Kaitlin Hoffmann

Office Hours: SH 243 Mondays 1:15 - 3:15PM. Tuesdays 2:45 - 4:45PM.
Email: hoffmank4@newpaltz.edu

NETWORKING BASICS

NETWORK SECURITY

OBJECTIVES

- What is Networking
- TCP/IP Protocols

WHAT IS NETWORKING?

- **Networking**, also known as computer networking, is the practice of transporting and exchanging data between nodes over a shared medium in an information system.
- **Nodes** can be physical networked devices, such as modems, PCs and printers. A node checks for identification, such as an IP address, to grant access to the node.
- Nodes **connect** over a link or communication channel. In a computer network these may be cable, fiber optic or wireless connections.

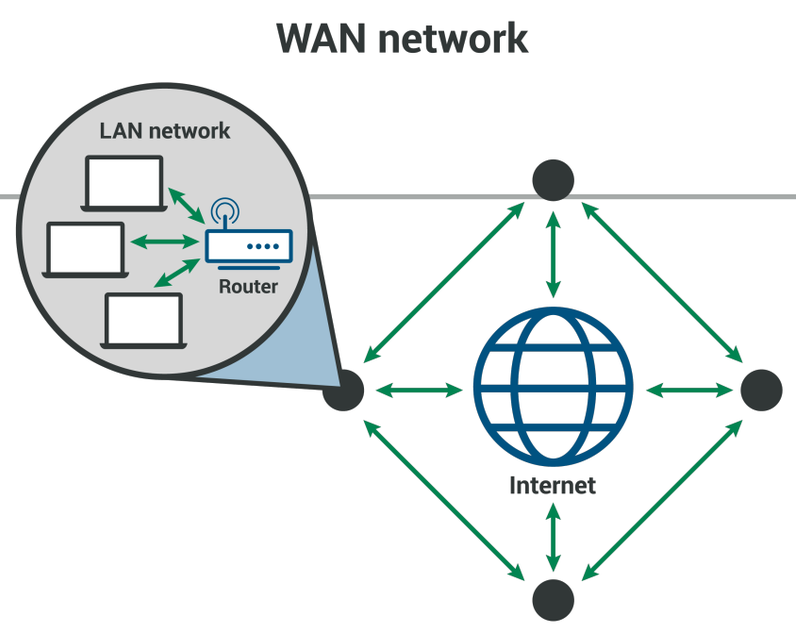
WHAT IS NETWORKING?

Some of the uses of computer networks are:

- Communicating using email, video, instant messaging, etc.
- Sharing devices such as printers, scanners, etc.
- Sharing files
- Sharing software and operating programs on remote systems
- Allowing network users to easily access and maintain information




WHAT IS NETWORKING?



Types of Computer Networks:

- Personal Area Network (PAN)
- Local Area Network (LAN)
- Wide Area Network (WAN)
- Wireless Local Area Network (WLAN)
- Campus Area Network (CAN)
- Metropolitan Area Network (MAN)
- Storage Area Network (SAN)
- System-Area Network (SAN)
- Passive Optical Local Area Network (POLAN)
- Enterprise Private Network (EPN)
- Virtual Private Network (VPN)
- Home Area Network (HAN)

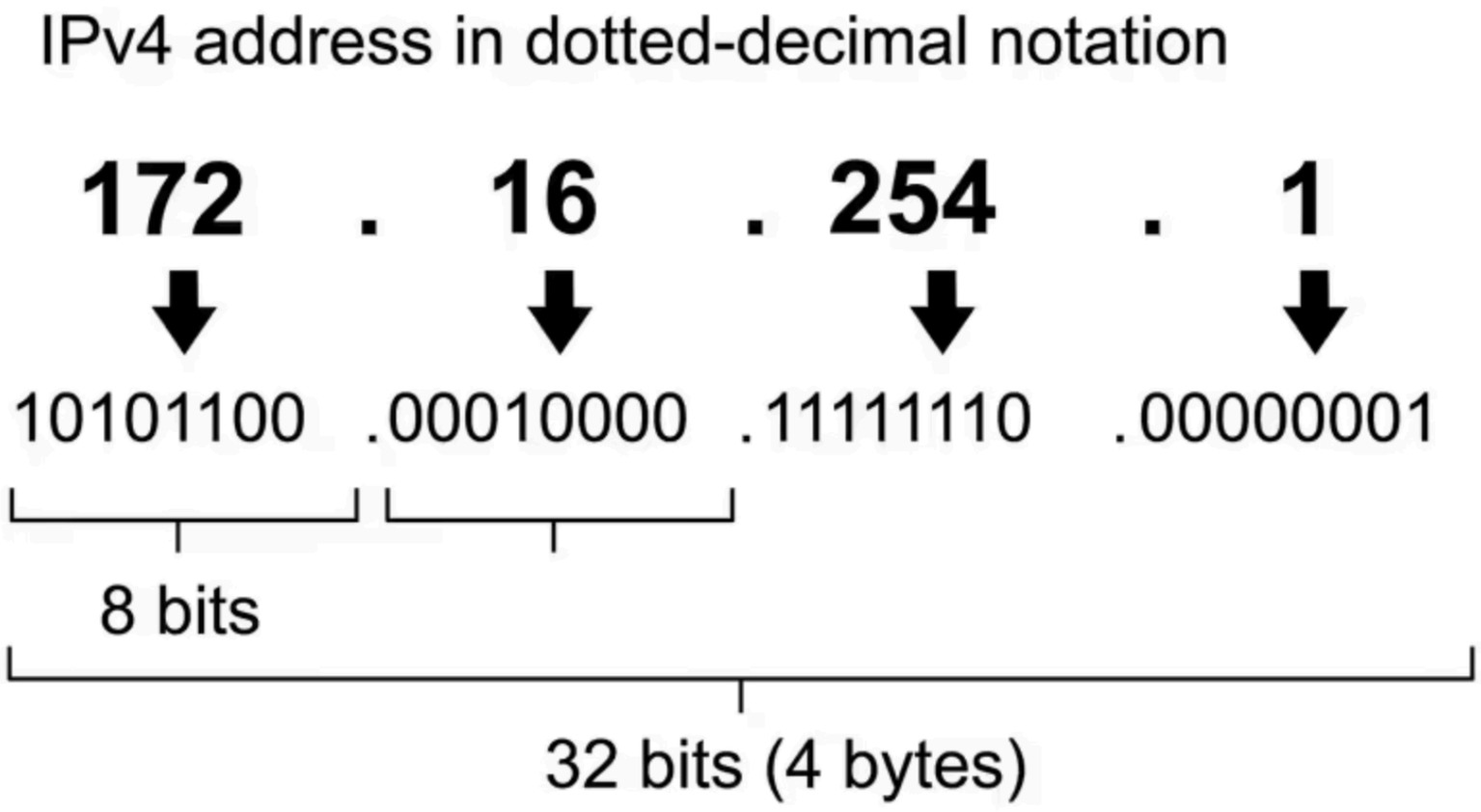
IP ADDRESS

- Every machine on a network has a unique identifier.
- Most networks today, including all computers on the internet, use the **TCP/IP** protocol as the standard for how to communicate on the network. In the TCP/IP protocol, the unique identifier for a computer is called its **IP address**.
- There are two standards for IP addresses: IP Version 4 (**IPv4**) and IP Version 6 (**IPv6**). All computers with IP addresses have an IPv4 address, and most use the new IPv6 address system as well.

Think of an IP address like a phone number to call a friend.

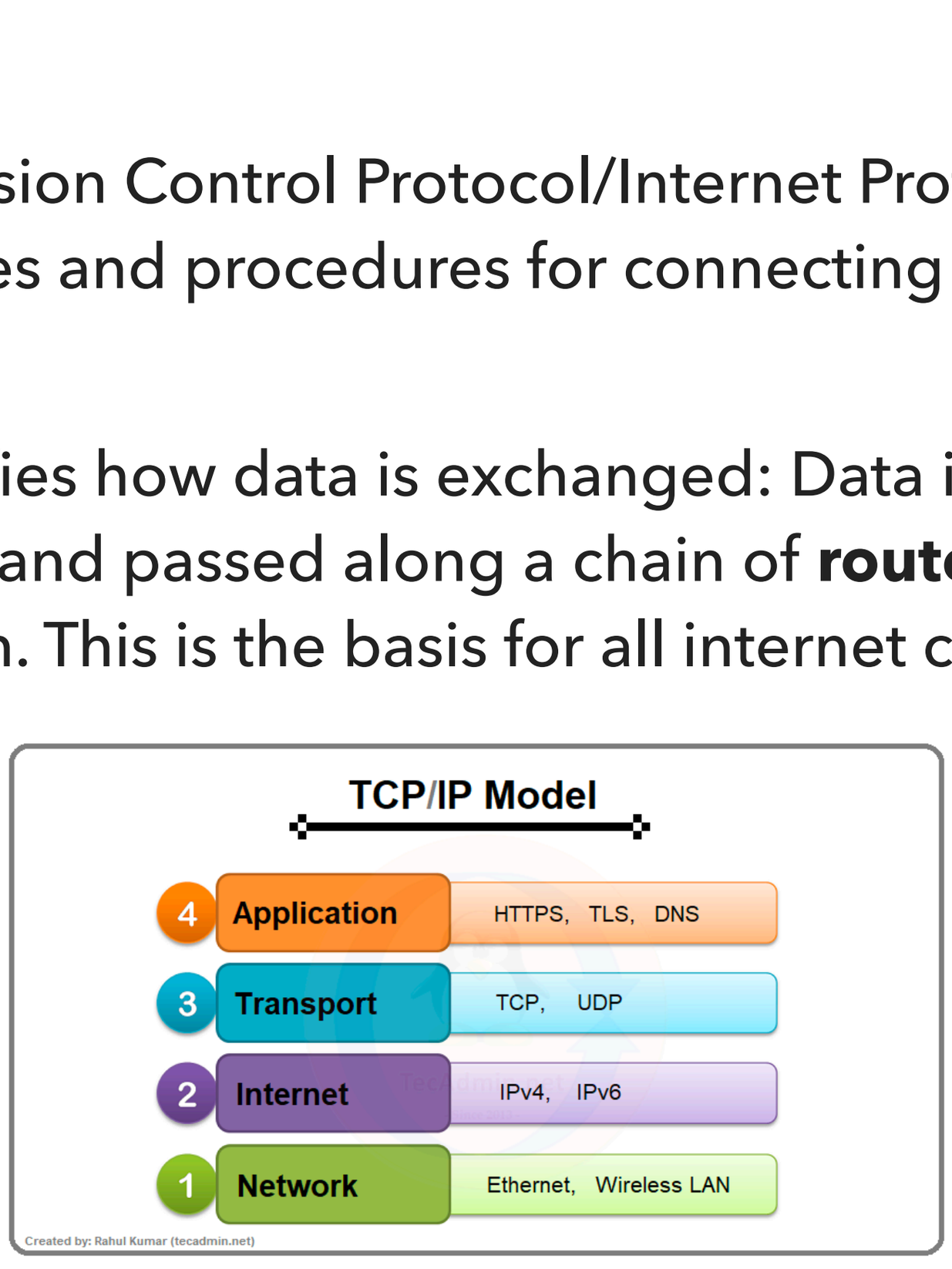
IP ADDRESS – STATIC VS DYNAMIC

- IP addresses can be permanent (**static**) or temporary (**dynamic**). The difference between static and dynamic IP addresses is that while the former never change, the latter can and do.
- Static addresses are mostly used by businesses, since their websites and web applications must be reliably accessible at all times. But your home IP address doesn't have to stay the same, since it's only needed when you're using the internet.



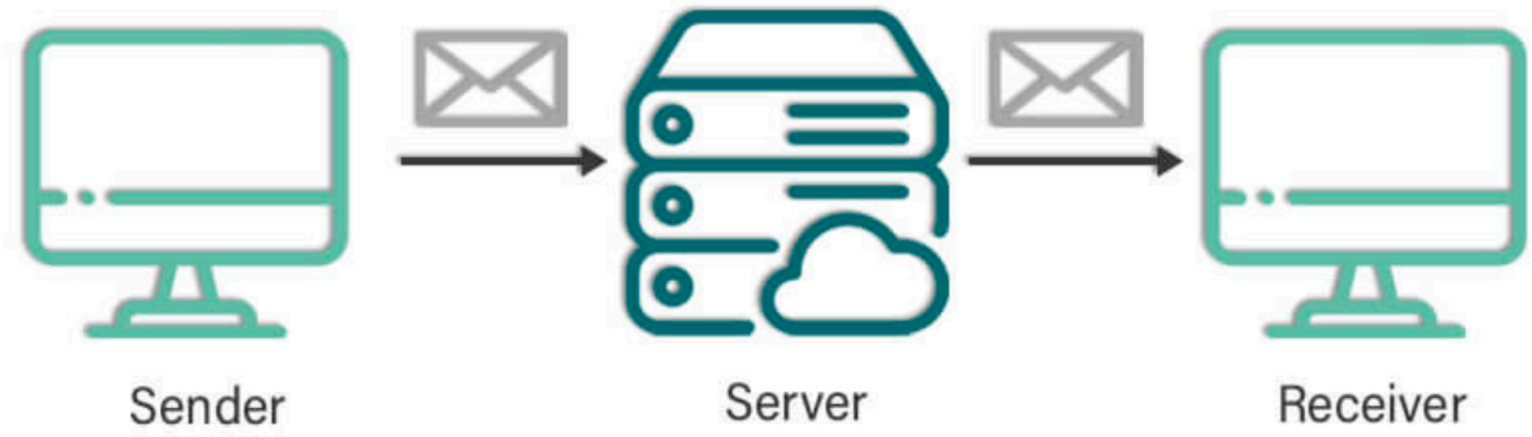
TCP/IP

- The Transmission Control Protocol/Internet Protocol (**TCP/IP**) is a set of rules and procedures for connecting devices across the internet.
- TCP/IP specifies how data is exchanged: Data is broken down into **packets** and passed along a chain of **routers** from origin to destination. This is the basis for all internet communication.



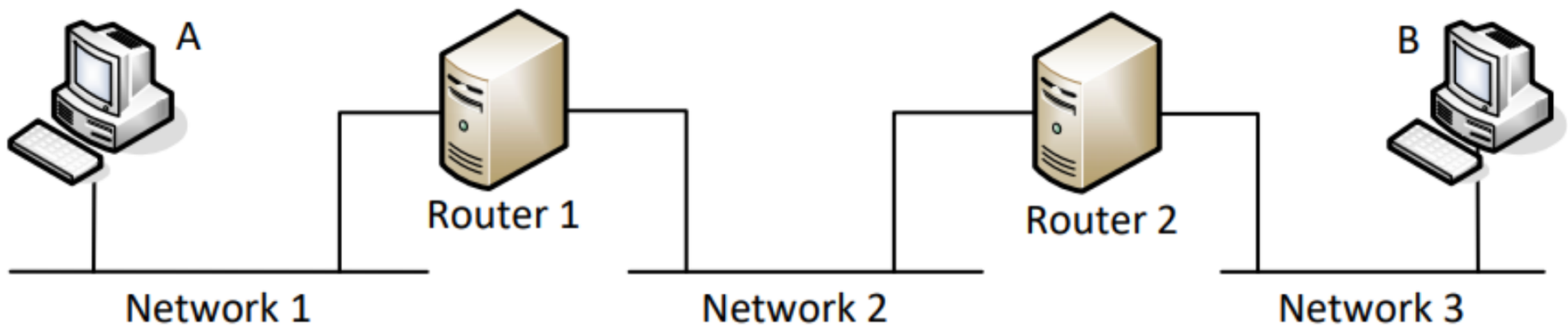
TCP VS IP

- **TCP** defines ***how*** applications communicate across the network. It manages how a message is broken down into a series of smaller packets, which are then transmitted over the internet and reassembled in the right order at the destination address.
- The **IP** portion of the protocol directs each packet to the right **destination**. Each gateway computer on the network checks this IP address to determine where to forward the message.



WHAT ARE PACKETS?

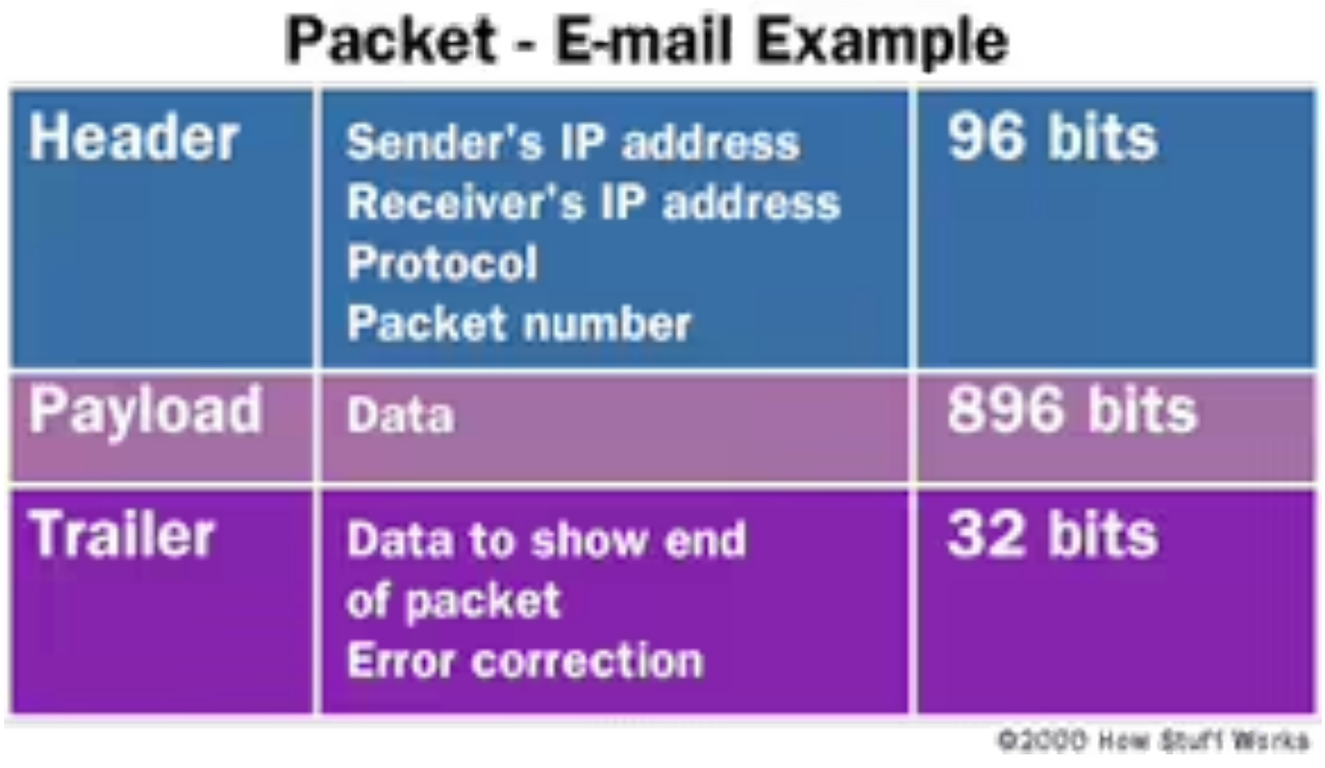
- Everything you do on the Internet involves **packets**. For example, every web page that you receive comes as a series of packets, and every e-mail you send leaves as a series of packets.
- On the Internet, the network breaks an e-mail message into parts of a certain size in bytes. These are the packets. Each packet carries the information that will help it get to its destination.



MOST NETWORK PACKETS ARE SPLIT INTO THREE PARTS:

Header - The header contains instructions about the data carried by the packet. These instructions may include:

- Length of packet
- Synchronization (help the packet match up to the network)
- Packet number
- Protocol (Type of packet is being transmitted: e-mail, web page, video)
- Destination address (where the packet is going)
- Originating address (where the packet came from)



MOST NETWORK PACKETS ARE SPLIT INTO THREE PARTS:

- **Payload** - Also called the **body** or **data** of a packet. This is the actual data that the packet is delivering to the destination.
- If a packet is fixed-length, then the payload may be **padded** with blank information to make it the right size.

Packet - E-mail Example

Header	Sender's IP address Receiver's IP address Protocol Packet number	96 bits
Payload	Data	896 bits
Trailer	Data to show end of packet Error correction	32 bits

MOST NETWORK PACKETS ARE SPLIT INTO THREE PARTS:

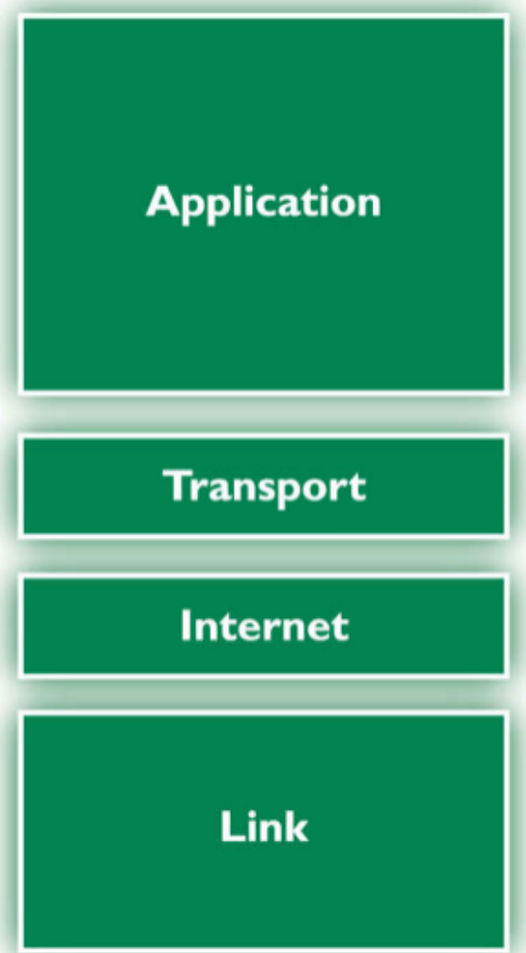
- **Trailer** - The trailer, sometimes called the **footer**, typically contains a couple of bits that tell the receiving device that it has reached the end of the packet.
- It may also have some type of error checking.

Packet - E-mail Example

Header	Sender's IP address Receiver's IP address Protocol Packet number	96 bits
Payload	Data	896 bits
Trailer	Data to show end of packet Error correction	32 bits

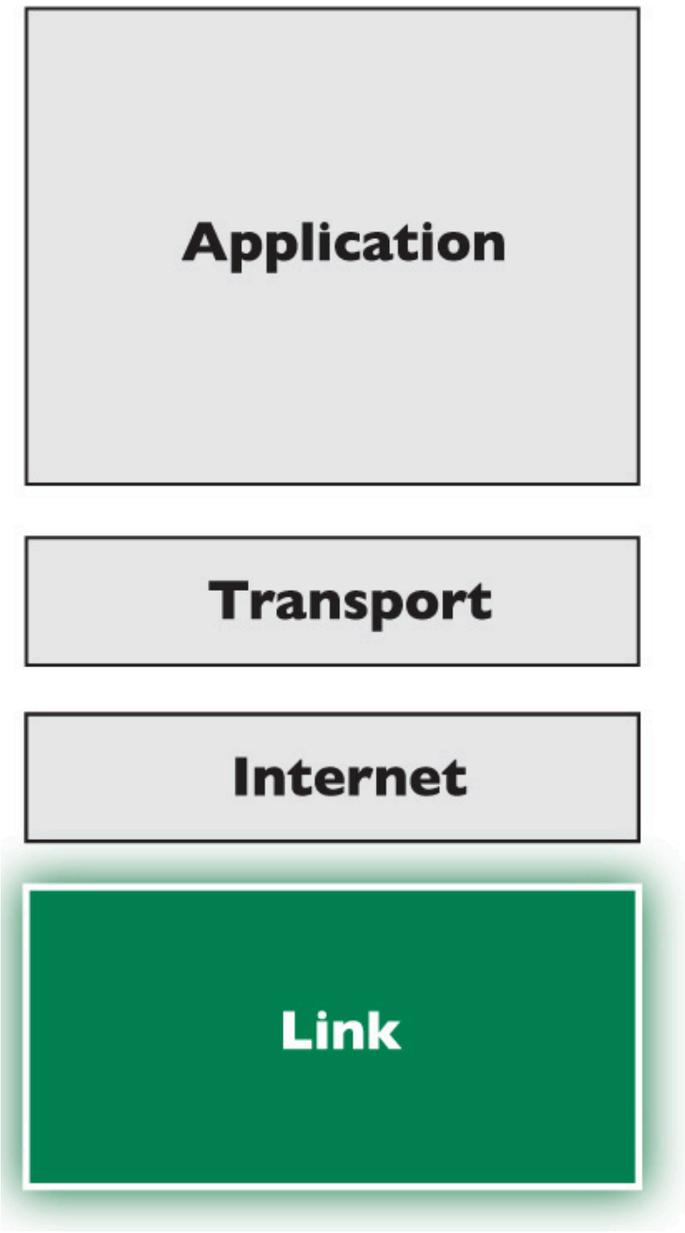
BACK TO TCP/IP

- The TCP/IP model consists of four layers:
  1. Link (a.k.a. Network Interface Layer)
  2. Internet
  3. Transport
  4. Application
- Let's talk about each part and how data is sent…



THE TCP/IP MODEL – THE LINK LAYER

- The **Link/Network layer** is the lowest layer of the TCP/IP model.
- It defines how the data should be sent ***physically*** through the network.
- This layer is mainly responsible for the transmission of the data between two devices on the same network.
- **Examples**: NIC Cards, ethernet cables, wifi, etc.

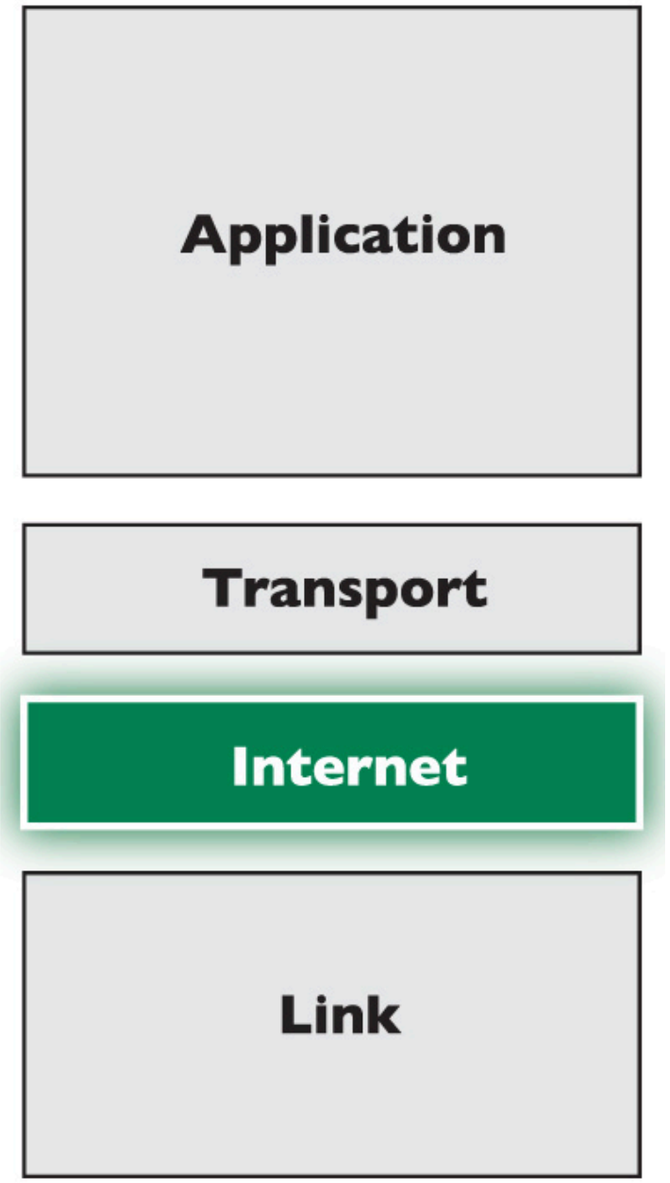


NIC CARD AND MAC ADDRESS

- A Media Access Control address (**MAC address**) is a hardware identifier that uniquely identifies each device on a network. Primarily, the manufacturer assigns it. They are often found on a device's **network interface controller** (NIC) **card**.
- An internet service provider (ISP) or network administrator assigns an IP address. An IP address helps identify a device connected to a network.

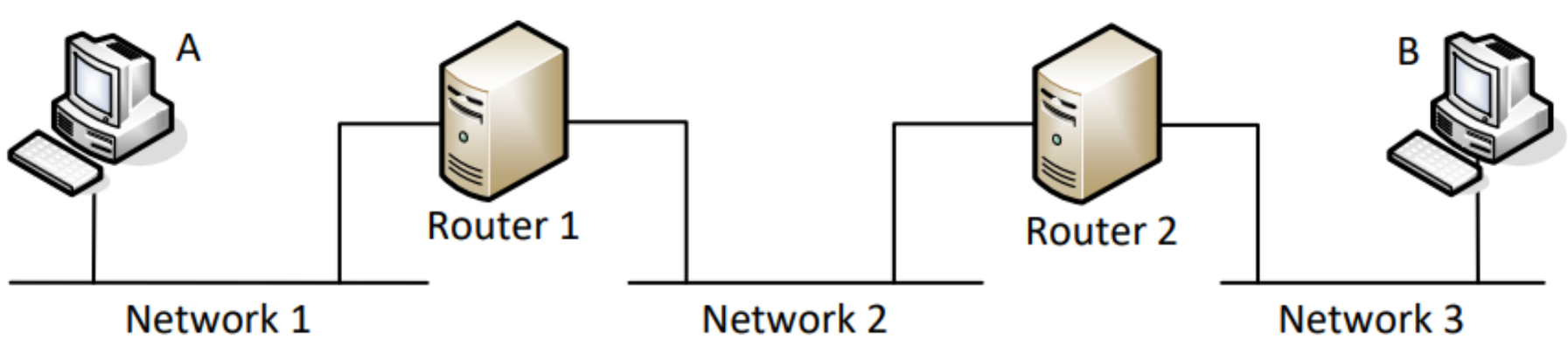
THE TCP/IP MODEL – THE INTERNET LAYER

- Any device or protocol that deals with pure IP packets – getting an IP packet to its destination – sits in the Internet layer.
- The internet layer also consists of **IP addressing** and **routers**. IP packets are also created at this layer.
- The Internet layer doesn't care about the type of data an IP packet carries, nor does it care whether the data gets there in good order or not.



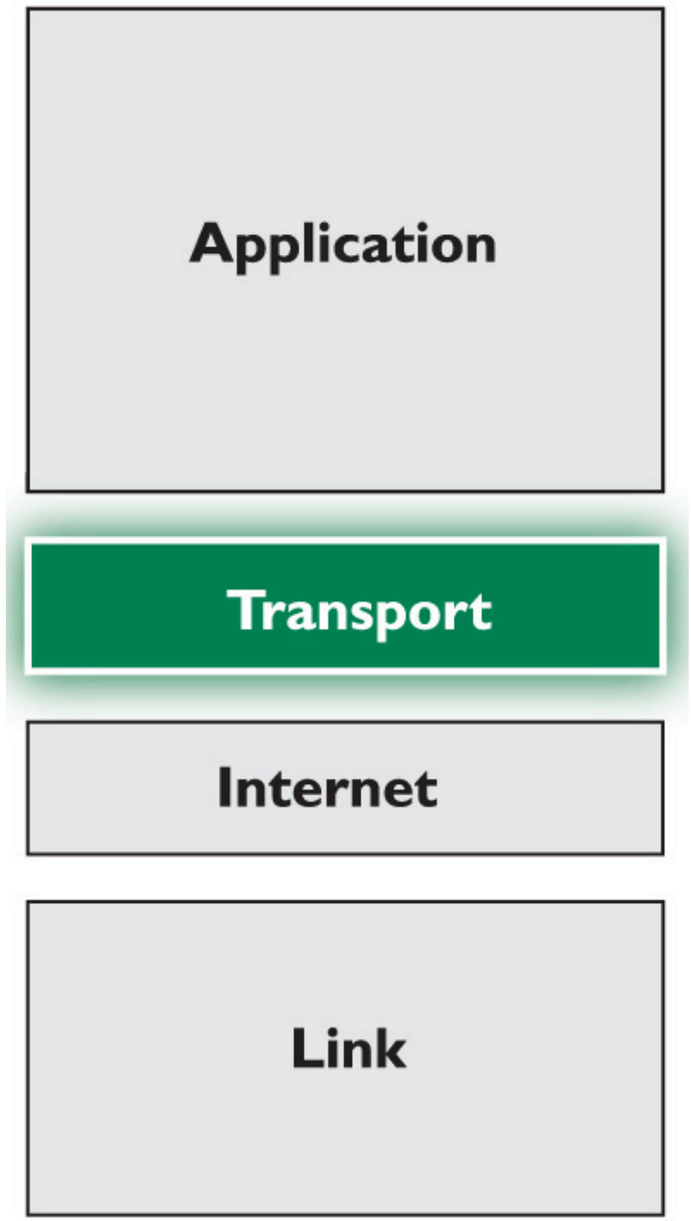
ROUTERS

- A **router** is a networking device that forwards data packets between computer networks.
- In order to direct packets effectively, a router uses an **internal routing table** – a list of paths to various network destinations.
- The router reads a packet's header to determine where it is going, then consults the routing table to figure out the most efficient path to that destination. It then forwards the packet to the next network in the path.



THE TCP/IP MODEL – THE TRANSPORT LAYER

- The **transport layer** is responsible for the reliability, flow control, and correction of data which is being sent over the network.
- The two protocols used in the transport layer are User Datagram protocol (**UDP**) and Transmission control protocol (**TCP**).



THE TCP/IP MODEL – THE TRANSPORT LAYER

- UDP is **connectionless** – sends data without first waiting to verify that the receiving system is ready.
- TCP is **connection-oriented** – verifies that there is a good connection (example, emails).

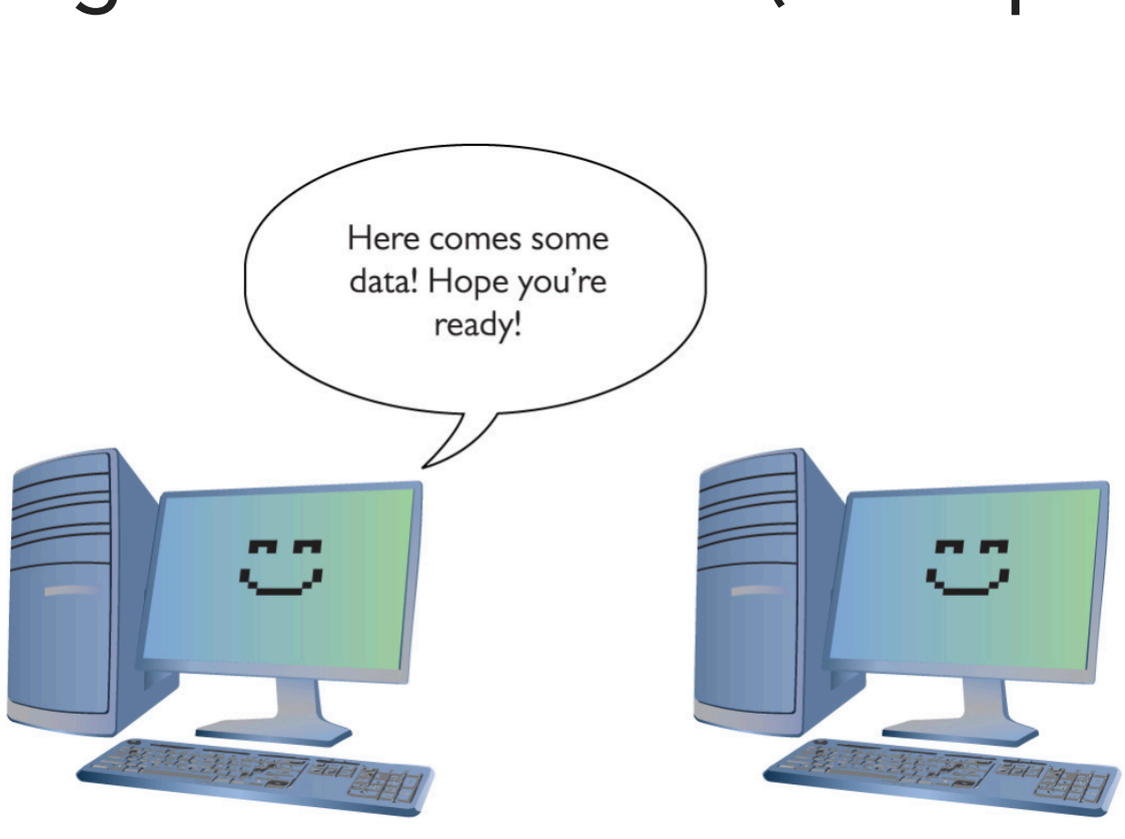


Figure 1-45 Connectionless communication

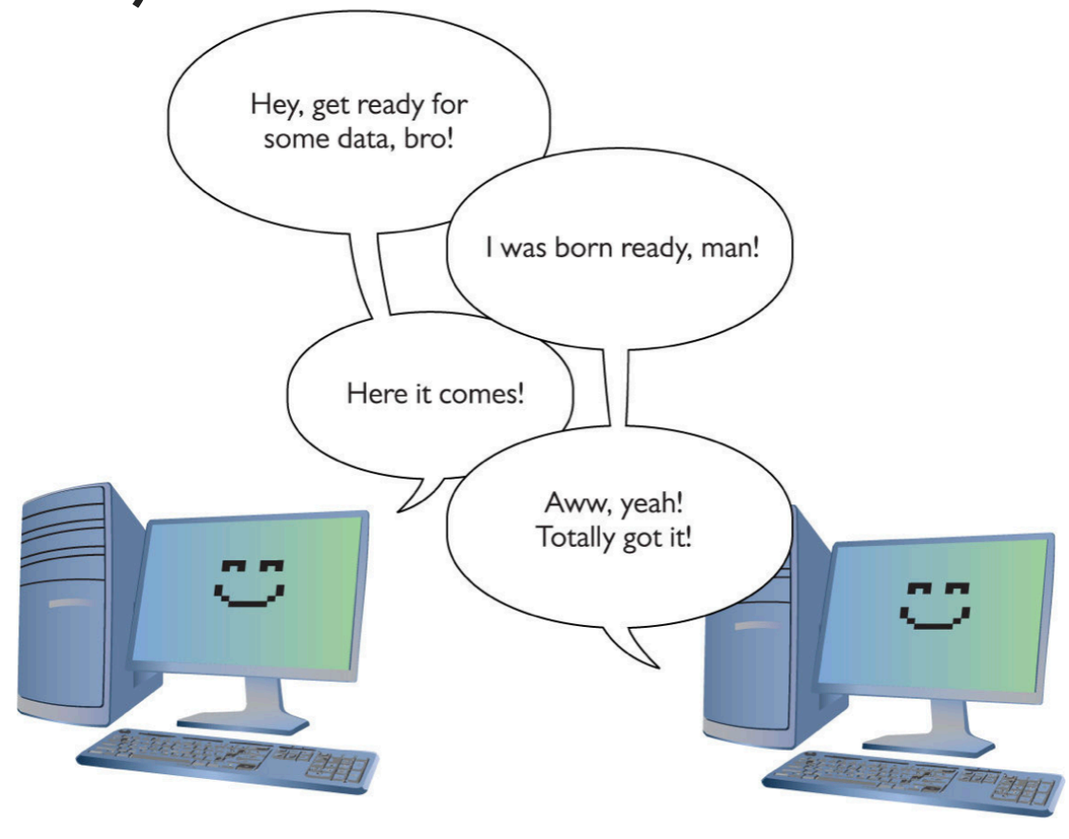


Figure 1-44 Connection between e-mail client and server

TCP 3 WAY HANDSHAKE

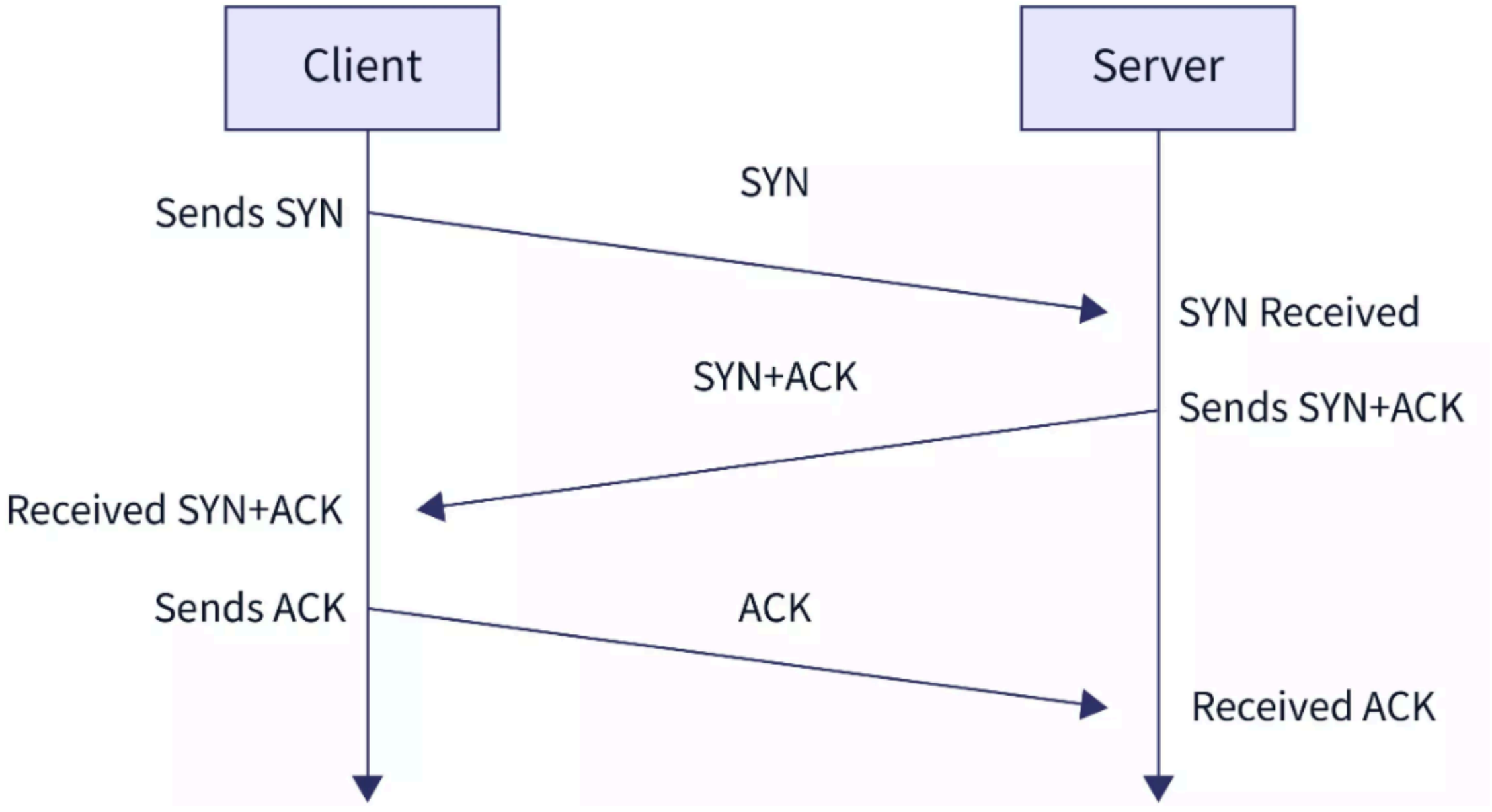
- The 3 way handshake is used for establishing and terminating the connection between the client and server (Remember, TCP is **connection-oriented!)**.
- There are three steps in the 3 way handshake (hence the name). Let's break it down...

TCP 3 WAY HANDSHAKE – ESTABLISHING CONNECTION SIMPLIFIED

1. The **client** sends the **SYN** (synchronize) message to the **server**.
2. The **server** responds *with* the **SYN** and the **ACK** (synchronize-acknowledge) message to the **client**.
3. The **client** sends the **ACK** (acknowledge) message to the **server**.

- Note: Simply put, the SYN and ACK are numbers of some kind.

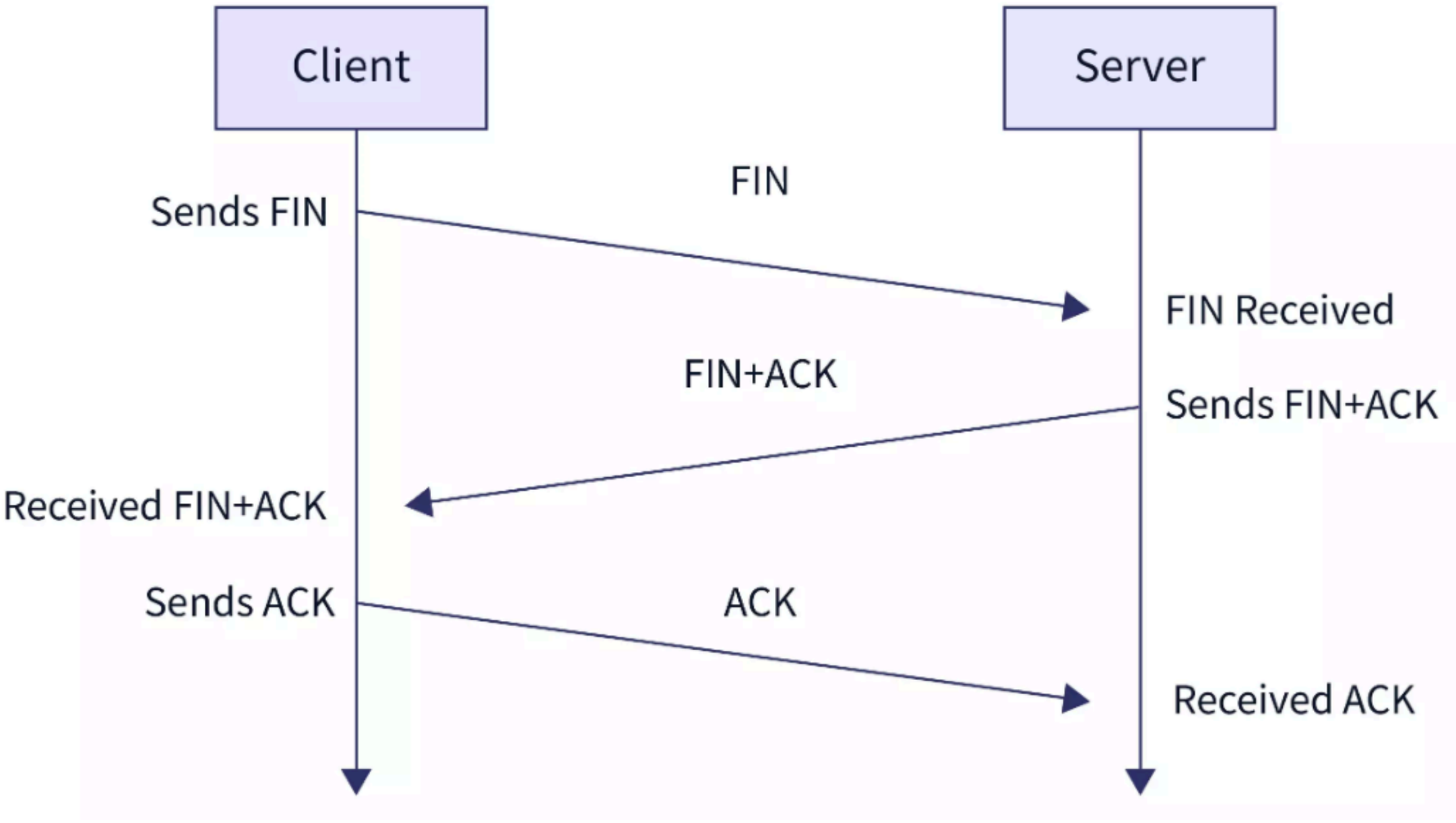
TCP 3 WAY HANDSHAKE – ESTABLISHING CONNECTION SIMPLIFIED



TCP 3 WAY HANDSHAKE – TERMINATING CONNECTION SIMPLIFIED

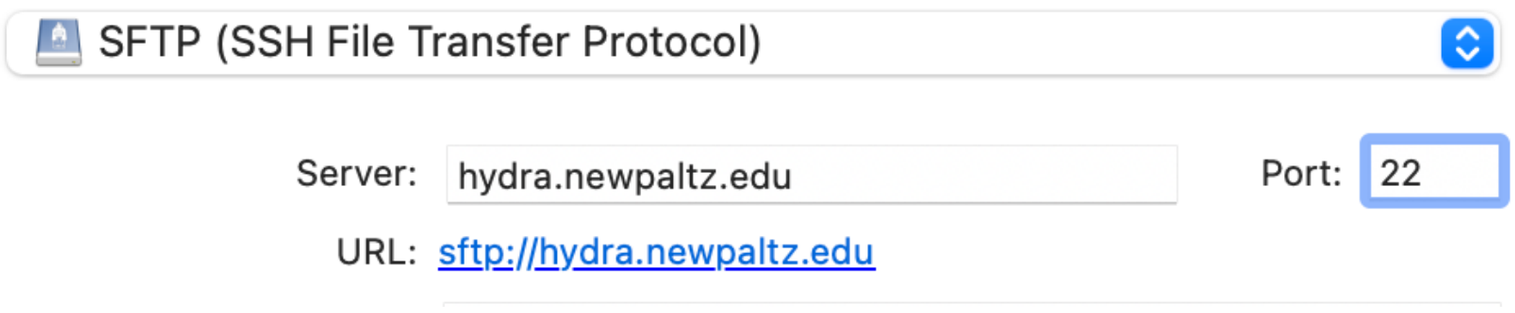
1. The **client** sends the **FIN** (finish) message to the **server**.
2. The **server** responds with the **FIN** and the **ACK** (finish-acknowledge) message to the **client**.
3. The **client** sends the **ACK** (acknowledge) message to the **server**.

TCP 3 WAY HANDSHAKE – TERMINATING CONNECTION SIMPLIFIED



PORTS

- A **port** is a virtual point where network connections start and end. TCP and UDP protocol will contain the port.
- Each port is associated with a specific process or service.
- Ports allow computers to easily differentiate between different kinds of traffic: emails go to a different port than webpages, for instance, even though both reach a computer over the same Internet connection.
- The use of ports helps computers understand what to do with the data they receive.



DIFFERENT PORT NUMBERS

There are 65,535 possible port numbers, although not all are in common use. Some of the most commonly used ports, along with their associated networking protocol, are:

- **Ports 20 and 21**: File Transfer Protocol (FTP). FTP is for transferring files between a client and a server.
- **Port 22**: Secure Shell (SSH).
- **Port 25**: Historically, Simple Mail Transfer Protocol (SMTP). SMTP is used for email.

DIFFERENT PORT NUMBERS CONTINUED

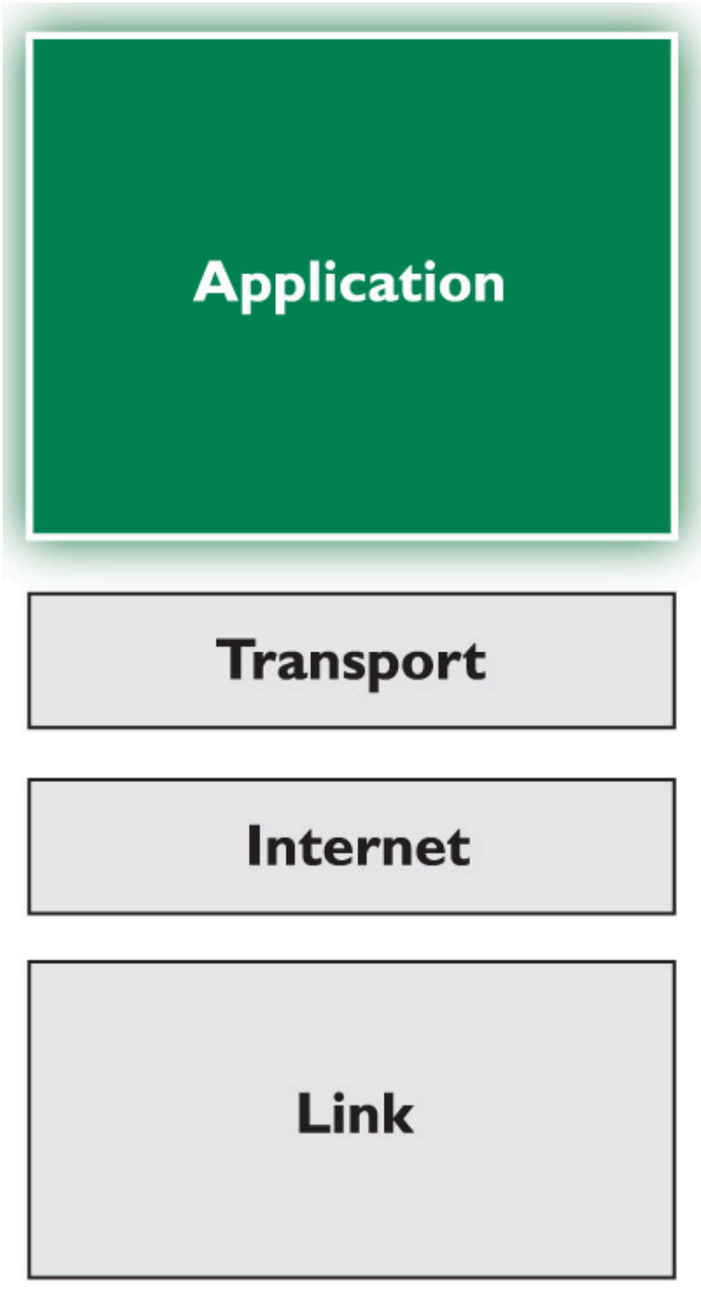
- **Port 53**: Domain Name System (DNS).
- **Port 80**: Hypertext Transfer Protocol (HTTP).
- **Port 443**: HTTP Secure (HTTPS). HTTPS is the secure and encrypted version of HTTP.
- **Port 587**: Modern, secure SMTP that uses encryption.
- **Port 3389**: Remote Desktop Protocol (RDP). RDP enables users to remotely connect to their desktop computers from another device.

SEE WHICH PORTS ARE OPEN ON SYSTEM

```
kaitlin@hydra:~$ sudo lsof -i -P -n
[sudo] password for kaitlin:
COMMAND      PID            USER   FD   TYPE  DEVICE SIZE/OFF NODE NAME
systemd-n   1081 systemd-network   21u  IPv4   23716      0t0  UDP 192.168.1.160:68
systemd-n   1081 systemd-network   24u  IPv6   23711      0t0  UDP [fe80::b645:6ff:fe
systemd-r   1084 systemd-resolve   13u  IPv4   31336      0t0  UDP 127.0.0.53:53
systemd-r   1084 systemd-resolve   14u  IPv4   31337      0t0  TCP 127.0.0.53:53 (LIS
sshd        1180            root    3u  IPv4   35626      0t0  TCP *:22 (LISTEN)
sshd        1180            root    4u  IPv6   35628      0t0  TCP *:22 (LISTEN)
mariadbd    1253           mysql   20u  IPv4   31987      0t0  TCP 127.0.0.1:3306 (LI
apache2    23202            root    4u  IPv6  225449      0t0  TCP *:80 (LISTEN)
apache2    23202            root    6u  IPv6  225453      0t0  TCP *:443 (LISTEN)
apache2   316757        www-data    4u  IPv6  225449      0t0  TCP *:80 (LISTEN)
apache2   316757        www-data    6u  IPv6  225453      0t0  TCP *:443 (LISTEN)
apache2   316758        www-data    4u  IPv6  225449      0t0  TCP *:80 (LISTEN)
apache2   316758        www-data    6u  IPv6  225453      0t0  TCP *:443 (LISTEN)
apache2   316759        www-data    4u  IPv6  225449      0t0  TCP *:80 (LISTEN)
apache2   316759        www-data    6u  IPv6  225453      0t0  TCP *:443 (LISTEN)
apache2   316760        www-data    4u  IPv6  225449      0t0  TCP *:80 (LISTEN)
```

THE TCP/IP MODEL – THE APPLICATION LAYER

- The **application layer** allows the user to interact with the application.
- When one application layer protocol wants to communicate with another application layer, it forwards its data to the transport layer.



THE TCP/IP MODEL – THE APPLICATION LAYER

The following are the main protocols used in the application layer:

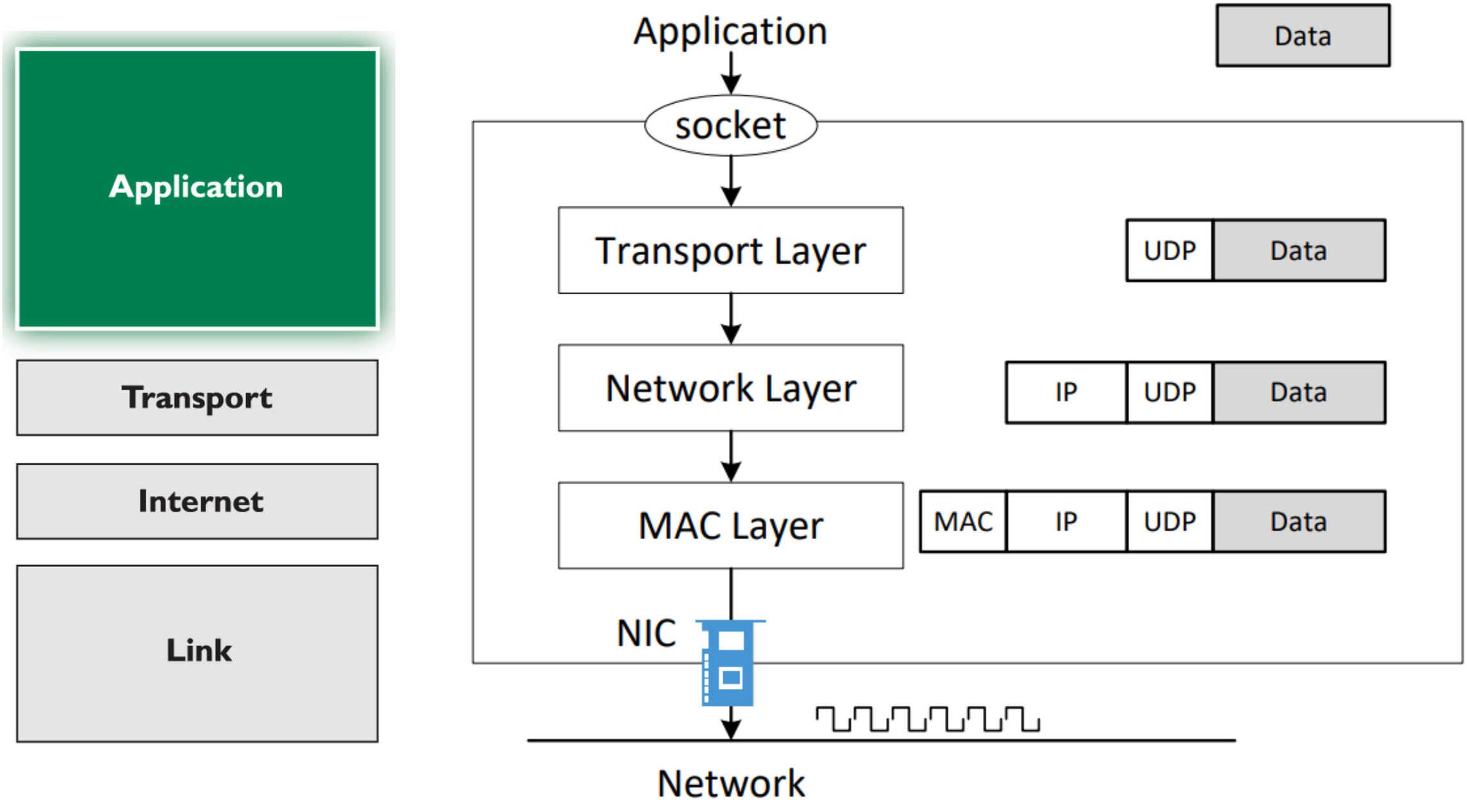
- **HTTP** – Hypertext transfer protocol. Allows us to access the data over the world wide web. It transfers the data in the form of plain text, audio, video.
- **SNMP** – Simple Network Management Protocol. Framework used for managing the devices on the internet by using the TCP/IP protocol suite.
- **SMTP** – Simple mail transfer protocol. Used to send the data to another e-mail address

THE TCP/IP MODEL – THE APPLICATION LAYER

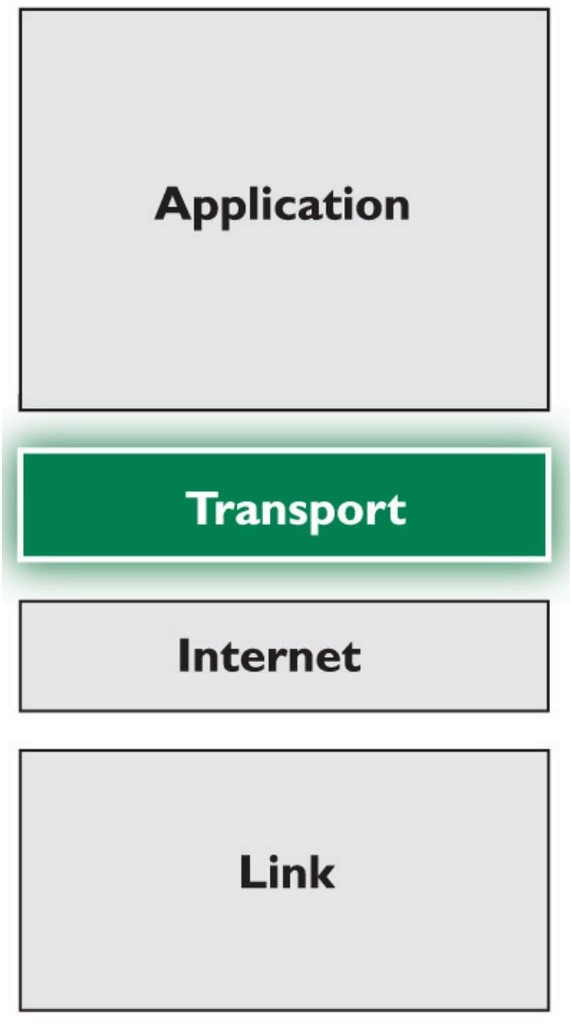
The following are the main protocols used in the application layer, continued:

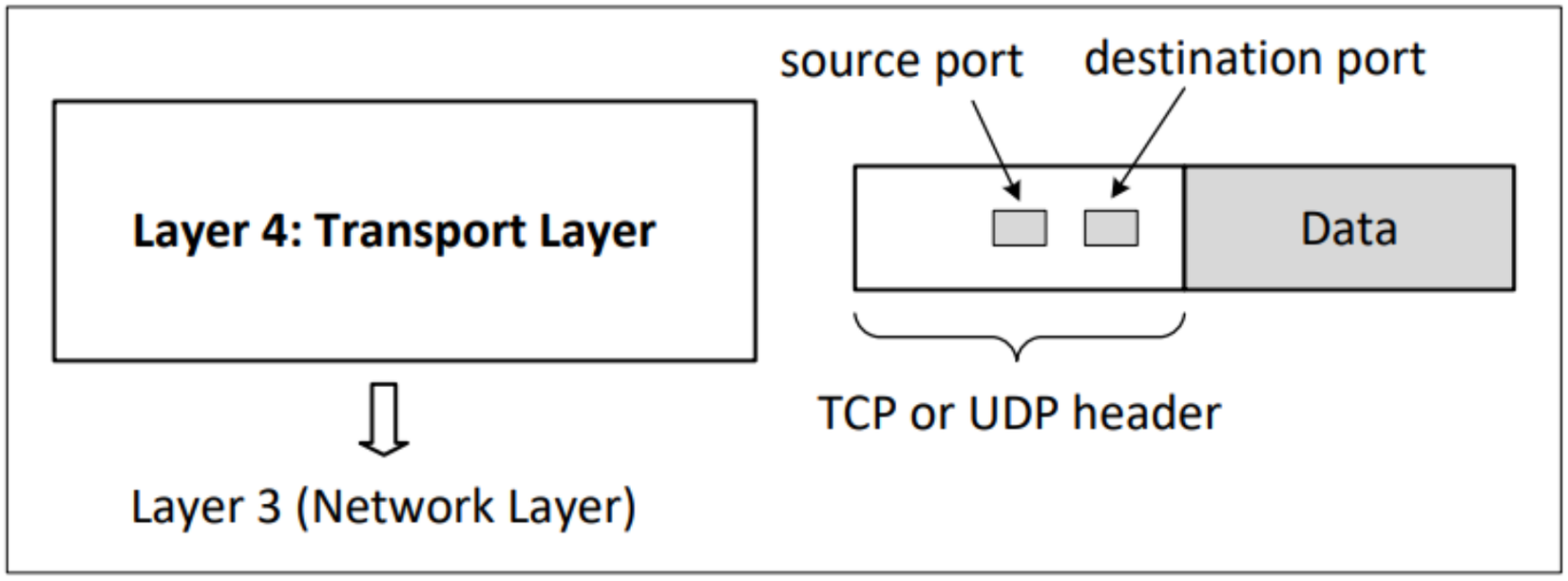
- **DNS** – Domain Name System. The system that maps a website name to an IP address.
- **TELNET** – Terminal Network. Establishes the connection between a local computer and remote computer.
- **FTP** – File Transfer Protocol. Standard internet protocol used for transmitting files from one computer to another.

HOW A PACKET IS CONSTRUCTED



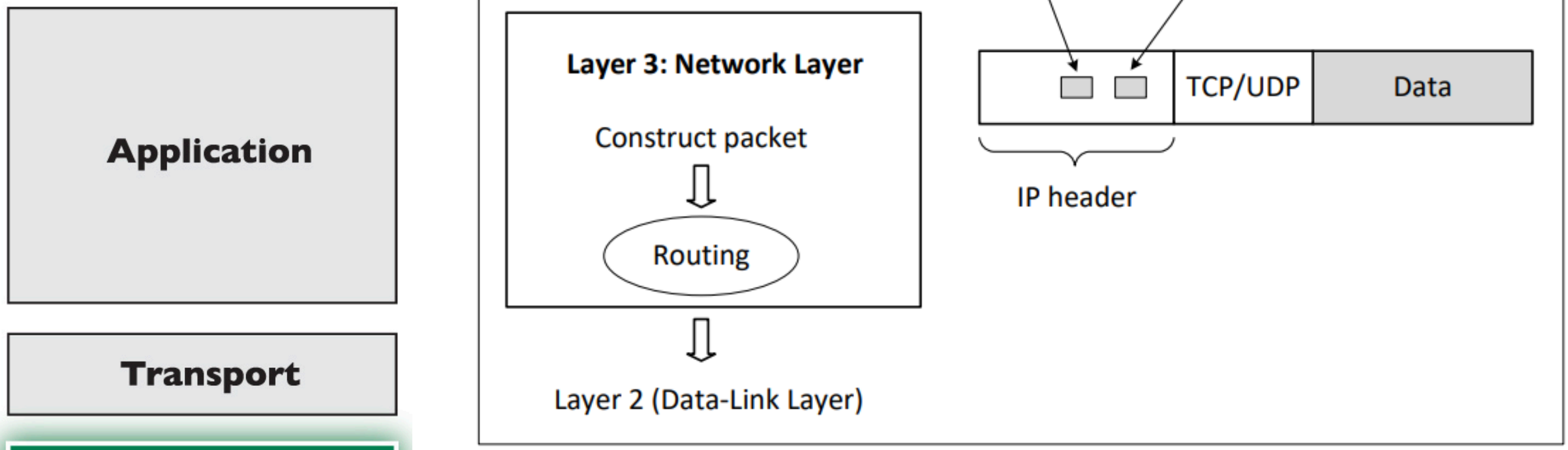
HOW A PACKET IS CONSTRUCTED





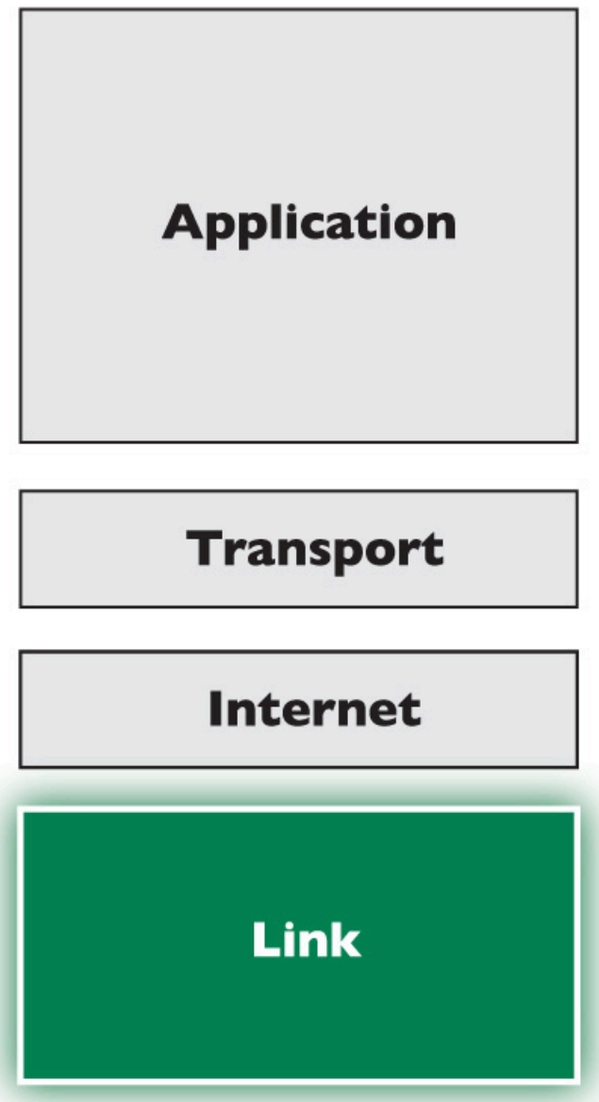

```
#!/usr/bin/python3

import socket

IP   = "127.0.0.1"
PORT = 9090
data = b'Hello, World!'

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.sendto(data, (IP, PORT))
```

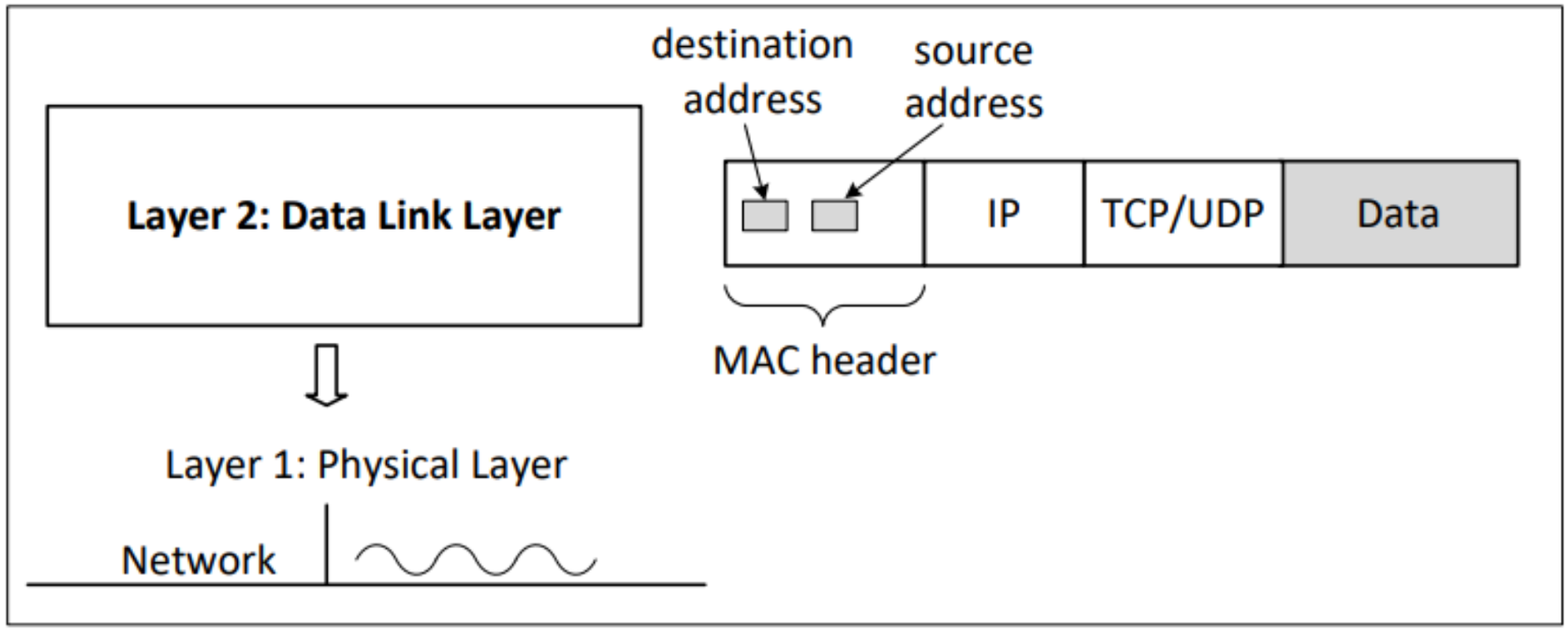
HOW A PACKET IS CONSTRUCTED




```python
#!/usr/bin/python3

import socket

IP   = "127.0.0.1"
PORT = 9090
data = b'Hello, World!'

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.sendto(data, (IP, PORT))
```

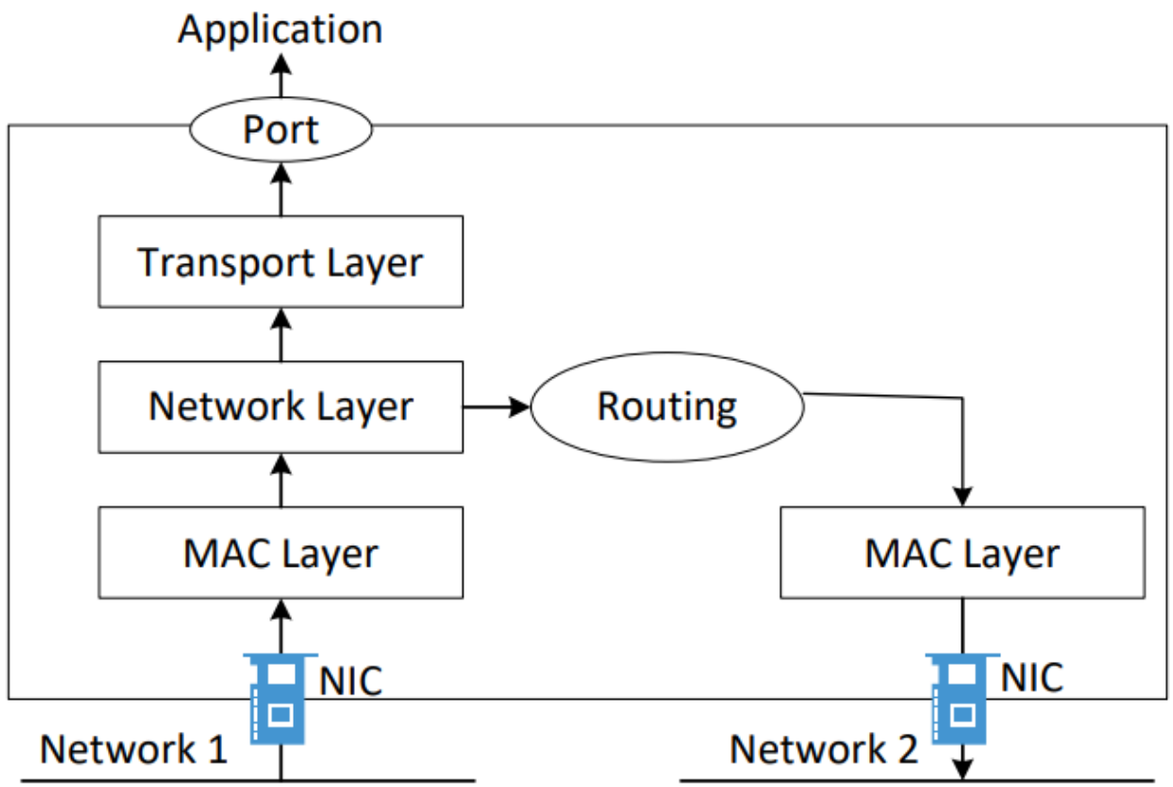
HOW A PACKET IS CONSTRUCTED

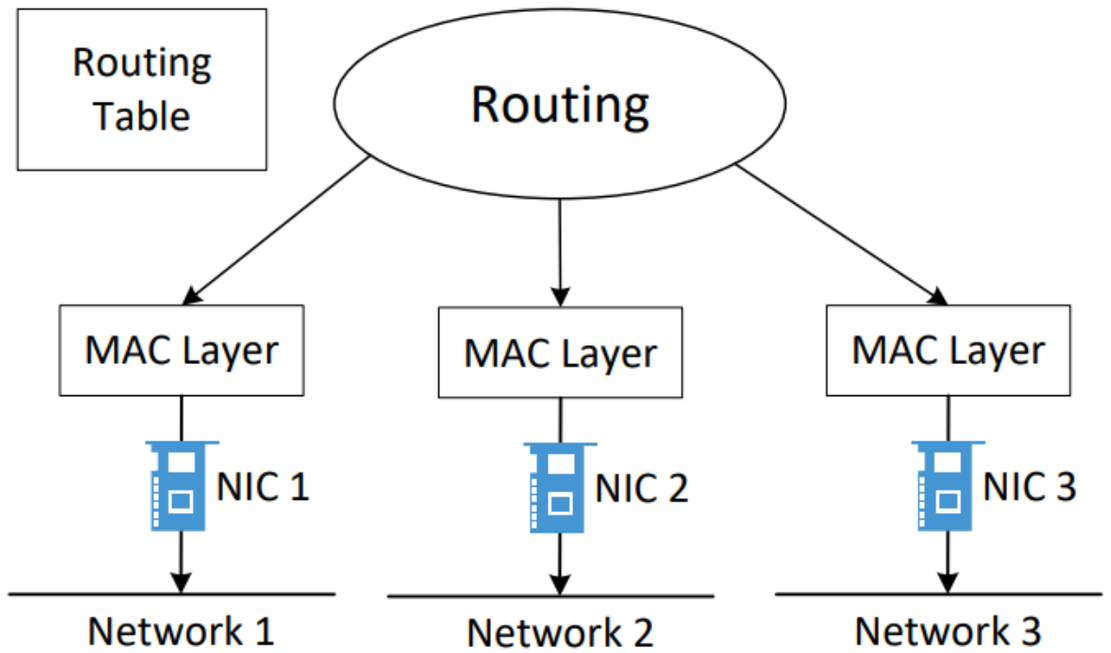





```python
#!/usr/bin/python3

import socket

IP   = "127.0.0.1"
PORT = 9090
data = b'Hello, World!'

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.sendto(data, (IP, PORT))
```

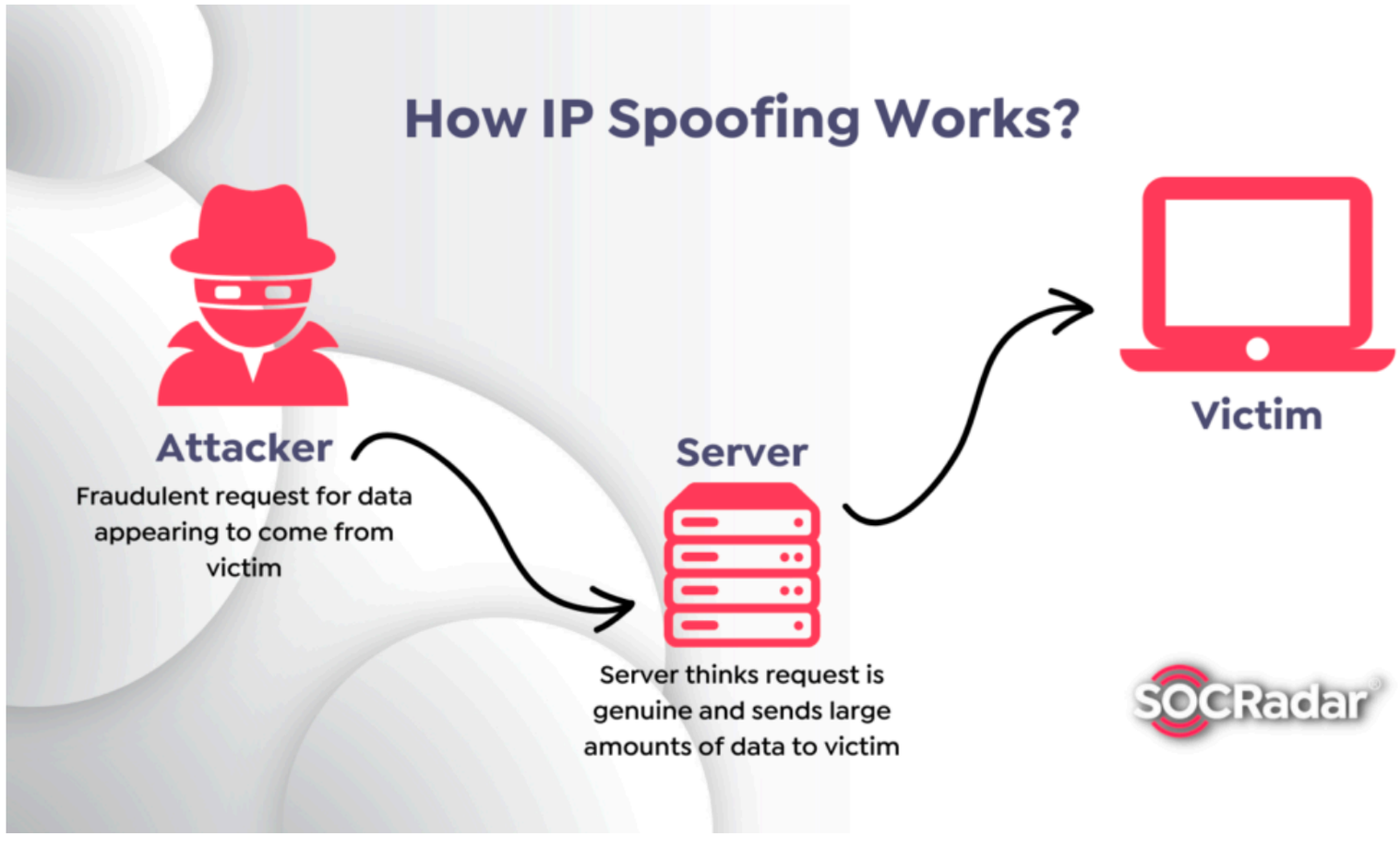
HOW PACKETS ARE RECEIVED





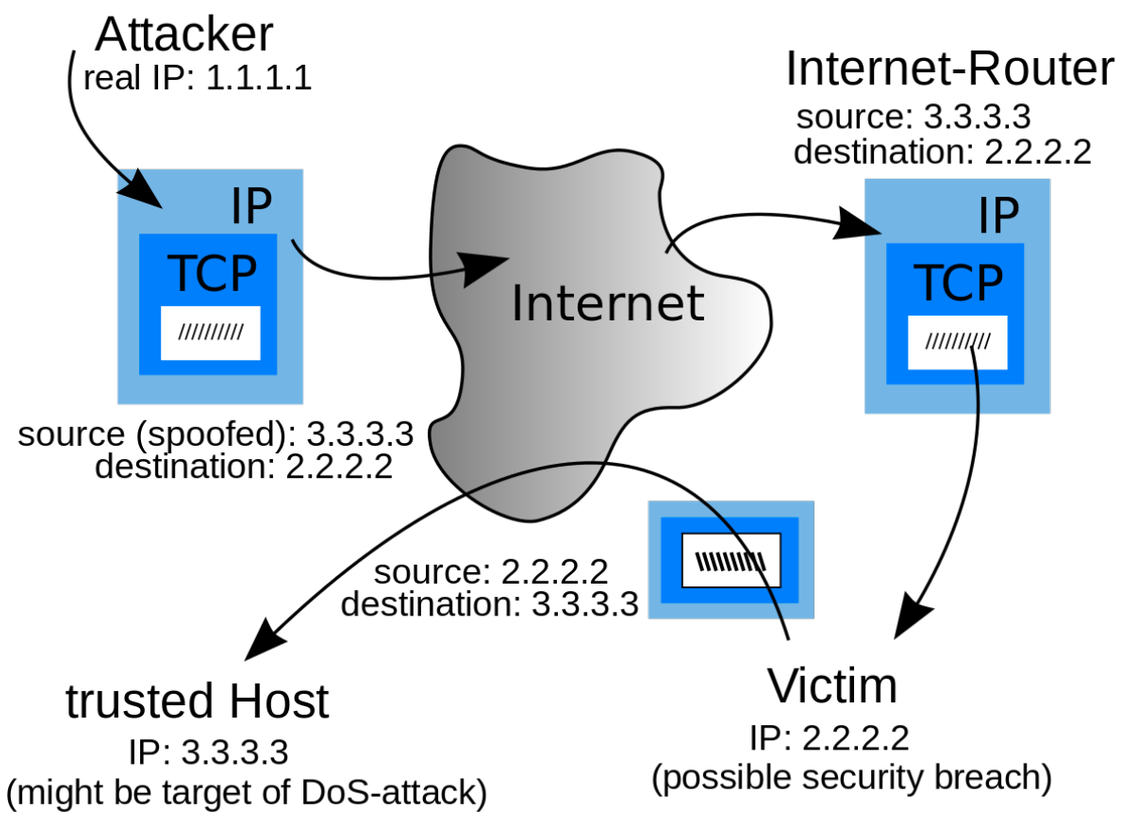
PACKET SPOOFING

- **Packet spoofing** (a.k.a. Internet Protocol (IP) spoofing) is a type of malicious attack where the threat actor hides the true source of IP packets to make it difficult to know where they came from.



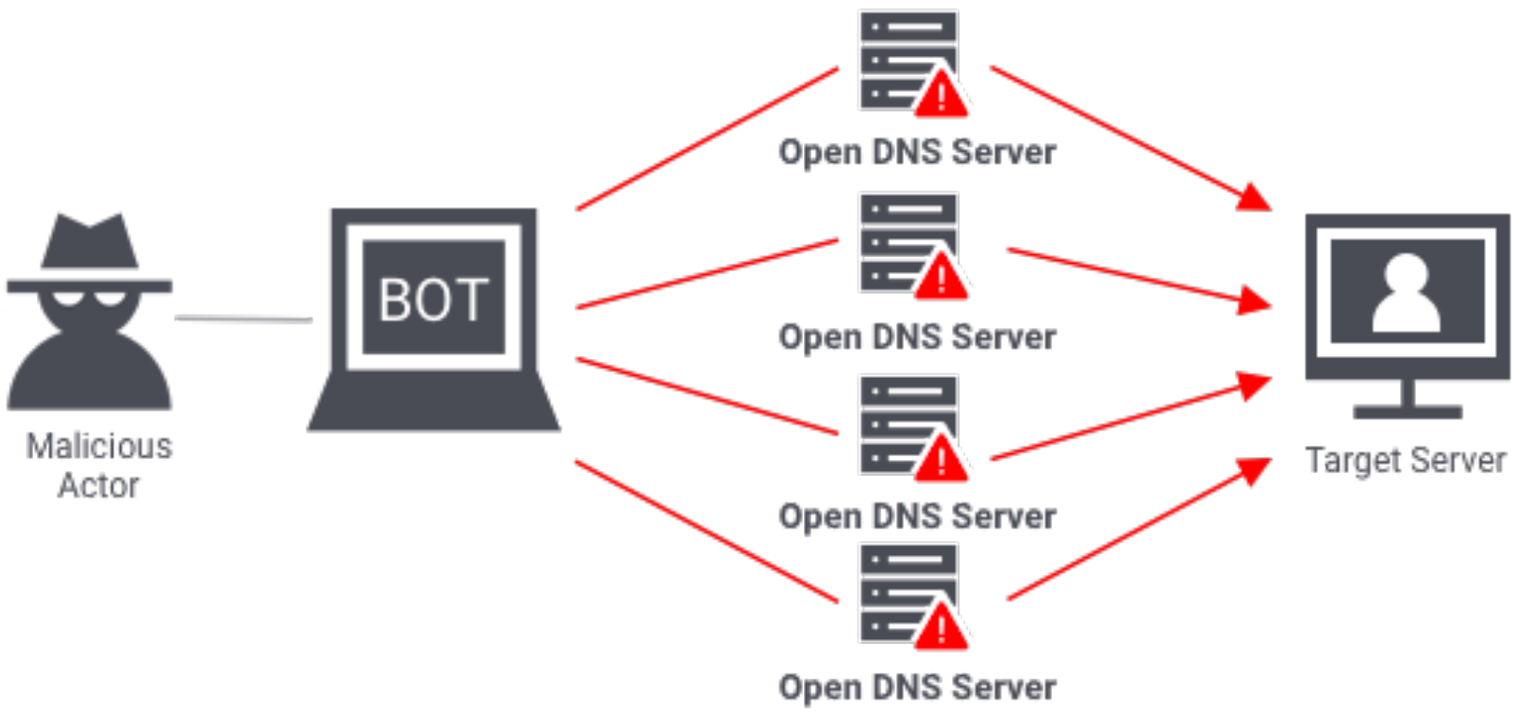
PACKET SPOOFING

- The attacker creates packets, changing the source IP address to impersonate a different computer system, disguise the sender's identity or both.
- The spoofed packet's header field for the source IP address contains an address that is different from the actual source IP address.



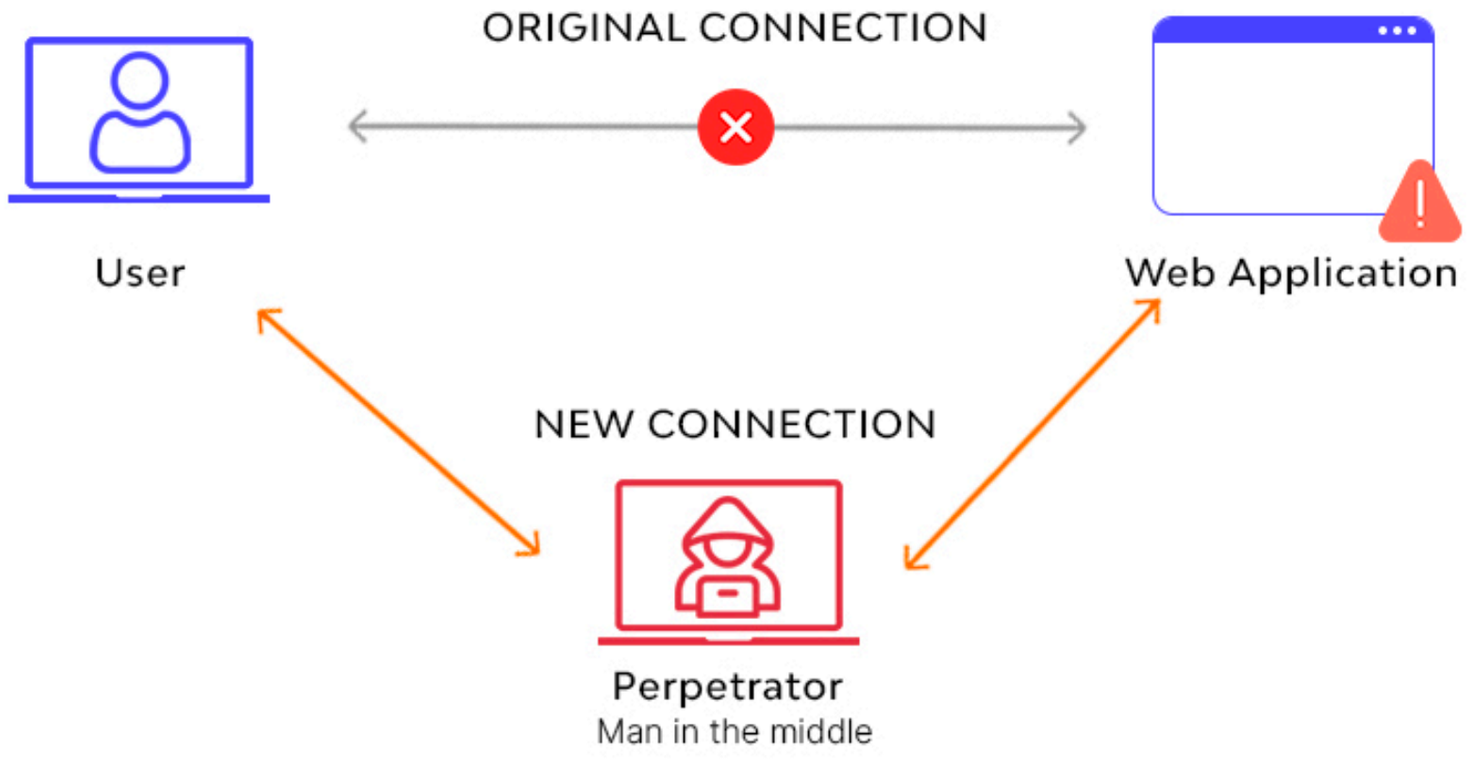
PACKET SPOOFING

- IP spoofing is a technique often used by attackers to launch **distributed denial of service** (DDoS) attacks and **man-in-the-middle** attacks against targeted devices or the surrounding infrastructures.
  - The goal of **DDoS** attacks is to overwhelm a target with traffic while hiding the identity of the malicious source, preventing mitigation efforts.



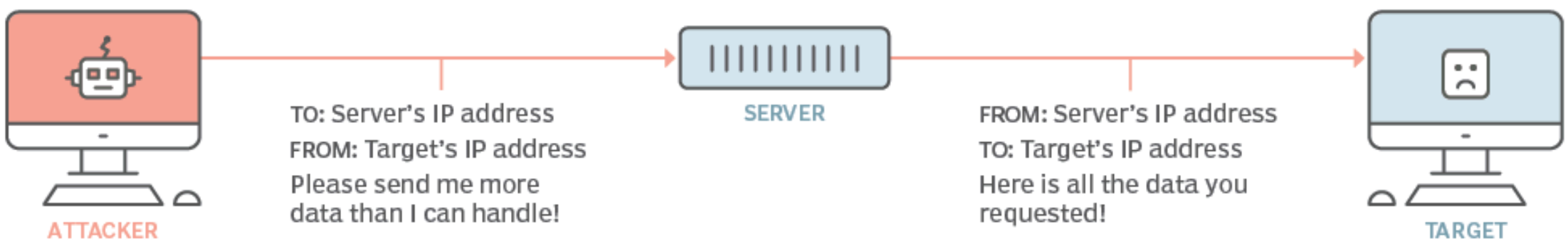
PACKET SPOOFING

- **man-in-the-middle** (MiTM) attack is when the attacker secretly intercepts and relays messages between two parties who believe they are communicating directly with each other. The attack is a type of eavesdropping in which the attacker intercepts and then controls the entire conversation.



PACKET SPOOFING — DDOS

- An attacker using IP spoofing pings the server repeatedly, using the target's IP address as the source. The server thinks the target is making legitimate requests and overwhelms the target with responses, causing it to malfunction.



PACKET SPOOFING

- Using spoofed IP addresses enables attackers to do the following:

    1. Keep authorities from discovering who they are and implicating them in the attack.
    2. Prevent targeted devices from sending alerts about attacks in which they are unwilling participants.
    3. Bypass security scripts, devices and services that blocklist IP addresses known to be sources of malicious traffic.

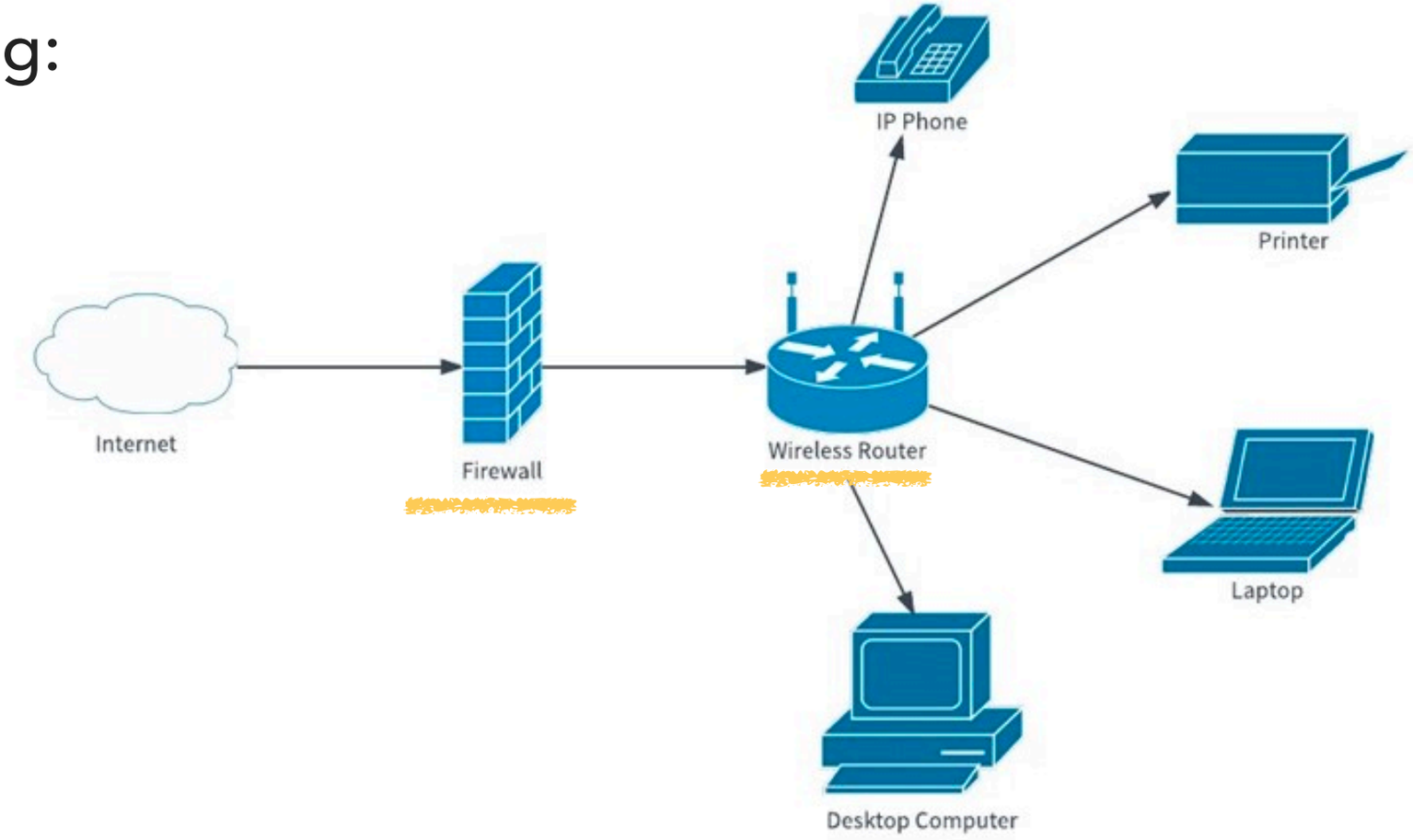
PACKET SPOOFING

- Attackers may generate fraudulent packet headers by falsifying and continuously randomizing the source address using a tool.
- They may also use the IP address of another existing device so that responses to the spoofed packet go there instead.
- To carry out IP spoofing, attackers need the following:
    1. A trusted IP address that the receiving device would permit to enter the network.
    2. The ability to intercept the packet and swap out the real IP header for the fraudulent one.

PACKET SPOOFING – HOW TO DETECT

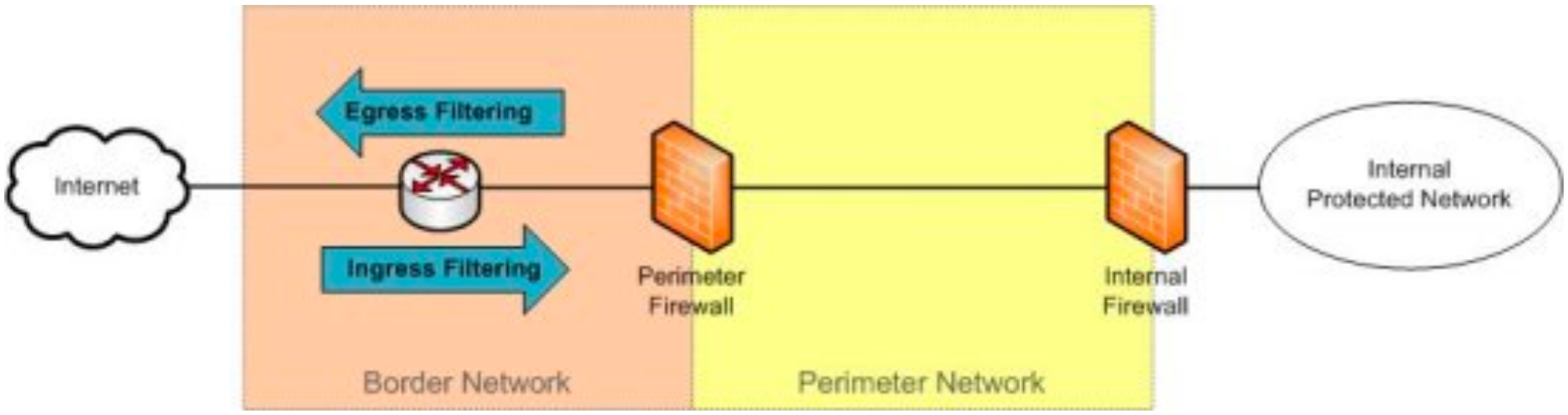
- **Packet filtering systems** can be used to detect packet spoofing.
- Packet filtering systems are often contained in **routers** and **firewalls**. They detect inconsistencies between the packet's IP address and desired IP addresses contained on access control lists (ACLs). Two types of packet filtering:

1. **Ingress filtering**
2. **Egress filtering**



PACKET SPOOFING – HOW TO DETECT

- **Ingress filtering** examines ***incoming*** packets to see if the source IP header matches a permitted source address. It rejects any that don't match or that display other suspicious behavior.
- **Egress filtering** examines ***outgoing*** IT scans for source IP addresses that don't match those on the company's network. This approach prevents insiders from launching an IP spoofing attack.



PACKET SPOOFING – HOW TO PROTECT AGAINST

- Use **strong verification** and authentication methods for all remote access. Do not authenticate devices and users solely based on IP address.
- Make an **ACL** of IP addresses.
- Use both ingress and egress packet **filtering**.
- Use **antivirus** and other security software that watches for suspicious network activity.
- Use IP-level **encryption protocols** to protect traffic going to and from the enterprise server. This approach keeps attackers from reading potential IP addresses to impersonate.
- Keep network software **updated**, and practice good patch management.
- Performing ongoing network **monitoring**.

WHAT IS PACKET SNIFFING

- A packet sniffer itself is passive. It observes messages being sent and received by applications and protocols running on your computer, but never sends packets itself.
- Similarly, received packets are never explicitly addressed to the packet sniffer. Instead, a packet sniffer receives a copy of packets that are sent/received from/by application and protocols executing on your machine.
- Wireshark is a very popular packet sniffing application.

WHAT IS PACKET SNIFFING

- The basic tool for observing the messages exchanged between executing protocol entities is called a **packet sniffer**. A packet sniffer captures ("sniffs") messages being sent/received from/by your computer.
- It will store and/or display the contents of the various **protocol** fields in these captured messages. A few Protocols (there are many more):
  - **TCP (Transmission Control Protocol)**: connection-based. used extensively by many internet applications, including the World Wide Web (WWW), email, File Transfer Protocol, Secure Shell, peer-to-peer file sharing…
  - **UDP (User Datagram Protocol)**: connectionless. Used in time-critical data transmissions such as DNS lookups, online gaming, and video streaming.
  - **ICMP (Internet Control Message Protocol)**: A network layer protocol used by routers, intermediary devices and hosts to communicate error information or updates to other routers, intermediary devices and hosts.

PACKET SNIFFING TOOLS

- **Tcpdump**
  - Command line
  - Good choice for containers (used in some of the labs)
- **Wireshark**
  - GUI
  - Good choices for the environment supporting GUI (not containers)
- **Scapy**
  - Implement your own sniffing tools

WIRESHARK

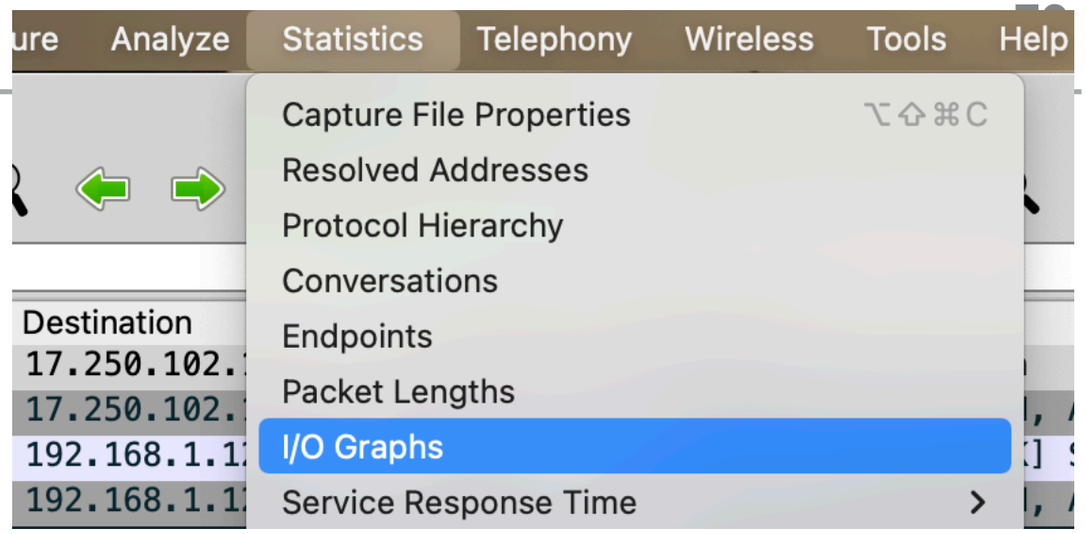
- **Wireshark** is a network protocol analyzer, or an application that captures packets from a network connection, such as from your computer to your home office or the internet.
- Packets are displayed at a granular level.
- Allows you to zoom in on the root cause of problems, assisting with network analysis and ultimately network security.

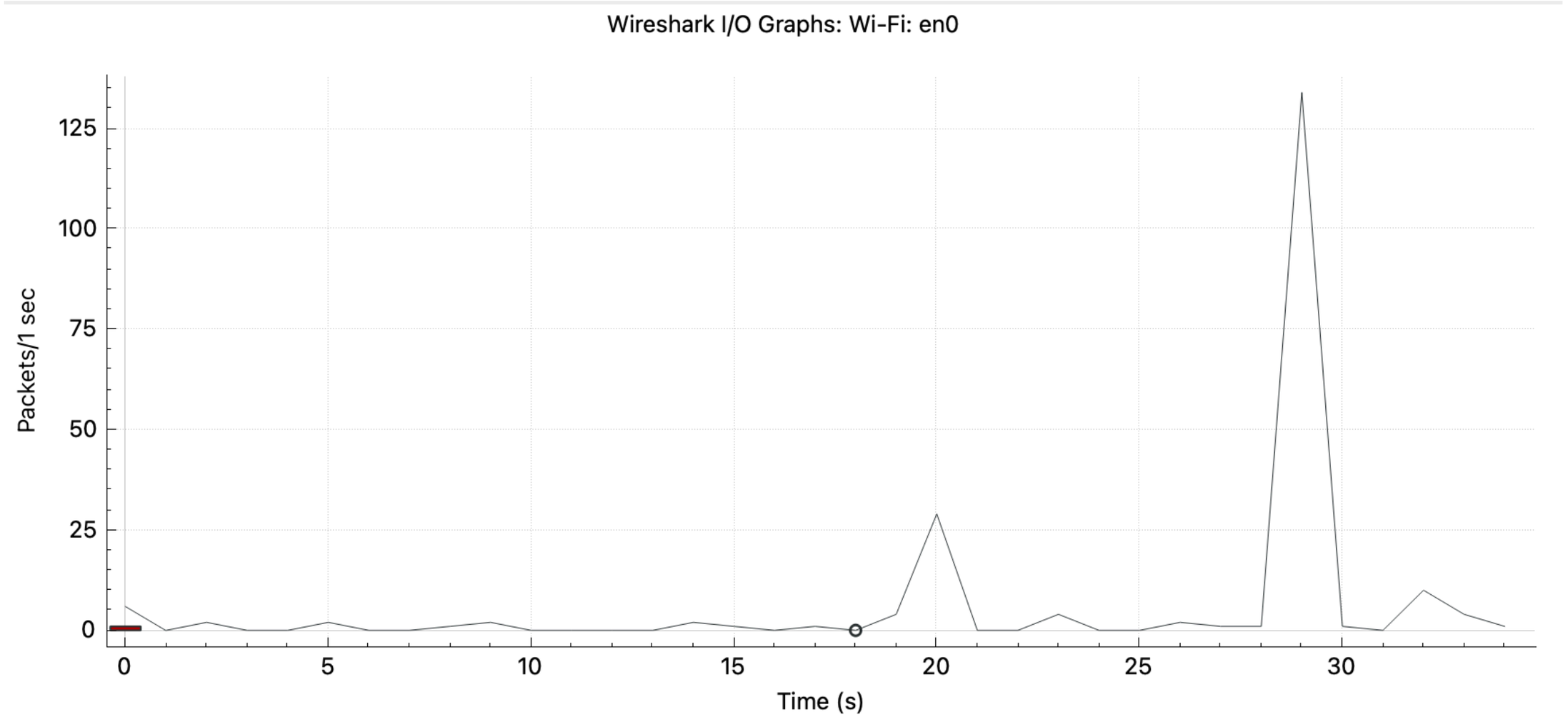
WIRESHARK – WHAT IT DOES

- **Packet Capture**: Wireshark listens to a network connection in real time and then grabs entire streams of traffic – quite possibly tens of thousands of packets at a time.
- **Filtering**: Wireshark is capable of slicing and dicing all of this random live data using filters. By applying a filter, you can obtain just the information you need to see.
- **Visualization**: Wireshark allows you to dive right into the very middle of a network packet. It also allows you to visualize entire conversations and network streams.

WIRESHARK – CAPTURE

- To view the input/output (I/O) statistics of an entire packet capture, *Statistics > I/O Graphs*:





Hover over the graph for details.

Enabled	Graph Name	Display Filter	Color	Style	Y Axis	Y Field	SMA Period	Y Axis Factc
☑	All Packets			Line	Packets		None	1
☑	TCP Errors	tcp.analysis.f...		Bar	Packets		None	1

WIRESHARK – CAPTURE

- To capture the amount of traffic generated between one system and another, go to *Statistics* and then select *Conversations;* you will see a summary of conversations between end points:

Wireshark · Conversations · Wi-Fi: en0

Ethernet · 12 | IPv4 · 16 | IPv6 · 2 | TCP · 2 | UDP · 22

Address A	Address B	Packets	Bytes	Packets A → B	Bytes A → B	Packets B → A	Bytes B → A	Rel Start	Duration
04:d9:f5:1c:cf:35	ff:ff:ff:ff:ff:ff	3	4.283 KiB	3	4.283 KiB	0	0 bytes	2.071182	25.3956
08:7c:39:19:5c:2b	01:00:5e:00:00:fb	1	60 bytes	1	60 bytes	0	0 bytes	28.693484	0.0000
1a:f6:b3:b8:9f:99	01:00:5e:00:00:fb	3	455 bytes	3	455 bytes	0	0 bytes	32.684850	1.1263
1a:f6:b3:b8:9f:99	33:33:00:00:00:fb	2	435 bytes	2	435 bytes	0	0 bytes	32.889512	0.9216
1a:f6:b3:b8:9f:99	ff:ff:ff:ff:ff:ff	6	642 bytes	6	642 bytes	0	0 bytes	32.583620	1.6407
1c:1b:0d:9f:6b:af	ff:ff:ff:ff:ff:ff	1	86 bytes	1	86 bytes	0	0 bytes	15.483988	0.0000
8c:79:f5:e3:f3:90	ff:ff:ff:ff:ff:ff	6	462 bytes	6	462 bytes	0	0 bytes	2.478807	30.0037
c4:91:0c:af:0c:c9	01:00:5e:00:00:fb	1	961 bytes	1	961 bytes	0	0 bytes	32.998509	0.0000
c4:91:0c:af:0c:c9	1a:f6:b3:b8:9f:99	1	401 bytes	1	401 bytes	0	0 bytes	32.998383	0.0000
c4:91:0c:af:0c:c9	33:33:00:00:00:fb	1	981 bytes	1	981 bytes	0	0 bytes	32.998605	0.0000
c4:91:0c:af:0c:c9	a4:cf:d2:b6:dd:11	182	120.255 KiB	115	109.828 KiB	67	10.427 KiB	0.000000	30.0200
d2:be:4b:3d:f0:b5	01:00:5e:00:00:fb	1	103 bytes	1	103 bytes	0	0 bytes	17.327189	0.0000

WIRESHARK – FILTER AND INSPECT PACKETS

- You can apply Wireshark filters in two ways:
  1. In the Display Filter window, at the top of the screen.
  2. By highlighting a packet (or a portion of a packet) and right-clicking on the packet.
- Wireshark filters use key phrases, such as the following:

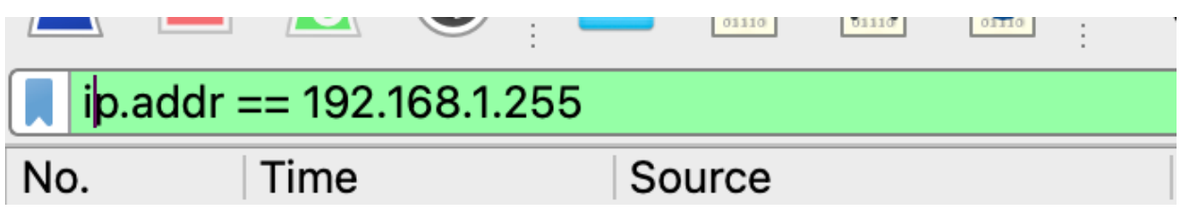
`ip.addr`	Specifies an IPv4 address
`ipv6.addr`	Specifies an IPv6 address
`src`	Source - where the packet came from
`dst`	Destination - where the packet is going

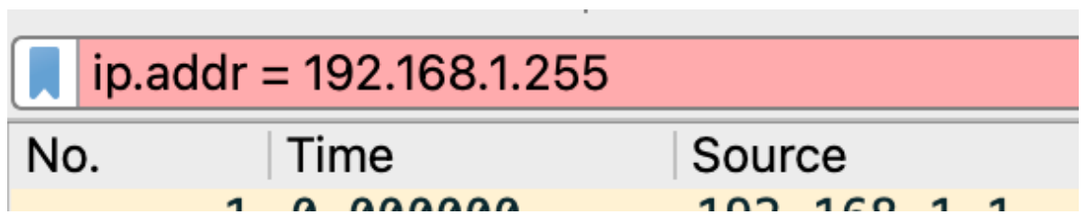
WIRESHARK – FILTER AND INSPECT PACKETS

- You can also use the following values:

&&	Means “and,” as in, “Choose the IP address of 192.168.2.1 and 192.168.2.2”
==	Means “equals,” as in “Choose only IP address 192.168.2.1”
!	Means “not,” as in, do not show a particular IP address or source port

- Valid filter rules are always colored green. If you make a mistake on a filter rule, the box will turn a vivid pink.





WIRESHARK – FILTER AND INSPECT PACKETS

- Filter example: Let's say you want to see packets that have only the IP address of *192.168.1.124* somewhere inside. You would create the following command line, and put it into the Filter window:

ip.addr == 192.168.1.124

No.	Time	Source	Destination	Protocol	Length	Info
5	7.885444	142.250.65.234	192.168.1.124	UDP	74	443 → 65418 Len=32
6	7.985669	192.168.1.124	142.250.65.234	UDP	75	65418 → 443 Len=33
10	10.921008	192.168.1.124	192.168.1.187	MDNS	401	Standard query response 0x0000 PTR
18	15.939397	192.168.1.124	172.253.122.188	TCP	54	58630 → 5228 [ACK] Seq=1 Ack=1 Win=
19	15.976426	172.253.122.188	192.168.1.124	TCP	66	[TCP ACKed unseen segment] 5228 → 5
21	17.297078	192.168.1.124	34.120.116.101	QUIC	1292	Initial, DCID=2097726be41c5b2f, PKN
22	17.335581	34.120.116.101	192.168.1.124	QUIC	1292	Protected Payload (KP0)
23	17.336929	192.168.1.124	34.120.116.101	QUIC	1292	Handshake, DCID=e097726be41c5b2f
24	17.336981	192.168.1.124	34.120.116.101	QUIC	203	Protected Payload (KP0), DCID=e0977
25	17.337333	192.168.1.124	34.120.116.101	QUIC	805	Protected Payload (KP0), DCID=e0977
26	17.363172	34.120.116.101	192.168.1.124	QUIC	1292	Protected Payload (KP0)
27	17.363173	34.120.116.101	192.168.1.124	QUIC	162	Protected Payload (KP0)
28	17.363174	34.120.116.101	192.168.1.124	QUIC	69	Protected Payload (KP0)

WIRESHARK – FILTER AND INSPECT PACKETS

- At the bottom, you can click around to see the Packet contents. This is mostly out of scope for the class, however, if you are interested in Networking, knowing how to use and interpret Wireshark is an excellent skill!

```
> Frame 2555: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface en0, id 0
> Ethernet II, Src: Apple_af:0c:c9 (c4:91:0c:af:0c:c9), Dst: UbeeInte_b6:dd:11 (a4:cf:d2:b6:dd:11)
> Internet Protocol Version 4, Src: 192.168.1.124, Dst: 34.197.167.79
v Transmission Control Protocol, Src Port: 55353, Dst Port: 443, Seq: 0, Len: 0
    Source Port: 55353
    Destination Port: 443
    [Stream index: 14]
    [Conversation completeness: Complete, WITH_DATA (63)]
    [TCP Segment Len: 0]
    Sequence Number: 0    (relative sequence number)
    Sequence Number (raw): 3823055263
    [Next Sequence Number: 1    (relative sequence number)]
    Acknowledgment Number: 0
    Acknowledgment number (raw): 0
    1011 .... = Header Length: 44 bytes (11)
  > Flags: 0x002 (SYN)
    Window: 65535
    [Calculated window size: 65535]
    Checksum: 0x54fb [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  > Options: (24 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operatior
  > [Timestamps]
```

WIRESHARK – FILTER AND INSPECT PACKETS

- Additional Filters:

`tcp.port==8080`	Filters packets to show a port of your own choosing – in this case, port 8080
`!(ip.src == 162.248.16.53)`	Shows all packets except those originating from 162.248.16.53
`!(ipv6.dst == 2607:f8b0:400a:15::b)`	Shows all packets except those going to the IPv6 address of 2607:f8b0:400a:15::b
`ip.addr == 192.168.4.1 && ip.addr == 192.168.4.2`	Shows both 192.168.4.1 and 192.168.4.2
`http.request`	Shows only http requests – useful when troubleshooting or visualizing web traffic

IF INTERESTED – TESTING OUT WIRESHARK

- Download Wireshark: https://www.wireshark.org/
- Go through the lecture slides and play around with Wireshark. Click on an IP address, try filtering, create a graph, etc.
- What kind of protocols do you see? What do the packet contents contain?

ASSIGNMENT

Complete the following labs. Take a screenshot of the completed labs with your avatar and green checks included. For your submission, add each screenshot to one file and submit as a PDF.

- Complete ALL tasks: https://tryhackme.com/r/room/whatisnetworking
- Only need to complete tasks 1 through 4: https://tryhackme.com/r/room/introtonetworking