inet 192.168.66.89/24 brd 192.168.66.255 scope global dynamic noprefixroute
wlo1
       valid_lft 36795sec preferred_lft 36795sec
    inet6 fe80::73e1:ca5e:3f93:b1b3/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

The terminal output above indicates the following:

- The host (laptop) IP address is `192.168.66.89/24`
- The broadcast address is `192.168.66.255`

If you are wondering, a subnet mask of `255.255.255.0` can also be written as `/24`. The `/24` means that the leftmost 24 bits within the IP address do not change across the network, i.e., the subnet. In other words, the leftmost three octets are the same across the whole subnet; therefore, we can expect to find addresses that range from `192.168.66.1` to `192.168.66.254`. Similar to what was mentioned earlier, `192.168.66.0` and `192.168.66.255` are the network and broadcast addresses, respectively.

Private Addresses

There are two types of IP addresses:

- Public IP addresses
- Private IP addresses

RFC 1918 defines the following three ranges of private IP addresses:

- `10.0.0.0` - `10.255.255.255` (`10/8`)
- `172.16.0.0` - `172.31.255.255` (`172.16/12`)
- `192.168.0.0` - `192.168.255.255` (`192.168/16`)

1. **10.x.x.x** → All addresses starting with 10
    - Example: `10.0.0.1`, `10.55.32.100`
2. **172.16.x.x – 172.31.x.x** → Only this range in 172
    - Example: `172.16.0.1`, `172.25.100.50`
    - Anything outside 172.16–31 is **public**
3. **192.168.x.x** → All addresses starting with 192.168
    - Example: `192.168.0.1`, `192.168.1.100`

UDP And TCP

The IP protocol allows us to reach a destination host on the network; the host is identified by its IP address. We need protocols that would enable processes on networked hosts to communicate with each other. There are two transport protocols to achieve that: UDP and TCP.

UDP

UDP (User Datagram Protocol) allows us to reach a specific process on this target host. UDP is a simple connectionless protocol that operates at the transport layer, i.e., layer 4. Being connectionless means that it does not need to establish a connection. UDP does not even provide a mechanism to know that the packet has been delivered.

An IP address identifies the host; we need a mechanism to determine the sending and receiving process. This can be achieved by using port numbers. A port number uses two octets; consequently, it ranges between 1 and 65535; port 0 is reserved.

TCP

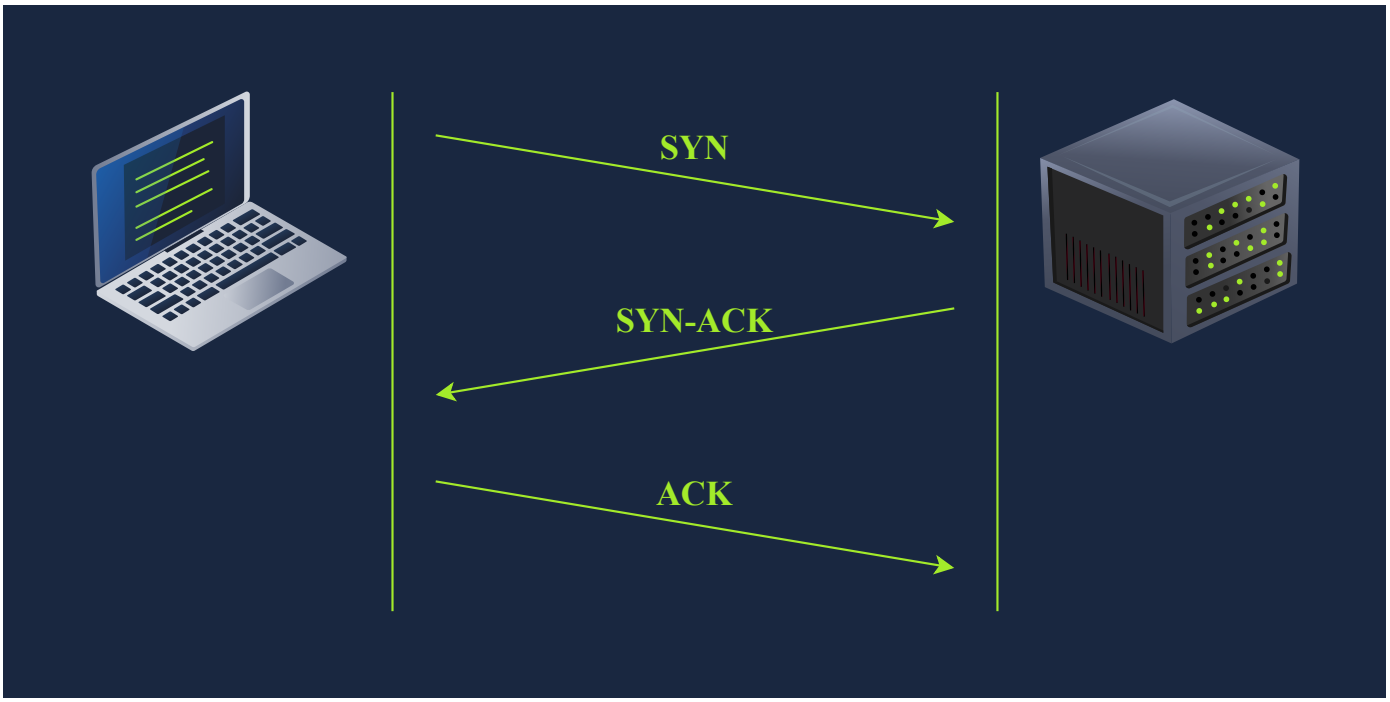
TCP (Transmission Control Protocol) is a connection-oriented transport protocol. It uses various mechanisms to ensure reliable data delivery sent by the different processes on the networked hosts. Like UDP, it is a layer 4 protocol. Being connection-oriented, it requires the establishment of a TCP connection before any data can be sent.

In TCP, each data octet has a sequence number; this makes it easy for the receiver to identify lost or duplicated packets. The receiver, on the other hand, acknowledges the reception of data with an acknowledgement number specifying the last received octet.

A TCP connection is established using what's called a three-way handshake. Two flags are used: SYN (Synchronise) and ACK (Acknowledgment). The packets are sent as follows:

1. SYN Packet: The client initiates the connection by sending a SYN packet to the server. This packet contains the client's randomly chosen initial sequence number.

2. SYN-ACK Packet: The server responds to the SYN packet with a SYN-ACK packet, which adds the initial sequence number randomly chosen by the server.
3. ACK Packet: The three-way handshake is completed as the client sends an ACK packet to acknowledge the reception of the SYN-ACK packet.

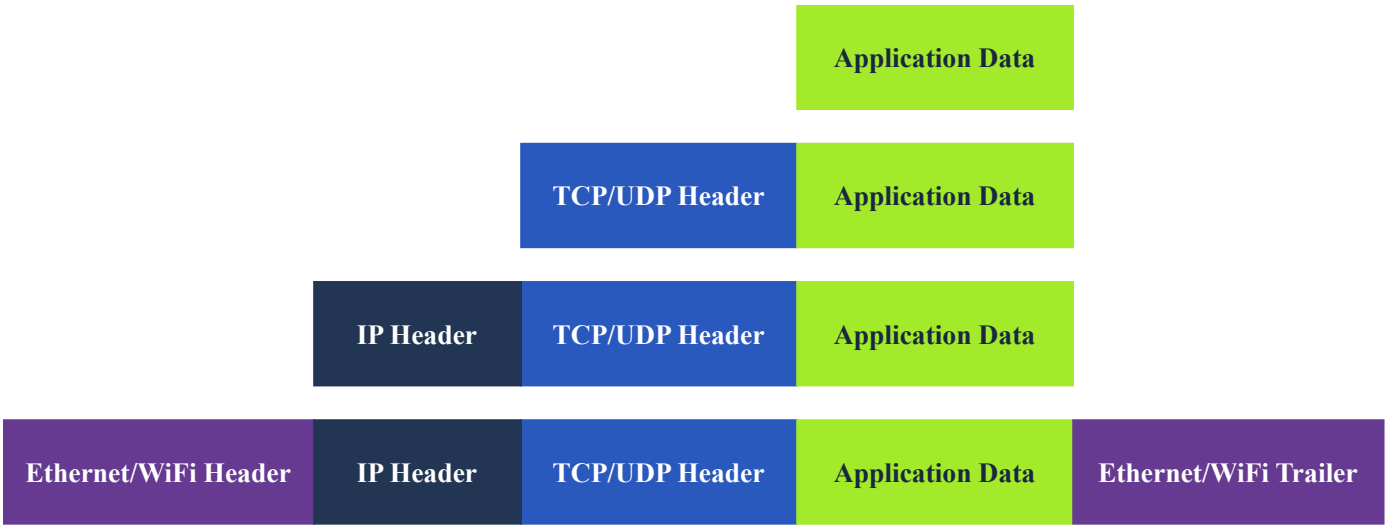


Similar to UDP, TCP identifies the process of initiating or waiting (listening) for a connection using port numbers. As stated, a valid port number ranges between 1 and 65535 because it uses two octets and port 0 is reserved.

Encapsulation

- **Application data**: It all starts when the user inputs the data they want to send into the application. For example, you write an email or an instant message and hit the send button. The application formats this data and starts sending it according to the application protocol used, using the layer below it, the transport layer.
- **Transport protocol segment or datagram**: The transport layer, such as TCP or UDP, adds the proper header information and creates the TCP **segment** (or UDP **datagram**). This segment is sent to the layer below it, the network layer.
- **Network packet**: The network layer, i.e. the Internet layer, adds an IP header to the received TCP segment or UDP datagram. Then, this IP **packet** is sent to the layer below it, the data link layer.
- **Data link frame**: The Ethernet or WiFi receives the IP packet and adds the proper header and trailer, creating a **frame**.

We start with application data. At the transport layer, we add a TCP or UDP header to create a **TCP segment** or **UDP datagram**. Again, at the network layer, we add the proper IP header to get an **IP packet** that can be routed over the Internet. Finally, we add the appropriate header and trailer to get a WiFi or Ethernet frame at the link layer.



Telnet

The TELNET (Teletype Network) protocol is a network protocol for remote terminal connection. In simpler words, `telnet`, a TELNET client, allows you to connect to and communicate with a remote system and issue text commands. Although initially it was used for remote administration, we can use `telnet` to connect to any server listening on a TCP port number.

On the target virtual machine, different services are running. We will experiment with three of them:

- Echo server: This server echoes everything you send it. By default, it listens on port 7.
- Daytime server: This server listens on port 13 by default and replies with the current day and time.
- Web (HTTP) server: This server listens on TCP port 80 by default and serves web pages.

Echo and daytime servers are considered security risks and should not be run

In the terminal below, we connect to the target VM at the echo server's TCP port number 7. To close the connection, press the `CTRL` + `]` keys simultaneously.

Terminal

```
user@TryHackMe$ telnet 10.201.48.206 7
telnet 10.201.48.206 7
Trying 10.201.48.206...
Connected to 10.201.48.206.
Escape character is '^]'.
Hi
```

```
Hi
How are you?
How are you?
Bye
Bye
^]

telnet> quit
Connection closed.
```

In the terminal below, we use `telnet` to connect to the daytime server listening at port 13. We noticed that the connection closes once the current date and time are returned.

Terminal

```
user@TryHackMe$ telnet 10.201.48.206 13
Trying 10.201.48.206...
Connected to 10.201.48.206.
Escape character is '^]'.
Thu Jun 20 12:36:32 PM UTC 2024
Connection closed by foreign host.
```