

see [ZCC] Part II  
or { [PF6.1], [PF6.2],[PF6.4],[PF6.6],[PF6.9]}

# Lecture 5: Network Security

5.1 Background

5.2 Attack on Naming

5.3 Denial of Service attack

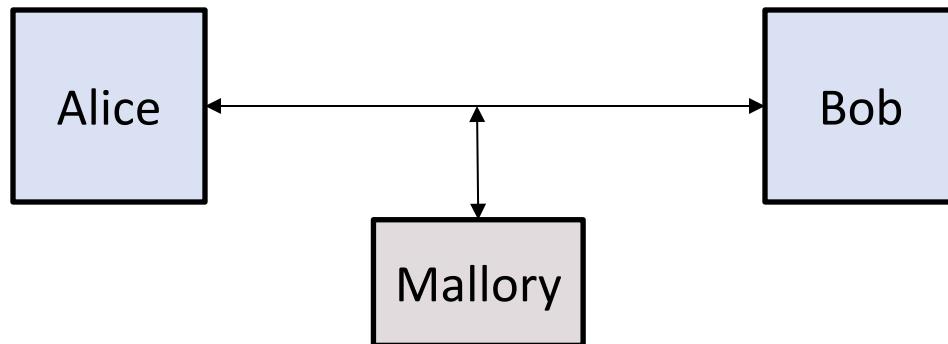
5.4 Useful tools

5.5 Protection: Securing the Channel using Cryptography

5.6 Protection: Firewall, Intrusion Detection System

5.7 Protection: Management

- Previous lectures illustrate that, using cryptographic techniques + PKI, we can secure the public communication channel between Alice and Bob, so as to achieve ***authenticity & confidentiality***.



- Mission accomplished? There are many other issues:
  1. Other security requirements (e.g. availability, anonymity, access control);
  2. Other information, in particular networking information, to be protected (e.g. DNS, routing information);
  3. Access control, monitoring and other management issue (e.g. firewall) ;
  4. etc

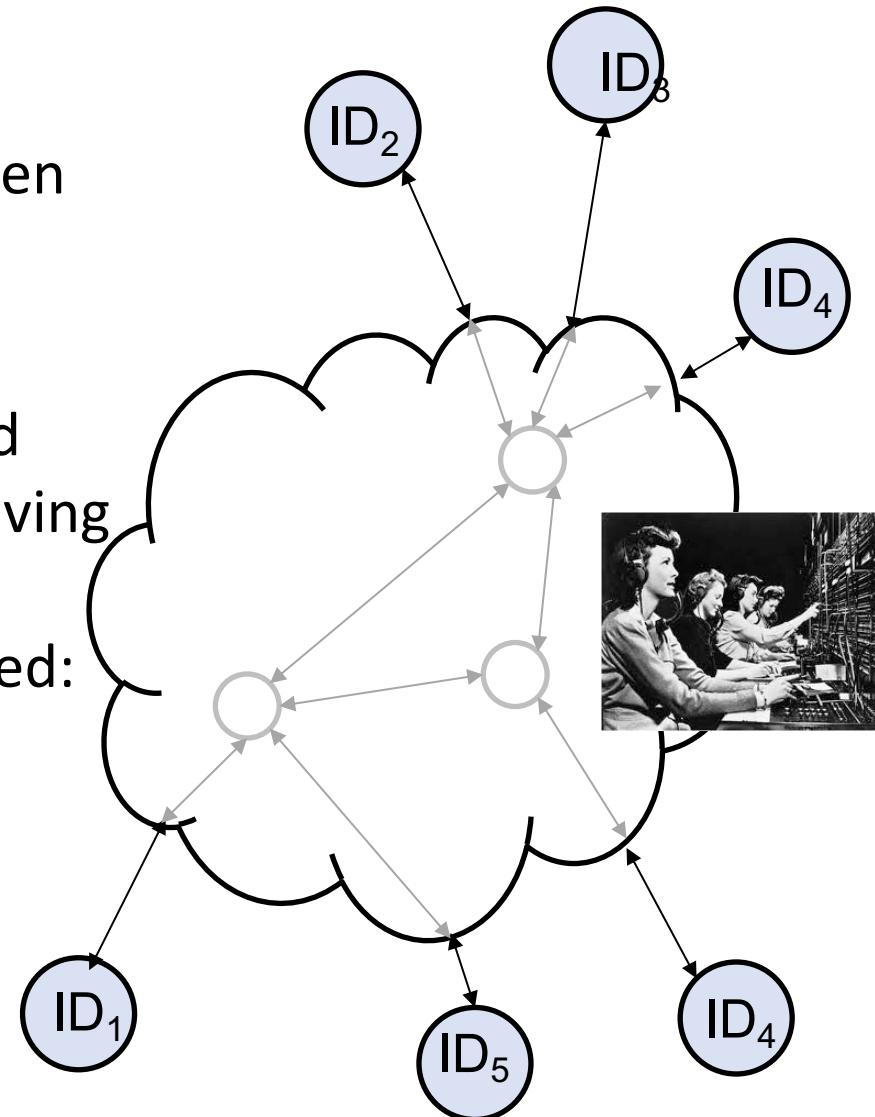
# 5.1 Background on Networking

Important notions for this module:

- Layers
- Naming
- Ports

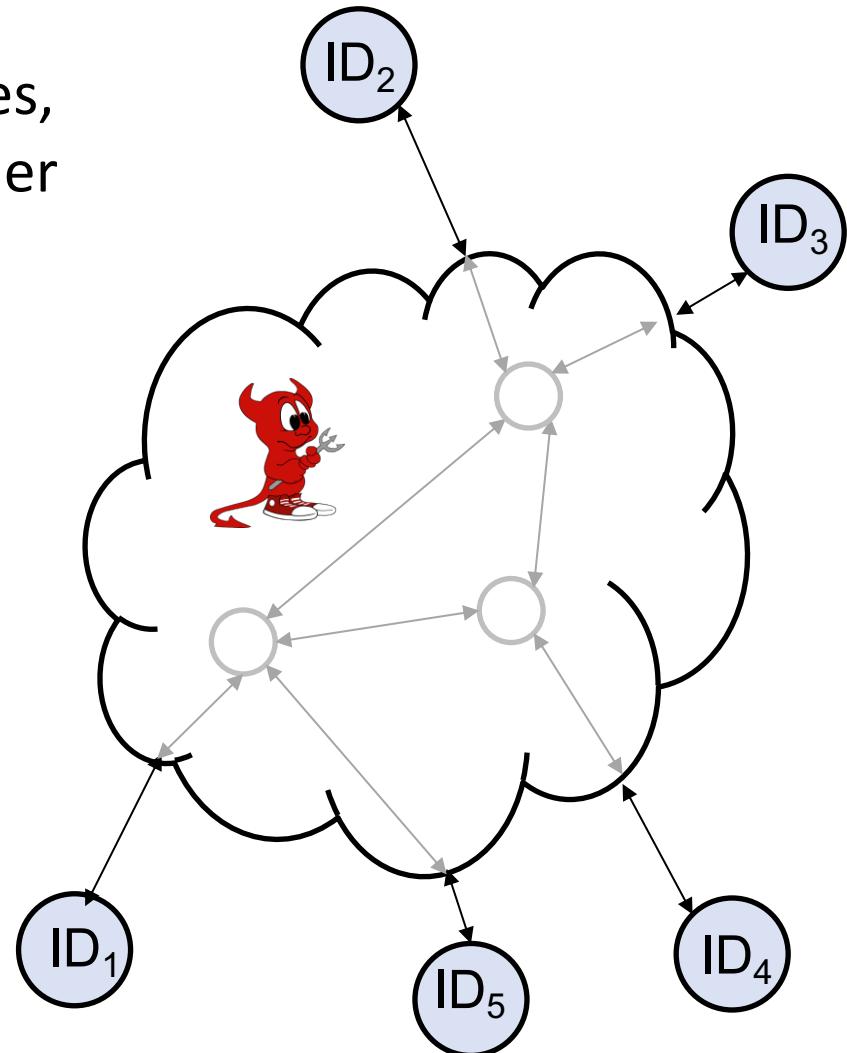
# Computer Network

- Computer Network establishes communicating connection between entities.
- To share networking resources and enhance robustness, instead of having dedicated line between any two nodes, **packet switching** is deployed:
  1. Messages route via multiple switches and routers.
  2. Messages are broken into “packets/frames”.



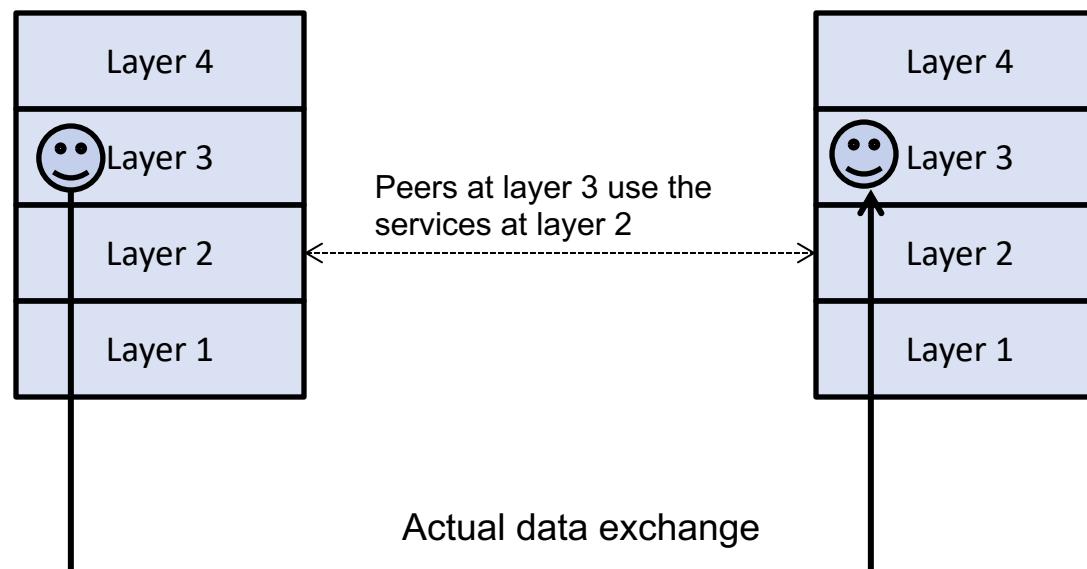
# Network Security

- While networking technologies focus on how to deliver messages, network security focus on another aspect. Under the presences of attackers:
  - Preserving confidentiality and authenticity of the messages and various entities;
  - Maintaining availability of the network;
  - Controlling and monitoring the traffic flow.



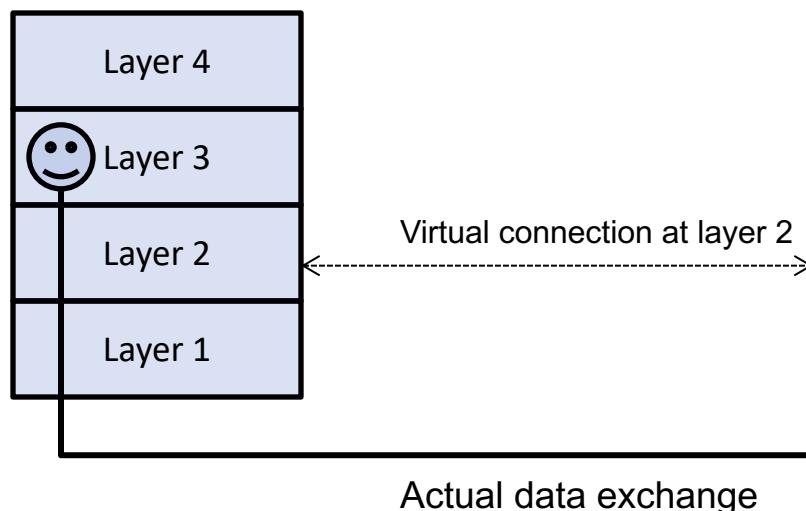
# Network layers

- Network protocols are abstracted as layers.
- Conceptually, layer (N-1) provides services for entities in layer N. The *peer entities* in the N layer communicate by executing protocols provided by layer (N-1).



# Network layers

- In other words, the layer N services/protocols are built on top of services provided by layer (N-1). We can view the services at (N-1) as a virtual connection.
- In turn, the layer N can provide services for (N+1).
- The actual data sent by layer N are often divided into smaller units, and each unit is called a PDU (protocol data unit).



A hypothetical example:

A Layer 3 services (called **handshake**) is designed as follow,

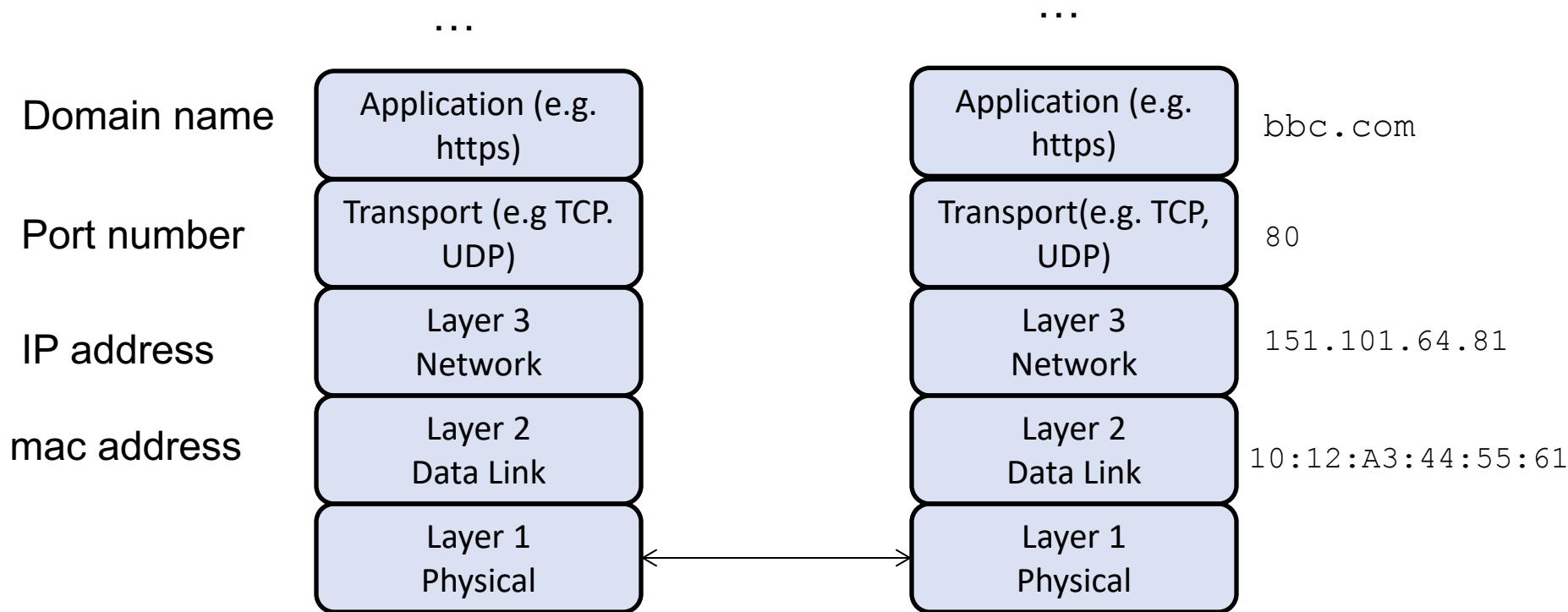
**Handshake** (A,B):

- (1) A → B: “hello”
- (2) A ← B: certificate of B.

The connection “A → B” is provided by Layer 2. The “virtual connection” at layer 2 sends the message “hello” from A to B in step (1), and sends the certificate in step (2).

Now, another entity in Layer 4 can use the layer 3’s service *handshake*.

# Internet layers:



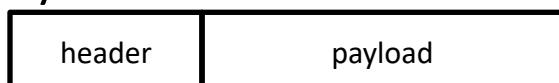
While the Open Systems Interconnection model (OSI model) gives 7 layers. In practice and this module, we are interested in the above layering.

# Naming

- A node in a network has different names in different layers.
- Four types of names are of particular concern in here.
  - Domain Name:                   bbc.com
  - Port Number:                   80
  - IP-address:                   151.101.64.81
  - mac-address:                   10:12:A3:44:55:61

# Datagram, Packet, Frame (PDU)

- When a layer-N service is invoked to send a message, the protocol in layer N might divide/merge the message into smaller units (i.e. the PDU).
  - In transport layer (layer 4), each portion is called a “datagram”.
  - In network layer (layer 3), it is called a “packet”.
  - In datalink layer (layer 2), it is called a “frame”.
- A packet/frame consists of two parts:
  - Header: this contains at least two pieces of information
    - The source (src) address, i.e. the sender’s address;
    - The destination (dest) address, i.e. the receiver’s address;
  - The payload: the data to be sent.

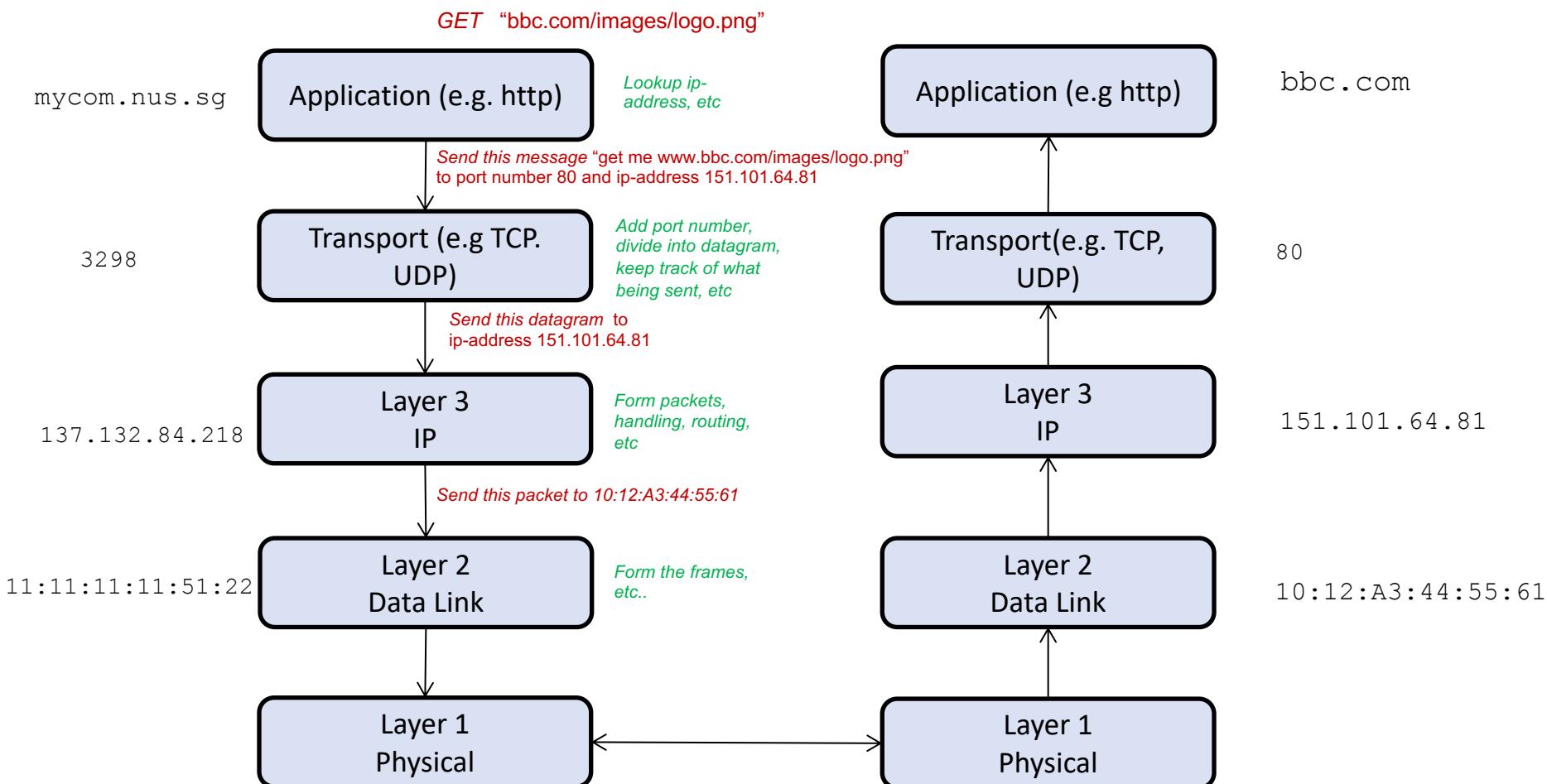


header: meta data

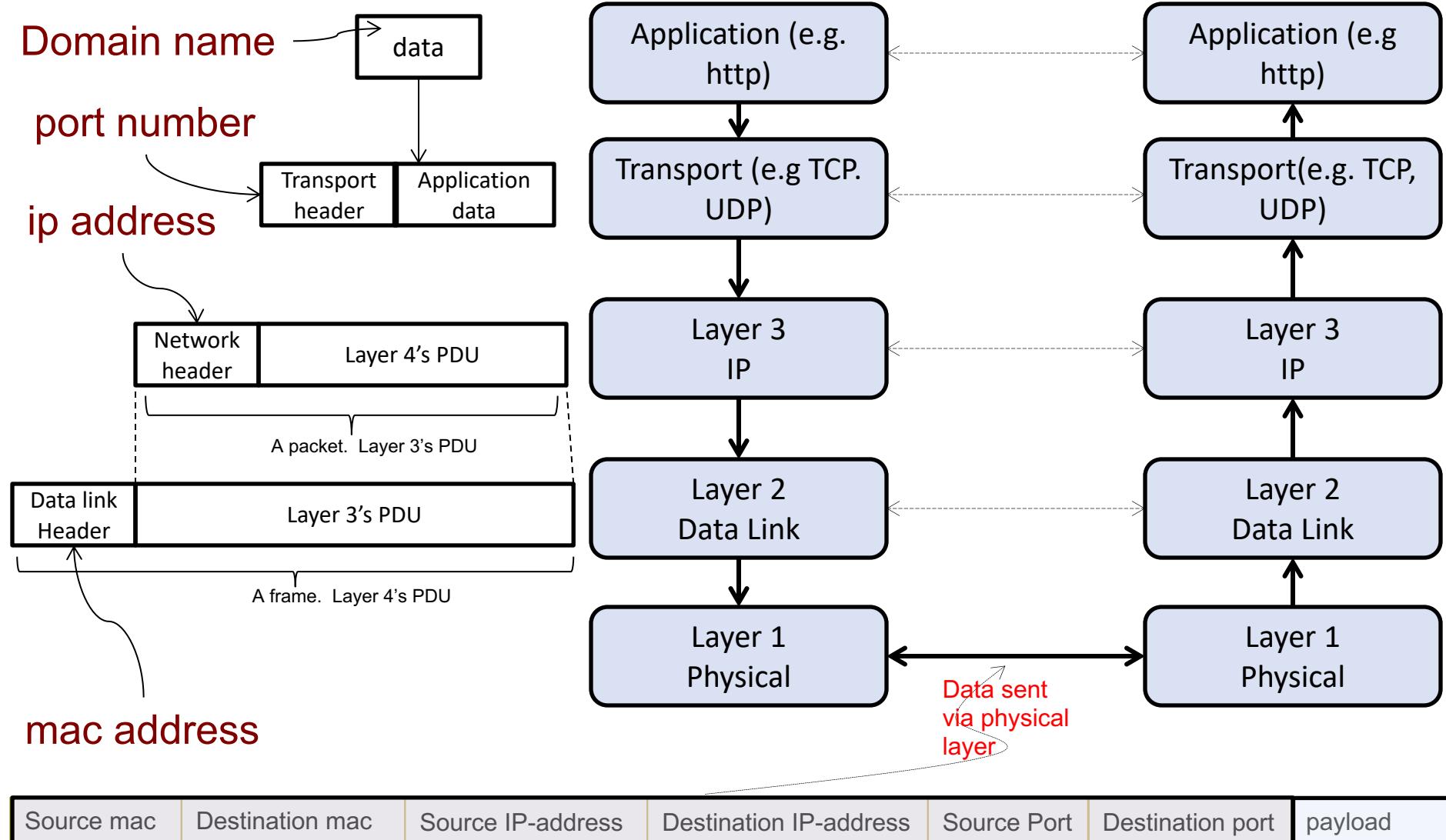
Payload: the message/information intend to be sent

# Data flow across the layers between two nodes

- When a node sends a message to another, the information flow “down” across the layers within the sender, and then “up” within the receiver.

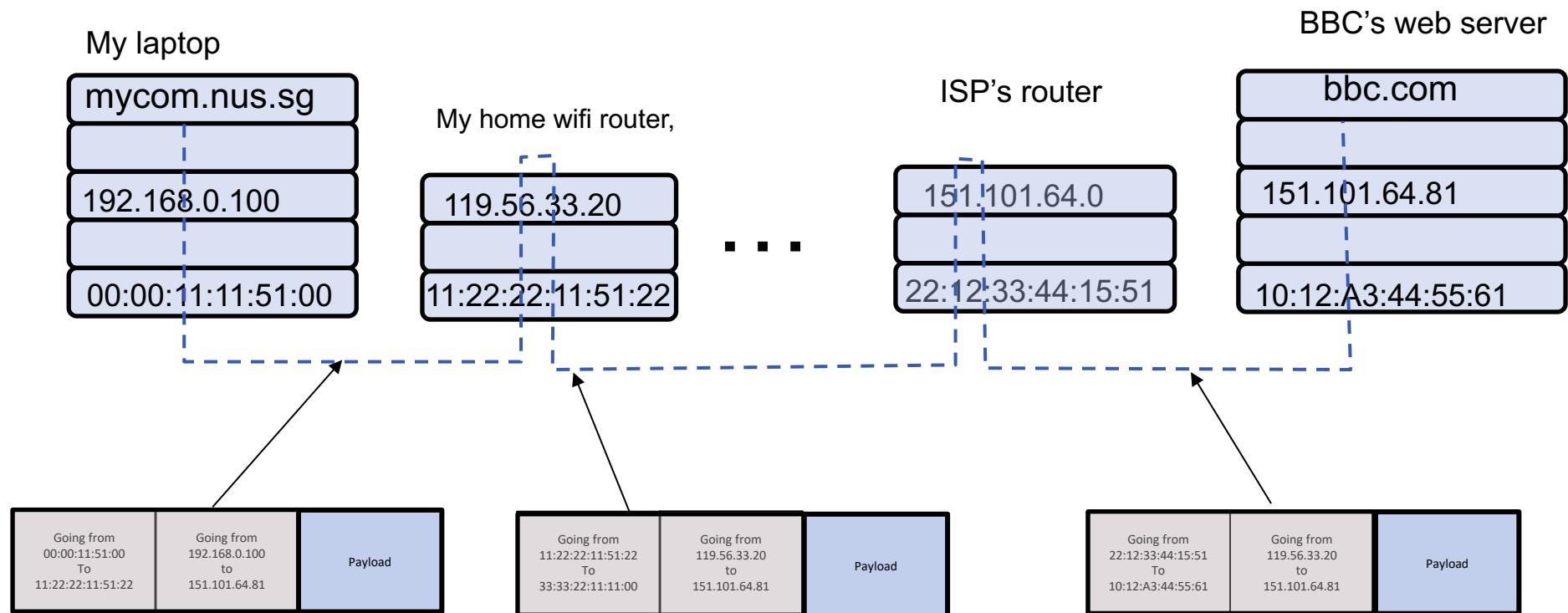


# Data flow across the layers between two nodes



# Multiple hops

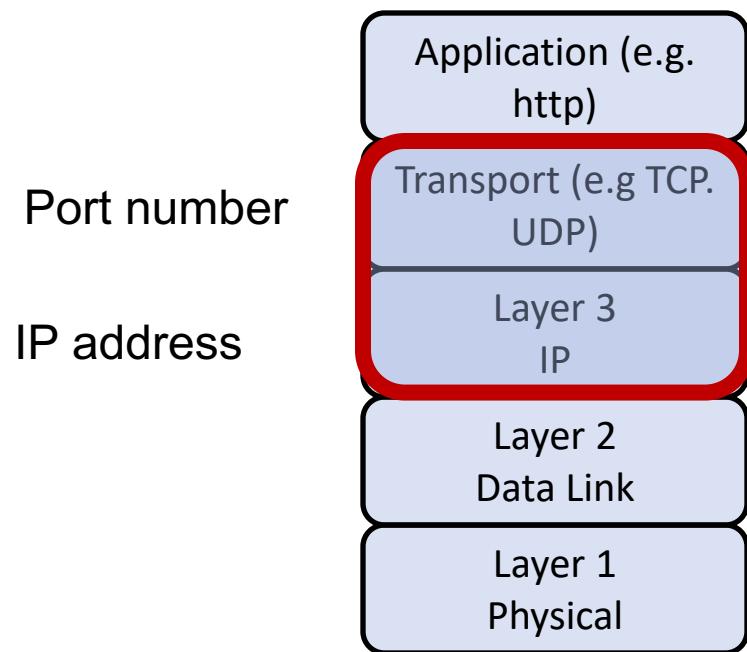
Data likely would go through multiple hops. Intermediate nodes could be owned by different semi-trusted third parties, e.g. Internet service provider (ISP), company's firewall. Intermediate nodes might change header information, translate the address etc, typically up to layer 3. Nonetheless, some intermediate nodes might change data in layer 4 and even application layer data.



# Challenges in Network Security

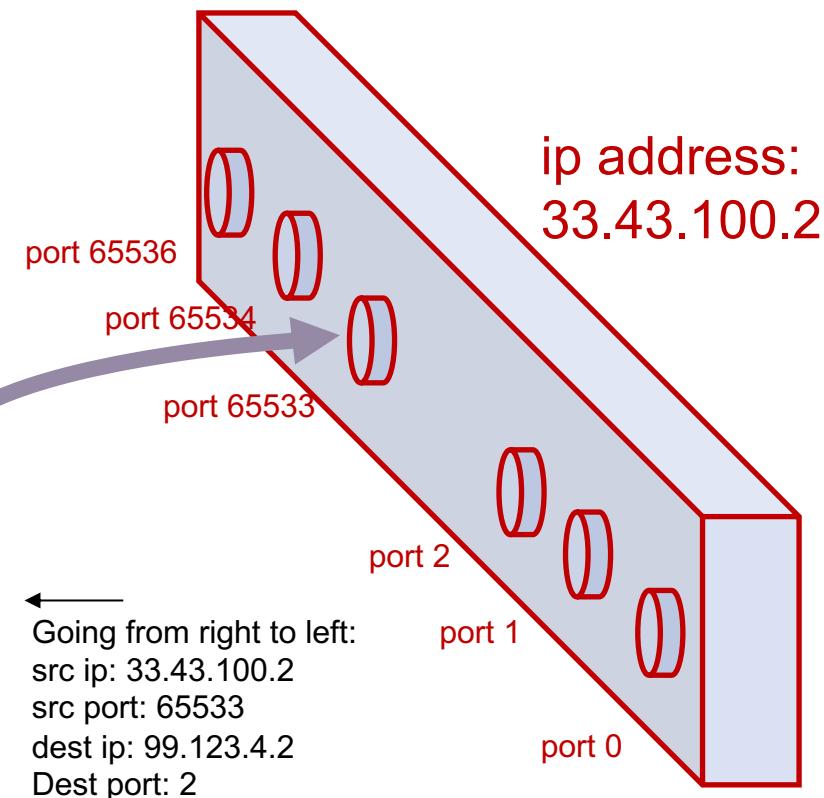
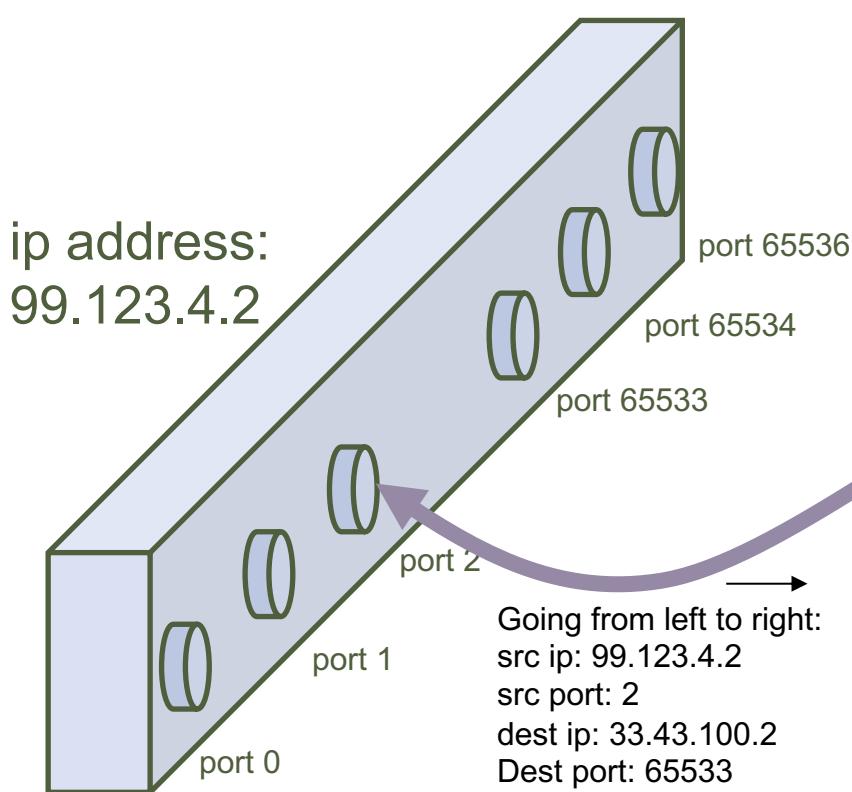
- (Complexity) Nodes are own by many different semi-trusted parties.
  - Intermediate node has access to both the header and the payload, and can modify (authenticity), read (confidentiality), drop (availability). An attacker at a certain layer can access all info in and below the layer.
- (Security Requirements) Many different security requirements.
  - In particular, availability can't be handled by crypto alone. Other requirements: anonymity, accountability, routing integrity, etc.
- (Legacy & security tradeoff) Initial design of many networking protocols did not consider intentional attacks.
  - For e.g., source ip-address in the IP header. Without additional protection mechanisms, a malicious sender can send packets with spoofed source ip-address.
  - For better performance and usability, many services do not employ strong protection mechanisms (e.g. network printer, DNS).
- (Management) There is a need to isolate and control information flow.

# Remarks on TCP/IP and UDP/IP



# Transport + IP layer

- Very often, the transport layer and IP layer are treated as one single layer. In this combined layer, the address of a communicating entity in a particular channel is an ***ip-address*** and a ***port***.
- Each node in the network has 65535 ports.
- A communication channel between two nodes is established by connecting a port in each node. In the e.g. between **99.123.4.2:2** and **33.43.100.2:65533**.



# UDP/IP

- If an application/program wants to invoke the UDP/IP protocol to send a message, the library call (depending on the programming languages) is typically of the form:

```
DatagramSend ( 2, "33.43.100.2", 65533, message)  
src ip is implicit and  
thus no need to specify      src port      dest ip      dest port
```

- The library call would construct an *IP datagram/packet*, and sends it using data-link layer protocol.



- There is limit on the size of message, at most ~65,000 bytes.
- The library call does not return a result indicating whether the destination has indeed received the packet. There is a possibility that the packet is lost! This is the property of UDP protocol. It is
  - *an unreliable, and*
  - *connectionless communication.*

# TCP/IP: *reliable* communication

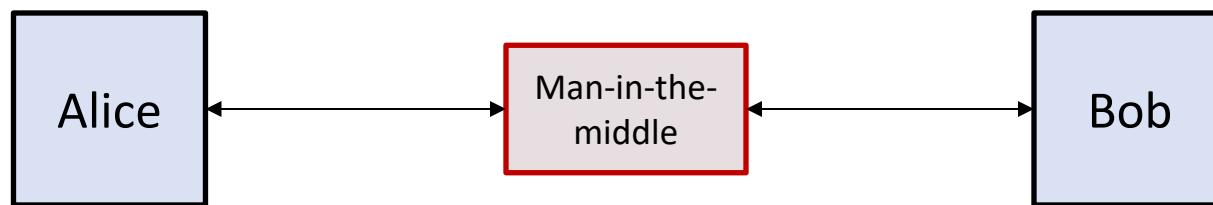
- In contrast, TCP/IP is *reliable*.
- An application/program would typically make the library calls (depending on programming languages) of the form and in the following order:
  - `P = open_connect (2, "33.43.100.2", 65533)`
  - `send (P, out_message)`
  - `read (P, in_message)`
  - `close_connection(P)`

*There could be multiple rounds of send/read.*
- `open_connect` would carry out some form of hand-shake protocol (known as TCP 3-way handshake) between the two nodes.
- `send` would construct *ip packets* of the following form and pass them to the datalink layer. In case the message is too long, multiple ip packets will be formed. The protocol also employs some mechanism of re-sending, re-ordering, acknowledgement to verify that the destination has indeed received the full message in the correct order.

src ip	dest ip	src port	dest port	message
--------	---------	----------	-----------	---------

# Reliability does not imply Security

- TCP/IP is reliable but not “secure”. Intermediate nodes along the communication route can still modify data in the header and payload.
- For example, a malicious intermediate node can
  - spoof an IP packet to inform one node to close the connection, while still communicating with the other node.
  - change information on the packets ordering, so that dest reconstruct a scrambled message.
- The intermediate nodes can act as a *man-in-the-middle* in the ip layer.

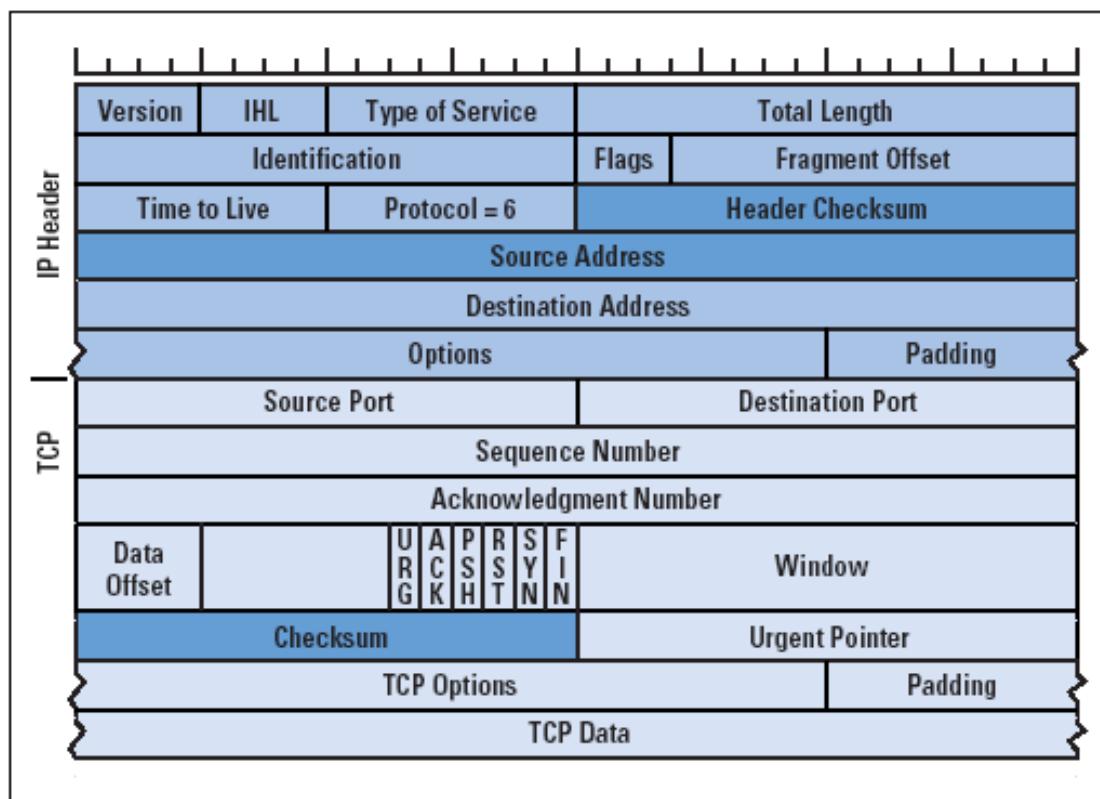


Optional

- Image from

<https://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-29/anatomy.html>

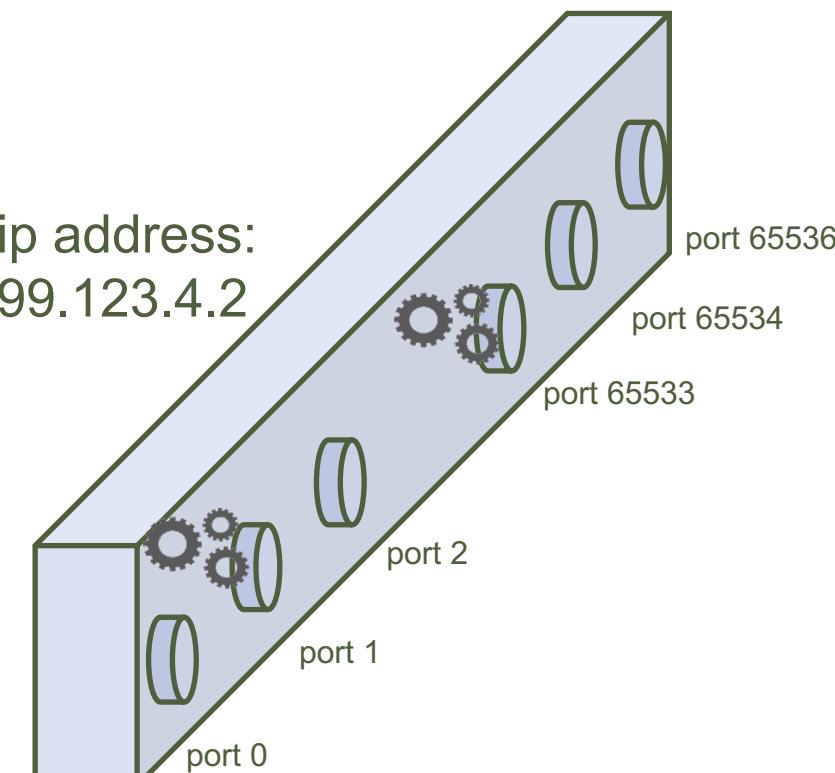
Figure 1: TCP/IP Header  
Fields Altered by NATs  
(Outgoing Packet)



# Listening Port

# What do we mean by “listening to a port”, “closed port”?

- We can imagine that behind certain ports (e.g. 1, 65533 in the e.g.), there are applications waiting to process data coming via the respective port. In such cases, we say that the node/process is “listening” to the port, and the port is a “listening port”. If the port is not listening, then it is a “closed port”.
- Data sent to a closed port will be dropped.



- There are security implications! We will revisit this in firewall design.
- Certain well-known services use widely agreed port. E.g. web server (HTTP) listen to 80, and hence the web server process is ready to process any request coming through 80.

## 5.2 Name resolution and attacks

- Each peer entity has a name. In a single node, at different layer, the name can be different.



- For a peer entity to use the virtual connection provided by layers below, it needs to find the corresponding name. E.g. finding the associated ip address for the domain name. Method of finding the name is called “*resolution*” protocol.
  - **DNS (Domain Name System):** resolution of Domain name to IP address
  - **ARP (Address Resolution Protocol):** resolution of IP address to mac address
- Many initial design of the resolution protocol didn't take security into account, and thus easy for attacker to manipulate the outcome.

- ARP attack targets at the association of ip-addr with mac-addr.
- DNS attack targets at the association of domain name with ip addr.

In this module, we only consider a basic type of DNS attack, and ARP poisoning.

**www.comp.nus.edu.sg** Domain name

## **ARP (Address Resolution Protocol)**

**132.127.12.3** IP address

## **DNS (Domain Name System)**

**01-02-03-04-05-06** mac address

# DNS (Domain Name System)

Given a domain name (e.g. [www.comp.nus.edu.sg](http://www.comp.nus.edu.sg)), its ip address can be found by looking up a locally stored table, or by querying a remote DNS server. The process is known as *resolution*. The client who initiates the query is called the *resolver*. If the address is found, we say that the domain name is *resolved*.

Demo:

```
$ nslookup www.comp.nus.edu.sg  
Server: 192.168.1.1  
Address: 192.168.1.1#53
```

Non-authoritative answer:

```
www.comp.nus.edu.sg canonical name =  
www0.comp.nus.edu.sg.
```

```
Name: www0.comp.nus.edu.sg
```

```
Address: 137.132.80.57
```

```
$
```

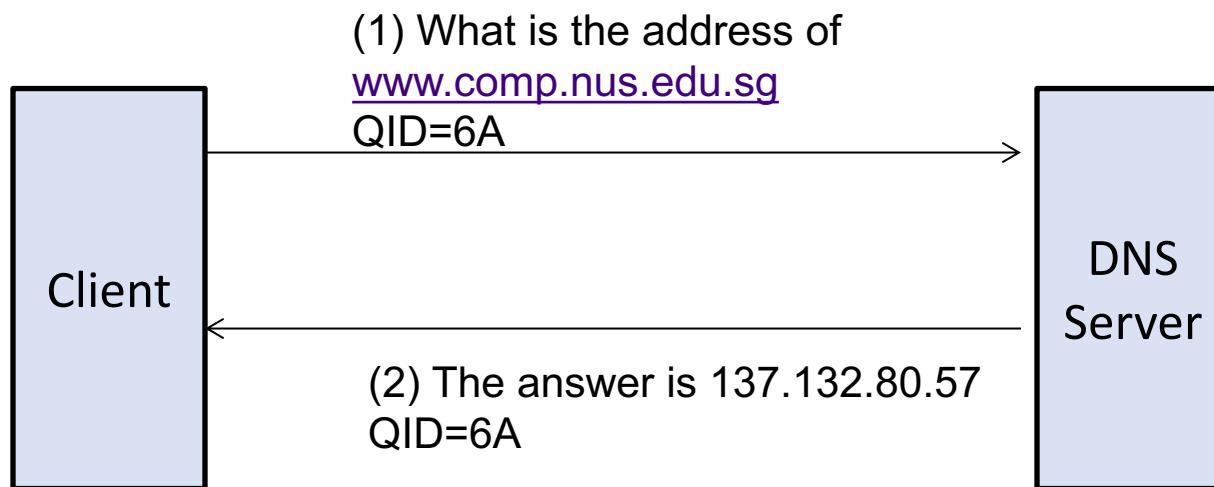
The domain name to lookup

Address of  
the DNS server

result of  
the query

# DNS query and answer

- Current DNS protocol does not protect (confidentiality and authenticity) the query nor the answer.



## Some technical details:

- The query contains a 16-bit number, known as QID (query ID).
- The response from the server must also contain a QID.
- If the QID in the response doesn't match the QID in the query, the client rejects the answer. The design consideration of having QID probably is to match the query result to multiple queries sent out by the client. Nonetheless, this can be treated as a form of "light-weight" authentication.

# DNS spoofing: Attack Scenario

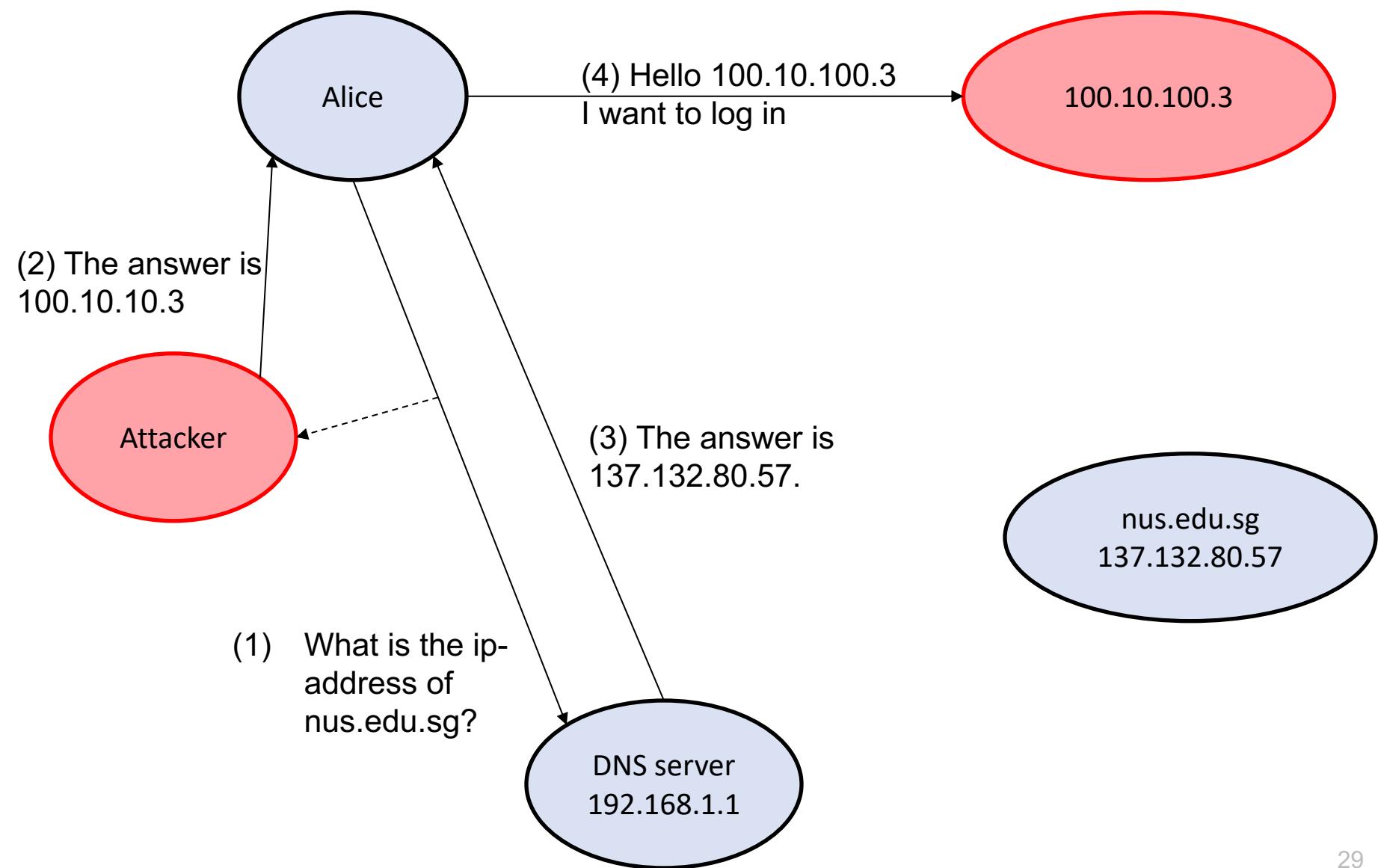
- Alice is using a café free wifi to surf the web.
- Alice wants to visit and login to [nus.edu.sg](http://nus.edu.sg)
- She types the domain name into the browser address bar.
- The browser makes a query to a DNS server to determine the ip address.  
(e.g. by executing nslookup)
- After the browser obtained the ip address, it connects to the ip address.

## Attack:

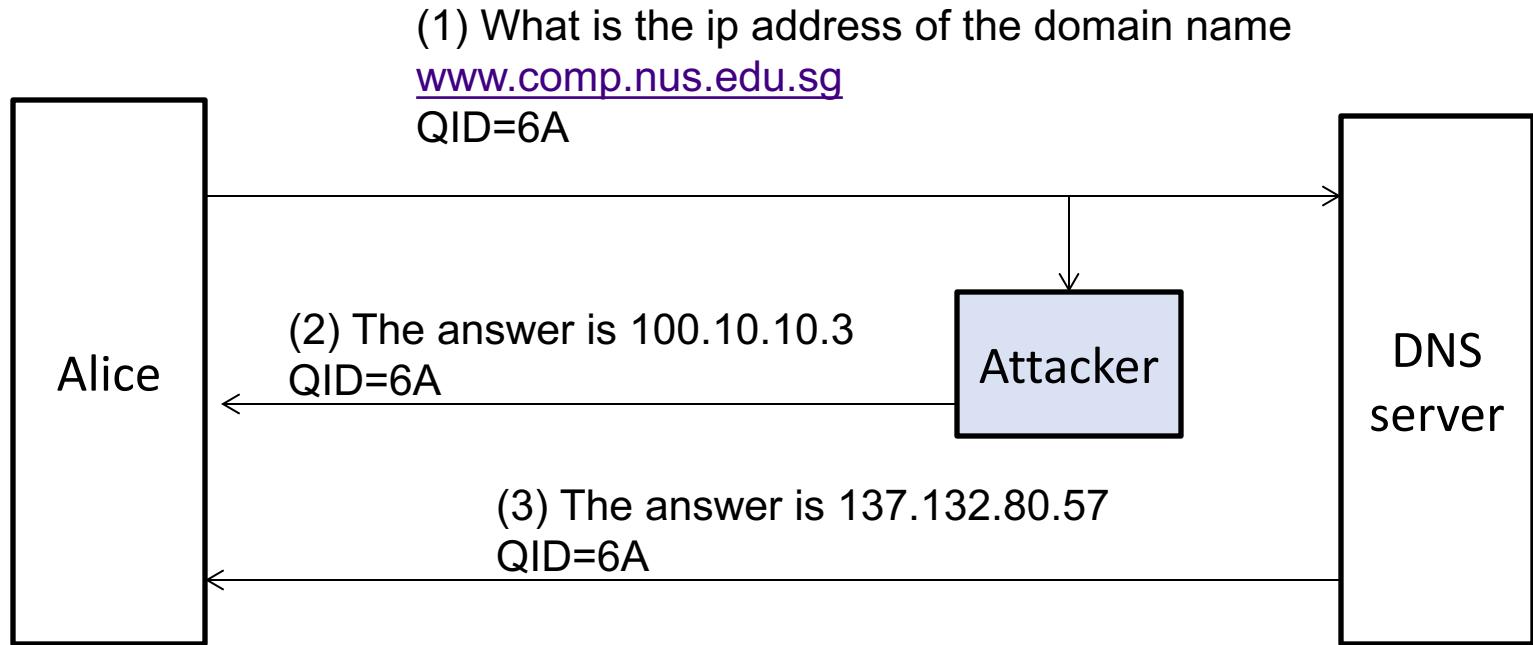
- We consider an attacker who is also in the café. Since the wifi is not protected, the attacker can
  - sniff data from the communication channel.
  - Inject spoofed data into the communication channel.

However, the attacker can't remove/modify data sent by Alice.  
(the attacker sits in the physical layer).

- Attacker owns a webserver at 100.10.10.3 which is a spoofed NUS website.



# The attack



Step (1) Alice asks for the address.

Step (2) Attacker sniffs and knows about it. Attacker quickly spoofs a reply with the same QID.

Step (3) DNS server also sends a reply. Since Attacker is closer to Alice, Attacker's reply likely to reach Alice first.

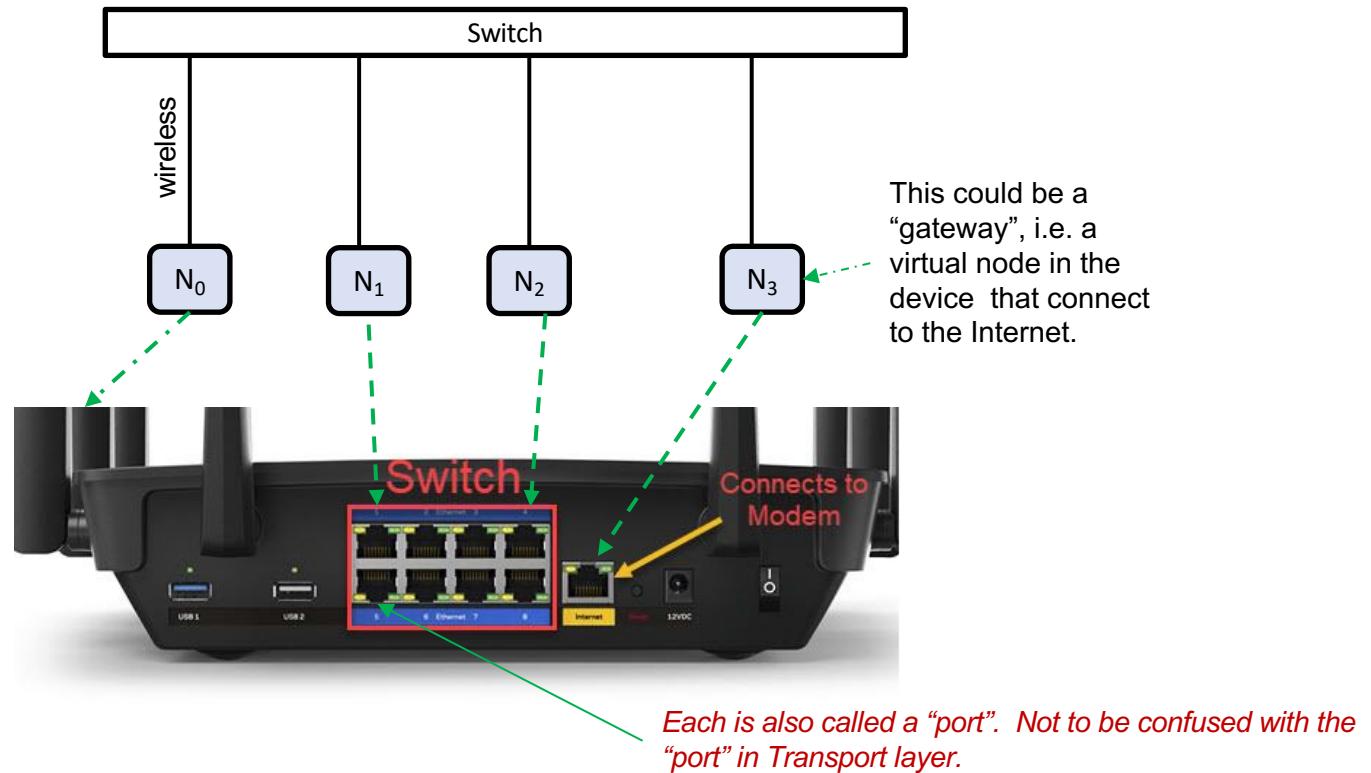
Alice takes the first reply as answer, and connect to 100.10.10.3

DNS spoofing does not work on https since the attacker wouldn't know the original domain's private key

- DNS is an important component as it resolves the domain name. Hence, an DNS server can be a “single-point-of-failure” of the network.
- A denial of service (DOS) attacks on a web service, instead of directly attack the web server, could conduct DOS to the DNS server instead. When DNS server is downed, the web service is not longer reachable.
- E.g. see attack on wikiLeak (<http://www.independent.co.uk/life-style/gadgets-and-tech/news/twitter-down-is-it-not-working-wikileaks-ddos-us-election-japan-a7402081.html> and [PF6.5]pg 414, [Pf]page 485).

# Poisoning attack on ARP table

- A switch connects a few nodes. (Note that *switch* handles mac-addresses, *router* handles ip-addresses.)



- E.g. our home wifi “router”, although often called as a “router”, it also functions as a switch.

- The role of the switch is similar to the telephone switchboard shown earlier. It connects two “ports”.
- Switch doesn’t understand ip-addresses and doesn’t store ip-addresses. It connects ports based on mac-addresses.
- The switch keeps a table that associates the port to the mac-addresses.
- Resolution of ip-address to mac-address is done by the nodes (e.g. desktop, mobile phone). Each node keeps a table that associates ip-address to mac-addresses.
- The nodes update each other using some protocols, regarding info on the ip-addresses and mac addresses.
- **ARP poisoning** is an attack that modifies (aka “poisons”) the tables so as to gain Man-In-The-Middle access.

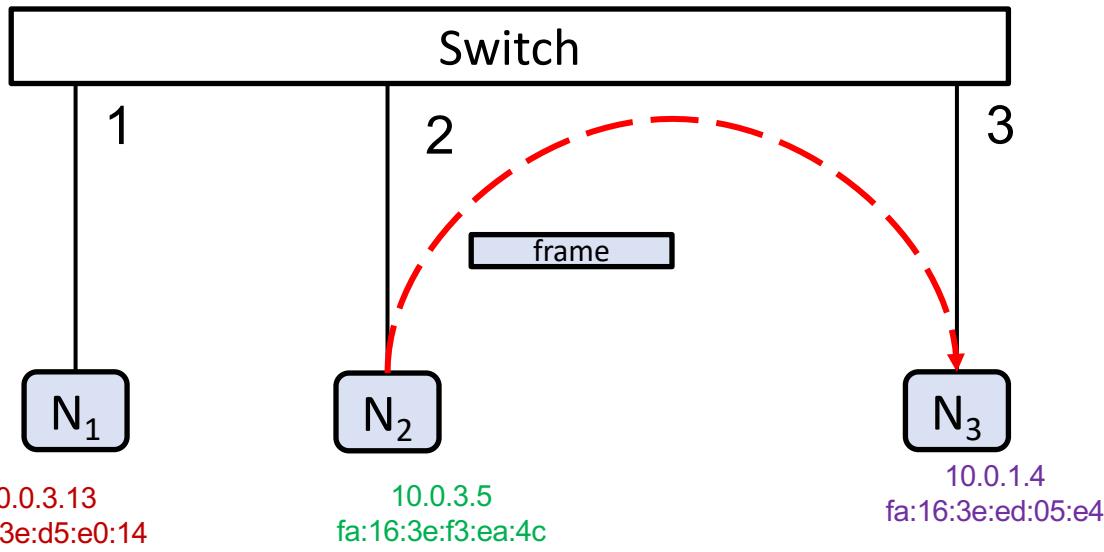
# When $N_2$ want to send to ip address 10.0.1.4

**T0**

Switch's port	Mac-address
1	fa:16:3e:d5:e0:14
2	fa:16:3e:f3:ea:4c
3	fa:16:3e:ed:05:e4

Under normal circumstances, these are carried out when  $N_2$  sends a packet to 10.0.1.4

1.  $N_2$  looks up the table **T2**. Resolve to fa:16:3e:ed:05:e4
2.  $N_2$  sends the frame to the switch, specifying destination fa:16:3e:ed:05:e4
3. Switch looks up the table **T0**, redirect the frame to port 3.



Ip-address	Mac-address
10.0.3.13	fa:16:3e:d5:e0:14
10.0.1.4	fa:16:3e:ed:05:e4

**T2**

Ip-address	Mac-address
10.0.3.13	fa:16:3e:d5:e0:14
10.0.3.5	fa:16:3e:f3:ea:4c

**T3**

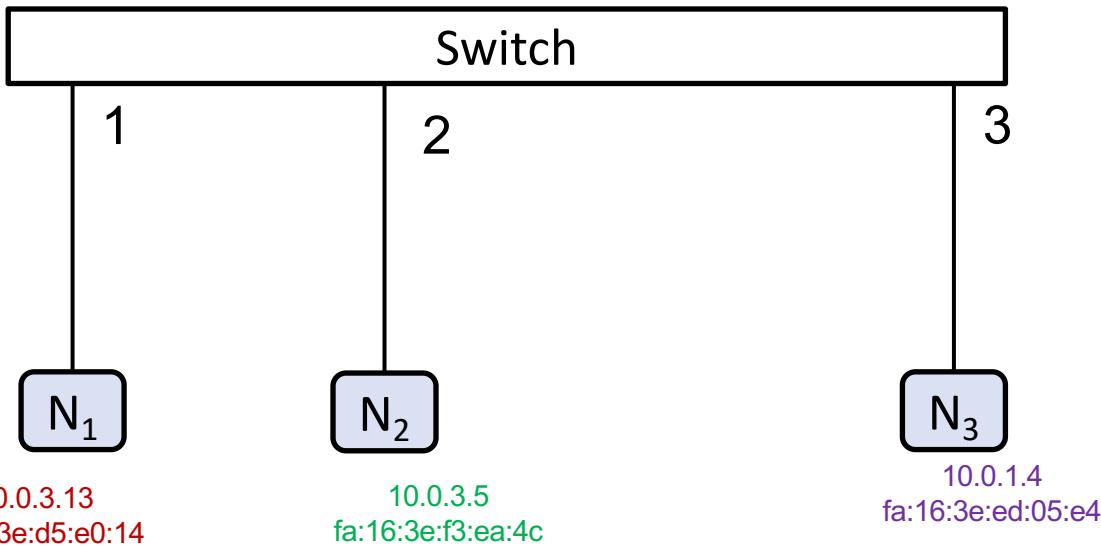
*Note:*

- $T0$  stored in switch.
- $T2$  stored in  $N_2$
- $T3$  stored in  $N_3$

# Attack: $N_1$ wants to be MITM between 10.0.3.5 and 10.0.1.4 $T_0$

Switch's port	Mac-address
1	fa:16:3e:d5:e0:14
2	fa:16:3e:f3:ea:4c
3	fa:16:3e:ed:05:e4

1.  $N_1$  informs  $N_2$  that mac address of 10.0.1.4 is fa:16:3e:d5:e0:14
2.  $N_1$  informs  $N_3$  that mac address of 10.0.3.5 is fa:16:3e:d5:e0:14



Ip-address	Mac-address
10.0.3.13	fa:16:3e:d5:e0:14
10.0.1.4	fa:16:3e:ed:05:e4

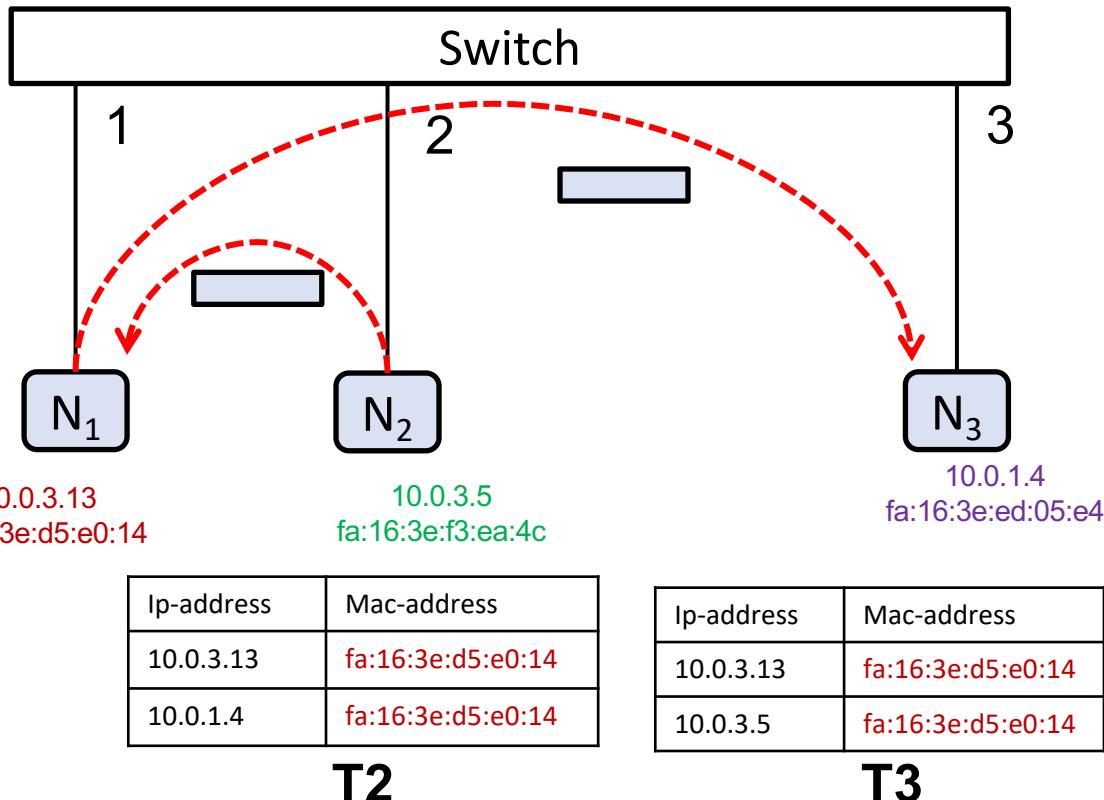
Ip-address	Mac-address
10.0.3.13	fa:16:3e:d5:e0:14
10.0.3.5	fa:16:3e:f3:ea:4c

$T_2$        $T_3$

# Attack: N<sub>1</sub> wants to be MITM between 10.0.3.5 and 10.0.1.4 T<sub>0</sub>

Switch's port	Mac-address
1	fa:16:3e:d5:e0:14
2	fa:16:3e:f3:ea:4c
3	fa:16:3e:ed:05:e4

- After the tables are poisoned, all frames will be sent to N<sub>1</sub>.
- N<sub>1</sub> can relay the frames, or modify the frames before relaying.
- Hence, N<sub>1</sub> become the MITM in layer 2.



## **5.3 Denial of Service Attack**

DOS is an attack on availability.

**Availability:** The property of being accessible and usable upon demand by an authorized entity.

**Denial of service (DOS):** The prevention of authorized access to resources or the delaying of time-critical operations.

Many successful DOS attacks simply flood the victims with overwhelming requests/data.

## Example of DOS attack

- Simply flood a web-server with http requests.

E.g. MyDoom worm which targeted SCO's website. Attacks starts on Feb 12, 2004.

- Sending large number of DNS requests to the DNS server.

For DOS to be effective, large number of attackers are required.  
(a single attacker can send requests only at a low rate).

When DOS is carried out by a large number of attackers, this is called ***DDOS: Distributed Denial of Service.***

# Reflection and Amplification attack

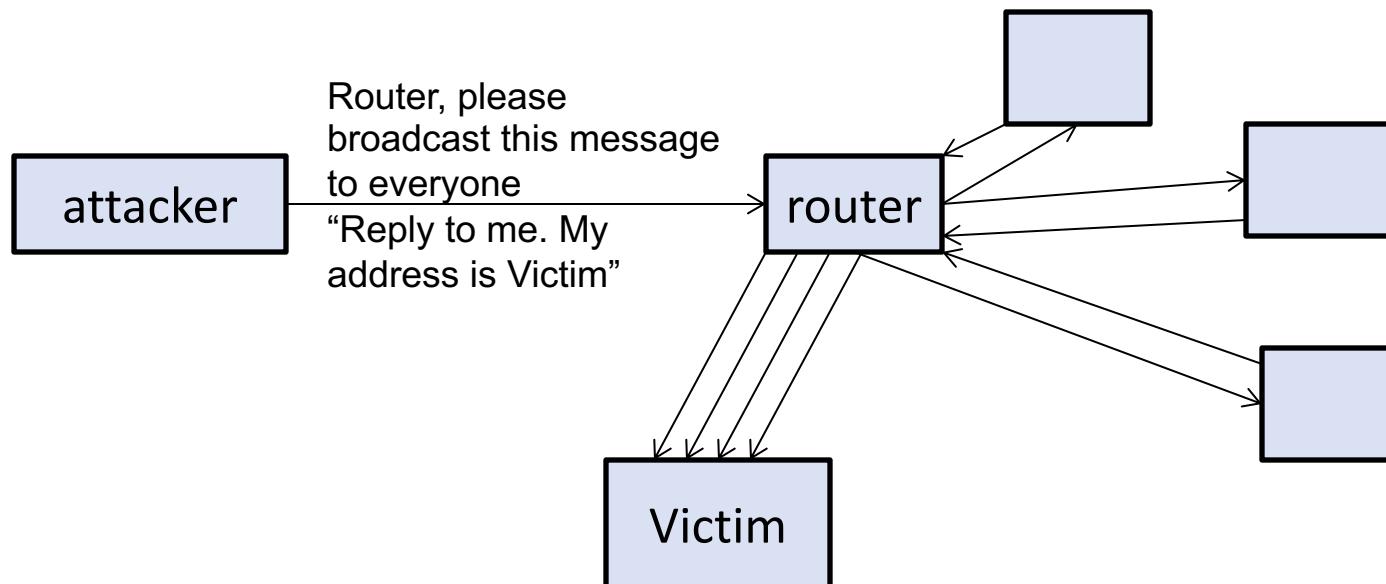
- Reflection attack is a type of DOS in which the attackers send requests to intermediate nodes, which in turn send overwhelming traffic to the victim.
- Indirect, and thus more difficult to trace.
- Very often, the reflected traffic could be amplified (a single request could trigger multiple responses from the intermediate nodes). Hence, these are also known as *Amplification attacks*.

# Eg of Reflection attacks: ICMP/Smurf flood

[PF] page 404

- Step 1. An attacker sends the request “ICMP PING” to a router, instructing the router to broadcast this request. The source ip-address of this request is spoofed with the victim ip address.
- Step 2. The router broadcasts this request.
- Step 3. Each entity who has received this request, replies to it by sending an “echo reply” to the source, which is the victim.

The victim’s network is overwhelmed with the “echo reply”.



## Smurf flood preventive measure.

- This attack is no longer effective. Most router are now configured not to broadcast.
- To prevent attack, this measure simply disabled a feature that previously thought to be useful.

See IP broadcasting, [http://en.wikipedia.org/wiki/Broadcast\\_address](http://en.wikipedia.org/wiki/Broadcast_address)

# Other examples of Reflection attack

- DNS reflection attack.

“During a DNS amplification attack, the perpetrator sends out a DNS query with a forged IP address (the victim’s) to an open DNS resolver, prompting it to reply back to that address with a DNS response. With numerous fake queries being sent out, and with several DNS resolvers replying back simultaneously, the victim’s network can easily be overwhelmed by the sheer number of DNS responses.”

<https://security.stackexchange.com/questions/93820/dns-reflection-attack-vs-dns-amplification-attack>

UDP-based Amplification Attacks

Protocol	Bandwidth Amplification Factor
Memcache	50000
NTP	556.9
CharGen	358.8
DNS	up to 179 <small>[50]</small>
QOTD	140.3
Quake Network Protocol	63.9
BitTorrent	4.0 - 54.3 <small>[51]</small>
SSDP	30.8
Kad	16.3
SNMPv2	6.3
Steam Protocol	5.5
NetBIOS	3.8

What was the largest DDOS attacks  
(as of Nov 2018)?

Targeted at Github using Memcache on 5 March 2018. Generated data at a rate of 1.7 Terabits per second.

[https://en.wikipedia.org/wiki/Denial-of-service\\_attack](https://en.wikipedia.org/wiki/Denial-of-service_attack)

# Botnet

- A **bot** (aka *zombie*) is a compromised machine.
- A **botnet** (aka *zombie army*) is a large collection of connected bots, communicating via covert channels.
- A botnet has a command-and-control mechanism, and thus can be controlled by an individual to carry out DDOS.
- Possible usages of a botnet:
  - DDoS flooding, vulnerability scanning, anonymizing HTTP proxy, email address harvesting, cipher breaking!

**Question:** Why covert channels are used by a botnet?

Covert channel: Even if communication is carried out with the bot, it is unknown.

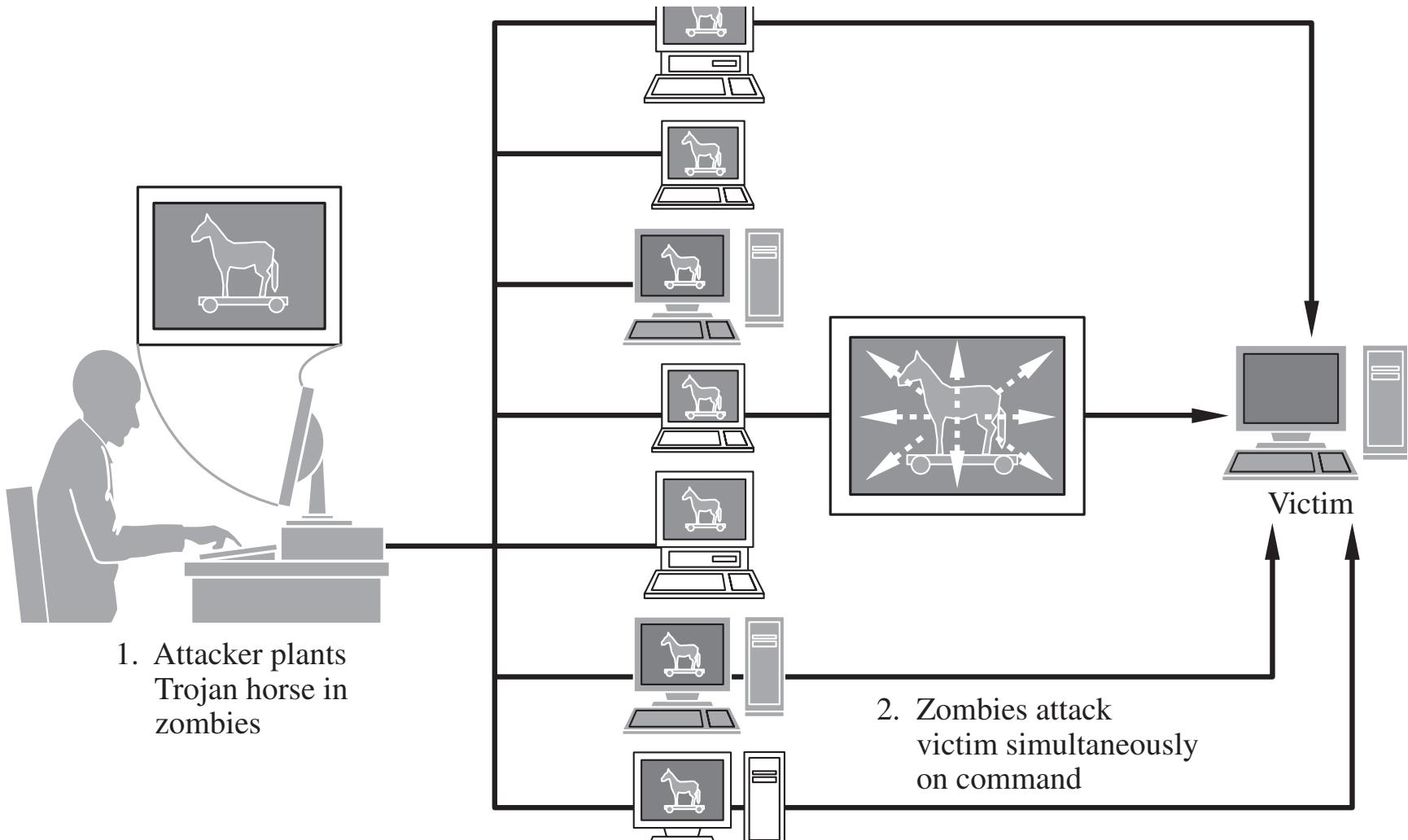
Don't want bot to know that it has been compromised.

# Botnet

- Size of known botnets. <http://en.wikipedia.org/wiki/Botnet>

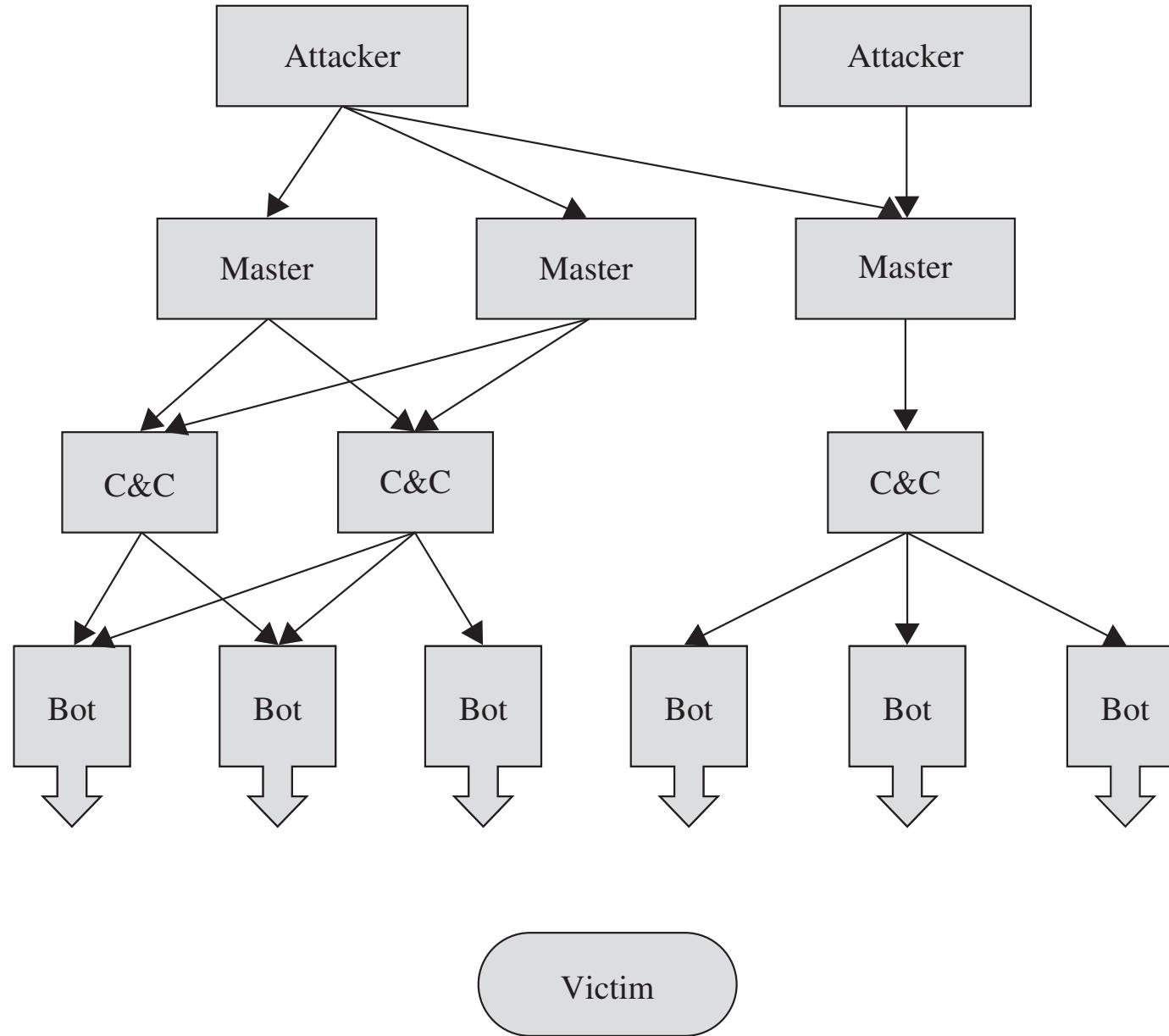
Date created	Date dismantled	Name	Estimated no. of bots	Spam capacity (bn/day)	Aliases
2009 (May)	November 2010 (not complete)	BredoLab	30,000,000 <sup>[51]</sup>	3.6	Oficla
2008 (around)	2009-Dec	Mariposa	12,000,000 <sup>[46]</sup>		
2008 (November)		Conficker	10,500,000+ <sup>[47]</sup>	10	DownUp, DownAndUp, DownAdUp, Kido
		Marina Botnet	6,215,000 <sup>[37]</sup>	92	Damon Briant, BOB.dc, Cotmonger, Hacktool.Spammer, Kraken
2010 (around)		TDL4	4,500,000 <sup>[56]</sup>		TDSS, Alureon
		Zeus	3,600,000 (US only) <sup>[57]</sup>		Zbot, PRG, Wsnpoem, Gorhax, Kneber
2011 or earlier	2015-02	Ramnit	3,000,000 <sup>[58]</sup>		
2007 (around)		Cutwail	1,500,000 <sup>[42]</sup>	74	Pandex, Mutant (related to: Wigon, Pushdo)

# Distributed Denial of Service (DDoS) using botnet



From *Security in Computing, Fifth Edition*, by Charles P. Pfleeger, et al. (ISBN: 9780134085043). Copyright 2015 by Pearson Education, Inc. All rights reserved.

# Botnet



# DDoS Attacks trend

- There is a new trend of using “non-PC”, such as IoT devices, as bots in launching DDOS. A well-known example is the Mirai attack.
- List of well-known DDOS:
  - **4 March, 2018.** Victim Github. 1.7 tera bits per second (tpbs). Memcached. Record breaking.
  - **February 28, 2018.** Victim Github: 1.35 tbps. Memcached. Record breaking.
  - **October, 2016.** Using Mirai to coordinate a botnet of IoT devices. A series of attacks. Victim Dyn (a DNS service providers), French web host OVH (~1 tpbs).
  - **March 2015,** Victim Github. Believed to be originated from China. Javascript injection by “Great Firewall”.
  - **2013,** Victim Spamhaus. 0.3 tpbs. Record breaking. See <https://blog.cloudflare.com/the-ddos-that-almost-broke-the-internet/>
  - **27 April, 2007,** Victim Estonia’s government services. Believe to be politically motivated (state sponsored attack?)
  - **January, 2004.** Mydoom. Infected machines sent https requests to the victim, SCO’s websites `www.sco.com`. Spread infection by email. Estimated to generate 1 in every 5 emails at its peak.

To prevent DDoS, hosts can use cloud services, or track incoming packets

## **5.4 Useful tools**

# Wireshark (packets analyzer)

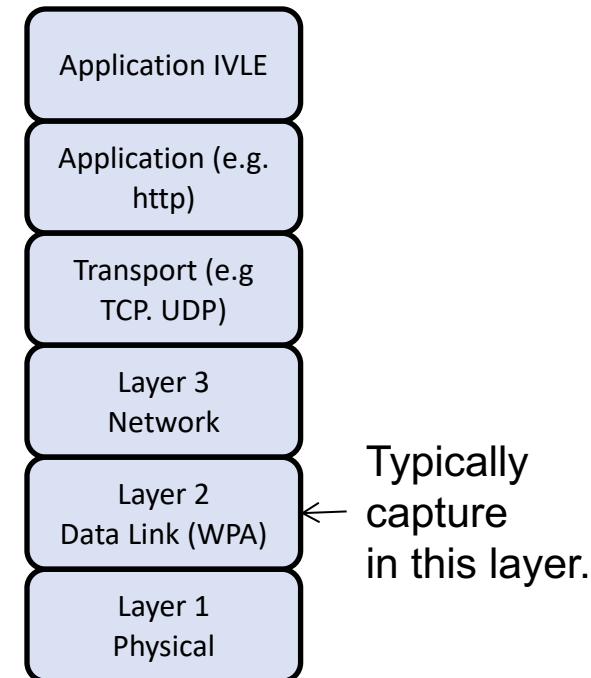
Wireshark: a free open-sourced packets analyzer.

<https://www.wireshark.org/>

Exactly what does wireshark capture?

- Wireshark listens to “interactions” between the OS and the network card driver. (In other words, it is a MITM between OS and network card).
- Hence, header added by the network card, or modification made by the network card, may not be captured by Wireshark. This depends on the OS and the hardware.
- see the following FAQ for more info.

<https://ask.wireshark.org/questions/22956/where-exactly-wireshark-does-captures-packets>



(demo: Wireshark)

tv-netflix-problems-2011-07-06.pcap

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/> Expression... +

No.	Time	Source	Destination	Protocol	Length	Info
343	65.142415	192.168.0.21	174.129.249.228	TCP	66	40555 → 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=491519346 TSecr=551811827
344	65.142715	192.168.0.21	174.129.249.228	HTTP	253	GET /clients/netflix/application.swf?flash_version=flash_lite_2.1&v=1.5&n...
345	65.230738	174.129.249.228	192.168.0.21	TCP	66	80 → 40555 [ACK] Seq=1 Ack=188 Win=6864 Len=0 TSval=551811850 TSecr=491519347
346	65.240742	174.129.249.228	192.168.0.21	HTTP	828	HTTP/1.1 302 Moved Temporarily
347	65.241592	192.168.0.21	174.129.249.228	TCP	66	40555 → 80 [ACK] Seq=188 Ack=763 Win=7424 Len=0 TSval=491519446 TSecr=551811852
348	65.242532	192.168.0.21	192.168.0.1	DNS	77	Standard query 0x2188 A cdn-0.netfliximg.com
349	65.276870	192.168.0.1	192.168.0.21	DNS	489	Standard query response 0x2188 A cdn-0.netfliximg.com CNAME images.netflix.com.edges...
350	65.277992	192.168.0.21	63.80.242.48	TCP	74	37063 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM=1 TSval=491519482 TSecr=...
351	65.297757	63.80.242.48	192.168.0.21	TCP	74	80 → 37063 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 SACK_PERM=1 TSval=3295...
352	65.298396	192.168.0.21	63.80.242.48	TCP	66	37063 → 80 [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSval=491519502 TSecr=3295534130
353	65.298687	192.168.0.21	63.80.242.48	HTTP	153	GET /us/nrd/clients/flash/814540.bun HTTP/1.1
354	65.318730	63.80.242.48	192.168.0.21	TCP	66	80 → 37063 [ACK] Seq=1 Ack=88 Win=5792 Len=0 TSval=3295534151 TSecr=491519503
355	65.321733	63.80.242.48	192.168.0.21	TCP	1514	[TCP segment of a reassembled PDU]

> Frame 349: 489 bytes on wire (3912 bits), 489 bytes captured (3912 bits)  
> Ethernet II, Src: Globalsc\_00:3b:0a (f0:ad:4e:00:3b:0a), Dst: Vizio\_14:8a:e1 (00:19:9d:14:8a:e1)  
> Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.21  
> User Datagram Protocol, Src Port: 53 (53), Dst Port: 34036 (34036)  
▼ Domain Name System (response)  
[Request In: 348]  
[Time: 0.034338000 seconds]  
Transaction ID: 0x2188  
Flags: 0x8180 Standard query response, No error  
Questions: 1  
Answer RRs: 4  
Authority RRs: 9  
Additional RRs: 9  
▼ Queries  
> cdn-0.netfliximg.com: type A, class IN  
> Answers  
> Authoritative nameservers  

0020	00 15 00 35 84 f4 01 c7 83 3f 21 88 01 80 00 01	...5.... .?.....
0030	00 04 00 09 00 09 05 63 64 6e 2d 30 07 6e 66 6c	.....c dn-0.netfliximg.com .....
0040	78 69 6d 67 03 63 6f 6d 00 00 01 00 01 c0 0c 00	.....). ".images
0050	05 00 01 00 00 05 29 00 22 06 69 6d 61 67 65 73	.netflix .com.edgesuite.net/...
0060	07 6e 65 74 66 6c 69 78 03 63 6f 6d 09 65 64 67	
0070	65 73 75 69 74 65 03 6e 65 74 00 c0 2f 00 05 00	

Identification of transaction (dns.id), 2bytes || Packets: 10299 | Displayed: 10299 (100.0%) | Load time: 0:0.182 | Profile: Default

# E.g. on SSL/TLS

Display filter. Note: there are two types of filter in wireshark: display and capture filter. (don't get confused).

ssl and ip.addr==202.79.210.117

No.	Time	Source	Destination	Protocol	Length	Info
474	8.777327	172.26.188.83	202.79.210.117	TLSv1.2	583	Client Hello
475	8.781729	202.79.210.117	172.26.188.83	TLSv1.2	1402	Server Hello
479	8.781918	202.79.210.117	172.26.188.83	TLSv1.2	1333	Certificate, Server Hello Done
481	8.782389	172.26.188.83	202.79.210.117	TLSv1.2	408	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
482	8.788428	202.79.210.117	172.26.188.83	TLSv1.2	141	Change Cipher Spec, Encrypted Handshake Message
485	8.789308	172.26.188.83	202.79.210.117	TLSv1.2	335	Application Data
488	8.808773	202.79.210.117	172.26.188.83	TLSv1.2	1399	Application Data
489	8.808800	202.79.210.117	172.26.188.83	TLSv1.2	695	Application Data
701	13.407585	172.26.188.83	202.79.210.117	TLSv1.2	335	Application Data
703	13.427872	202.79.210.117	172.26.188.83	TLSv1.2	1399	Application Data
704	13.427900	202.79.210.117	172.26.188.83	TLSv1.2	695	Application Data

Encrypted data

Wireshark - Packet 474 - wireshark\_en9\_20190314085258\_2TIWA

Frame 474: 583 bytes on wire (4664 bits), 583 bytes captured (4664 bits) on interface 0

Internet Protocol Version 4, Src: 172.26.188.83, Dst: 202.79.210.117

Transmission Control Protocol, Src Port: 54403, Dst Port: 443, Seq: 1, Ack: 1, Len: 517

Sequence number: 1 (relative sequence number)

0000 00 00 0c 07 ac 00 00 e0 4c 13 5a b8 08 00 45 00	.....L.Z...E.
0010 02 39 00 00 40 00 40 06 33 8c ac 1a bc 53 ca 4f	.9...@. 3....S.0
0020 d2 75 d4 83 01 bb 69 04 7d 29 69 a0 3a 80 18	.....i. }..j.::.
0030 08 04 4f aa 00 00 01 01 08 0a 1f 8e 66 1a 0c 60	.....0.....f..`
0040 8d 87 16 03 01 02 00 01 00 01 fc 03 03 66 4b 4a	.....fKJ
0050 ed 4b 06 eb 0c 5c cc 6a 7b d3 9c be 96 06 65 8c	.K...\\j {....e.
0060 43 40 da 68 bb 71 74 fa 9a d1 da ca c4 20 dc 36	C@.h.qt. ....6
0070 38 c1 fb 1e 67 3b 88 58 33 82 3c 92 34 84 1a 52	8...g;X 3.<..4..R
0080 98 7e f8 15 eb c1 9d 36 6b c9 fa 92 ac ed 00 22	~....6 k. ...."
0090 ca ca 13 01 13 02 13 03 02 2b c0 2c c0 30	.....+./.,0
00a0 cc a9 cc a8 c3 c0 14 00 9c 09 0d 00 2f 00 35	...../.5
00b0 00 00 01 00 91 1a 1a 00 00 00 00 17 00 15	.....0
00c0 00 00 12 62 73 2e 73 65 72 76 69 6e 67 2d 73 79	...bs.s rving-sy
00d0 73 2e 63 6f 6d 00 17 00 00 ff 01 00 01 00 00 0a	s.com... .....
00e0 00 00 00 08 aa aa 0d 00 17 00 18 00 0b 00 02	.....

No: 474 - Time: 8.777327 - Source: 172.26.188.83 - Destination: 202.79.210.117 - Protocol: TLSv1.2 - Length: 583 - Info: Client Hello

Wireshark - Packet 479 - wireshark\_en9\_20190314085258\_2TIWA

Certificates (3828 bytes)

Secure Sockets Layer

Handshake Protocol: Server Hello Done

Content Type: Handshake (22)

Version: TLS 1.2 (0x0303)

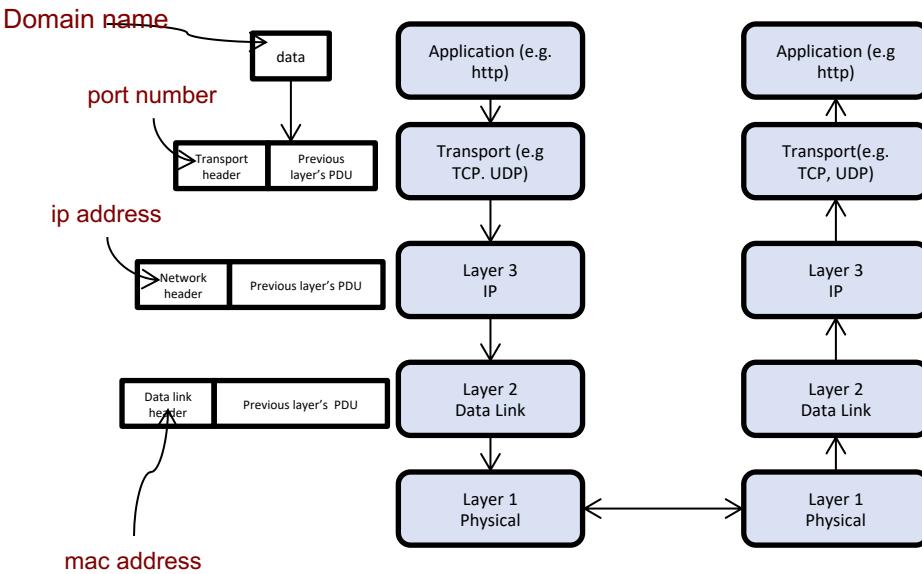
Length: 4

00f0 64 79 20 47 72 6f 75 70 2c 20 49 6e 63 2e 31 31 dy Group , Inc.11
0100 30 2f 06 03 55 04 0b 13 28 47 6f 20 44 61 64 64 0/.U... (Go Dadd
0110 79 26 43 6c 61 73 73 20 32 20 43 65 72 74 69 66 y Class 2 Certif
0120 69 63 61 74 69 6f 6e 20 41 75 74 68 6f 72 69 74 ication Authorit
0130 79 30 1e 17 0d 31 34 30 31 30 31 30 37 30 30 30 y0...140 10107000
0140 30 5a 17 0d 33 31 30 35 33 30 37 30 30 30 30 30 0Z..3105 30070000
0150 5a 30 81 83 31 0b 06 03 55 04 06 13 02 55 ZO..1.0. ..U....U
0160 53 31 10 30 06 03 55 04 08 13 07 41 72 69 7a S1.0...U ....Ariz
0170 6f 6d 31 31 30 11 06 03 55 04 07 13 0a 53 63 onal1.0. ..U....Sc
0180 6f 74 73 64 61 66 65 31 1a 30 18 06 03 55 04 ottsdale 1.0...U.
0190 0a 13 11 47 6f 44 61 64 64 79 2e 63 6f 6d 2c 20 ...GoDad dy.com,
01a0 49 6e 63 2e 31 31 30 2f 06 03 55 04 03 13 28 47 Inc.110/ ..U....G
01b0 6f 20 44 61 64 79 20 52 6f 6f 74 20 43 65 72 o Daddy Root Cer
01c0 74 69 66 69 63 61 74 65 20 41 75 74 68 6f 72 69 tificate Authori

Bytes 66-1323: TCP segment data (tcp.segment\_data)

# Useful Tools: Nmap (port scanning)

- Why port scanning?
- There are multiple processes in running in a server. When the server receives a packet, base on the port number, it will decide which process handle that packet. So, by saying that a process/service is “*listening*” to a particular *port*, we mean that the process is running and ready to handle arriving packets with that particular port number.
- When a port is “*open*”, there exist such a process running in the server. When a port is “*closed*”, no process is listening to that port.
- If a port is “*closed*”, attacker is unable to feed malicious data to that port.
- see  
[https://en.wikipedia.org/wiki/List\\_of\\_TCP\\_and\\_UDP\\_port\\_numbers#Well-known\\_ports](https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers#Well-known_ports) for well-known port number.



69	Assigned	Yes	Official	Trivial File Transfer Protocol (TFTP) <sup>[10][37][38][39]</sup>
70	Yes	Assigned	Official	Gopher protocol <sup>[40]</sup>
71–74	Yes	Yes	Official	NETRJS protocol <sup>[41][42][43]</sup>
79	Yes	Assigned	Official	Finger protocol <sup>[10][44][45]</sup>
	Yes, and SCTP <sup>[11]</sup>	Assigned	Official	Hypertext Transfer Protocol (HTTP) <sup>[10][46][47][48]</sup>
80	No	Yes	Unofficial	Quick UDP Internet Connections (QUIC), a transport protocol over UDP (still in draft as of July 2018), using stream multiplexing, encryption by default with TLS, and currently supporting HTTP/2. <sup>[49]</sup>
81	Yes		Unofficial	TorPark onion routing <sup>[verification needed]</sup>
82		Yes	Unofficial	TorPark control <sup>[verification needed]</sup>
88	Yes	Assigned	Official	Kerberos <sup>[10][50][51]</sup> authentication system
90	Yes	Yes	Unofficial	PointCast (dotcom) <sup>[1][third-party source needed]</sup>
101	Yes	Assigned	Official	NIC host name <sup>[52]</sup>
102	Yes	Assigned	Official	ISO Transport Service Access Point (TSAP) Class 0 protocol. <sup>[53][54]</sup>
104	Yes	Yes	Official	Digital Imaging and Communications in Medicine (DICOM; also port 11112)
105	Yes	Yes	Official	CCSO Nameserver <sup>[55]</sup>
107	Yes	Yes	Official	Remote User Telnet Service (RTelnet) <sup>[56]</sup>
108	Yes	Yes	Official	IBM Systems Network Architecture (SNA) gateway access server
109	Yes	Assigned	Official	Post Office Protocol, version 2 (POP2) <sup>[57]</sup>
110	Yes	Assigned	Official	Post Office Protocol, version 3 (POP3) <sup>[10][58][59]</sup>
111	Yes	Yes	Official	Open Network Computing Remote Procedure Call (ONC RPC, sometimes referred to as Sun RPC)

- Port scanning: The process of determining which ports are open in a network.
- Port scanner: A tool for port scanning. E.g. Nmap.
- Port scanner is a useful tool for attacker, and network administrator to scan for vulnerabilities.
- Is port scanning illegal?

(Read if you want to use it) <https://nmap.org/book/legal-issues.html>

(demo: Nmap)

# Nmap scan report on my desktop.

```
> nmap -v -p 1-65535 -sV -O -sS -T4 172.26.
```

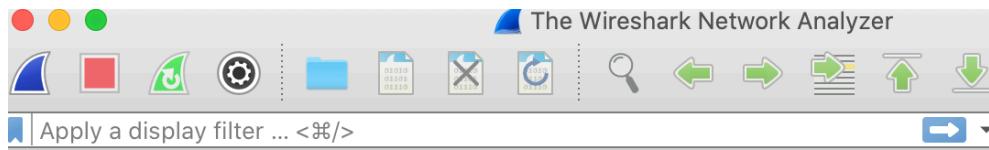
```
Starting Nmap 7.60 ( https://nmap.org ) at 2019-03-13 16:31 +08
NSE: Loaded 42 scripts for scanning.
Initiating ARP Ping Scan at 16:31
Scanning 172.26. [1 port]
Completed ARP Ping Scan at 16:31, 0.01s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 16:31
Completed Parallel DNS resolution of 1 host. at 16:31, 0.00s elapsed
Initiating SYN Stealth Scan at 16:31
Scanning [REDACTED] ports [65535]
Discovered open port 139/tcp on 172.26.
Discovered open port 135/tcp on 172.26.
Discovered open port 445/tcp on 172.26.
Discovered open port 10600/tcp on 172.26.
Discovered open port 49154/tcp on 172.26.
SYN Stealth Scan Timing: About 22.53% done; ETC: 16:33 (0:01:47 remaining)
Discovered open port 17500/tcp on 172.26.
Discovered open port 49155/tcp on 172.26.
SYN Stealth Scan Timing: About 56.84% done; ETC: 16:32 (0:00:46 remaining)
Discovered open port 49153/tcp on 172.26.
Discovered open port 2869/tcp on 172.26.
Completed SYN Stealth Scan at 16:32, 91.72s elapsed (65535 total ports)
Initiating Service scan at 16:32
Scanning 9 services on changec-d9010.d2.comp.nus.edu.sg
(172.26.188.231)
Completed Service scan at 16:33, 58.54s elapsed (9 services on 1 host)
Initiating OS detection (try #1) against changec-
NSE: Script scanning 172.26 .
Initiating NSE at 16:33
Completed NSE at 16:33, 7.09s elapsed
Initiating NSE at 16:33
Completed NSE at 16:33, 0.02s elapsed
Nmap scan report for [REDACTED]
(172.26. [REDACTED])
Host is up (0.00064s latency).
```

```
Not shown: 65526 filtered ports
PORT      STATE SERVICE      VERSION
135/tcp    open  msrpc        Microsoft Windows RPC
139/tcp    open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp    open  microsoft-ds?
2869/tcp   open  http         Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
10600/tcp  open  ssl/http    Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
17500/tcp  open  ssl-db-lsp?
49153/tcp  open  msrpc        Microsoft Windows RPC
49154/tcp  open  msrpc        Microsoft Windows RPC
49155/tcp  open  msrpc        Microsoft Windows RPC
MAC Address: B8:CA:3A:B5:0B:E0 (Dell)
Warning: OSScan results may be unreliable because we could not find at least 1
open and 1 closed port
Device type: general purpose|phone
Running: Microsoft Windows 2008|8.1|7|Phone|Vista
OS CPE: cpe:/o:microsoft:windows_server_2008:r2 cpe:/o:microsoft:windows_8.1
cpe:/o:microsoft:windows_7::professional cpe:/o:microsoft:windows_8
cpe:/o:microsoft:windows cpe:/o:microsoft:windows_vista::-
cpe:/o:microsoft:windows_vista::sp1
OS details: Microsoft Windows Server 2008 R2 or Windows 8.1, Microsoft Windows 7
Professional or Windows 8, Microsoft Windows Phone 7.5 or 8.0, Microsoft Windows
Vista SP0 or SP1, Windows Server 2008 SP1, or Windows 7, Microsoft Windows Vista
SP2, Windows 7 SP1, or Windows Server 2008
Uptime guess: 5.115 days (since Fri Mar 8 13:47:55 2019)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=257 (Good luck!)
IP ID Sequence Generation: Incremental
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Read data files from: /opt/local/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 159.90 seconds
Raw packets sent: 131169 (5.773MB) | Rcvd: 89 (4.238KB)
```

# Wireshark Demo

# (1) Choose the “network interface”



(2) Set a capture filter if required.  
e.g simply leave it empty or “tcp”

This interface ip-address  
Is 192.168.50.183.

Its mac address  
(not shown in this screenshot)  
Is 78:4f:43:8b:47:aa

(3) Too many info. Use the display filter to select what to see. Note the difference between display and capture filter.

(4) Click here to see various options for display filter.

Wi-Fi: en0

arp

No.	Time	Source	Destination	Protocol	Length	Info
1529	1.898372	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
2871	3.740251	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
3869	5.889540	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
5168	7.174376	04:d4:c4:bd:ba:48	Apple_8b:47:aa	ARP	42	Who has 192.168.50.183? Tell 192.168.50.1
5169	7.174416	Apple_8b:47:aa	04:d4:c4:bd:ba:48	ARP	42	192.168.50.183 is at 78:4f:43:8b:47:aa
5507	7.734178	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228

Frame 1529: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0

Ethernet II, Src: 64:1c:b0:45:eb:96 (64:1c:b0:45:eb:96), Dst. Broadcast (ff:ff:ff:ff:ff:ff)

Address Resolution Protocol (request)

0000 ff ff ff ff ff ff 64 1c b0 45 eb 96 08 06 00 01 .....

0010 08 00 06 04 00 01 64 1c b0 45 eb 96 c0 a8 32 e4 .....

0020 00 00 00 00 00 00 c0 a8 32 01 00 00 00 00 00 .....

0030 00 00 00 00 00 00 00 00 00 00 00 00 .....

wireshark en0 20200311210531 mM4kT6

Packets: 6045 · Displayed: 6 (0.1%)

Profile: Default

(4) Use this to start and stop capture, change interface.

# Display filter.

Wireshark · Display Filter Expression

Field Name

- ▶ 104apci · IEC 60870-5-104-Apci
- ▶ 104asdu · IEC 60870-5-104-Asdu
- ▶ 29West · 29West Protocol
- ▶ 2dparityfec · Pro-MPEG Code of Practice #3...
- ▶ 3COMXNS · 3Com XNS Encapsulation
- ▶ 3GPP2 A11 · 3GPP2 A11
- ▶ 6LoWPAN · IPv6 over Low power Wireless P...
- ▶ 802.11 Radio · 802.11 radio information
- ▶ 802.11 Radiotap · IEEE 802.11 Radiotap Capt...
- ▶ 802.11 RSNA EAPOL · IEEE 802.11 RSNA EA...
- ▶ 802.3 Slow protocols · Slow Protocols
- ▶ 9P · Plan 9
- ▶ A-bis OML · GSM A-bis OML
- ▶ A21 · A21 Protocol
- ▶ AAF · AVTP Audio Format
- ▶ AAL1 · ATM AAL1
- ▶ AAL3/4 · ATM AAL3/4
- ▶ AARP · Appletalk Address Resolution Protocol
- ▶ AASP · Astra Signalling Protocol
- ▶ ACAP · Application Configuration Access Pr...
- ▶ ACN · Architecture for Control Networks
- ▶ ACP133 · ACP133 Attribute Syntaxes
- ▶ ACR 122 · Advanced Card Systems ACR122
- ▶ ACSE · ISO 8650-1 OSI Association Control ...
- ▶ ACtrace · AudioCodes Trunk Trace
- ▶ ADB · Android Debug Bridge
- ▶ ADB CS · Android Debug Bridge Client-Server
- ▶ ADB Service · Android Debug Bridge Service
- ▶ ADP · Aruba Discovery Protocol
- ▶ ADwin · ADwin communication protocol
- ▶ ADwin-Config · ADwin configuration protocol
- ▶ Aeron · Aeron Protocol

Relation

- is present
- ==
- !=
- >
- <
- >=
- <=
- contains
- matches

Value

Predefined Values

Range (offset:length)

Search:

No display filter

A hint.

Cancel OK

Profile: Default

ssl

No. Time

133	0.950965
134	0.954629
5595	17.456059
5598	17.466147

Domain Name System (r  
[Request In: 5595]  
[Time: 0.010088000  
Transaction ID: 0x  
Flags: 0x8180 Stan  
Questions: 1  
Answer RRs: 2  
Authority RRs: 0  
Additional RRs: 0  
Queries  
www.comp.nus.edu  
Name: www.com  
[Name Length:  
[Label Count:  
Type: A (Host  
Class: IN (0  
Answers  
www.comp.nus.edu  
t36chsc.x.incap  
Name: t36chsc  
Type: A (Host  
Class: IN (0  
Time to live:  
Data length:  
Address: 45.6  
0010 00 75 00 00 40 00  
0020 32 b7 00 35 d5 af  
0030 00 02 00 00 00 00  
0040 6e 75 73 03 65 64  
0050 0c 00 05 00 01 00  
0060 68 73 63 01 78 08  
0070 65 74 00 c0 31 00  
0080 3c 23 e1

Text item (text), 16 b

ARP: recall the query-answer in arp. What is the frame number 5168 and 5169?

Wi-Fi: en0

arp

No.	Time	Source	Destination	Protocol	Length	Info
1529	1.898372	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
2871	3.740251	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
3869	5.889540	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
5168	7.174376	04:d4:c4:bd:ba:48	Apple_8b:47:aa	ARP	42	Who has 192.168.50.183? Tell 192.168.50.1
5169	7.174416	Apple_8b:47:aa	04:d4:c4:bd:ba:48	ARP	42	192.168.50.183 is at 78:4f:43:8b:47:aa
5507	7.734178	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228

► Frame 1529: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0  
► Ethernet II, Src: 64:1c:b0:45:eb:96 (64:1c:b0:45:eb:96), Dst: Broadcast (ff:ff:ff:ff:ff:ff)  
► Address Resolution Protocol (request)

Expand this to see more details.

0000 ff ff ff ff ff ff 64 1c b0 45 eb 96 08 06 00 01  
0010 08 00 06 04 00 01 64 1c b0 45 eb 96 c0 a8 32 e4  
0020 00 00 00 00 00 00 c0 a8 32 01 00 00 00 00 00 00  
0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Actual byte values in ASCII format.

Actual byte values in hexdecimals.  
What is “ff ff ff ff ff ff”, and “64 1c b0 45 eb 96”?

Packets: 6045 · Displayed: 6 (0.1%) Profile: Default

Wi-Fi: en0

arp

No.	Time	Source	Destination	Protocol	Length	Info
48	0.192123	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
1036	2.334681	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
1821	4.177717	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
2636	6.331676	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
3257	8.171254	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
4174	10.321718	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
4639	12.164889	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
4714	12.465663	04:d4:c4:bd:ba:48	Apple_8b:47:aa	ARP	42	Who has 192.168.50.183? Tell 192.168.50.1
4716	12.465741	Apple_8b:47:aa	04:d4:c4:bd:ba:48	ARP	42	192.168.50.183 is at 78:4f:43:8b:47:aa
5411	14.315399	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228
6115	16.158589	64:1c:b0:45:eb:96	Broadcast	ARP	60	Who has 192.168.50.1? Tell 192.168.50.228

▶ Frame 4714: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0

▼ Ethernet II, Src: 04:d4:c4:bd:ba:48 (04:d4:c4:bd:ba:48), Dst: Apple\_8b:47:aa (78:4f:43:8b:47:aa)

- ▶ Destination: Apple\_8b:47:aa (78:4f:43:8b:47:aa)
- ▶ Source: 04:d4:c4:bd:ba:48 (04:d4:c4:bd:ba:48)
- Type: ARP (0x0806)

▼ Address Resolution Protocol (request)

Hardware type: Ethernet (1)  
Protocol type: IPv4 (0x0800)  
Hardware size: 6  
Protocol size: 4  
Opcode: request (1)  
Sender MAC address: 04:d4:c4:bd:ba:48 (04:d4:c4:bd:ba:48)  
Sender IP address: 192.168.50.1  
Target MAC address: 00:00:00\_00:00:00 (00:00:00:00:00:00)  
Target IP address: 192.168.50.183

0000	78	4f	43	8b	47	aa	04	d4	c4	bd	ba	48	08	06	00	01	x0C.G....H....
0010	08	00	06	04	00	01	04	d4	c4	bd	ba	48	c0	a8	32	01	.....H..2.
0020	00	00	00	00	00	00	c0	a8	32	b7							.....2.

Address Resolution Protocol (arp), 28 bytes

Packets: 44663 · Displayed: 69 (0.2%)

Profile: Default

Wi-Fi: en0

No.	Time	Source	Destination	Protocol	Length	Info
133	0.950965	192.168.50.183	192.168.50.1	DNS	87	Standard query 0x2fb7 PTR 183.50.168.192.in-addr.ar...
134	0.954629	192.168.50.1	192.168.50.183	DNS	113	Standard query response 0x2fb7 PTR 183.50.168.192.i...
5595	17.456059	192.168.50.183	192.168.50.1	DNS	79	Standard query 0xd81d A www.comp.nus.edu.sg
5598	17.466147	192.168.50.1	192.168.50.183	DNS	131	Standard query response 0xd81d A www.comp.nus.edu.s...

```
change -- bash -- 80x24
Last login: Wed Mar 11 14:59:24 on ttys000
Hello running .profile
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
OfficeLapTop:~ change$ nslookup www.comp.nus.edu.sg
Server: 192.168.50.1#53
Address: 192.168.50.1#53

Non-authoritative answer:
www.comp.nus.edu.sg canonical name = t36chsc.x.incapdns.net.
Name: t36chsc.x.incapdns.net
Address: 45.60.35.225

OfficeLapTop:~ change$ 
```

Hex dump of the captured DNS response:

0000	04 d4 c4 bd ba 48 78 4f	43 8b 47 aa 08 00 45 00	.....Hx0 C.G...E.
0010	00 49 96 5f 00 00 ff 11	3f 3b c0 a8 32 b7 c0 a8	.I._.... ?;..2...
0020	32 01 ee 28 00 35 00 35	74 42 2f b7 01 00 00 01	2..,(.5.5 tB/....
0030	00 00 00 00 00 00 03 31	38 33 02 35 30 03 31 36	.....1 83.50.16
0040	38 03 31 39 32 07 69 6e	2d 61 64 64 72 04 61 72	8.192.in -addr.ar...
0050	70 61 00 00 0c 00 01		pa.....

Wi-Fi: en0							
No.	Time	Source	Destination	Protocol	Length	Info	
133	0.950965	192.168.50.183	192.168.50.1	DNS	87	Standard query	0x2fb7 PTR 183.50.168.192.in-addr.arpa
134	0.954629	192.168.50.1	192.168.50.183	DNS	113	Standard query response	0x2fb7 PTR 183.50.168.192.in-addr.a...
5595	17.456059	192.168.50.183	192.168.50.1	DNS	79	Standard query	0xd81d A www.comp.nus.edu.sg
5598	17.466147	192.168.50.1	192.168.50.183	DNS	131	Standard query response	0xd81d A www.comp.nus.edu.sg CNAME ...

► Frame 5595: 79 bytes on wire (632 bits), 79 bytes captured (632 bits) on interface 0

▼ Ethernet II, Src: Apple\_8b:47:aa (78:4f:43:8b:47:aa), Dst: 04:d4:c4:bd:ba:48 (04:d4:c4:bd:ba:48)

- Destination: 04:d4:c4:bd:ba:48 (04:d4:c4:bd:ba:48)
- Source: Apple\_8b:47:aa (78:4f:43:8b:47:aa)
- Type: IPv4 (0x0800)

► Internet Protocol Version 4, Src: 192.168.50.183, Dst: 192.168.50.1

► User Datagram Protocol, Src Port: 54703, Dst Port: 53

▼ Domain Name System (query)

[\[Response In: 5598\]](#)

Transaction ID: 0xd81d

► Flags: 0x0100 Standard query

Questions: 1

Answer RRs: 0

Authority RRs: 0

Additional RRs: 0

▼ Queries

▼ www.comp.nus.edu.sg: type A, class IN

Name: www.comp.nus.edu.sg

[Name Length: 19]

[Label Count: 5]

Type: A (Host Address) (1)

Class: IN (0x0001)

```

0000 04 d4 c4 bd ba 48 78 4f  43 8b 47 aa 08 00 45 00  ....Hx0 C.G...E.
0010 00 41 4d d7 00 00 40 11  46 cc c0 a8 32 b7 c0 a8  .AM...@. F...2...
0020 32 01 d5 af 00 35 00 2d  da 7d d8 1d 01 00 00 01  2....5.- .}.....
0030 00 00 00 00 00 03 77 77 04 63 6f 6d 70 03  .....w ww.comp.
0040 6e 75 73 03 65 64 75 02  73 67 00 00 01 00 01  nus.edu. sg.....

```

No.	Time	Source	Destination	Protocol	Length	Info
133	0.950965	192.168.50.183	192.168.50.1	DNS	87	Standard query 0x2fb7 PTR 183.50.168.192.in-addr.arpa
134	0.954629	192.168.50.1	192.168.50.183	DNS	113	Standard query response 0x2fb7 PTR 183.50.168.192.in-addr...
5595	17.456059	192.168.50.183	192.168.50.1	DNS	79	Standard query 0xd81d A www.comp.nus.edu.sg
5598	17.466147	192.168.50.1	192.168.50.183	DNS	131	Standard query response 0xd81d A www.comp.nus.edu.sg CNAME

### ▼ Domain Name System (response)

[Request In: 5595]

[Time: 0.010088000 seconds]

Transaction ID: 0xd81d

► Flags: 0x8180 Standard query response, No error

Questions: 1

Answer RRs: 2

Authority RRs: 0

Additional RRs: 0

#### ▼ Queries

▼ www.comp.nus.edu.sg: type A, class IN

Name: www.comp.nus.edu.sg

[Name Length: 19]

[Label Count: 5]

Type: A (Host Address) (1)

Class: IN (0x0001)

#### ▼ Answers

► www.comp.nus.edu.sg: type CNAME, class IN, cname t36chsc.x.incapdns.net

▼ t36chsc.x.incapdns.net: type A, class IN, addr 45.60.35.225

Name: t36chsc.x.incapdns.net

Type: A (Host Address) (1)

Class: IN (0x0001)

Time to live: 30

Data length: 4

Address: 45.60.35.225

0010	00 75 00 00 40 00 40 11 54 6f c0 a8 32 01 c0 a8 .u..@. To...2...
0020	32 b7 00 35 d5 af 00 61 35 dc d8 1d 81 80 00 01 2.5...a 5.....
0030	00 02 00 00 00 00 03 77 77 77 04 63 6f 6d 70 03 .....w ww.comp.
0040	6e 75 73 03 65 64 75 02 73 67 00 00 01 00 01 c0 nus.edu. sg.....
0050	0c 00 05 00 01 00 00 40 83 00 18 07 74 33 36 63 .....@ ....t36c
0060	68 73 63 01 78 08 69 6e 63 61 70 64 6e 73 03 6e hsc.x.in capdns.n
0070	65 74 00 c0 31 00 01 00 01 00 00 00 1e 00 04 2d et.1.... ....-
0080	3c 23 e1 <#.

# TLS: set display filter to “ssl”. Find the TLS handshake (authenticated key ex

Wi-Fi: en0

ssl Expression... + Apply this filter

No.	Time	Source	Destination	Protocol	Length	Info
14	5.908086	192.168.50.183	137.132.1.170	TLSv1.2	103	Application Data
21	5.931462	192.168.50.183	103.1.139.51	TLSv1.2	583	Client Hello
25	5.940121	103.1.139.51	192.168.50.183	TLSv1.2	1514	Server Hello
31	5.941714	103.1.139.51	192.168.50.183	TLSv1.2	1514	Certificate[TCP segment of a reassembled PDU]
32	5.947015	103.1.139.51	192.168.50.183	TLSv1.2	900	Certificate Status, Server Key Exchange, Server He...
35	5.950806	192.168.50.183	103.1.139.51	TLSv1.2	192	Client Key Exchange, Change Cipher Spec, Hello Req...
37	5.957259	103.1.139.51	192.168.50.183	TLSv1.2	324	New Session Ticket, Change Cipher Spec, Encrypted ...
47	5.979696	192.168.50.183	103.1.139.51	TLSv1.2	243	Application Data
48	5.979787	192.168.50.183	103.1.139.51	TLSv1.2	1260	Application Data
49	5.991153	103.1.139.51	192.168.50.183	TLSv1.2	143	Application Data

Ethernet II, Src: 04:d4:c4:bd:ba:48 (04:d4:c4:bd:ba:48), Dst: Apple\_8b:47:aa (78:4f:43:8b:47:aa)

- ▶ Destination: Apple\_8b:47:aa (78:4f:43:8b:47:aa)
- ▶ Source: 04:d4:c4:bd:ba:48 (04:d4:c4:bd:ba:48)
- Type: IPv4 (0x0800)

Internet Protocol Version 4, Src: 103.1.139.51, Dst: 192.168.50.183

Transmission Control Protocol, Src Port: 443, Dst Port: 61643, Seq: 4097, Ack: 518, Len: 1448

[4 Reassembled TCP Segments (4349 bytes): #25(1365), #26(1448), #27(1200), #31(336)]

Secure Sockets Layer

TLSv1.2 Record Layer: Handshake Protocol: Certificate

- Content Type: Handshake (22)
- Version: TLS 1.2 (0x0303)
- Length: 4344

Handshake Protocol: Certificate

- Handshake Type: Certificate (11)
- Length: 4340
- Certificates Length: 4337

Certificates (4337 bytes)

- Certificate Length: 1909
- Certificate: 3082077130820659a003020102021100bcfa1cb4f9bf2c69... (id-at-commonName=www.dbs.com.sg,id-at-serialNumber=196800306E,id-at-expirationTime=2023-06-11T12:00:00Z,id-at-issuerName=Entrust Certification Authority - L1M,id-at-organizationName=Entrust Certification Authority - L1M,id-at-localityName=Singapore,id-at-stateOrProvinceName=Singapore,id-at-countryName=SG)
- Certificate Length: 1329
- Certificate: 3082052d30820415a003020102020c61a1e7d20000000051... (id-at-commonName=Entrust Certification Authority - L1M,id-at-organizationName=Entrust Certification Authority - L1M,id-at-localityName=Singapore,id-at-stateOrProvinceName=Singapore,id-at-countryName=SG)
- Certificate Length: 1090
- Certificate: 3082043e30820326a00302010202044a538c28300d06092a... (id-at-commonName=Entrust Root Certification Authority - G2,id-at-organizationName=Entrust Root Certification Authority - G2,id-at-localityName=Singapore,id-at-stateOrProvinceName=Singapore,id-at-countryName=SG)

Frame	Length	Hex	Dec	Text
0000	1514	78 4f 43 8b 47 aa 04 d4 c4 bd ba 48 08 00 45 00	x0C.G... .H..E.	
0010	1514	05 dc 0b 2e 40 00 3c 06 48 5a 67 01 8b 33 c0 a8	....@.<. Hzg..3..	
0020	1514	32 b7 01 bb f0 cb 73 1f 02 1a 4c 57 ff c0 80 10	2.....S. ..LW....	
0030	1514	00 eb d7 fb 00 00 01 01 08 0a 72 43 42 28 38 37	..... .rCB(87	
0040	1514	fb 2d 55 1d 0f 01 01 ff 04 04 03 02 01 06 30 0f	.-U..... .0.....0.	
0050	1514	06 03 55 1d 13 01 01 ff 04 05 30 03 01 01 ff 30	.U..... .0.....0.	

Frame (1514 bytes) Reassembled TCP (4349 bytes)

Packets: 6903 · Displayed: 4407 (63.8%) Profile: Default

Wi-Fi: en0

ssl

No.	Time	Source	Destination	Protocol	Length	Info
14	5.908086	192.168.50.183	137.132.1.170	TLSv1.2	103	Application Data
21	5.931462	192.168.50.183	103.1.139.51	TLSv1.2	583	Client Hello
25	5.940121	103.1.139.51	192.168.50.183	TLSv1.2	1514	Server Hello
31	5.941714	103.1.139.51	192.168.50.183	TLSv1.2	1514	Certificate[TCP segment of a reassembled PDU]
32	5.947015	103.1.139.51	192.168.50.183	TLSv1.2	900	Certificate Status, Server Key Exchange, Server He...
35	5.950806	192.168.50.183	103.1.139.51	TLSv1.2	192	Client Key Exchange, Change Cipher Spec, Hello Req...
37	5.957259	103.1.139.51	192.168.50.183	TLSv1.2	324	New Session Ticket, Change Cipher Spec, Encrypted ...
47	5.979696	192.168.50.183	103.1.139.51	TLSv1.2	243	Application Data
48	5.979787	192.168.50.183	103.1.139.51	TLSv1.2	1260	Application Data
49	5.991153	103.1.139.51	192.168.50.183	TLSv1.2	143	Application Data

► Frame 47: 243 bytes on wire (1944 bits), 243 bytes captured (1944 bits) on interface 0

▼ Ethernet II, Src: Apple\_8b:47:aa (78:4f:43:8b:47:aa), Dst: 04:d4:c4:bd:ba:48 (04:d4:c4:bd:ba:48)

- Destination: 04:d4:c4:bd:ba:48 (04:d4:c4:bd:ba:48)
- Source: Apple\_8b:47:aa (78:4f:43:8b:47:aa)
- Type: IPv4 (0x0800)

► Internet Protocol Version 4, Src: 192.168.50.183, Dst: 103.1.139.51

► Transmission Control Protocol, Src Port: 61643, Dst Port: 443, Seq: 644, Ack: 6637, Len: 177

▼ Secure Sockets Layer

- TLSv1.2 Record Layer: Application Protocol: http2
- Content Type: Application Data (23)
- Version: TLS 1.2 (0x0303)
- Length: 172
- Encrypted Application Data: 0000000000000001ca473341c448ac15f91eb54da8ccaccb...

Encrypted data

0000	04	d4	c4	bd	ba	48	78	4f	43	8b	47	aa	08	00	45	00	.....Hx0 C.G...E
0010	00	e5	00	00	40	00	40	06	54	7f	c0	a8	32	b7	67	01	....@. T...2.g.
0020	8b	33	f0	cb	01	bb	4c	58	00	3e	73	1f	0c	06	80	18	.3....LX .>s...
0030	08	00	35	b7	00	00	01	01	08	0a	38	37	fb	57	72	43	.5..... .87.WrC
0040	42	38	17	03	03	00	ac	00	00	00	00	00	01	ca	B8.....	.....	B8.....
0050	47	33	41	c4	48	ac	15	f9	1e	b5	4d	a8	cc	ac	cb	cd	G3A.H... .M...
0060	7f	0d	dd	43	ea	21	cd	10	8f	4f	2b	b0	2e	be	dc	b3	...C.!... .0+....
0070	d3	c7	dd	e7	a0	82	e0	7b	c4	f2	13	f3	58	a0	31	95	.....{ ....X.1.

# Looking at all IP packets from the ip.adder 103.1.139.51

Note that at TCP layer, there are many other more interactions (3 way-handshake, the “ACK” etc). Details not required in this module.

Wi-Fi 600

No.	Time	Source	Destination	Protocol	Length	Info
18	5.922834	192.168.50.183	103.1.139.51	TCP	78	61643 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS...
19	5.928521	103.1.139.51	192.168.50.183	TCP	74	443 → 61643 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 ...
20	5.928606	192.168.50.183	103.1.139.51	TCP	66	61643 → 443 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSva...
21	5.931462	192.168.50.183	103.1.139.51	TLSv1.2	583	Client Hello
24	5.940109	103.1.139.51	192.168.50.183	TCP	66	443 → 61643 [ACK] Seq=1 Ack=518 Win=30080 Len=0 TSv...
25	5.940121	103.1.139.51	192.168.50.183	TLSv1.2	1514	Server Hello
26	5.940215	103.1.139.51	192.168.50.183	TCP	1514	[TCP segment of a reassembled PDU]
27	5.940217	103.1.139.51	192.168.50.183	TCP	1266	[TCP segment of a reassembled PDU]
28	5.940344	192.168.50.183	103.1.139.51	TCP	66	61643 → 443 [ACK] Seq=518 Ack=2897 Win=128832 Len=0...
29	5.940345	192.168.50.183	103.1.139.51	TCP	66	61643 → 443 [ACK] Seq=518 Ack=4097 Win=127616 Len=0...
30	5.940606	192.168.50.183	103.1.139.51	TCP	66	[TCP Window Update] 61643 → 443 [ACK] Seq=518 Ack=4...
31	5.941714	103.1.139.51	192.168.50.183	TLSv1.2	1514	Certificate[TCP segment of a reassembled PDU]
32	5.947015	103.1.139.51	192.168.50.183	TLSv1.2	900	Certificate Status, Server Key Exchange, Server Hel...
33	5.947069	192.168.50.183	103.1.139.51	TCP	66	61643 → 443 [ACK] Seq=518 Ack=6379 Win=130176 Len=0...
35	5.950806	192.168.50.183	103.1.139.51	TLSv1.2	192	Client Key Exchange, Change Cipher Spec, Hello Requ...
37	5.957259	103.1.139.51	192.168.50.183	TLSv1.2	324	New Session Ticket, Change Cipher Spec, Encrypted H...
38	5.957365	192.168.50.183	103.1.139.51	TCP	66	61643 → 443 [ACK] Seq=644 Ack=6637 Win=130752 Len=0...
47	5.979696	192.168.50.183	103.1.139.51	TLSv1.2	243	Application Data

▶ Frame 18: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface 0

▼ Ethernet II, Src: Apple\_8b:47:aa (78:4f:43:8b:47:aa), Dst: 04:d4:c4:bd:ba:48 (04:d4:c4:bd:ba:48)

▶ Destination: 04:d4:c4:bd:ba:48 (04:d4:c4:bd:ba:48)

▶ Source: Apple\_8b:47:aa (78:4f:43:8b:47:aa)

Type: IPv4 (0x0800)

▶ Internet Protocol Version 4, Src: 192.168.50.183, Dst: 103.1.139.51

▶ Transmission Control Protocol, Src Port: 61643, Dst Port: 443, Seq: 0, Len: 0

0000	04 d4 c4 bd ba 48 78 4f 43 8b 47 aa 08 00 45 00	.....Hx0 C.G...E
0010	00 40 00 00 40 00 40 06 55 24 c0 a8 32 b7 67 01	@. @. @. U\$..2.g.
0020	8b 33 f0 cb 01 bb 4c 57 fd ba 00 00 00 00 b0 02	.3....LW .....
0030	ff ff e1 70 00 00 02 04 05 b4 01 03 03 06 01 01	....p.....
0040	08 0a 38 37 fb 26 00 00 00 00 04 02 00 00	..87.&.. .....

We can use nslookup to find out more about an ipaddress.

(the is called reverser DNS lookup where find the domain name from ip-address.) We can also find the geolocation by using some services from the web. Does not always work.

Wi-Fi: en0

ssl

No.	Time	Source	Destination	Protocol	Length	Info
1693	5.181749	192.168.50.183	137.132.21.14	TLSv1.2	97	Encrypted Alert
1695	5.188478	137.132.21.14	192.168.50.183	TLSv1.2	1299	Application Data
1697	5.188733	192.168.50.183	137.132.21.14	TLSv1.2	97	Encrypted Alert
1708	5.210304	192.168.50.183	137.132.21.14	TLSv1.2	299	Client Hello
1709	5.213711	137.132.21.14	192.168.50.183	TLSv1.2	1299	Application Data
1711	5.214000	192.168.50.183	137.132.21.14	TLSv1.2	97	Encrypted Alert
1719	5.231818	192.168.50.183	137.132.21.14	TLSv1.2	299	Client Hello
1720	5.233407	137.132.21.14	192.168.50.183	TLSv1.2	215	Server Hello, Change Cipher Spec, Encrypted Handsh...
1722	5.233601	192.168.50.183	137.132.21.14	TLSv1.2	72	Change Cipher Spec
1723	5.233607	192.168.50.183	137.132.21.14	TLSv1.2	111	Encrypted Handshake Message
1724	5.233655	192.168.50.183	137.132.21.14	TLSv1.2	712	Application Data
1725	5.244499	137.132.21.14	192.168.50.183	TLSv1.2	215	Server Hello, Change Cipher Spec, Encrypted Handsh...
1727	5.244876	192.168.50.183	137.132.21.14	TLSv1.2	72	Change Cipher Spec
1728	5.244877	192.168.50.183	137.132.21.14			

Session ID: 4988c84304472c403a8890e6ba91abc097a  
Cipher Suite: TLS\_ECDHE\_RSA\_WITH\_AES\_256\_GCM\_SHA384  
Compression Method: null (0)  
Extensions Length: 17  
► Extension: renegotiation\_info  
► Extension: ec\_point\_formats  
► Extension: Extended Master Secret  
▼ TLSv1.2 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec  
Content Type: Change Cipher Spec (20)  
Version: TLS 1.2 (0x0303)  
Length: 1  
▼ Change Cipher Spec Message  
[Expert Info (Note/Sequence): This session reuses previously negotiated keys (Session resumption)]  
[This session reuses previously negotiated keys (Session resumption)]  
[Severity level: Note]  
[Group: Sequence]  
▼ TLSv1.2 Record Layer: Handshake Protocol: Encrypted Handshake Message  
Content Type: Handshake (22)  
Version: TLS 1.2 (0x0303)  
Length: 40

0000	78 4f 43 8b 47 aa 04 d4 c4 bd ba 48 08 00 45 00	x0C.G... . . . H..E.
0010	00 c9 fc 76 40 00 2d 06 be c6 89 84 15 0e c0 a8	....v0.-. . . . . . . .
0020	32 b7 01 bb d7 56 84 52 d6 ed 62 01 9c 9d 80 18	2....V.R . . . b.....
0030	00 eb 68 60 00 00 01 01 08 0a f2 76 a6 bf 38 2c	...h` . . . . . . . . . .
0040	38 de 16 03 03 00 5d 02 00 00 59 03 03 c2 c3 c7	8.....]. . . Y.....
0050	4f dc cf bc c8 6e a3 c0 83 df e9 60 a1 6f 8f 73	0....n. . . . . o.s
0060	0e 28 de 0a 82 7d 93 ee ee 5f ce d3 a7 20 49 88	.{...}. . . . . I.
0070	c8 43 04 47 2c 40 3a 88 90 e6 ba 91 ab c0 97 a5	.C.G,@: . . . . . . . .

Frame (frame), 215 bytes

Packets: 6939 · Displayed: 2832 (40.8%) · Dropped: 0 (0.0%) · Profile: Default

There are many more tools and features. (e.g. the “follow”,  
“Analyze”, “export” etc)

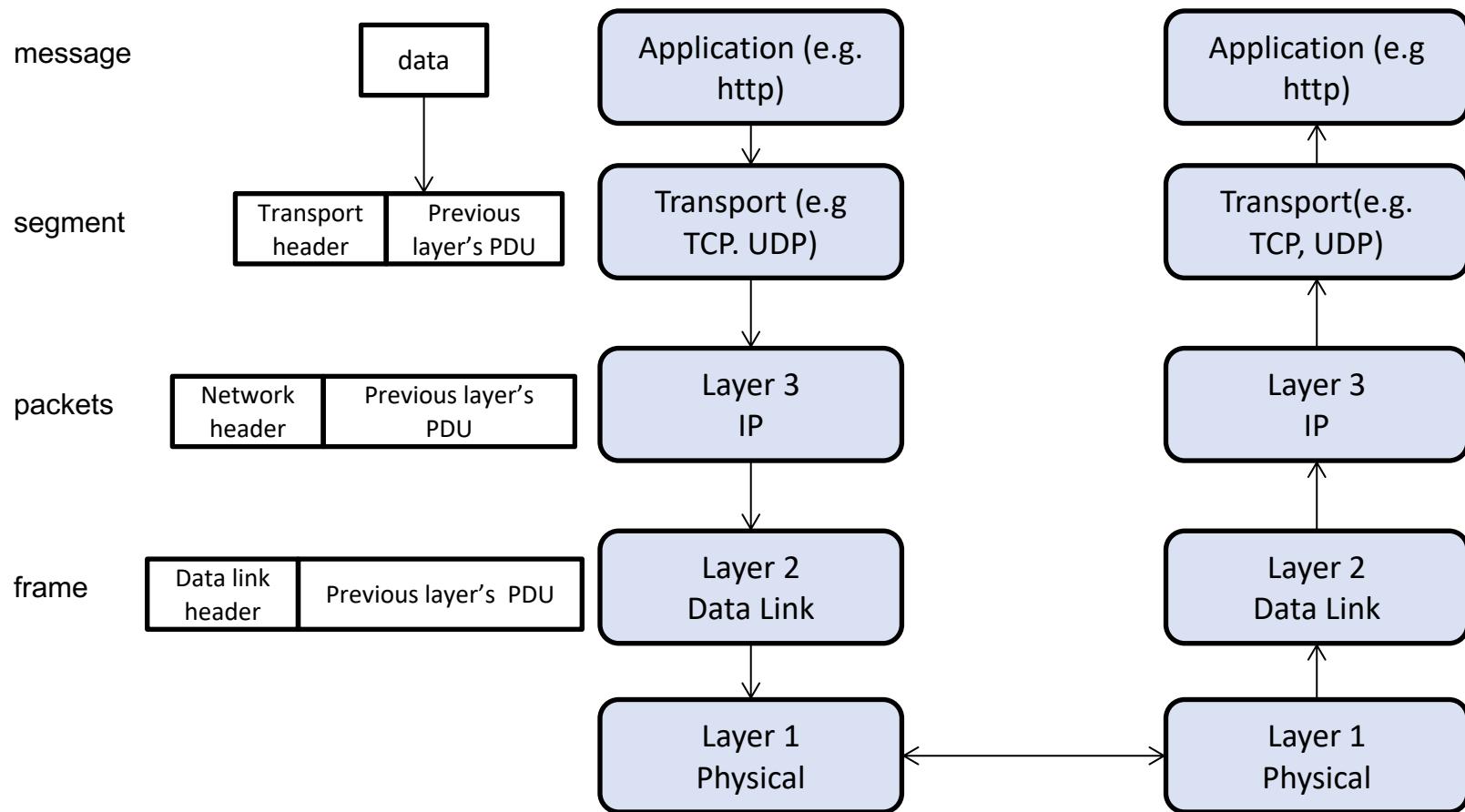
Explore to find out more.

## **5.5 Protection: Securing the communication channel using cryptography**

In the past few lectures, we have illustrated that cryptography techniques can be deployed to achieve confidentiality (encryption) and authenticity (mac, PKI, strong authentication) over a public communication channel, even if the adversary can sniff & spoof data.

There are many security protocols achieving that, but operates at different “layers”.

- The well-known TLS/SSL, WPA, IPSEC protect different layers.



## Remark on Layering

Very often, when referring to a security protocol, we indicate the “layer” the protocol targets to protect. (complication: some protections span across multiple layers, or do not provide full protection of the targeted layer.)

When analyzing an attack, it is also insightful to figure out at what layer the attacker resides. (complication: likewise, some attacks span across multiple layers. In such situations, trying hard to pin-point the layer could sometime be very confusing ).

A security protocol that protects layer k, would protect information in that layer and above.

Hence, if an attacker resides at layer 1, and there is a security protocol that protect layer 3, then information generated in layer 3 and above will be protected, but information generated in layer 2 would not be protected by the security protocol

# 1. SSL/TLS

- The SSL/TLS sit on top of Transport layer.
- We can imagine that, when an application (say browser or email agent) wants to send data to the other end point, it first passes the data and the address (ip address) to SSL/TLS. Next SSL/TLS first “protects” the data using encryption (confidentiality) and mac (authenticity), and then instructs the transport layer to send the protected data.

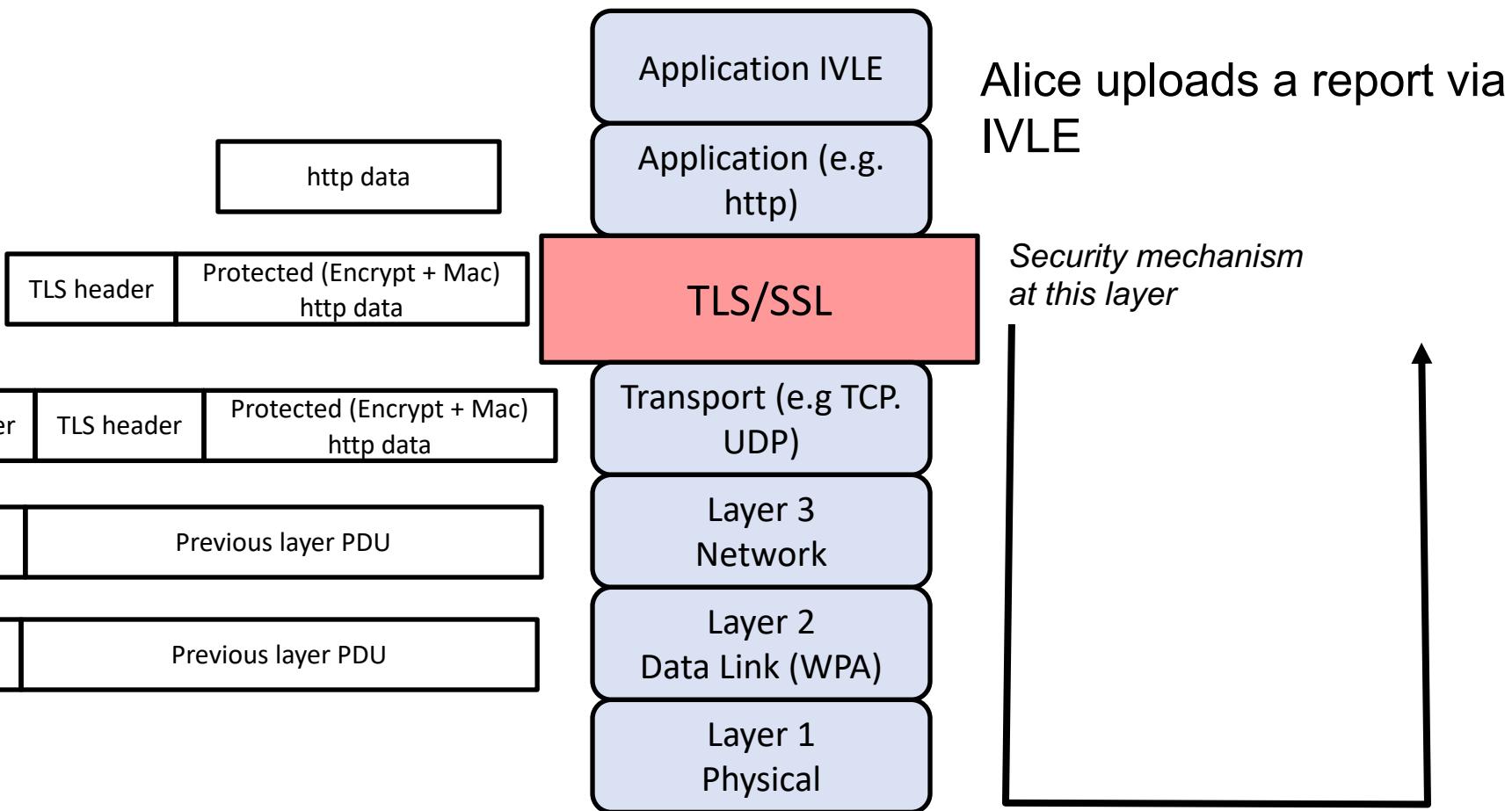
- E.g. Alice using IVLE to upload a report a.pdf to the IVLE server. Note that IVLE uses https, which in turn uses SSL/TLS.

Alice's machine carries the following:

1. The “IVLE” application passes the file a.pdf to https, and then to TLS.
2. TLS protects the data by encryption and mac.
3. TLS passes the protected data to the transport layer.

IVLE's server carries out the following:

1. The transport layer passes the protected data to TLS.
2. TLS decrypt the data and verify the mac for integrity.
3. TLS passes the decrypted data to IVLE's application.

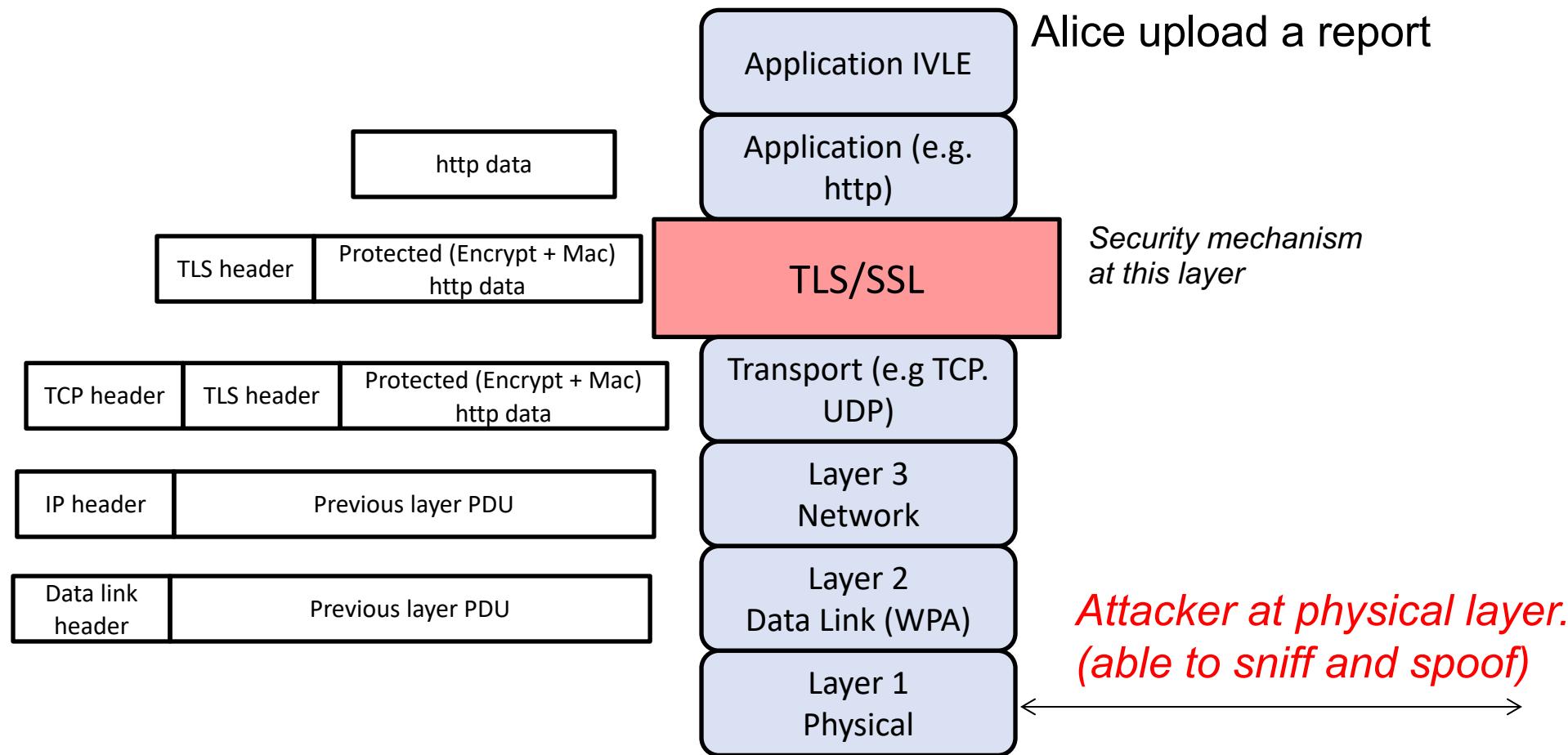


The receiver end-point decrypts the received data at the corresponding layer.

## Scenario 1

- Suppose that there is an attacker in the physical layer who can sniff and spoof message at that layer (i.e. MITM in the physical layer). For example, Alice uploading her report in a cafe using free wifi (without WPA protection). Hence anyone in the café has access to the physical layer and thus can sniff and spoof messages in that layer.
- Question: Can the attacker learn
  1. Alice's report?
  2. The fact that Alice is visiting IVLE website (i.e. can the attacker learn the ip-address)?

# Attacker at physical layer.



## Scenario 2

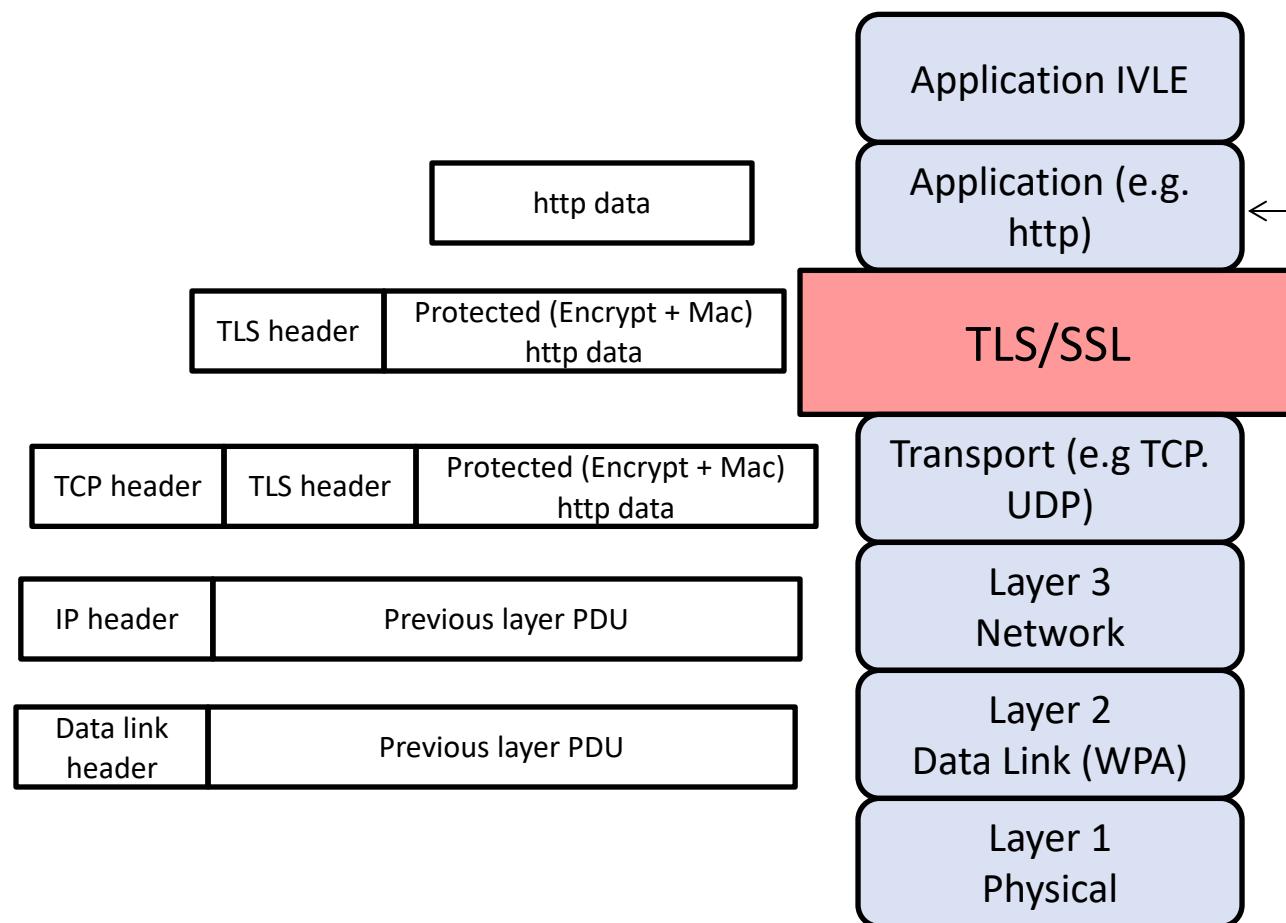
- Suppose that there is an adversary in the application layer.  
For example, a malicious java script injected into IVLE and being executed by Alice's browser.
- Question: Can the malicious script learn
  1. Alice's report?
  2. Alice's mac address? (yes and no)

# Attacker at application layer (e.g. Malware in browser)

Alice upload a report

*Malware in application  
(able to sniff and spoof)*

*Security mechanism  
at this layer*



## 2. WPA2

*Wifi Protected Access II (WPA2).* A popular protocol employed in home Wifi access point.

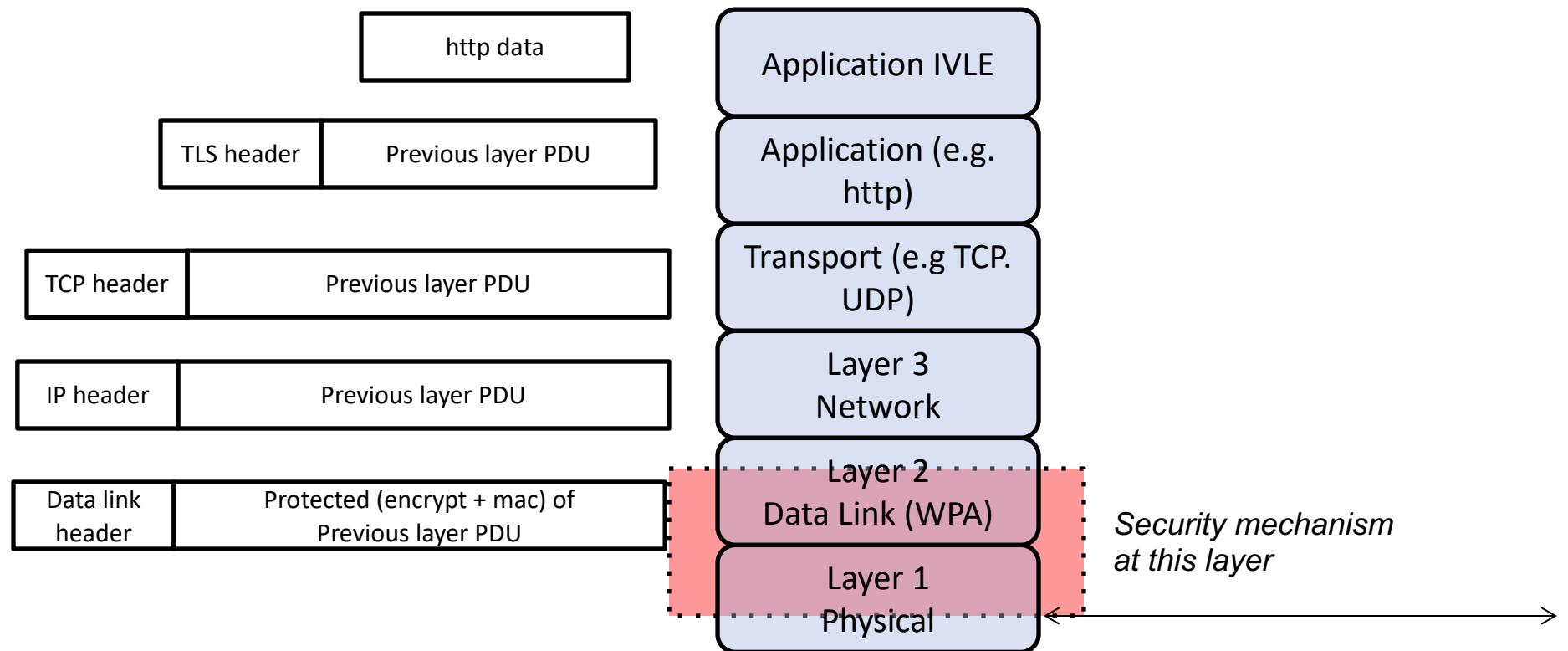
WPA2 provides protection at layer 2 (Link) and layer 1 (Physical). Not all information in layer 2 are protected.

See recent attack on WPA2.

**Key Reinstallation Attacks: Breaking WPA2 by forcing nonce reuse**  
<https://www.krackattacks.com/>

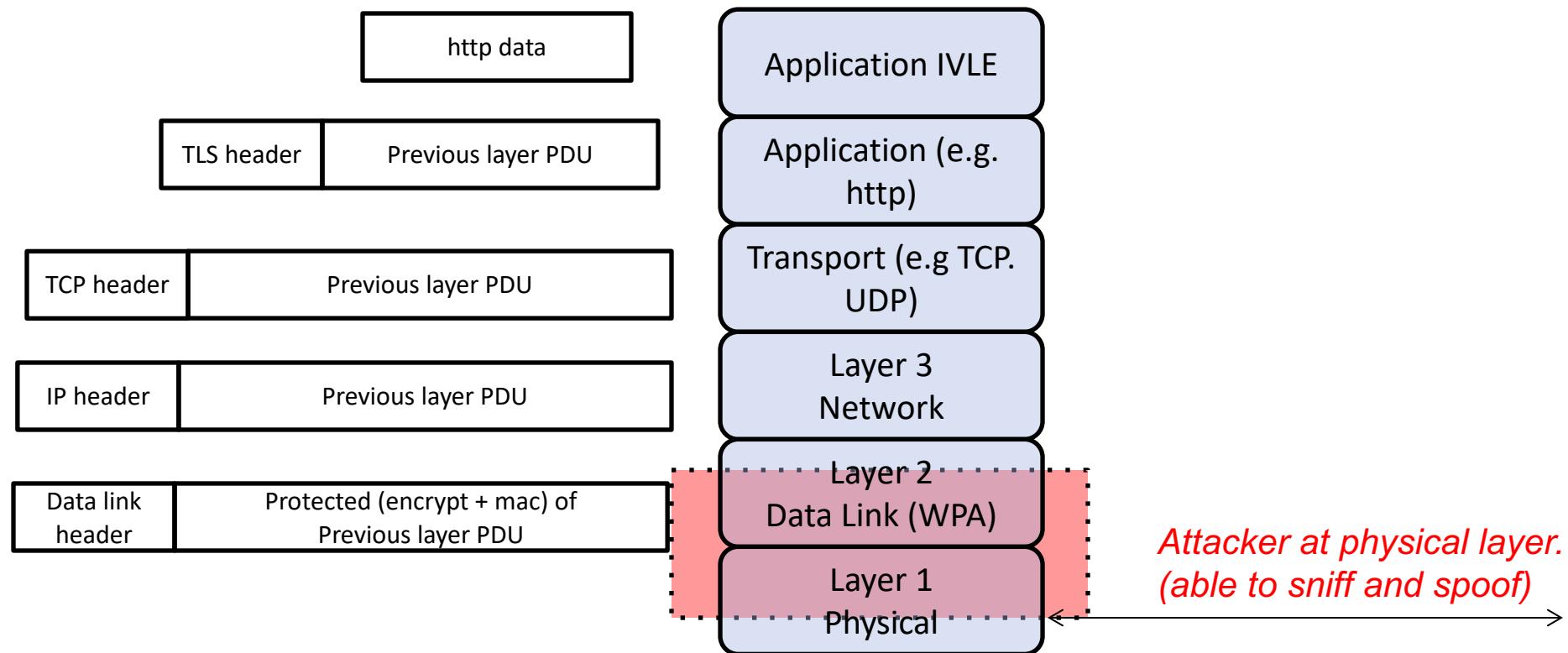
Research paper: <https://papers.mathyvanhoef.com/ccs2017.pdf>

## Alice uploading a report



# Attacker at physical layer

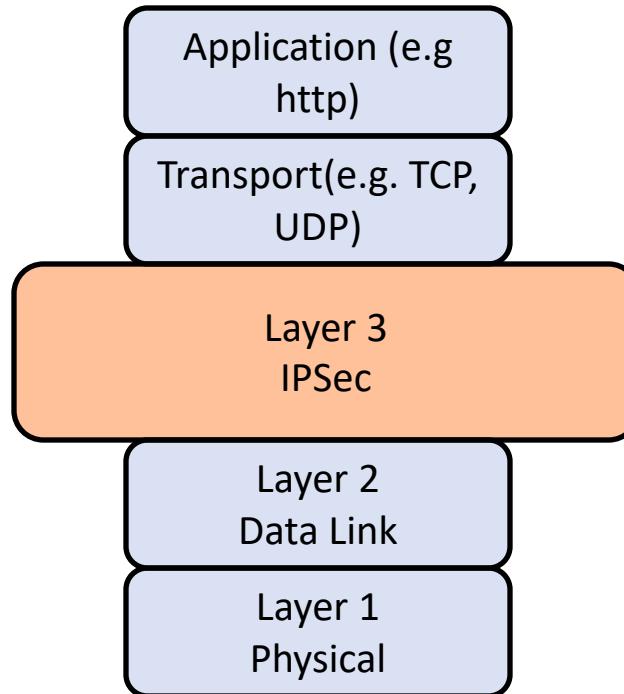
Alice uploading a report



Question: Can the attacker learn

1. Alice's report? No. Data is from Application layer and would be encrypted.
2. The fact that Alice is visiting IVLE website? No. Data is at Network layer and would be encrypted.
3. The mac address (link layer)? Not immediately clear from the figure above. Yes.

### 3. IPSec



E.g. IPSec provides “integrity/authenticity” protection of ip-address, but not confidentiality. Hence, attackers are unable to “spoof” the source ip-source, but can learn the source and destination ip-address of the sniffed packets.

# Remarks

- IPSec is a mechanism whose goal is to protect the IP layer.
- “Internet Protocol Security (IPsec) is a protocol suite for securing Internet Protocol (IP) communications by authenticating and encrypting each IP packet of a communication session. IPsec includes protocols for establishing mutual authentication between agents at the beginning of the session and negotiation of cryptographic keys to be used during the session. IPsec can be used in protecting data flows between a pair of hosts (*host-to-host*), between a pair of security gateways (*network-to-network*), or between a security gateway and a host (*network-to-host*).”
- “Internet Protocol security (IPsec) uses cryptographic security services to protect communications over Internet Protocol (IP) networks. IPsec supports network-level peer authentication, data origin authentication, data integrity, data confidentiality (encryption), and replay protection.”
- “IPsec is an end-to-end security scheme operating in the Internet Layer of the Internet Protocol Suite, while some other Internet security systems in widespread use, such as Transport Layer Security (TLS) and Secure Shell (SSH), operate in the upper layers at Application layer. Hence, only IPsec protects any application traffic over an IP network. Applications can be automatically secured by IPsec at the IP layer.”

-wiki

## 5.6. Firewall

[ZCC] ED Zwicky, S Cooper, DB Chapman, **Building Internet Firewalls: Internet and Web Security**, 2000

(free from books.google.com

[http://dbmanagement.info/Books/Others/O%27Reilly\\_Building\\_Internet\\_Firewalls\\_2nd\\_Edition.pdf](http://dbmanagement.info/Books/Others/O%27Reilly_Building_Internet_Firewalls_2nd_Edition.pdf) )

- Consider the computer network in an organization.
  - Some nodes contain more sensitive information than other. Example, Student examination record database server, vs public workstations in lecture hall.
  - Some nodes might have unpatched vulnerabilities. Example, when a patch is available, it might take some time (e.g. days, weeks, months) to patch all the systems.
  - Certain nodes might be compromised (e.g. An unhappy student who has access to the lab's workstation).
  - Certain protocols do not have protection mechanisms, or only light-weight protection. For example, DNS, arp, network printer, etc. We need to prevent attackers from accessing such protocol.
- We need to divide the computer network into segments and deny unnecessary access. (**Principle of least privilege, compartmentalization**)
- Firewall, Intrusion detection system (IDS) are tools to control access to the network.

## Remark: Principle of least privilege and compartmentalization

- The term “Principle of least privilege” was introduced in the design of OS, but was later adopted in other domain in security.
  - “The **principle of least privilege (PoLP**, also known as the **principle of minimal privilege** or the **principle of least authority**) requires that in a particular abstraction layer of a computing environment, every module (such as a process, a user, or a program, depending on the subject) must be able to access only the information and resources that are necessary for its legitimate purpose”  
[https://en.wikipedia.org/wiki/Principle\\_of\\_least\\_privilege#Implementation](https://en.wikipedia.org/wiki/Principle_of_least_privilege#Implementation)
- Compartmentalization: There isn’t a single well accepted definition. Generally, it refer to the notion of confining information within compartments.

# Firewall

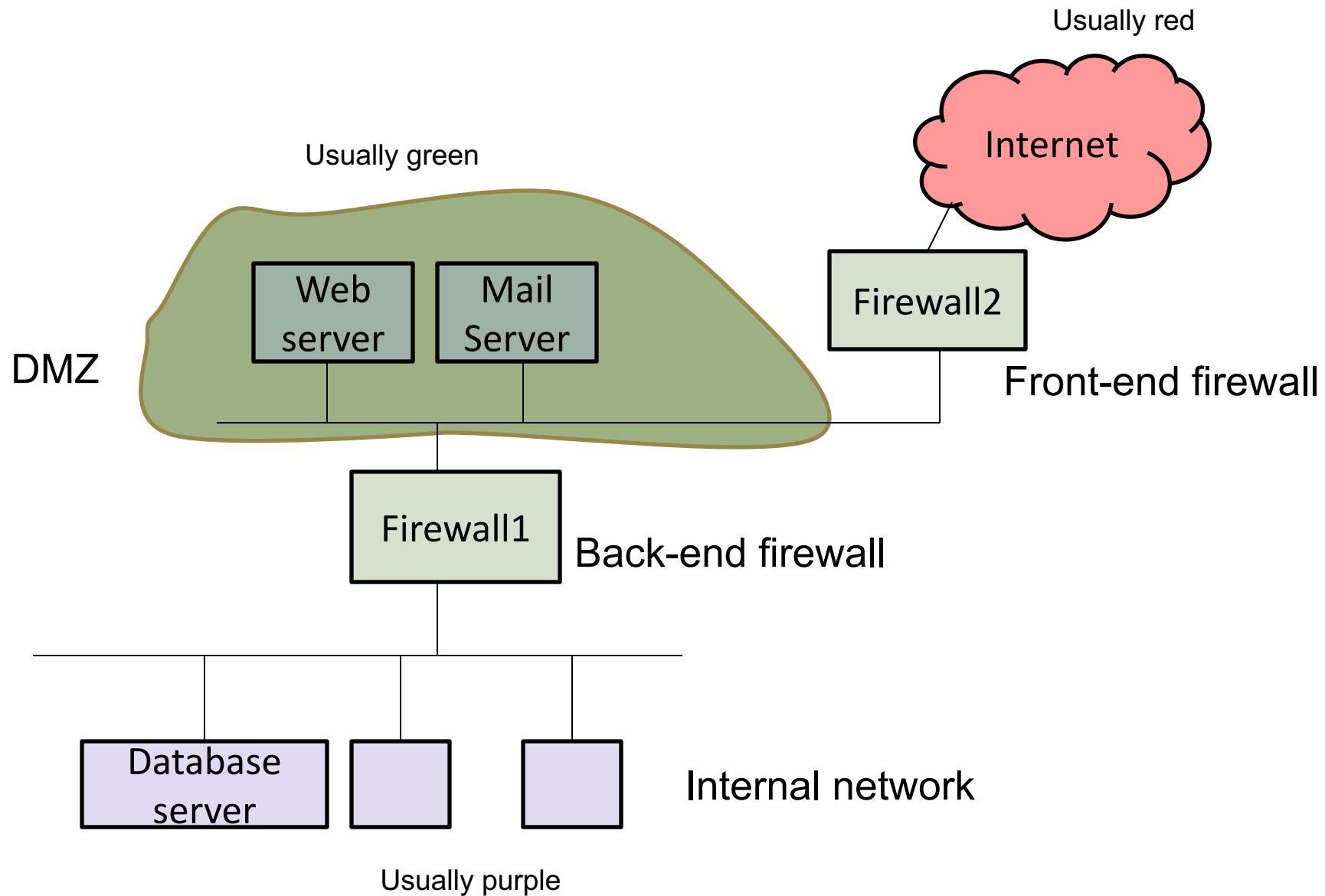
- Firewall: A Firewall controls what traffic is allowed to enter the network (*ingress* filtering), or leave the network (*egress* filtering).
- *Firewall are devices or programs that control the flow of network traffic between networks or hosts that employ differing security postures.*

(from Guidelines on Firewalls and Firewall Policy, NIST, special publication 800-41

<http://csrc.nist.gov/publications/nistpubs/800-41-Rev1/sp800-41-rev1.pdf> )

- DMZ: Demilitarized zone. A sub-network that exposes the organization's external service to the (untrusted) Internet.

# A typical 2-firewall setting.



## Packet filtering/screening ([ZCC Section 5.2])

Firewall's controls are achieved by “packets filtering” (aka screening). Filtering may occur in router, gateway/bridge, host, etc.

Packet filtering inspects every packet, typically only on the IP packet's header information. (if the payload is inspected, we call it ***deep packet inspection (DPI)***). Action taken after inspection could be

- Allow the packet to pass
- Just drop the packet
- Reject the packet (i.e. drop and inform the sender)
- Log info
- Notify system admin
- Modify the packet (for more advanced device).

# Example of firewall rules

- Drop packets with “source ip-address” not within the organization’s network. (*To stop attacks originated within the network*).
- Whitelist
  - Drop all packets except those specified in the white-list. (e.g. drop all except http, email protocol, and DNS)
- Blacklist
  - Accept all packets except those specified in the black-list. (e.g. allow https except ip-address in the black list).

## Firewall design (see [ZCC] ch8.5-12)

A firewall enforces a set of rules provided by the network administrator.

Example on the 2-firewall setting:

- Rules for Firewall1.
  - Block: HTTP.
  - Allow internal to Mail Server: SMTP, POP3
- Rules for Firewall2
  - Allow from anywhere to Mail Server: SMTP only

How the rules are to be specified differ on different devices and software. In this lecture, we consider a typical design (from [PF] pg 453)

Rule	Type	Direction	Source Address	Destination Address	Designation Port	Action
1	TCP	in	*	192.168.1.*	25	Permit
2	TCP	in	*	192.168.1.*	69	Permit
3	TCP	out	192.168.1.*	*	80	Permit
4	TCP	in	*	192.168.1.18	80	Permit
5	TCP	in	*	192.168.1.*	*	Deny
6	UDP	in	*	192.168.1.*	*	Deny

*Matching condition*

*action*

- The rules are processed sequentially starting from rule 1, 2, .... The first matching rule determines the action.
- The symbol “\*” matches any value. This is a symbol in “regular expression”.

# Types of firewall

- NIST's document (NIST 800-41) groups the firewalls into 3 types:
  1. Packet filters (Inspect packet header)
  2. Stateful Inspection (Deep packet inspection)
  3. Proxy (Modify packets)

# Intrusion Detection System (IDS)

An IDS system consists of a set of “sensors” gathering data. Sensors could be in the host, or network router. The data are analyzed for intrusion.

Three types of IDS:

- Attack signature Detection
  - The attack has specific, well-defined signature. For e.g. using certain port number, certain source ip address.
- Anomaly Detection
  - The IDS attempt to detect abnormal pattern. For e.g. a sudden surge of packets with certain port number.
- Behavior-based IDS
  - Can be viewed as a type of anomaly detection that focuses on human behavior. For e.g. The system might keep the profile of each user. It then tries to detect any user who deviates from the profile (e.g. start to download large files).

Popular open-source IDS: Snort. <https://www.snort.org/>

e.g. of snort rule (from <https://blog.rapid7.com/2016/12/09/understanding-and-configuring-snort-rules/> ):

```
log tcp !192.168.0/24 any -> 192.168.0.33 (msg: "mounted access" ; )
```

# **Other tools.**

## TLS/SSL scan

- Checking known vulnerability of a server.
- <https://www.ssllabs.com/ssltest/analyze.html?d=luminus.nus.edu.sg&hideResults=on>

## 5.7 Management

- Management needed in order to monitor and adjust network characteristic.
- Detail omitted. Optional. see [PF6.9]

Some terminologies:

- **Security Operations Center (SOC)**
  - a centralized unit in an organization that monitors the IT systems and deals with security issues.
- **Security Information and Event Management (SIEM)**
  - pronounced as “SIM”. Approaches and tools for SOC.
  - A popular system: Splunk ( <https://www.splunk.com/> )