Computer Networks-Lab 05

Instructor: Engr. Khuram Shahzad

CL30001 – Computer Networks-Lab

SEMESTER Fall 2021

JANUARY 28, 2022

NATIONAL UNIVERSTIY OF COMPUTER AND EMERGING SCIENCES, FAST- PESHAWAR CAMPUS
Department of Computer Science & Software Engineering

Computer Networks - Lab 05

OBJECTIVES

After these Lab students shall be able to perform

- Introduction with Wire shark
- Relation OSI and TCP/IP model
- OSI and TCP/IP Layer Analysis via Wireshark
- HTTP and HTTPS analysis using Wireshark
- Http packet sniffing on wire shark
- Http Get and Http Ok.

PRE-LAB READING ASSIGNMENT

Remember the delivered lecture carefully.

Table of Contents

OSI Network Layer Analysis via Wireshark

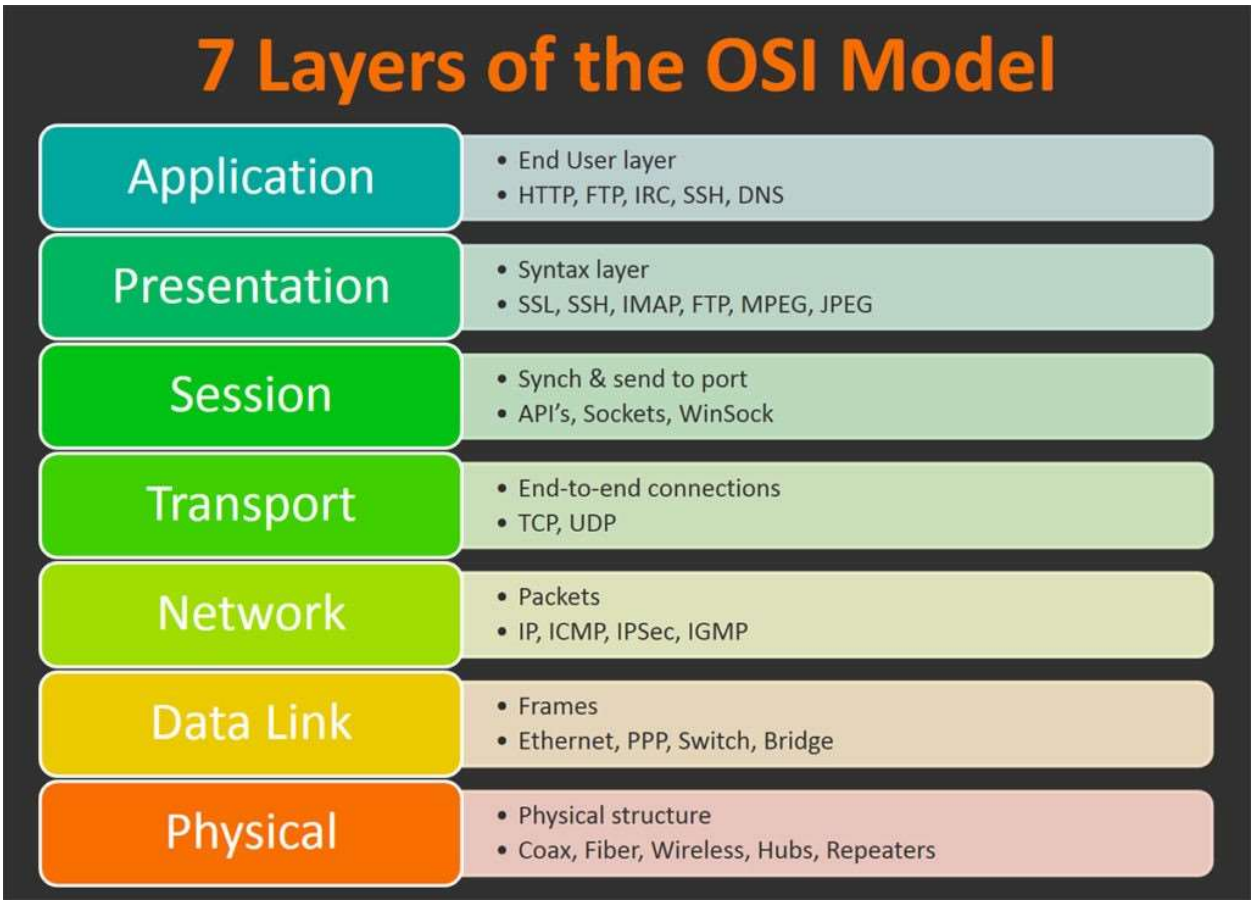
OSI model and TCP/IP model:

We all know that OSI (Open Systems Interconnection) is a reference model for how applications communicate over a network.
Here are the 7 layers according to OSI model:

Application Layer	**[Layer 7]**

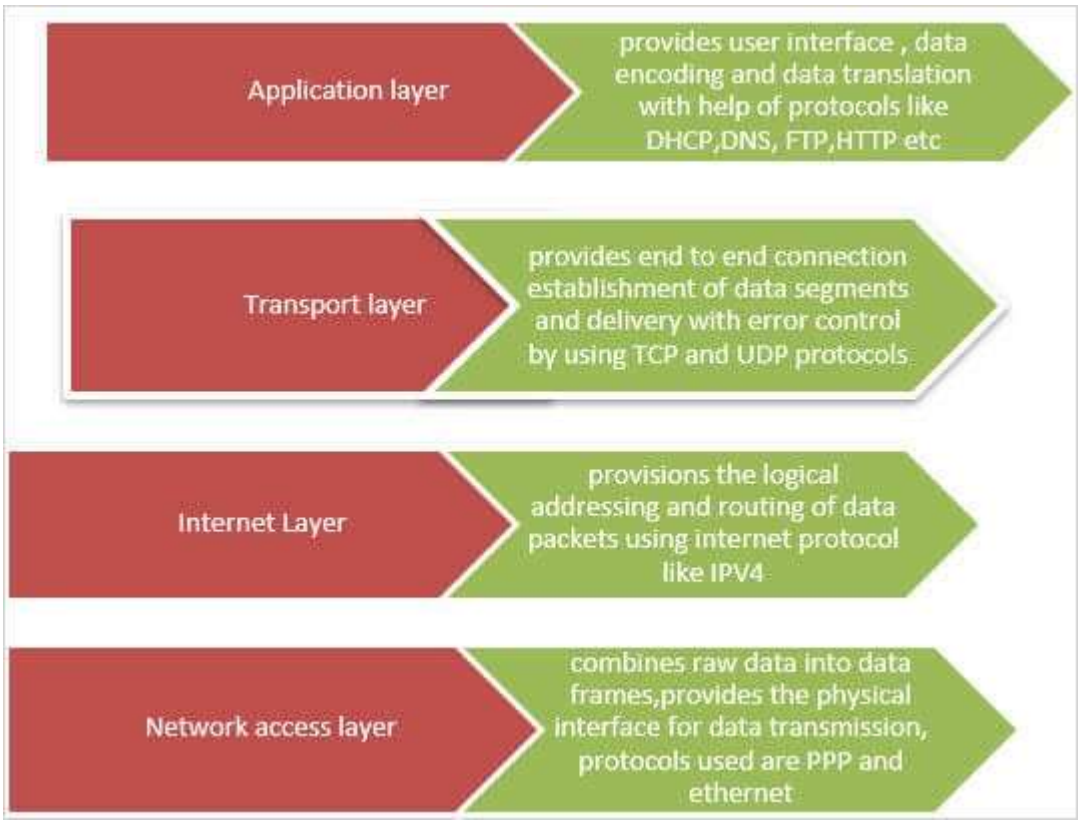
Presentation Layer	[Layer 6]
Session Layer	[Layer 5]
Transport Layer	[Layer 4]
Network Layer	[Layer 3]
Data Link Layer	[Layer 2]
Physical Layer	[Layer 1]

7 Application	**Key Responsibilities -** User Application Services **Common Protocols -** DNS; NFS; BOOTP; DHCP; SNMP; RMON; FTP; TFTP; SMTP; POP3; IMAP; NNTP; HTTP; Telnet **Scope-** Application data **Data Type can Handled -** User Data
6 Presentation	**Key Responsibilities -** Data Translation; Compression and Encryption **Common Protocols-** SSL; Shells and Redirectors; MIME **Scope-** Application data representations **Data Type can Handled -** Encoded User Data
5 Session	**Key Responsibilities -** Session Establishment, Management and Termination **Common Protocols-** NetBIOS, Sockets, Named Pipes, RPC **Scope-** Sessions between local or remote devices **Data Type can Handled -** Session
4 Transport	**Key Responsibilities** - Process-Level Addressing; Multiplexing/Demultiplexing; Connections; Segmentation and Reassembly Acknowledgments and Retransmissions, Flow Control **Common Protocols-** TCP and UDP; SPX; NetBEUI/NBF **Scope-** Communication between software processes **Data Type can Handled -** Datagram and Packets
3 Network	**Key Responsibilities-**Logical Addressing; Routing; Datagram Encapsulation; Fragmentation and Reassembly; Error Handling and Diagnostics **Common Protocols-** IP; IPv6; IP NAT; IPsec; Mobile IP; ICMP; IPX; DLC; PLP; Routing protocols such as RIP and BGP **Scope-** Messages between local or remote devices **Data Type can Handled-** Datagram and Packets
2 Datalink	**Key Responsibilities-** Logical Link Control; Media Access Control; Data Framing; Addressing; Error Detection and Handling; Defining Requirements of Physical Layer **Common Protocols-** EEE 802.2 LLC, Ethernet Family; Token Ring; FDDI and CDDI; IEEE 802.11 (WLAN, Wi-Fi); HomePNA; HomeRF; ATM; SLIP and PPP **Scope-** Low-level data messages between local devices **Data Type can Handled -** Frames
1 Physical	**Key Responsibilities-** Encoding and Signaling; Physical Data Transmission; Hardware Specifications; Topology and Design **Common Protocols-** (Physical layers of most of the technologies listed for the data link layer) **Scope-** Electrical or light signals sent between local devices **Data Type can Handled** - Bits



There is another network model which is TCP/IP.

Application Layer	**[Layer 4]**
Transport Layer	[Layer 3]
Internet Layer	[Layer 2]
Network Access Layer	[Layer 1]



Here are the 4 layers according to TCP/IP model:

Relation OSI and TCP/IP model:

Below is the relation between OSI model and TCP/IP model.

OSI Model	TCP/IP Model
Application Layer	Application Layer
Presentation Layer	
Session Layer	
Transport Layer	Transport Layer
Network Layer	Internet Layer
Data Link Layer	Network access Layer
Physical Layer	

Now the question comes, in **Wireshark what model we should be expecting?**
Actually in Wireshark we observe below layers

Application Layer [Layer 5]
Transport Layer [Layer 4]
Network Layer [Layer 3]
Data Link Layer [Layer 2]
Physical Layer [Layer 1]

Now we understand that the above layers are not exactly OSI or TCP/IP but a combination of both models.

Let's look into Wireshark capture and understand better.

What we see in Wireshark?

We will take some protocols as example and understand the layers through Wireshark. The interesting part is all protocol does not have all the layers.

Note:
As Wireshark decodes packets at Data Link layer so we will not get physical layer information always. In some cases, capturing adapter provides some physical layer information and can be displayed through Wireshark.
So here are the sequence layers seen in Wireshark

Data Link Layer

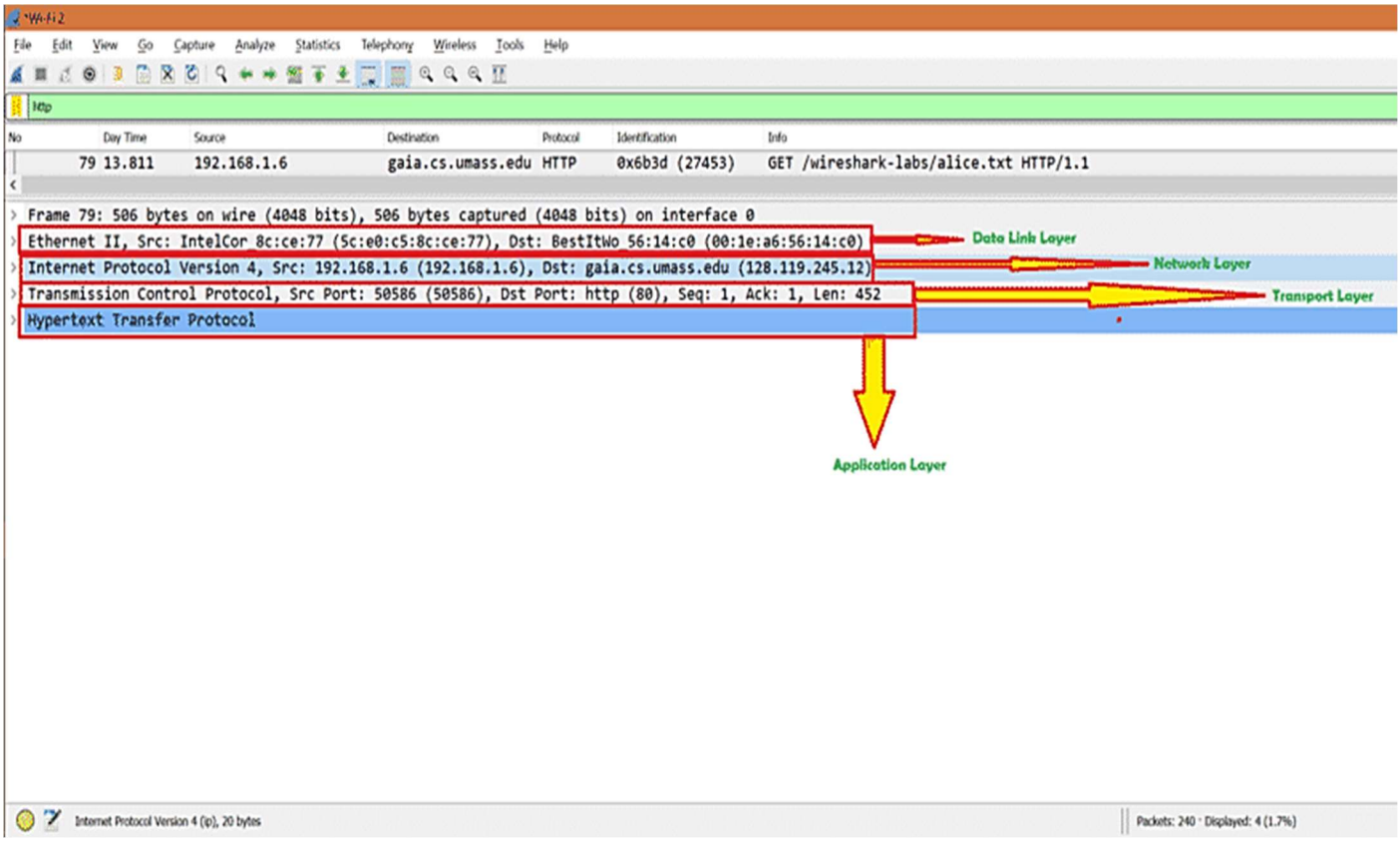
Network Layer
Transport Layer
Application Layer

Hope you understand that Wireshark is just showing in reverse order. If physical layer information is given to Wireshark then that time we should see physical layer information on top of Data link. See below picture.

HTTP	Physical Layer	*[It has 4*
	Data Link Layer	
	Network Layer	
	Transport Layer	
	Application Layer	

layers]:

You can follow below link to understand HTTP through WiresharkHere is the screenshot of a HTTP packet where we can see 4 layers.

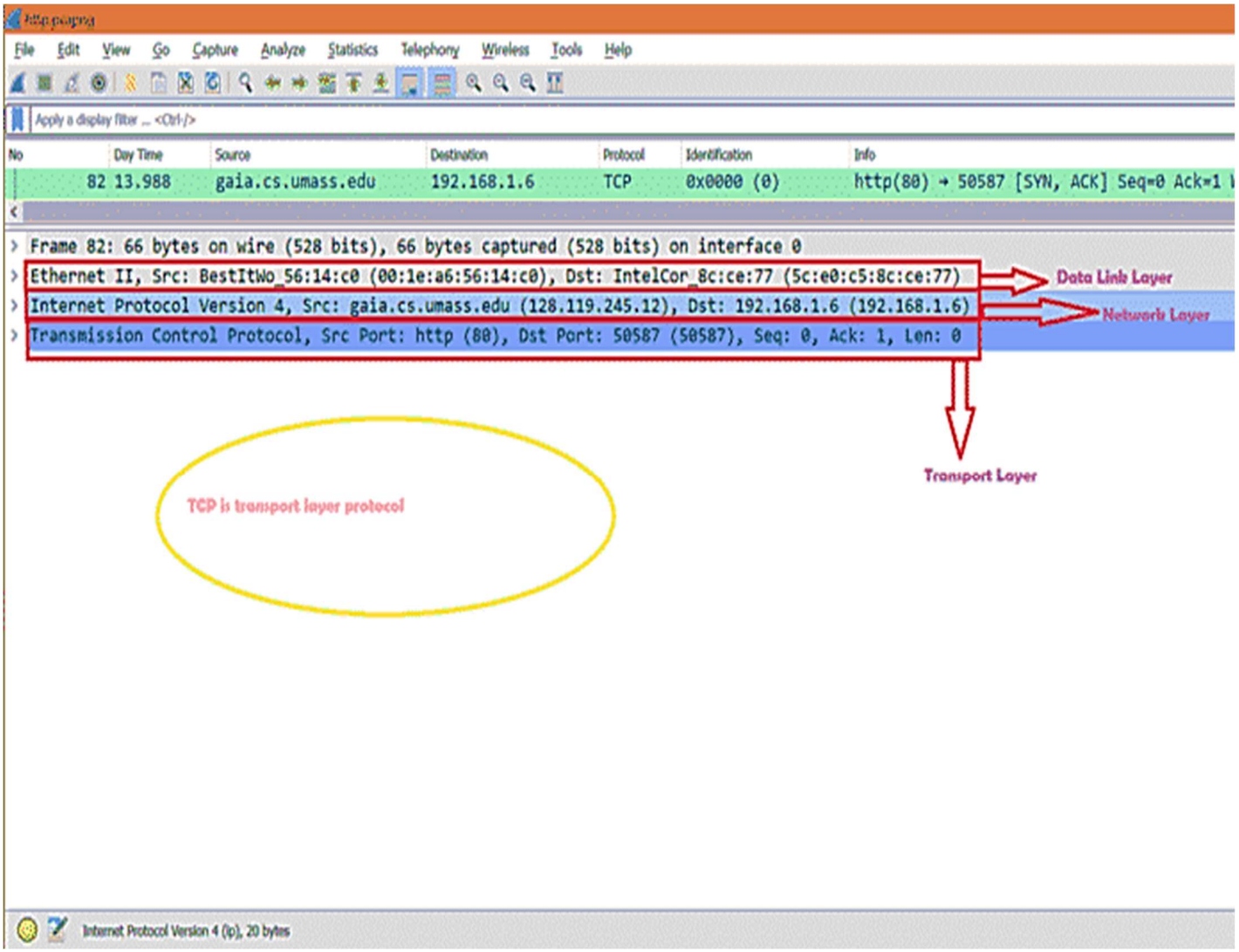


We know HTTP is an application layer so we see application layer also.

Now let's see a transport layer protocol in Wireshark.

TCP [It has 3 layers]:

Here is the screenshot of a TCP packet where we can see 3 layers.

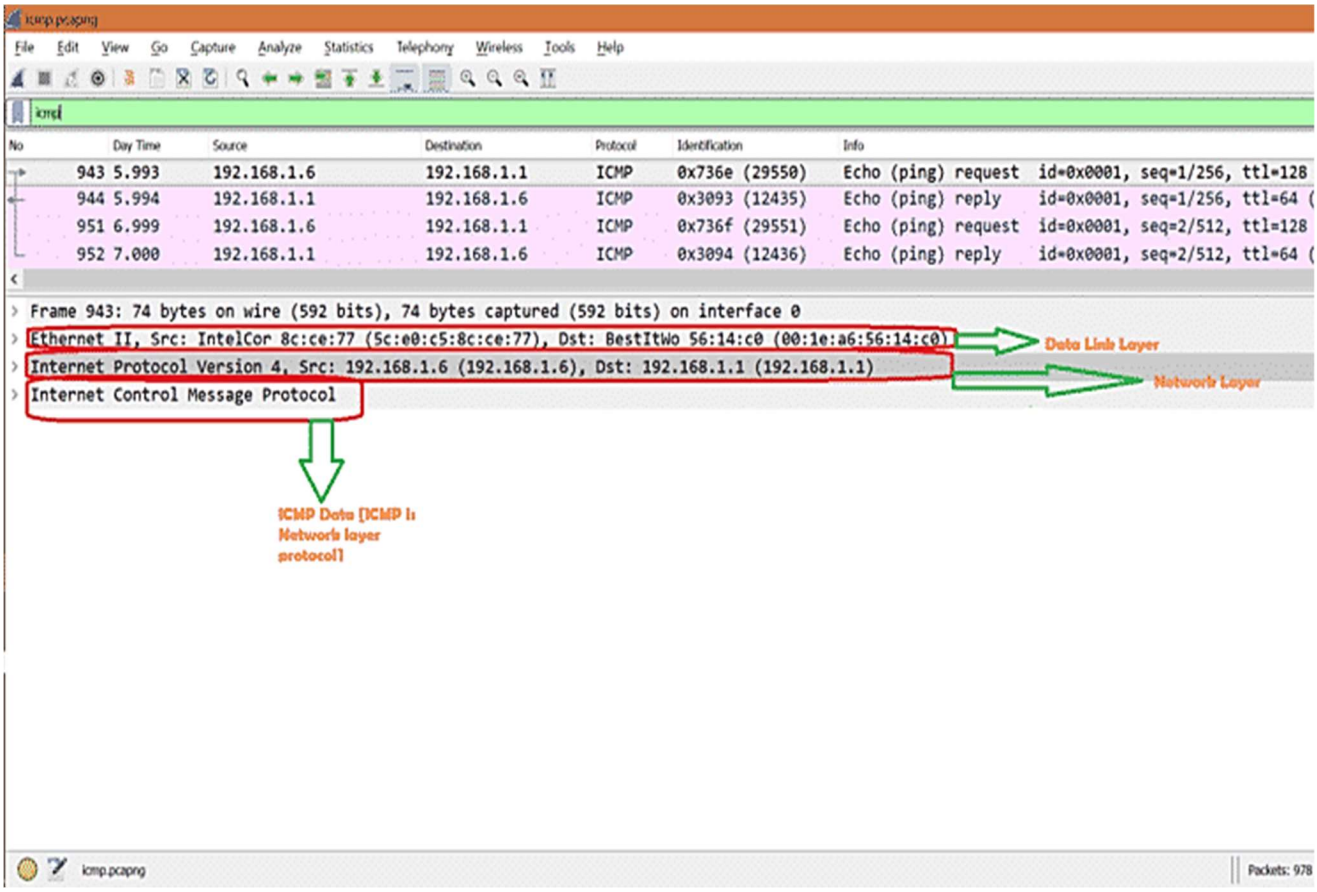


Let's see ICMP packet.

ICMP [It has 2 layers]:

Here is the screenshot of an ICMP frame where we can see 2 layers.

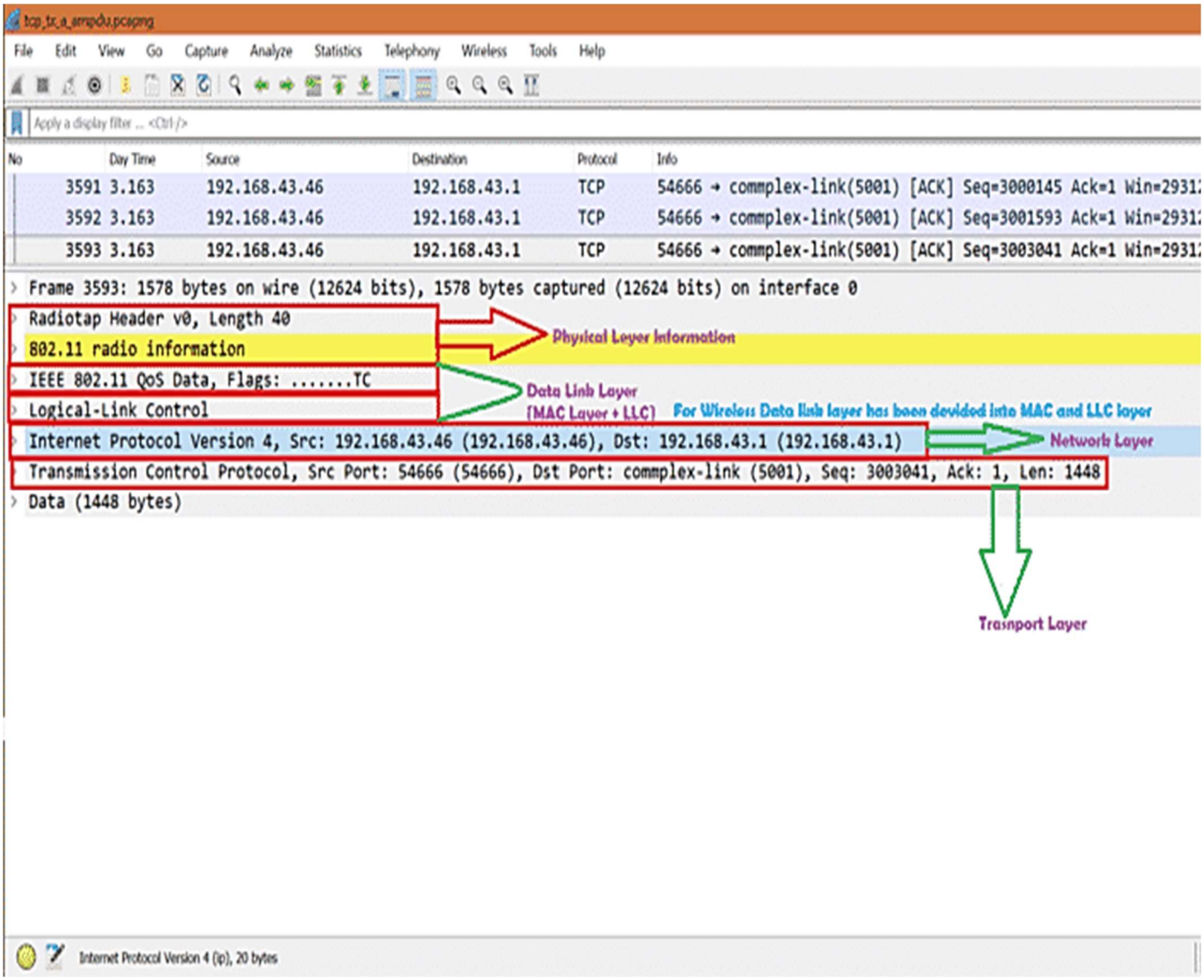
-



Now let's see one wireless TCP frame where we can see physical layer information.

Wireless TCP [It has 4 layers]:

Here is the screenshot of a TCP frame where we can see 4 layers including physical layer.

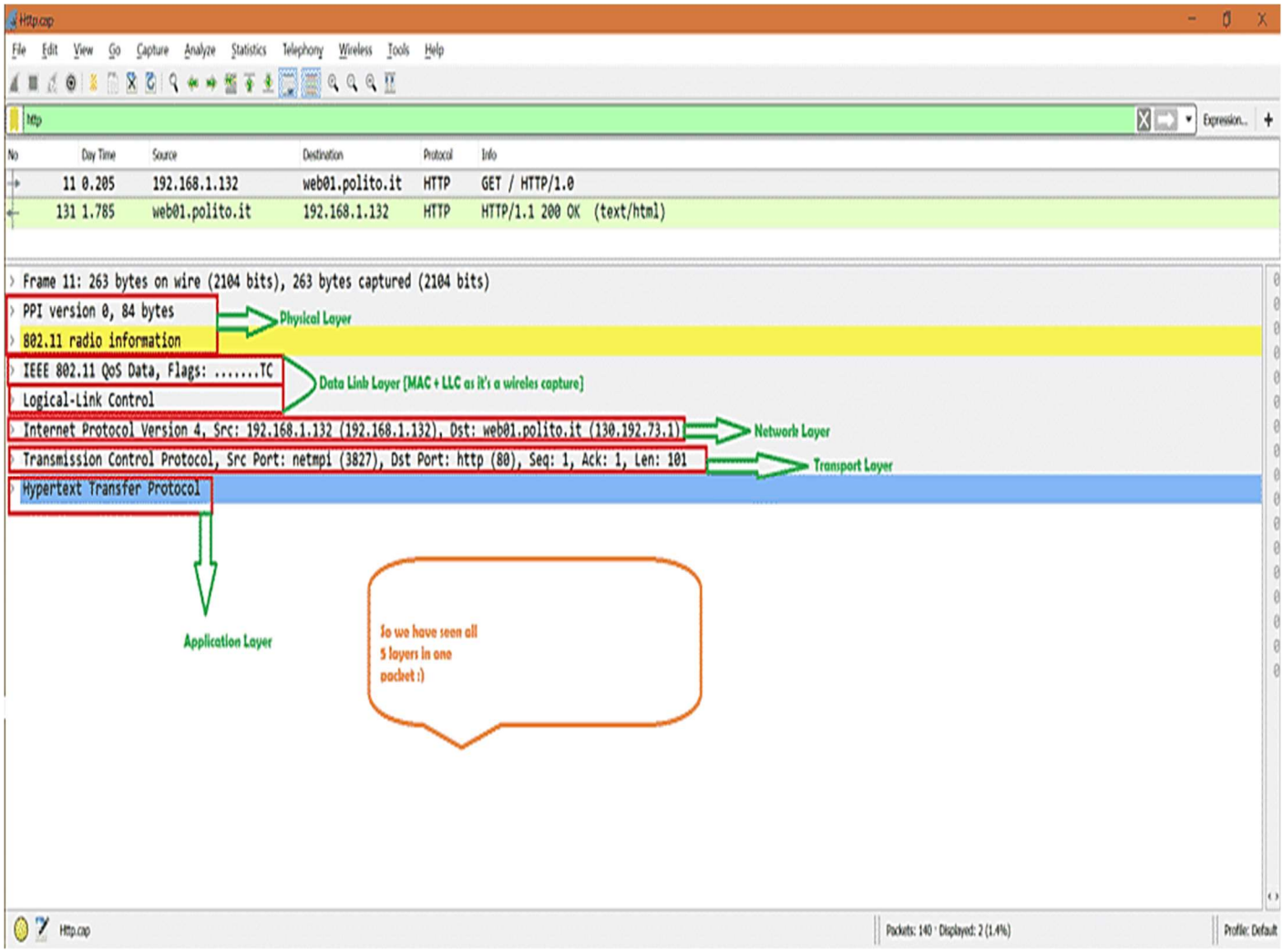


As TCP is a transport layer protocol so we did not see any application layer protocol.

Now let's see Wireless capture for HTTP and hope to see all 5 layers including Application layer and physical layer.

Wireless HTTp [It has all 5 layers]:

Here is the screenshot of a HTTP frame where we can see including Application layer and physical layer.



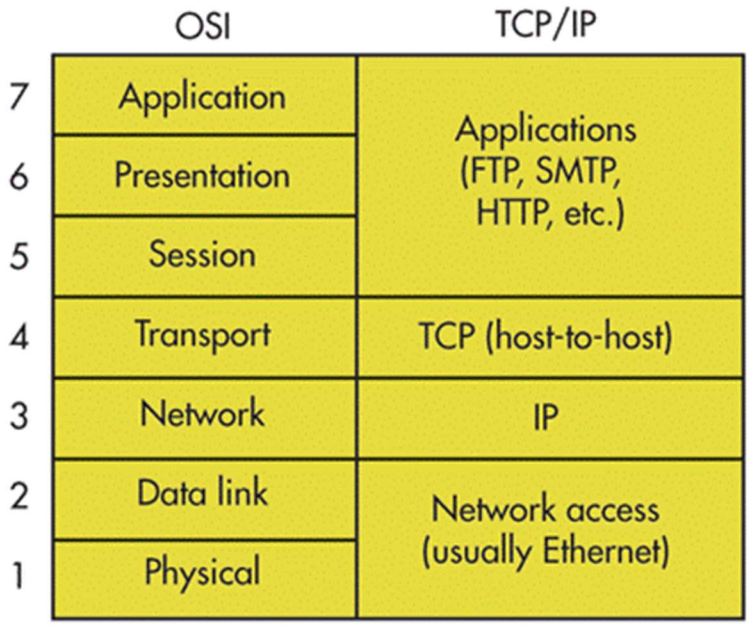
Summary:

In summary we can say that depending on protocol different layers can been seen in Wireshark.

HTTP analysis using Wireshark

What is HTTP?

First is all the full form of HTTP is HyperText Transfer Protocol. HTTP is an application layer protocol in ISO or TCP/IP model. See below picture to find out HTTP which resides under application layer.



HTTP is used by the World Wide Web (w.w.w) and it defines how messages are formatted and transmitted by browser. So HTTP define reules what action should be taken when a browser receives HTTP command. And also HTTP defines rules for transmitting HTTP command to get data from server.
For example, when you enter a url in browser (Internet explorer, Chrome, Firefox, Safari etc) it actually sends an HTTP command to server.And server replies with appropiate command.

HTTP Methods:

There are some set of methods for HTTP/1.1 (This is HTTP version)

GET, HEAD, POST, PUT, DELETE, CONNECT, OPTION and TRACE.

We will not go in details of each method instead we will get to know about the methods which are seen quite often.Such as

GET: GET request asks data from web server. This is a main method used document retrival. We will see one practical example of this method.
POST: POST method is used when it's required to send some data to server.

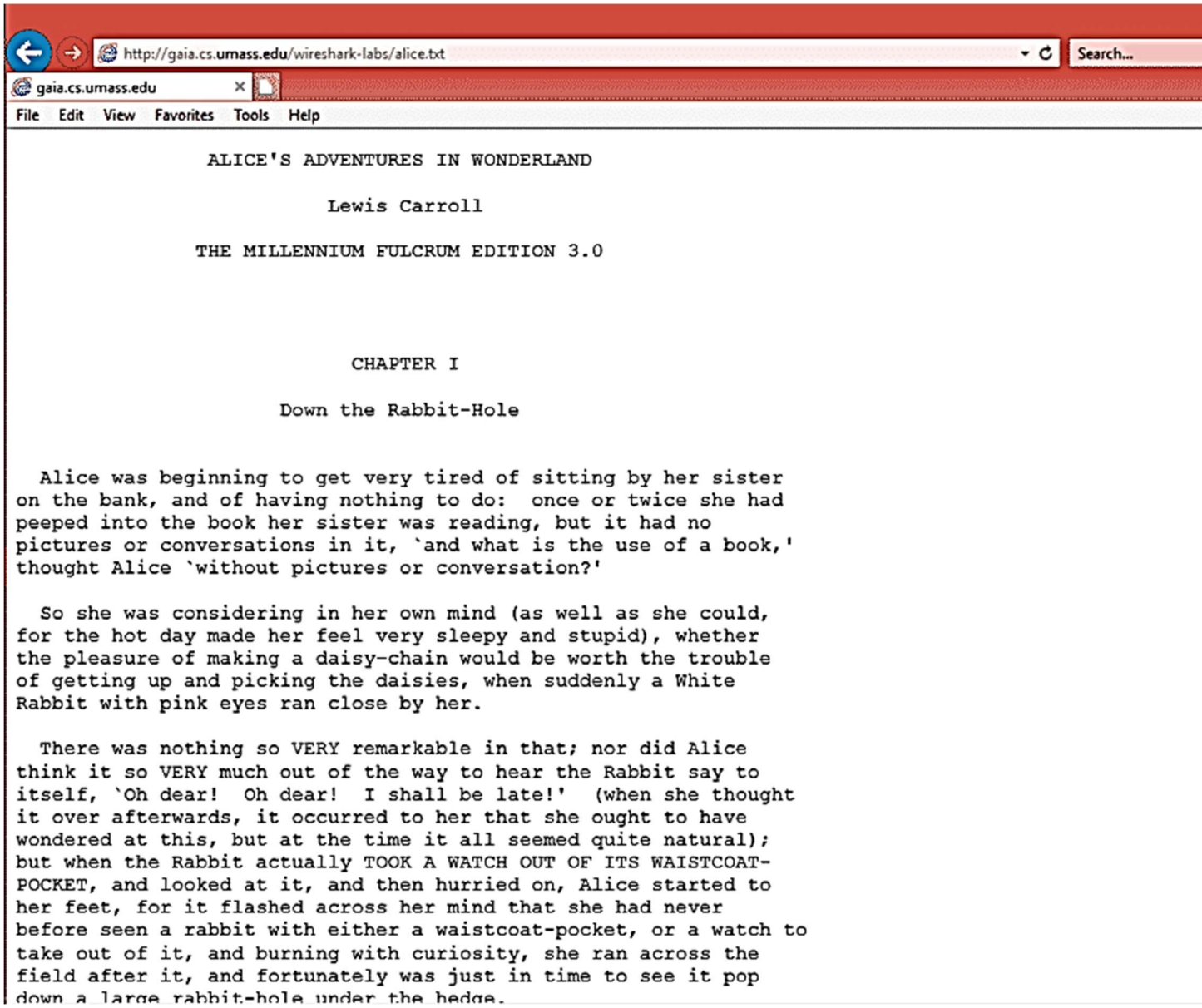
HTTP is Wiresahark:

Let's try something practical to understand how HTTP works ?

So in this example we will download **"alice.txt" (Data file present in server)** from **"gaia.cs.umass.edu"** server.

Setps:

1. Open the URL http://gaia.cs.umass.edu/wireshark-labs/alice.txt [We know the full url for downloading alice.txt] in computer browser.
2. Now we see the downloaded file in browser. Here is the screenshot



3. In parallel we have capture the packets in Wireshark.

HTTP packets exchanges in Wireshark:

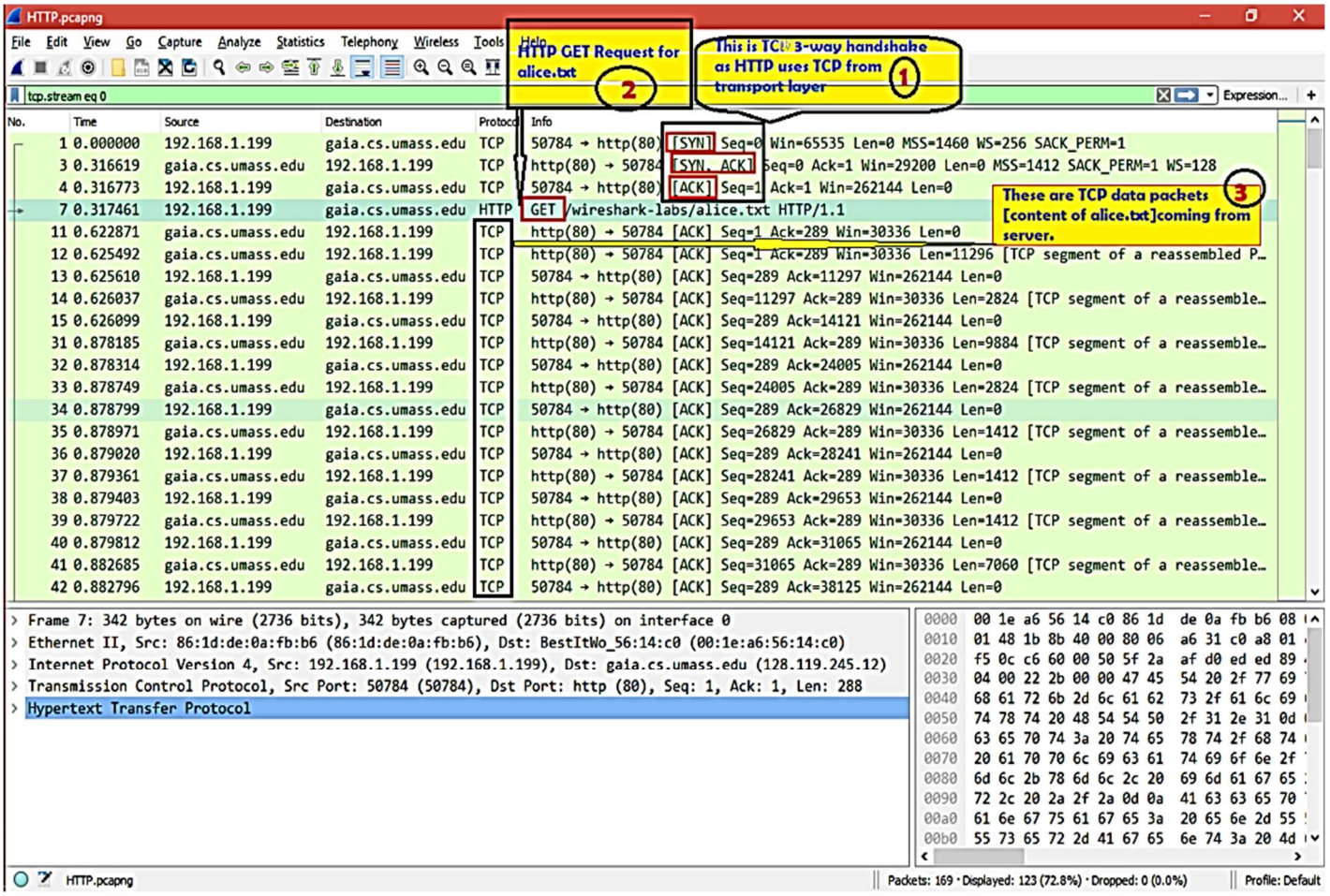
Before we go into HTTP we should know that HTTP uses port 80 and TCP as transport layer protocol [We will explain TCP in another topic discussion].

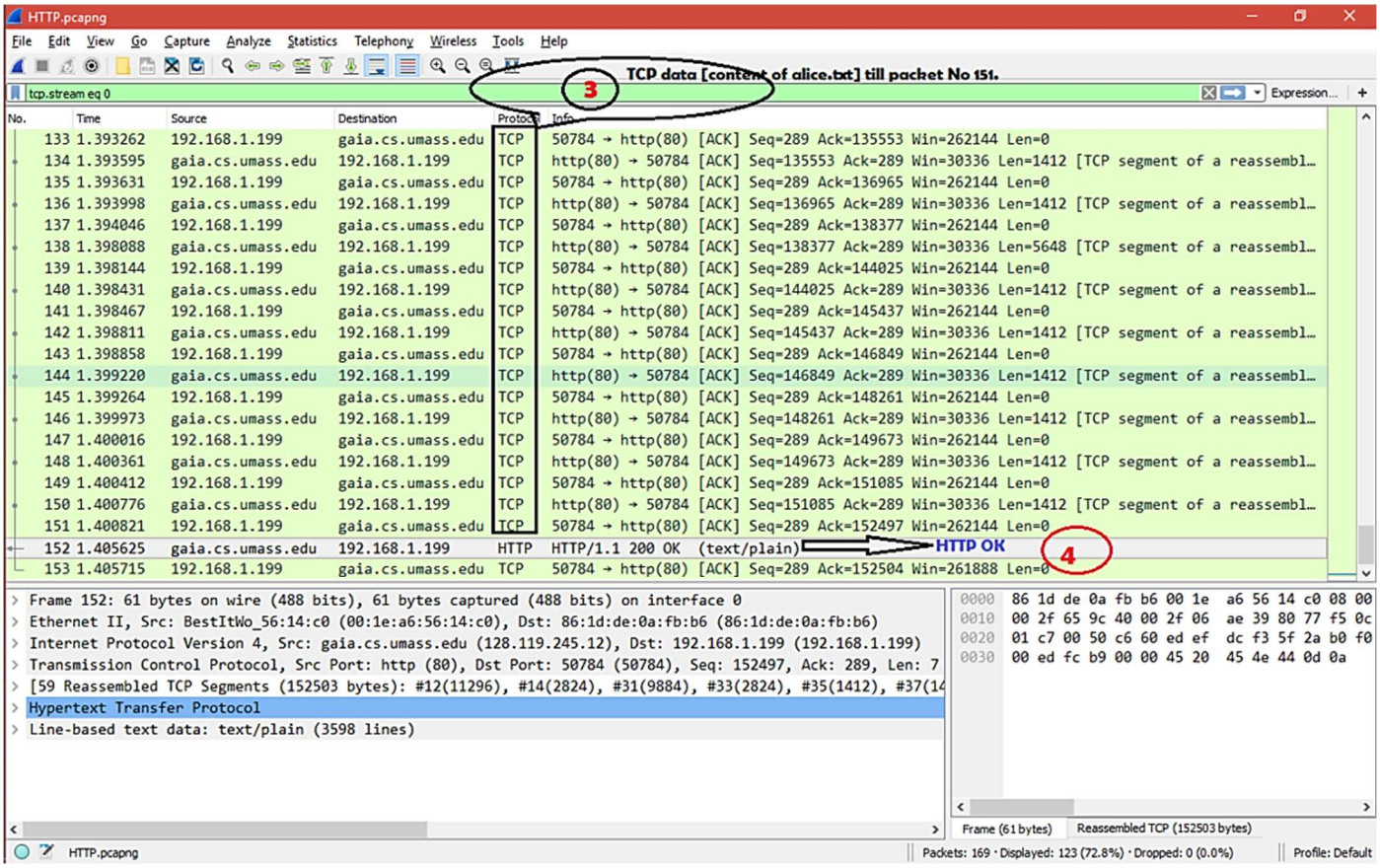
Now let's see what happens in network when we put that URL and press enter in browser.

Here is the screenshot for

TCP 3-way handshake ——-> HTTP OK ——-> TCP Data [content of alice.txt] ——->

HTTP-OK





Now let's see what's there inside HTTP GET and HTTP OK packets.

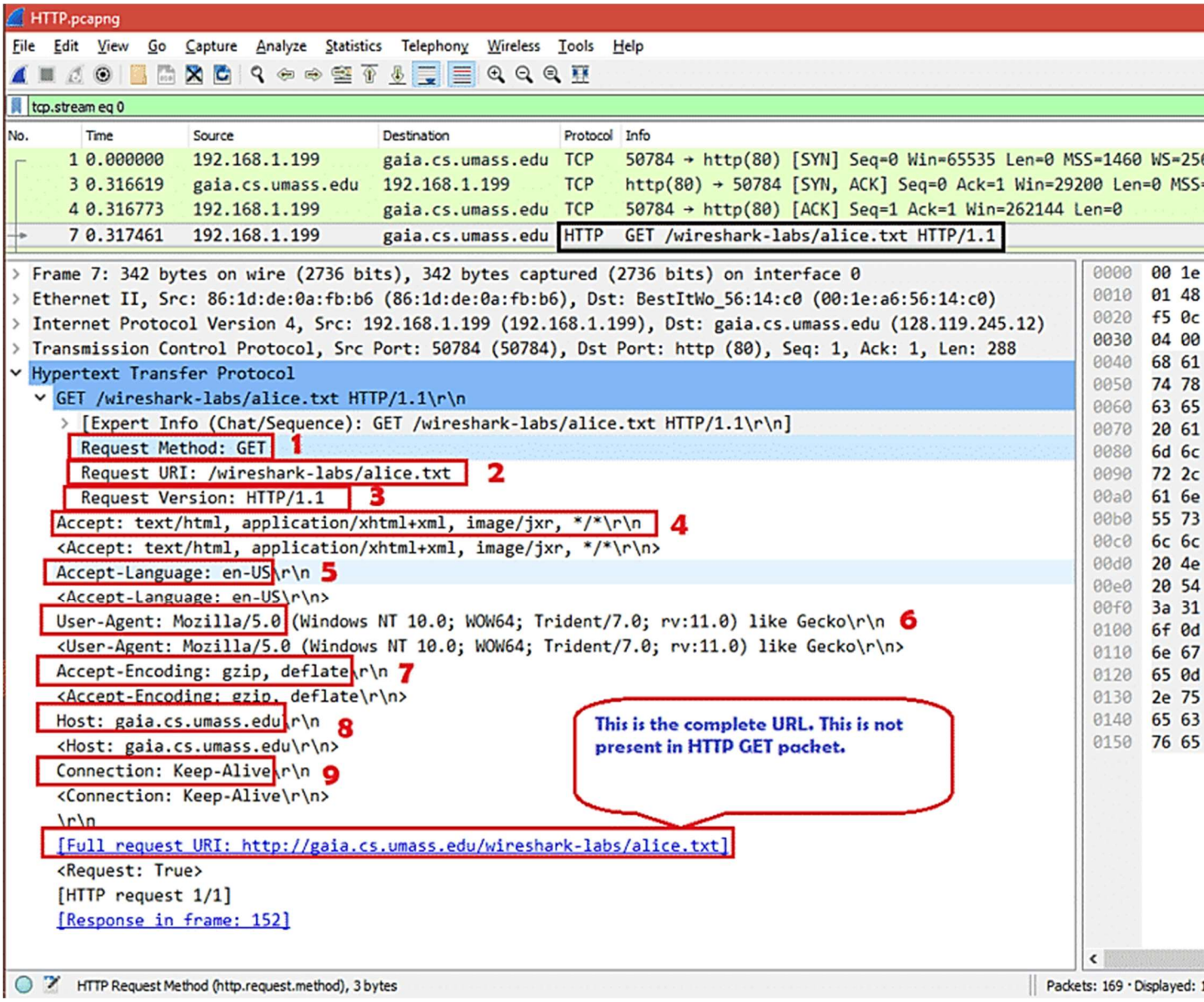
Note: We will explain TCP exchanges in another topic discussion.

HTTP GET:

After TCP 3-way handshake [SYN, SYN+ACK and ACK packets] is done HTTP GET request is sent to the server and here are the important fields in the packet.

1.Request Method: GET ==> The packet is a HTTP GET .
2.Request URI: /wireshark-labs/alice.txt ==> The client is asking for file alice.txt present under /Wireshark-labs
3.Request version: HTTP/1.1 ==> It's HTTP version 1.1
4.Accept: text/html, application/xhtml+xml, image/jxr, */* ==> Tells server about the type of file it [client side browser] can accept. Here the client is expecting alice.txt which is text type.
5.Accept-Language: en-US ==> Accepted language standard.

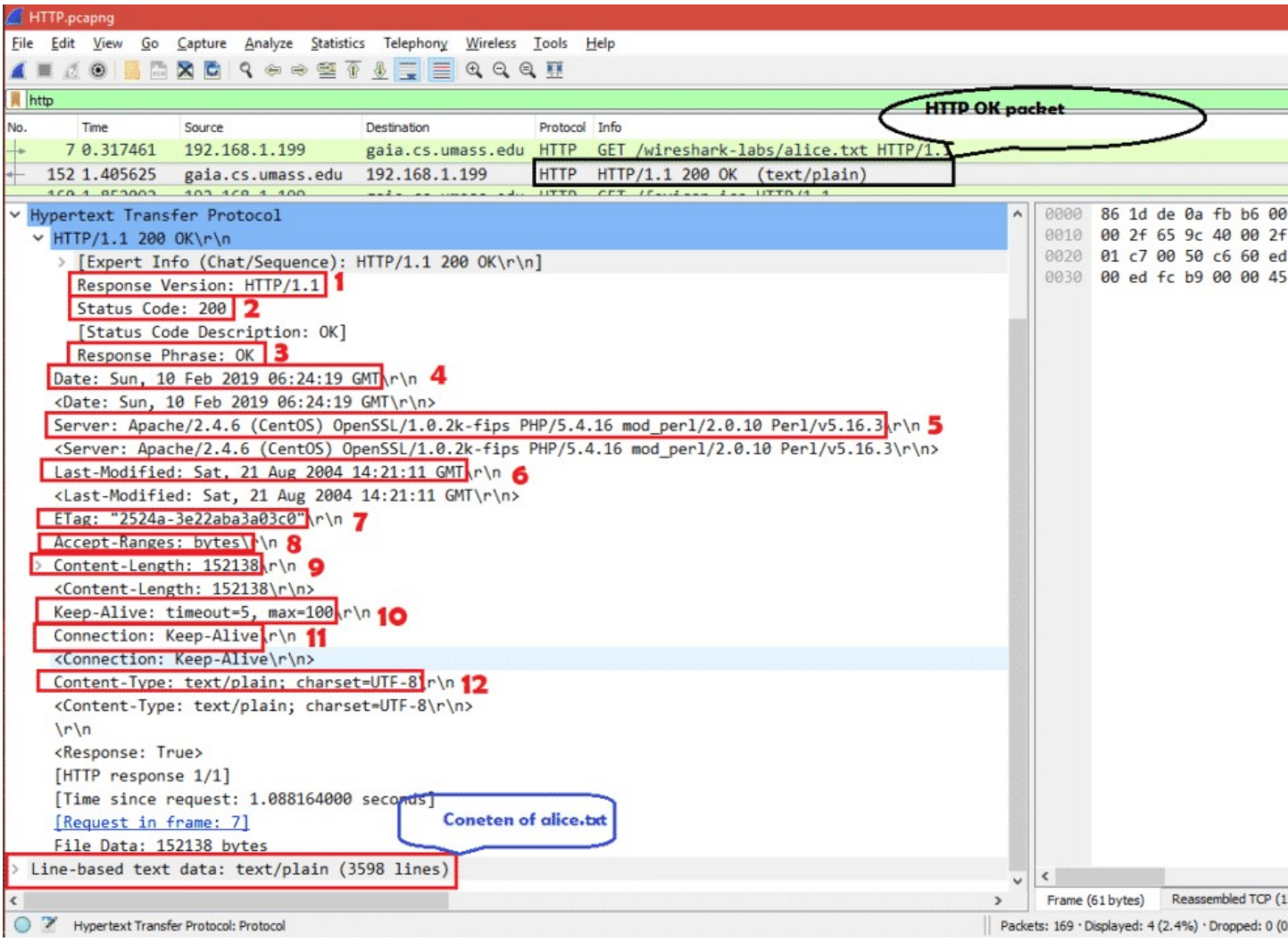
6.User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko ==> Client side browser type. Even if we used internet explorer but we see it always/maximum time says Mozilla
7.Accept-Encoding: gzip, deflate ==> Accepted encoding in client side.
8.Host: gaia.cs.umass.edu ==> This is the web server name where client is sending HTTP GET request.
9.Connection: Keep-Alive ==> Connection controls whether the network connection stays open after the current transaction finishes. Connection type is keep alive.
Here is the screenshot for HTTP-GET packet fields



HTTP OK:

After TCP data [content of alice.txt] is sent successfully HTTP OK is sent to the client and here are the important fields in the packet.
1. Response Version: HTTP/1.1 ==> Here server also in HTTP version 1.1
2.Status Code: 200 ==> Status code sent by server.
3.Response Phrase: OK ==> Response phrase sent by server.
So the from 2 and 3 we get 200 OK which means the request [HTTP GET] has succeeded.

4.Date: Sun, 10 Feb 2019 06:24:19 GMT ==> Current date , time in GMT when HTTP GET was received by server.
5.Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/5.4.16 mod_perl/2.0.10 Perl/v5.16.3 ==> Server details and configurations versions.
6.Last-Modified: **Sat, 21 Aug 2004 14:21:11 GMT** ==> Last modified date and time for the file “alice.txt”.
7.ETag: “2524a-3e22aba3a03c0” ==> The ETag indicates the content is not changed to assist caching and improve performance. Or if the content has changed, etags are useful to help prevent simultaneous updates of a resource from overwriting each other.
8. Accept-Ranges: bytes ==> Byte is the unit used in server for content.
9.Content-Length: 152138 ==> This is the total length of the alice.txt in bytes.
10. Keep-Alive: timeout=5, max=100 ==> Keep alive parameters.
11.Connection: Keep-Alive ==> Connection controls whether the network connection stays open after the current transaction finishes. Connection type is keep alive.
12.Content-Type: text/plain; charset=UTF-8 ==> The content [alice.txt] type is text and charset standard is UTF-8.
Here is the screenshot for different fields of HTTP OK packet.



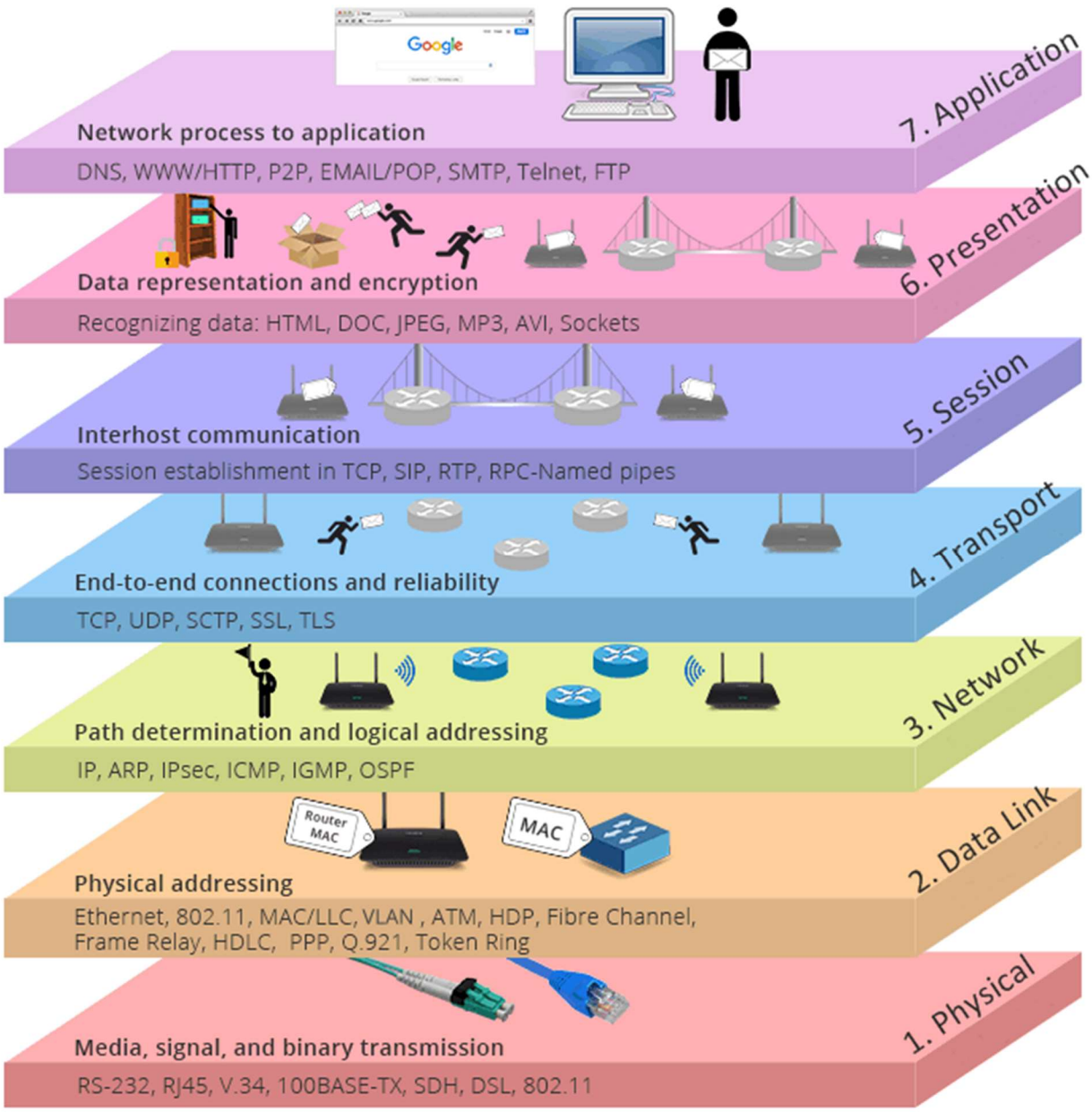
So now we know what happens when we request for any file that is present in web server.

Conclusion:

HTTP is simple application protocol that we use every day in our life. But it's not secure so HTTPS has been implemented. That "S" stands for secure. That's why you so maximum web server name start with http***s://[websitename]***. This means all communication between you and server are encrypted.

Computer Networks Lab 5

Prepared by: Engr. Khuram Shahzad


Google
7. Application
Network process to application
DNS, WWW/HTTP, P2P, EMAIL/POP, SMTP, Telnet, FTP
6. Presentation
Data representation and encryption
Recognizing data: HTML, DOC, JPEG, MP3, AVI, Sockets
5. Session
Interhost communication
Session establishment in TCP, SIP, RTP, RPC-Named pipes
4. Transport
End-to-end connections and reliability
TCP, UDP, SCTP, SSL, TLS
3. Network
Path determination and logical addressing
IP, ARP, IPsec, ICMP, IGMP, OSPF
Router MAC
MAC
2. Data Link
Physical addressing
Ethernet, 802.11, MAC/LLC, VLAN , ATM, HDP, Fibre Channel,
Frame Relay, HDLC, PPP, Q.921, Token Ring
1. Physical
Media, signal, and binary transmission
RS-232, RJ45, V.34, 100BASE-TX, SDH, DSL, 802.11