

## Computer Networks - Lab 04

---

### OBJECTIVES

After these Lab students shall be able to perform

- Understanding of Application layer and its protocols
- Configuring DHCP server on a Router..
- Configuring DHCP service on a generic server in Packet Tracer.
- Web Server Configuration in Cisco Packet Tracer
- Configuring DHCP, DNS and Web Server configuration in cisco packet tracer
- Wire shark
- OSI Network Layer Analysis via Wireshark
- Relation OSI and TCP/IP model
- HTTP and HTTPS analysis using Wireshark
- Http and Https packet sniffing on wire shark

### PRE-LAB READING ASSIGNMENT

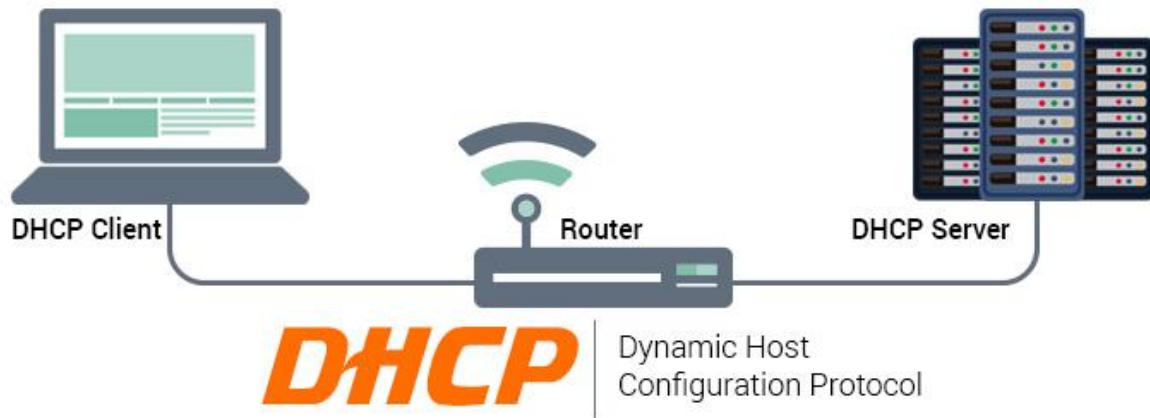
Remember the delivered lecture carefully.

## Table of Contents

Computer Networks - Lab 04 .....	1
OBJECTIVES .....	1
PRE-LAB READING ASSIGNMENT .....	1
▪ DHCP .....	3
▪ DNS Server. ....	3
▪ -Configuring DHCP server on a Router.....	6
• Router(dhcp-config)#ex .....	7
▪ Configuring DHCP service on a generic server in Packet Tracer. ....	9
- .....	11
▪ Configuring DHCP, DNS and Web Server configuration in cisco packet tracer .....	12
OSI Network Layer Analysis via Wireshark .....	18
OSI model and TCP/IP model: .....	18
Relation OSI and TCP/IP model: .....	19
What we see in Wireshark?.....	20
HTTP analysis using Wireshark .....	27
What is HTTP? .....	27
HTTP Methods: .....	27
HTTP is Wiresahark: .....	28
HTTP packets exchanges in Wireshark: .....	29
HTTP GET:.....	31
HTTP OK:.....	33

- **DHCP**

DHCP (**D**ynamic **H**ost **C**onfiguration **P**rotocol) Server allots the IP addresses to computers, while DNS server resolves them. You need DHCP Server if you do not want to manually maintain IP Addresses or you have less IP Addresses than number of machines you have, as dynamic DHCP Server will recycle IP Addresses on machines. DHCP Features support Static and Dynamic 125 DHCP Ranges, Range Filters, Relay Agents and BOOT, Options can be specified for DHCP Ranges, Global or for Static Hosts.

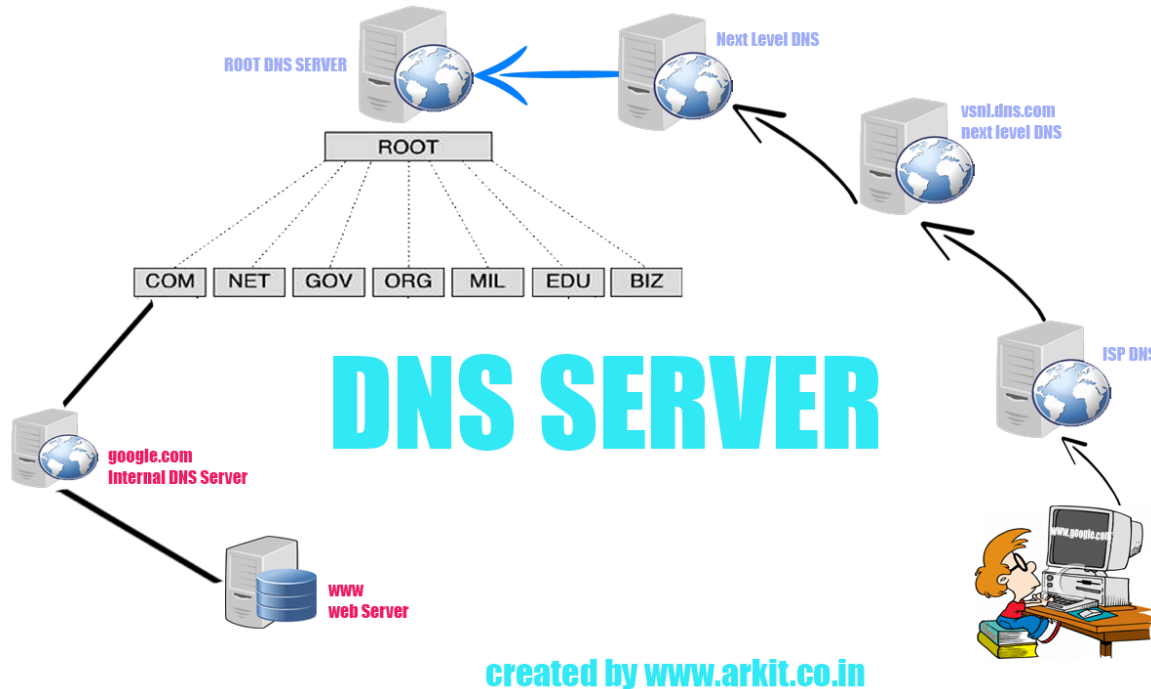


- **Website Server.**

The server where whole website is location

- **DNS Server.**

DNS Server is needed for resolving hostnames to their IP addresses. Normally your ISP will provide you with DNS Service. You may have your own DNS Server, which will resolve hostnames by forwarding them to ISP's DNS Server and cache the addresses also.

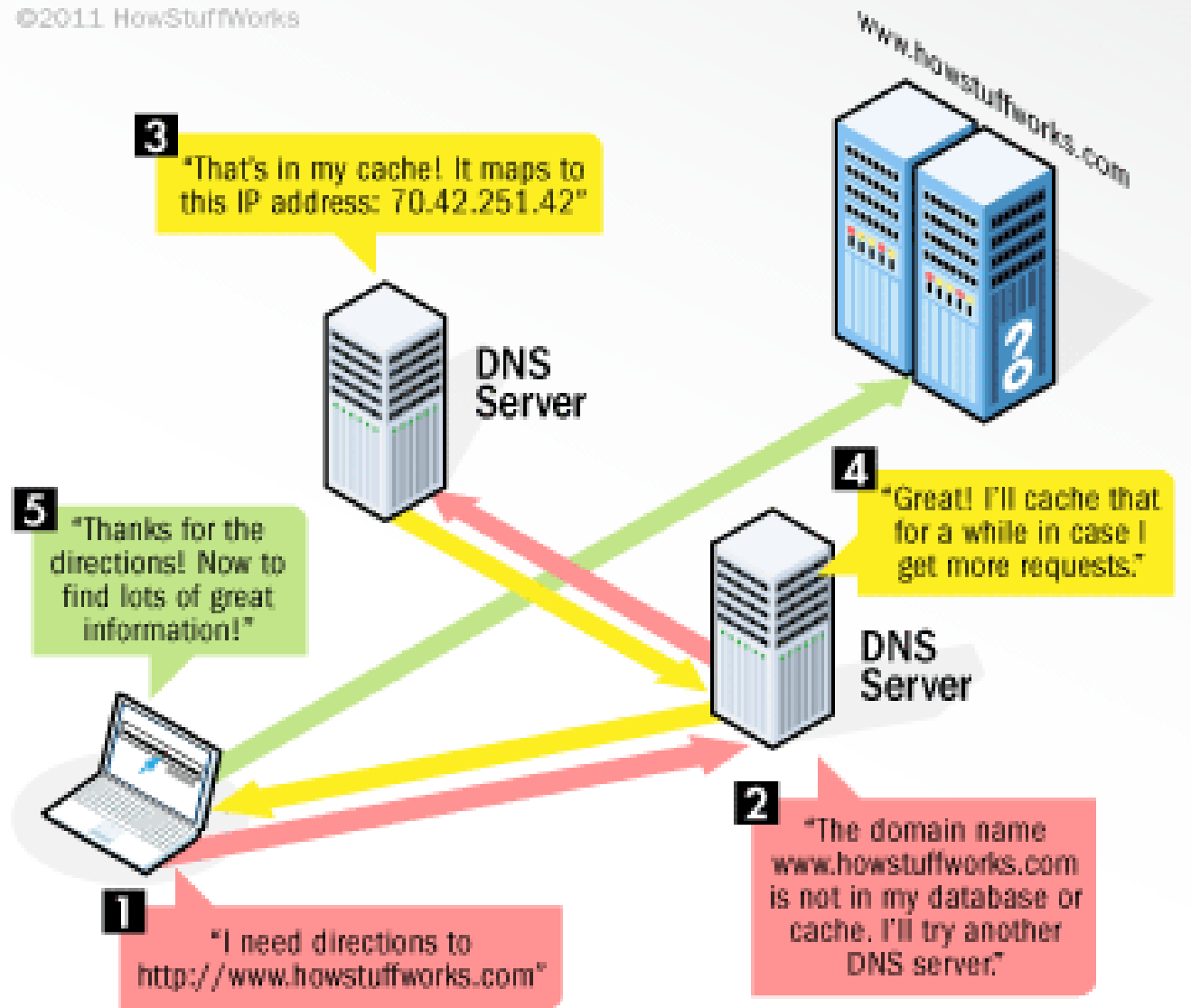


If you have home/small office network with Unix/Linux machines, these machines will not be resolved from each other, as Unix/Linux machines do not support NBNS protocol and you need your own DNS Server. But how about resolving your local machines ?. Your ISP's DNS Server will not have this list and your own DNS Server won't have them either. Most DNS Servers cannot do this.(unless you configure dynamic updates, or use static IP addresses and manually enter them).

This server resolves dhcp allotted local machines automatically in addition to external hosts, with the added advantage being both dhcp and dns server are always in sync. Also there is no need to create and maintain cumbersome zone files. Dual DHCP DNS Server is an Open Source Freeware. In addition, this server is designed for Load Sharing Replicated Operation

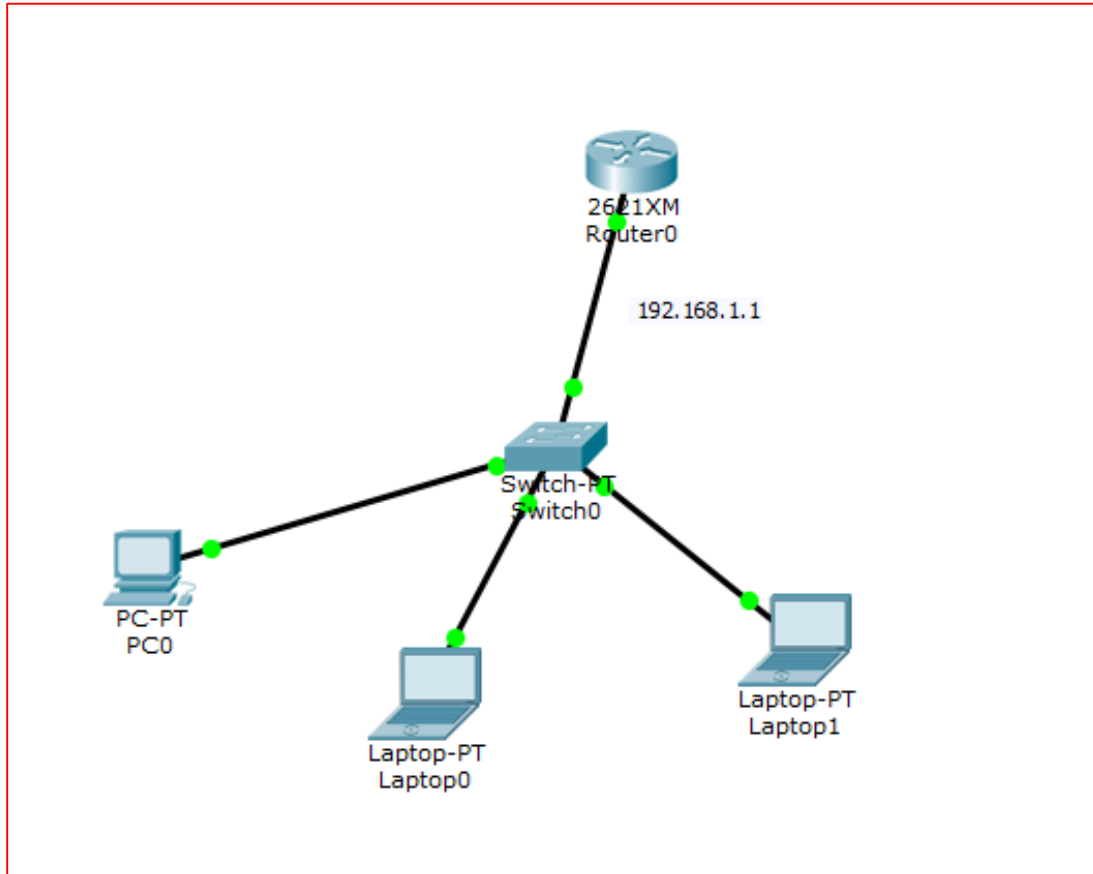
DNS Features include Forward and Reverse Lookup, Zone Transfer, Primary/Secondary Mode of Operation, MX Records, Wildcard Records, Conditional and Default forwarding.

Either DHCP or DNS Service can be used. If both services are used, DHCP allotted hosts are automatically added in DNS zones.



## ▪ -Configuring DHCP server on a Router

1. Build the network topology:



2. On the router, configure interface fa0/0 to act as the default gateway for our LAN.

1. Router>enable
2. Router#configure terminal
3. Enter configuration commands, one per line. End with CNTL/Z.
4. Router(config)#interface FastEthernet0/0
5. Router(config-if)#ip address 192.168.1.1 255.255.255.0
6. Router(config-if)#no shutdown

```
Router>enable
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface FastEthernet0/0
Router(config-if)#ip address 192.168.1.1 255.255.255.0
Router(config-if)#no shutdown
```

3. Configure DHCP server on the Router. In the server we will define a DHCP pool of IP addresses to be assigned to hosts, a Default gateway for the LAN and a DNS Server.

- Router(config-if)#ex
- Router(config)#ip dhcp pool P1
- Router(dhcp-config)#network 192.168.1.1 255.255.255.0
- Router(dhcp-config)#default-router 192.168.1.1
- Router(dhcp-config)#dns-server 192.168.1.10
- Router(dhcp-config)#ex

```
Router(config-if)#ex
Router(config)#ip dhcp pool P1
Router(dhcp-config)#network 192.168.1.1 255.255.255.0
Router(dhcp-config)#default-router 192.168.1.1
Router(dhcp-config)#dns-server 192.168.1.10
Router(dhcp-config)#ex
```

We can add **ip dhcp excluded-address** command to our configuration so as to configure the router to exclude addresses **192.168.1.1** through **192.168.1.10** when assigning addresses to clients. The **ip dhcp excluded-address** command may be used to reserve addresses that are statically assigned to key hosts.

So add the above command under the **global configuration mode**.

- **Router(config)#ip dhcp excluded-address 192.168.1.1 192.168.1.10**
4. Now go to every PC and on their IP configuration tabs, enable DHCP. Every PC should be able to obtain an IP address, default gateway and DNS server, as defined in step 2.

For example, to enable DHCP on PC1:

**Click PC1->Desktop->IP configuration. Then enable DHCP:**

Physical Config Desktop Custom Interface

**IP Configuration** [X]

IP Configuration

☒ DHCP ☐ Static

IP Address: 192.168.1.13

Subnet Mask: 255.255.255.0

Default Gateway: 192.168.1.1

DNS Server: 192.168.1.10

Do this for the other PCs.

You can test the configuration by pinging PC2 from PC1. Ping should succeed.

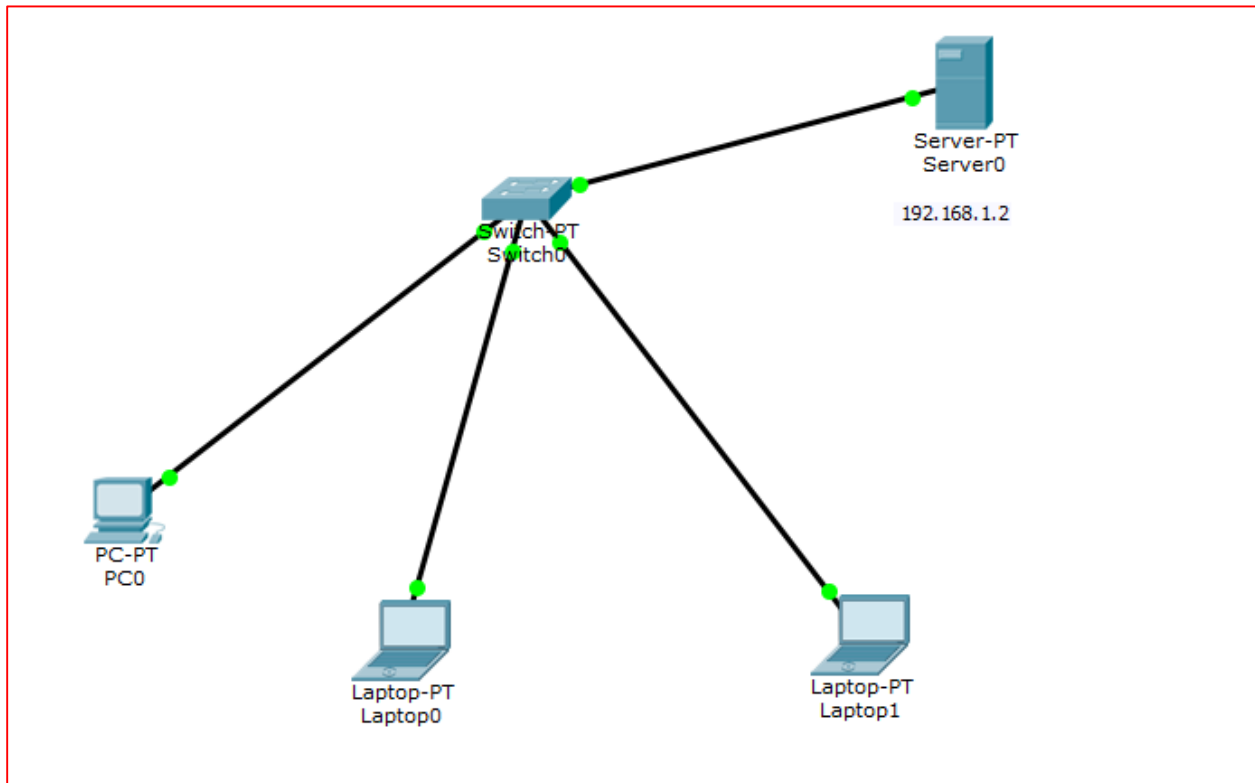
It's that simple!

Now let's do the same thing using a Generic server in place of a router:

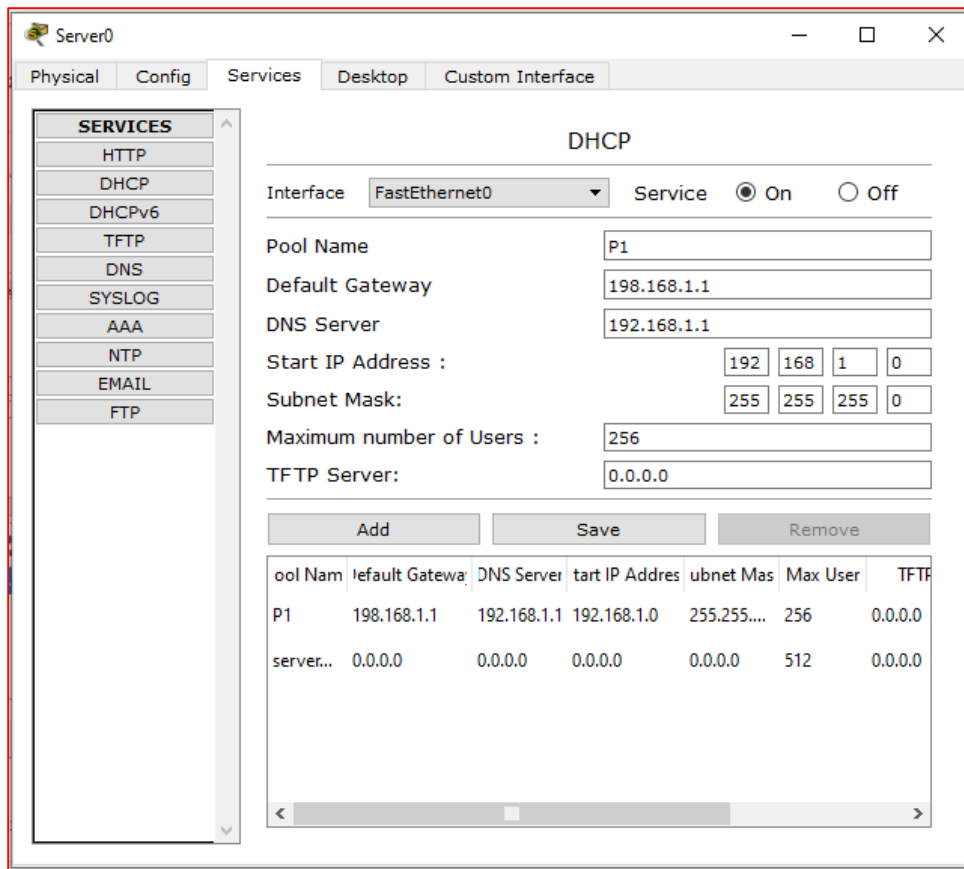


▪ **Configuring DHCP service on a generic server in Packet Tracer.**

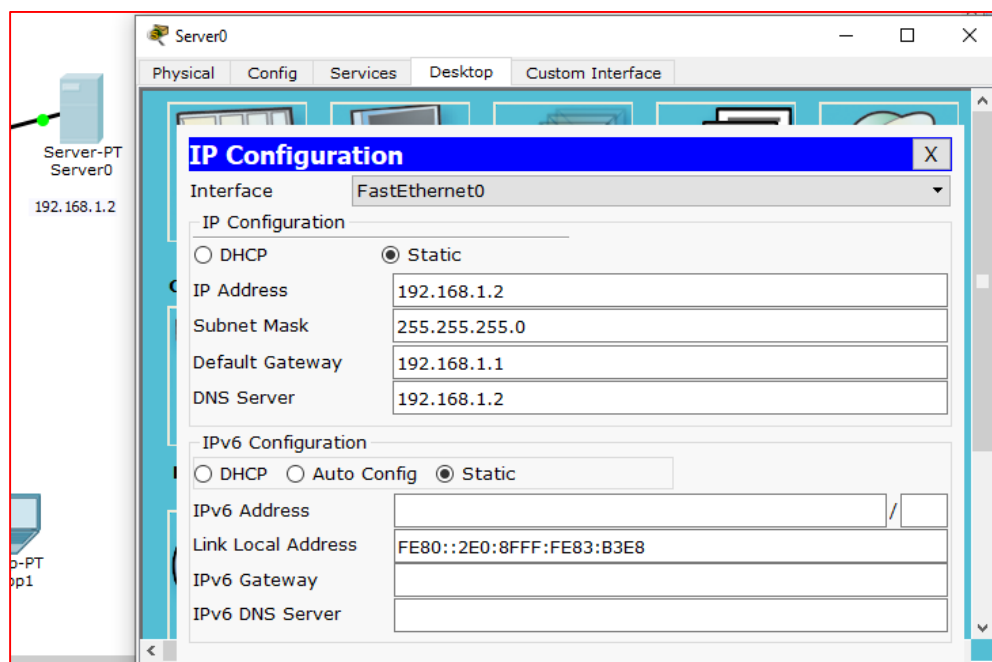
1. Build the network topology:



2. Configure static IP address on the server (192.168.1.2/24).
3. Now configure DHCP service on the generic server.
4. To do this, click on the server, then click on Services tab. You will pick DHCP on the menu. Then proceed to define the DHCP network parameters as follows:
  - ❖ Pool name: MY\_LAN
  - ❖ Default Gateway: 192.168.1.1
  - ❖ DNS Server: 192.168.1.2
  - ❖ Start IP Address: 192.168.1.0
  - ❖ Subnet Mask: 255.255.255.0
  - ❖ Maximum Number of users: 256



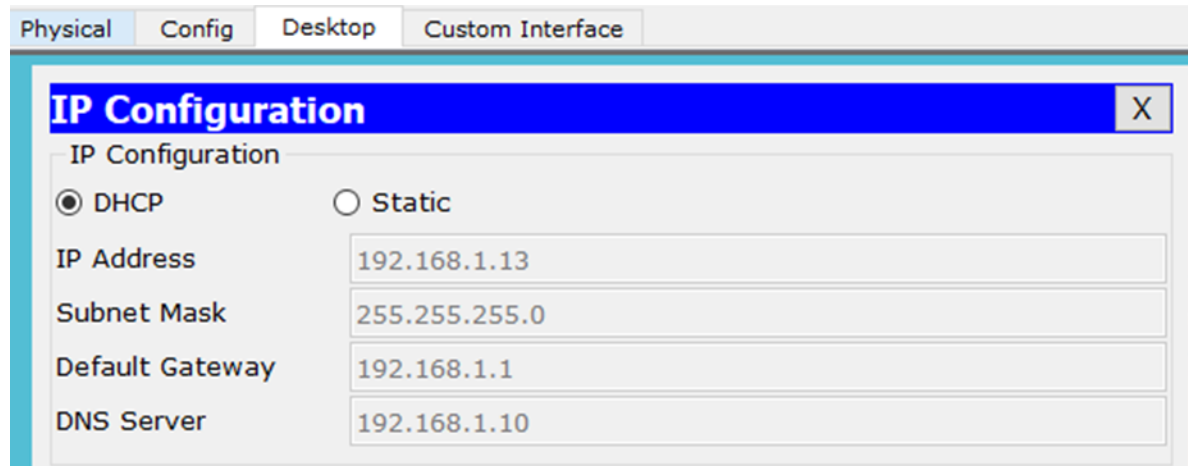
5. Assign the Ip Address to DHCP



- Now go to every PC and on their IP configuration tabs, enable DHCP. Every PC should be able to obtain an IP address, default gateway and DNS server, as defined in step 2.

For example, to enable DHCP on PC1:

**Click PC1->Desktop->IP configuration. Then enable DHCP:**



IP Configuration	
<input checked="" type="radio"/> DHCP <input type="radio"/> Static	
IP Address	192.168.1.13
Subnet Mask	255.255.255.0
Default Gateway	192.168.1.1
DNS Server	192.168.1.10

Do this for the other PCs.

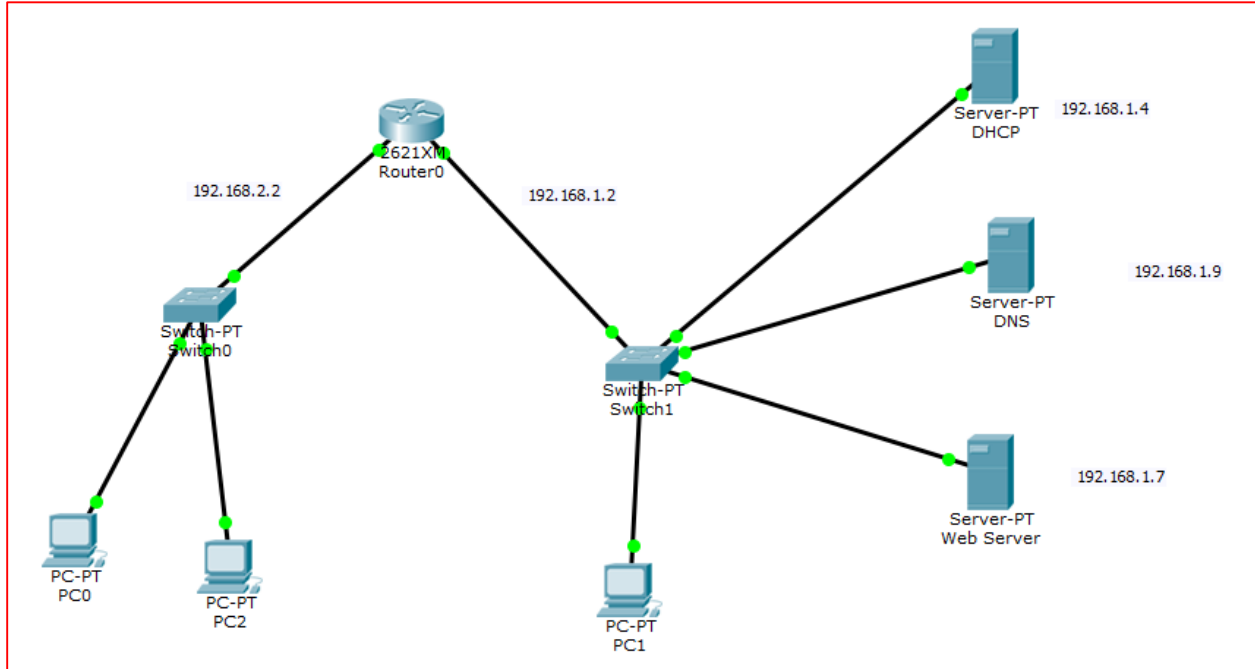
You can test the configuration by pinging PC2 from PC1. Ping should succeed.

It's that simple!

-

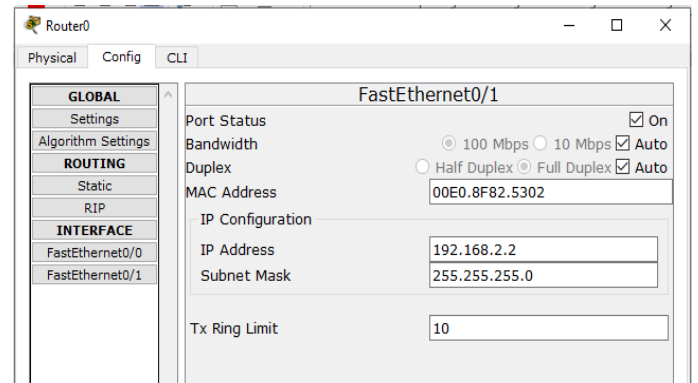
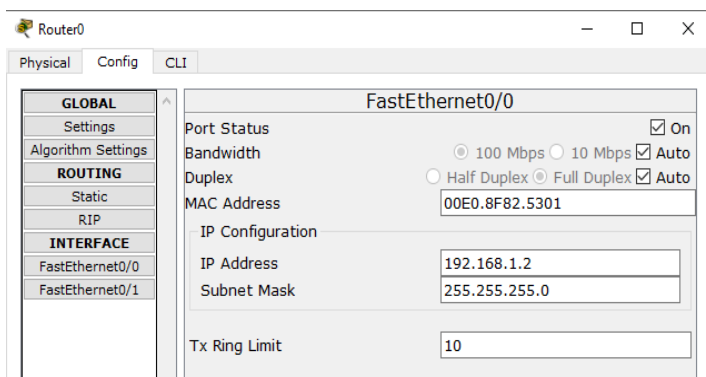
## ■ Configuring DHCP, DNS and Web Server configuration in cisco packet tracer

1. Build the network topology:

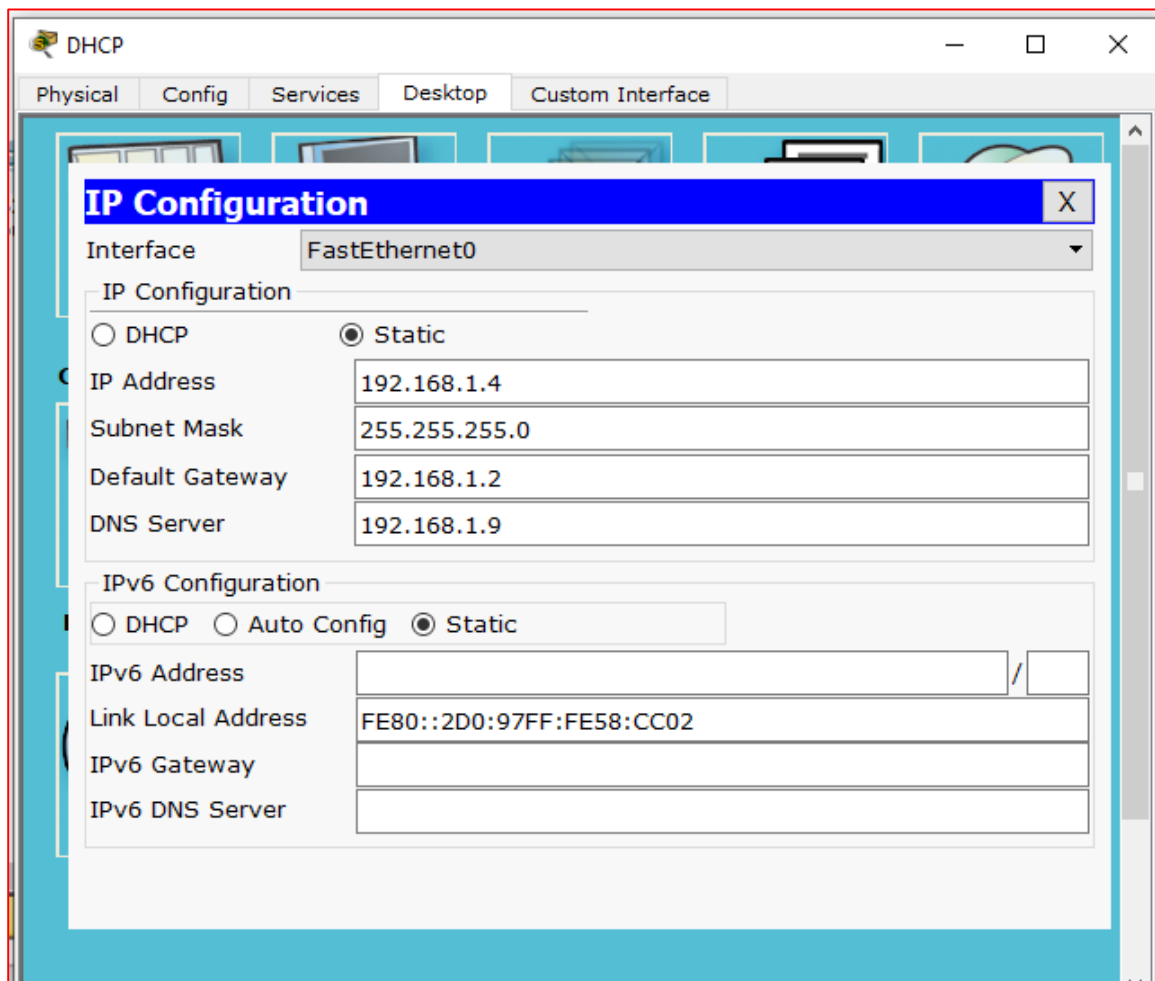


2. On the router, configure interface fa0/0 to act as the default gateway for our LAN.

Assign Ip address to F0/0 and F0/1 of router.



3. Apply follow commands in configuration mode on Router
  - ip dhcp pool P1
  - network 192.168.1.0 255.255.255.0
  - default-router 192.168.1.2
  - ip dhcp pool P2
  - network 192.168.2.0 255.255.255.0
  - default-router 192.168.2.2
4. Apply follows setting on DHCP IP Configurations



5. Enable DHCP Services and Add Pool P1 and Pool P2 with respective Ip Address

Interface: FastEthernet0 Service: ☒ On ☐ Off

Pool Name: P2

Default Gateway: 192.168.2.2

DNS Server: 192.168.1.9

Start IP Address : 192 168 1 0

Subnet Mask: 255 255 255 0

Maximum number of Users : 256

TFTP Server: 0.0.0.0

Pool Name	Default Gateway	DNS Server	Start IP Address	Subnet Mask	Max User	TFTP Server
P2	192.168.2.2	192.168.1.9	192.168.1.0	255.255.255.0	256	0.0.0.0
P1	192.168.1.2	192.168.1.9	192.168.1.0	255.255.255.0	256	0.0.0.0
serverPool	192.168.1.2	192.168.1.9	192.168.1.0	255.255.255.0	256	0.0.0.0

6. Apply follows setting on DNS IP Configurations

Interface: FastEthernet0

IP Configuration: ☐ DHCP ☒ Static

IP Address: 192.168.1.9

Subnet Mask: 255.255.255.0

Default Gateway: 192.168.1.2

DNS Server: 192.168.1.9

IPv6 Configuration: ☐ DHCP ☐ Auto Config ☒ Static

IPv6 Address: /

Link Local Address: FE80::20B:BEFF:FE9A:B827

IPv6 Gateway:

IPv6 DNS Server:

7. Enable DNS Services of DNS and add two resources records

The screenshot shows the 'DNS' configuration window. On the left, a 'SERVICES' list includes HTTP, DHCP, DHCPv6, TFTP, DNS, SYSLOG, AAA, NTP, EMAIL, and FTP. The 'DNS' service is selected. The main area shows 'DNS Service' set to 'On'. Below, 'Resource Records' are configured with 'Name' and 'Type' dropdowns. Two records are shown in a table:

No.	Name	Type	Detail
0	www.dhcp.com	A Record	192.168.1.4
1	www.webserver.com	A Record	192.168.1.7

Buttons for 'Add', 'Save', and 'Remove' are present. A 'DNS Cache' button is at the bottom.

8. Apply follows setting on Web Server

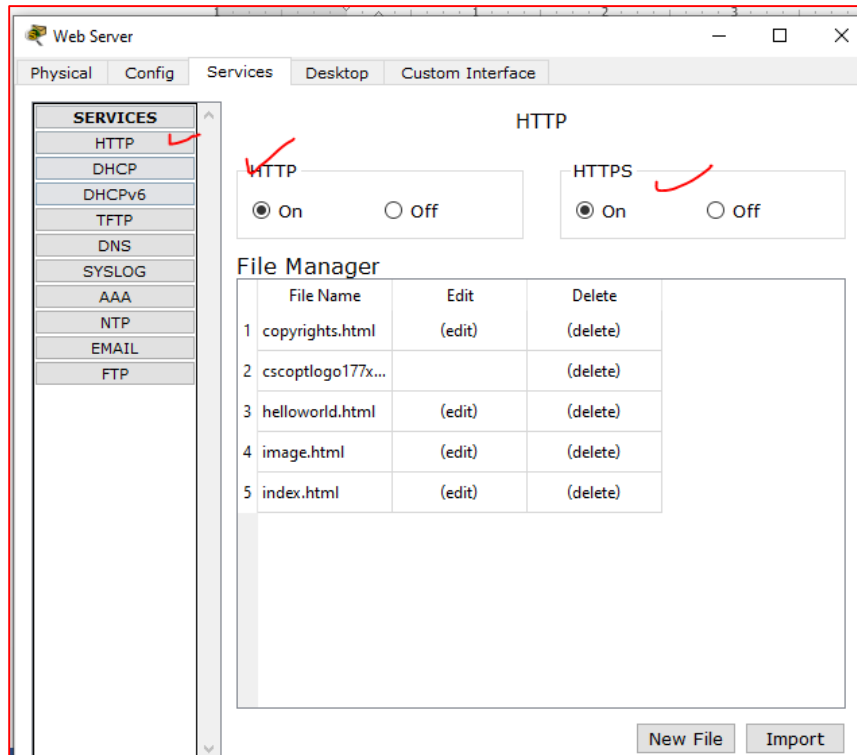
The screenshot shows the 'Web Server' configuration window. An 'IP Configuration' dialog box is open, showing settings for the 'FastEthernet0' interface. The 'IP Configuration' section has 'Static' selected. The settings are as follows:

Field	Value
Interface	FastEthernet0
IP Configuration	Static
IP Address	192.168.1.7
Subnet Mask	255.255.255.0
Default Gateway	192.168.1.2
DNS Server	192.168.1.9

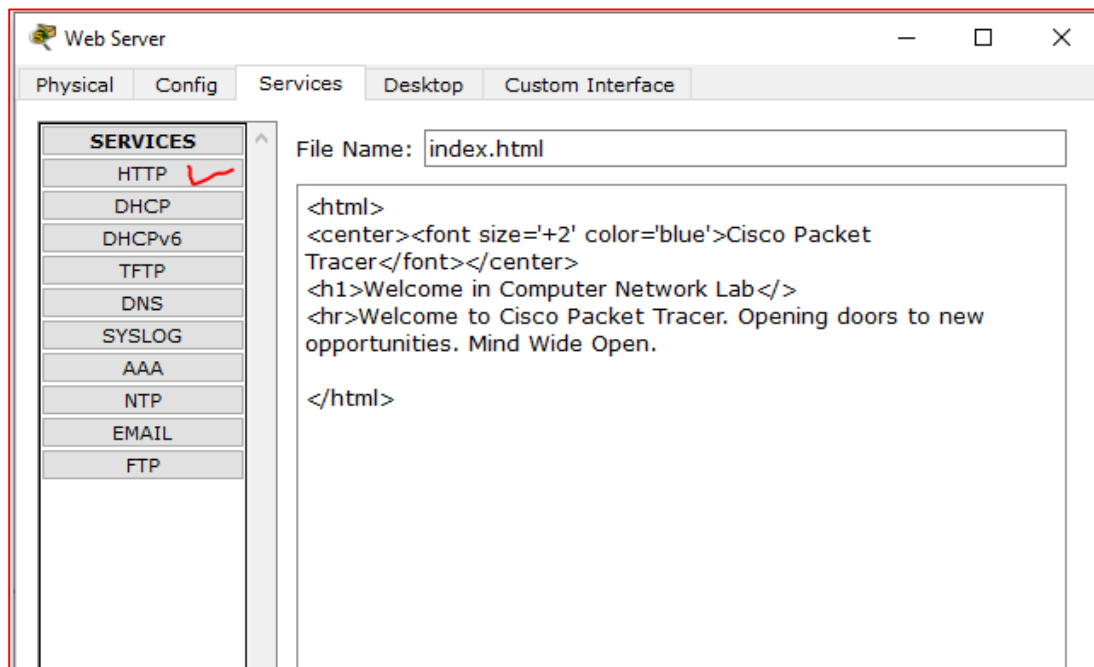
The 'IPv6 Configuration' section shows 'Static' selected, with fields for IPv6 Address, Link Local Address (FE80::20A:F3FF:FE90:3B5A), IPv6 Gateway, and IPv6 DNS Server.

Prepared by: Engr. Khuram Shahzad

## 9. Enable Http and Https in Web Server



## 10. Edit the Index .html and update it

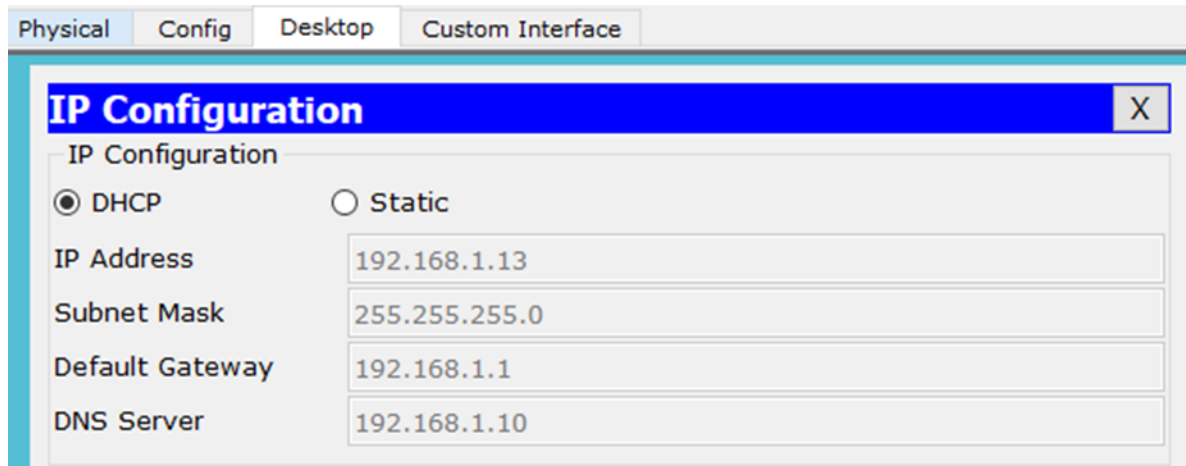




11. Now go to every PC and on their IP configuration tabs, enable DHCP. Every PC should be able to obtain an IP address, default gateway and DNS server, as defined in step 2.

For example, to enable DHCP on PC1:

**Click PC1->Desktop->IP configuration. Then enable DHCP:**



The screenshot shows a window titled "IP Configuration" with a close button (X) in the top right corner. The window has four tabs: "Physical", "Config", "Desktop", and "Custom Interface". The "Desktop" tab is selected. Inside the window, there is a section labeled "IP Configuration" with two radio buttons: "DHCP" (which is selected) and "Static". Below these are four input fields with their respective labels: "IP Address" (192.168.1.13), "Subnet Mask" (255.255.255.0), "Default Gateway" (192.168.1.1), and "DNS Server" (192.168.1.10).

Do this for the other PCs.

You can test the configuration by pinging PC2 from PC1. Ping should succeed.

It's that simple!

# 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:

<b>Application Layer</b>	<b>[Layer 7]</b>
<b>Presentation Layer</b>	<b>[Layer 6]</b>
<b>Session Layer</b>	<b>[Layer 5]</b>
<b>Transport Layer</b>	<b>[Layer 4]</b>
<b>Network Layer</b>	<b>[Layer 3]</b>
<b>Data Link Layer</b>	<b>[Layer 2]</b>
<b>Physical Layer</b>	<b>[Layer 1]</b>

There is another network model which is TCP/IP.

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

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

### 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]

**Prepared by: Engr. Khuram Shahzad**

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.

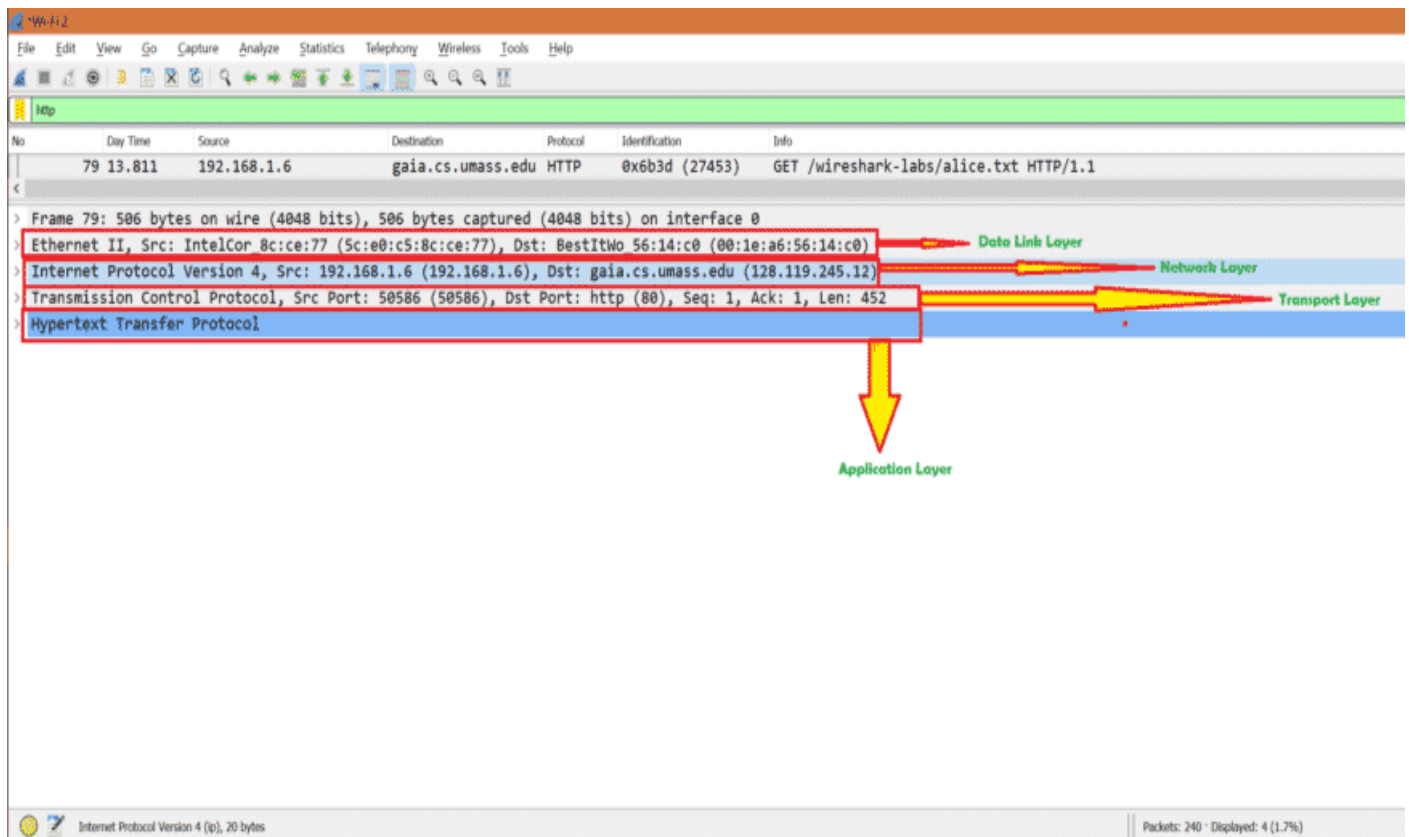
Physical Layer
Data Link Layer
Network Layer
Transport Layer
Application Layer

**1.1.1.1 HTTP [It has 4 layers]:**

You can follow below link to understand HTTP through Wireshark

[https://linuxhint.com/http\\_wireshark/](https://linuxhint.com/http_wireshark/)

Here is the screenshot of a HTTP packet where we can see 4 layers.

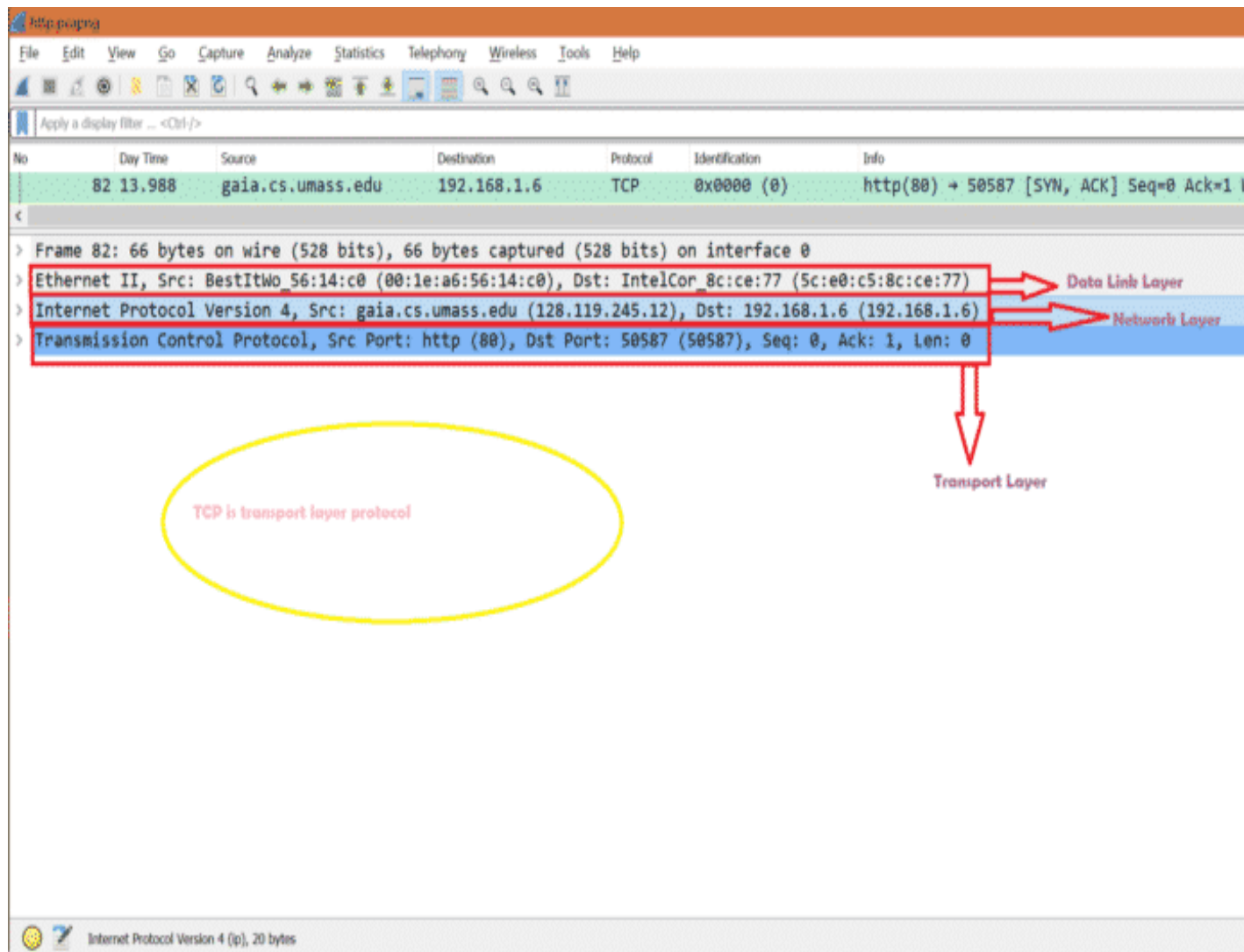


We know HTTP is an application layer so we see application layer also.

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

### **1.1.1.2 TCP [It has 3 layers]:**

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

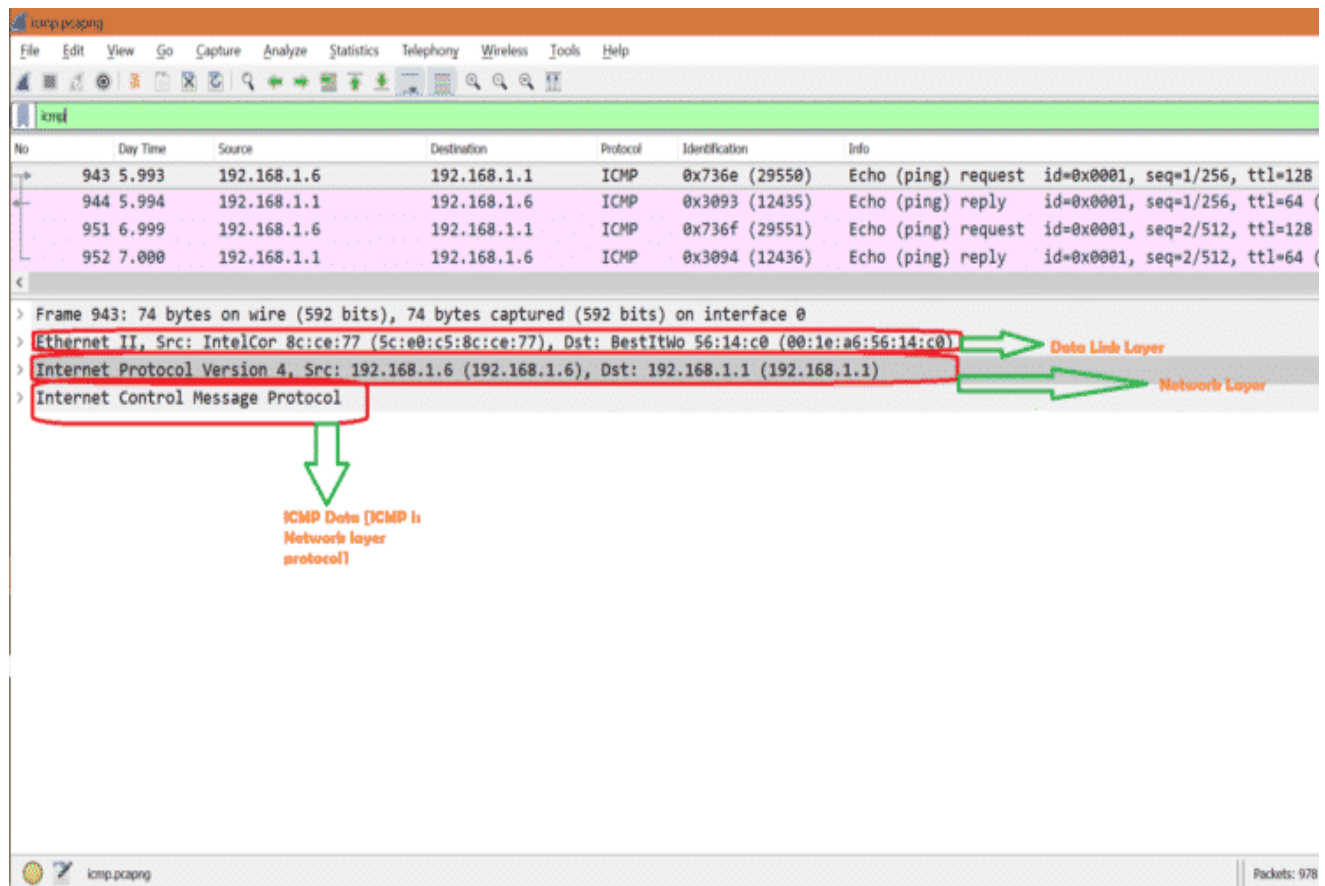


Let's see ICMP packet.

### 1.1.1.3 ICMP [It has 2 layers]:

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

-

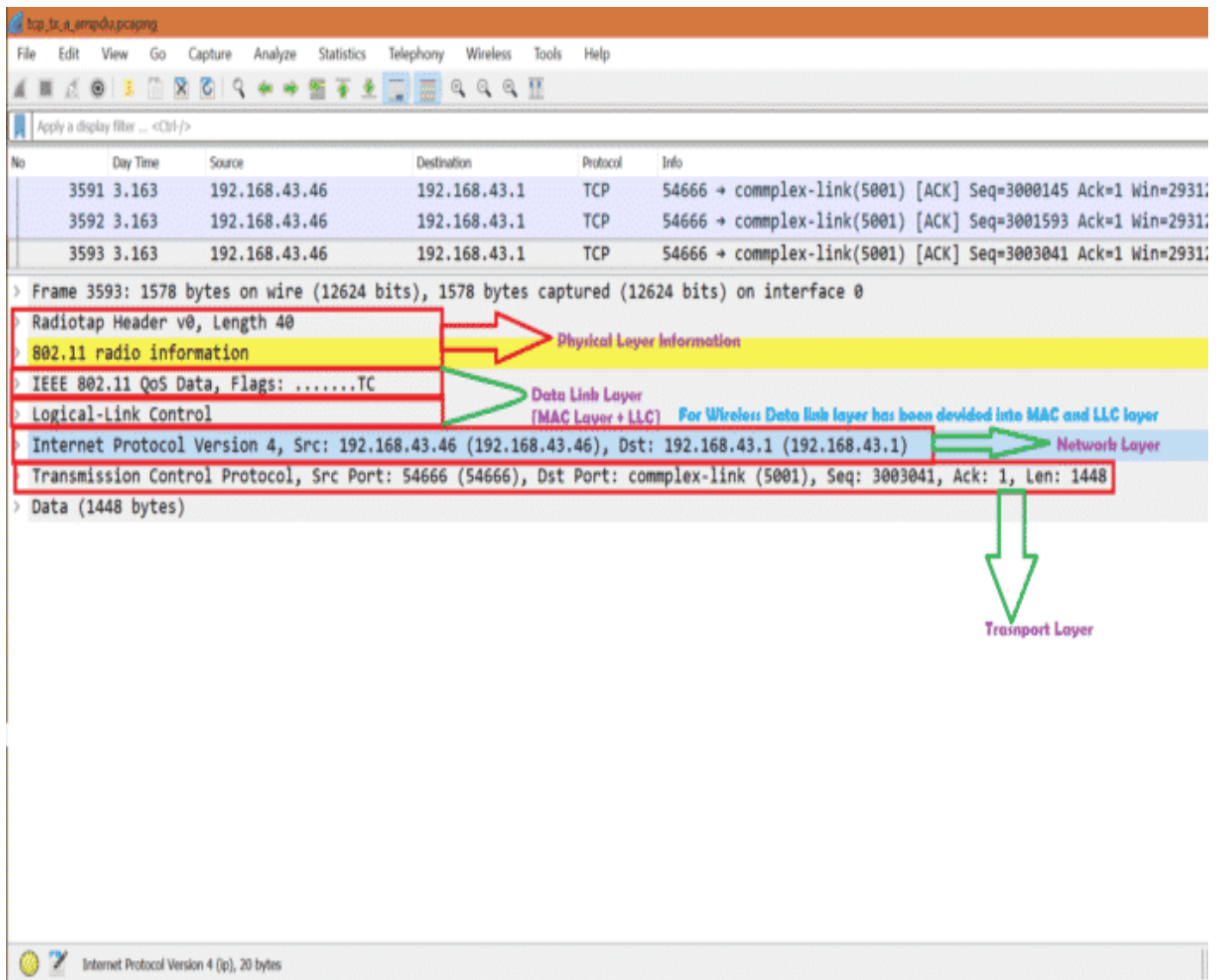


Now let's see one wireless TCP frame where we can see physical layer information.

#### 1.1.1.4 Wireless TCP [It has 4 layers]:

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



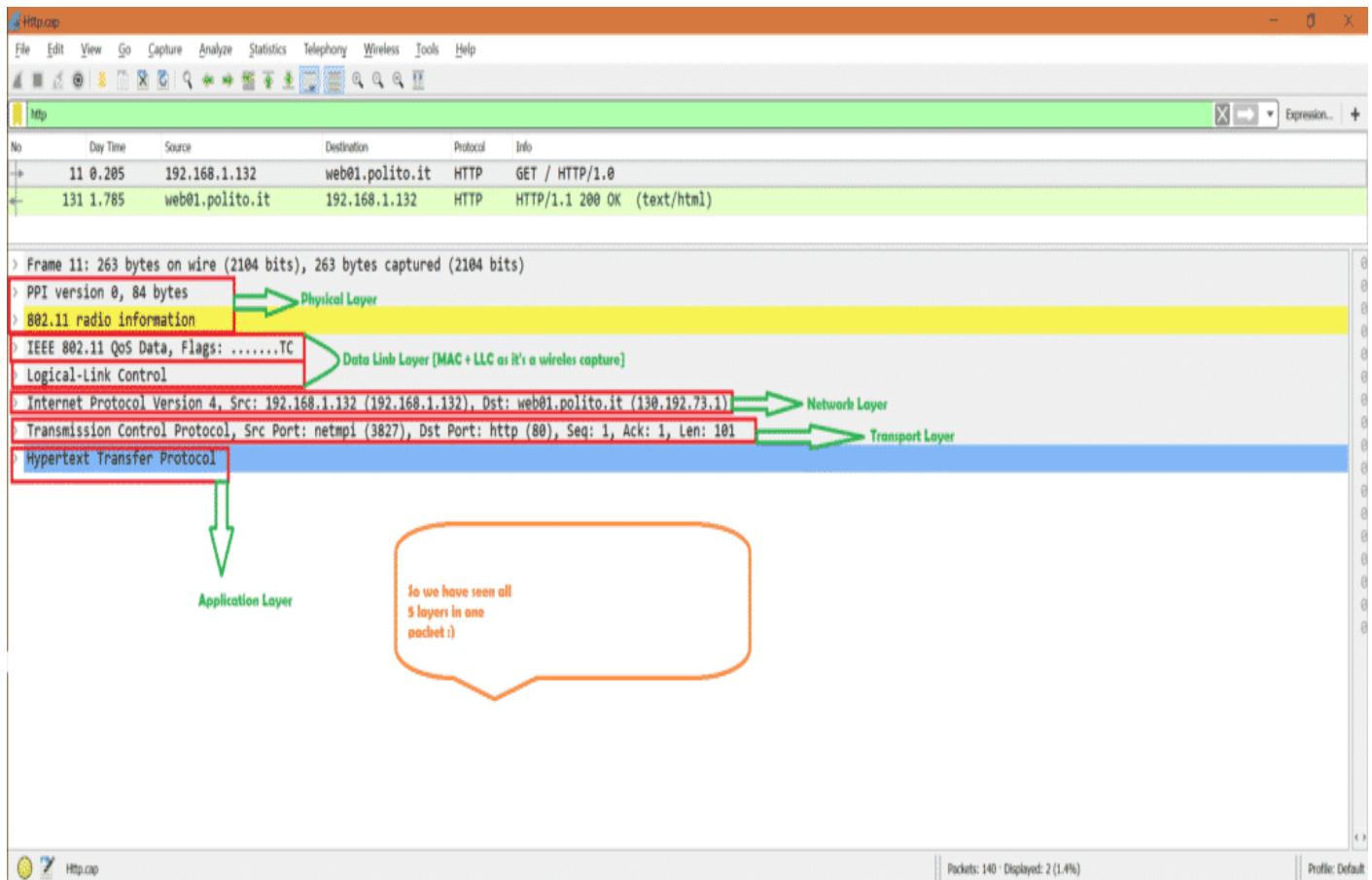


As TCP is a transport layer protocol so we did not see any application layer protocol.

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

#### **1.1.1.5 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.



### 1.1.1.6 Summary:

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

# HTTP analysis using Wireshark

## What is HTTP?

First of 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.

	OSI	TCP/IP
7	Application	Applications (FTP, SMTP, HTTP, etc.)
6	Presentation	
5	Session	
4	Transport	TCP (host-to-host)
3	Network	IP
2	Data link	Network access (usually Ethernet)
1	Physical	

HTTP is used by the [World Wide Web](http://www.w3.org/) (w.w.w) and it defines how messages are formatted and transmitted by browser. So HTTP defines rules 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 appropriate 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 retrieval. We will see one practical example of this method.

**POST:** POST method is used when it's required to send some data to server.

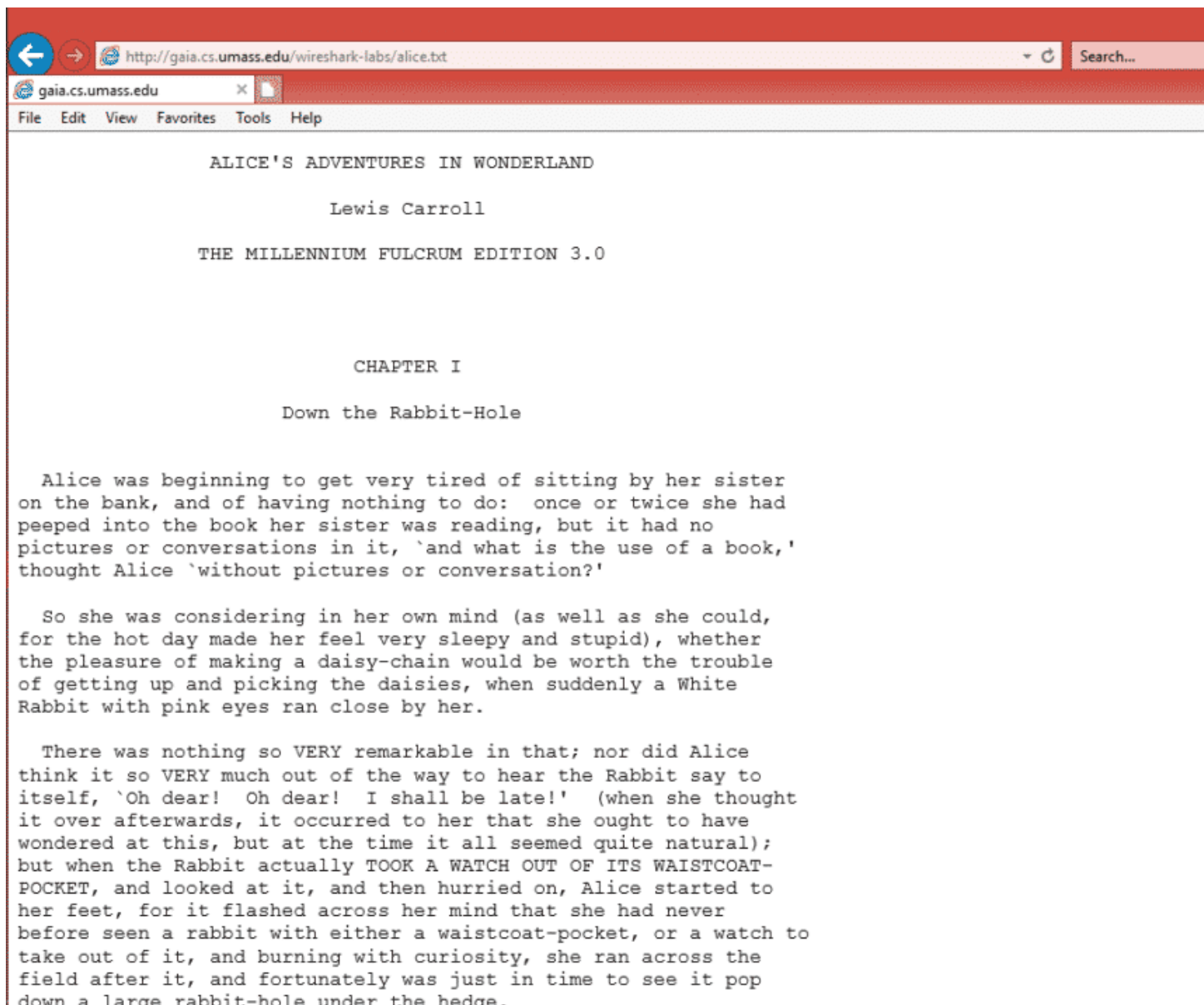
### **HTTP is Wireshark:**

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

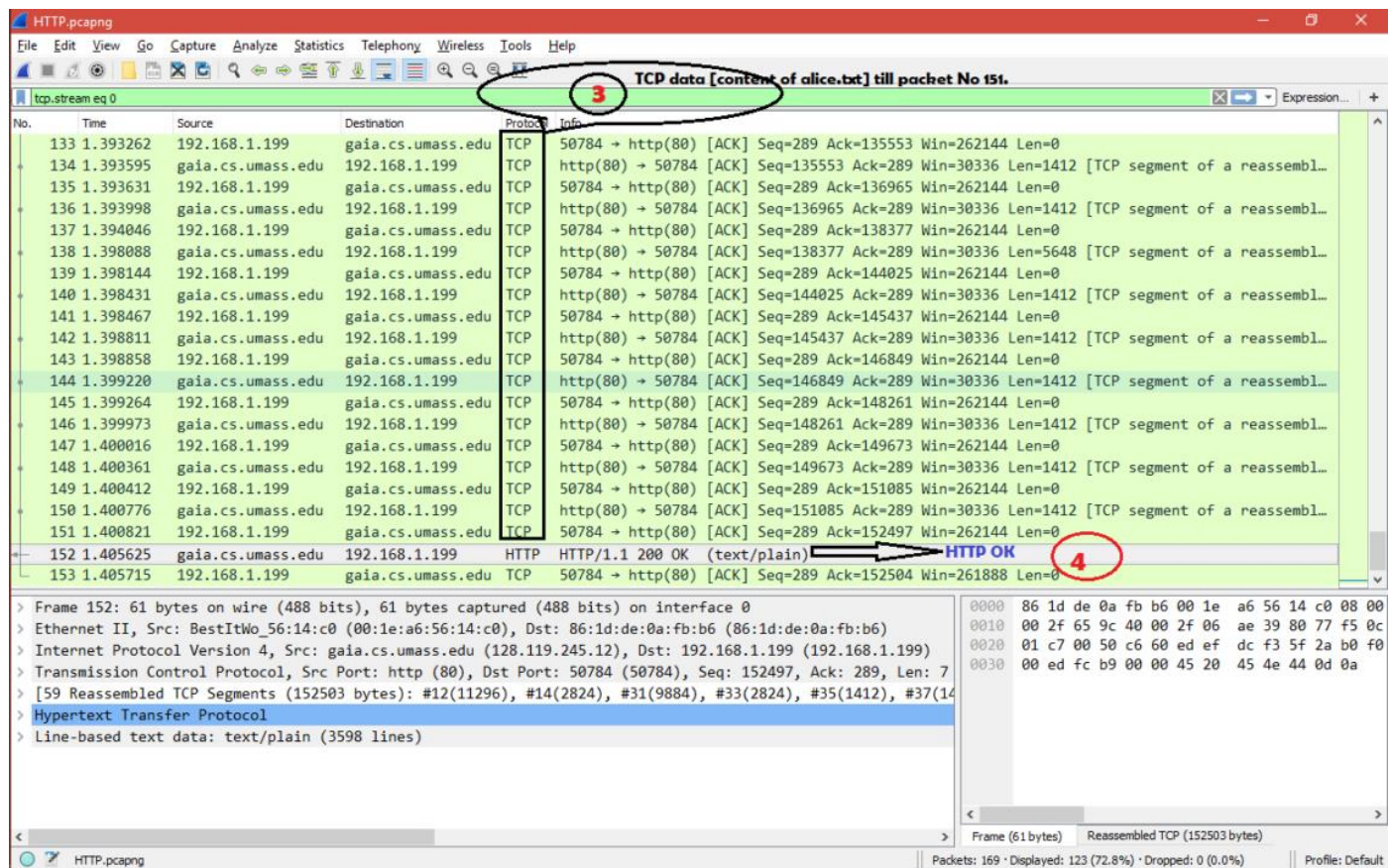
The image shows a Wireshark packet capture of an HTTP GET request for 'alice.txt'. The capture is filtered on 'tcp.stream eq 0'. The packet list shows the following packets:

- 1 0.000000 192.168.1.199 → gaia.cs.umass.edu TCP 50784 → http(80) [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK\_PERM=1
- 3 0.316619 gaia.cs.umass.edu → 192.168.1.199 TCP http(80) → 50784 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1412 SACK\_PERM=1 WS=128
- 4 0.316773 192.168.1.199 → gaia.cs.umass.edu TCP 50784 → http(80) [ACK] Seq=1 Ack=1 Win=262144 Len=0
- 7 0.317461 192.168.1.199 → gaia.cs.umass.edu HTTP GET /wireshark-labs/alice.txt HTTP/1.1
- 11 0.622871 gaia.cs.umass.edu → 192.168.1.199 TCP http(80) → 50784 [ACK] Seq=1 Ack=289 Win=30336 Len=0
- 12 0.625492 gaia.cs.umass.edu → 192.168.1.199 TCP http(80) → 50784 [ACK] Seq=1 Ack=289 Win=30336 Len=11296 [TCP segment of a reassembled P...]
- 13 0.625610 192.168.1.199 → gaia.cs.umass.edu TCP 50784 → http(80) [ACK] Seq=289 Ack=11297 Win=262144 Len=0
- 14 0.626037 gaia.cs.umass.edu → 192.168.1.199 TCP http(80) → 50784 [ACK] Seq=11297 Ack=289 Win=30336 Len=2824 [TCP segment of a reassembled...]
- 15 0.626099 192.168.1.199 → gaia.cs.umass.edu TCP 50784 → http(80) [ACK] Seq=289 Ack=14121 Win=262144 Len=0
- 31 0.878185 gaia.cs.umass.edu → 192.168.1.199 TCP http(80) → 50784 [ACK] Seq=14121 Ack=289 Win=30336 Len=9884 [TCP segment of a reassembled...]
- 32 0.878314 192.168.1.199 → gaia.cs.umass.edu TCP 50784 → http(80) [ACK] Seq=289 Ack=24005 Win=262144 Len=0
- 33 0.878749 gaia.cs.umass.edu → 192.168.1.199 TCP http(80) → 50784 [ACK] Seq=24005 Ack=289 Win=30336 Len=2824 [TCP segment of a reassembled...]
- 34 0.878799 192.168.1.199 → gaia.cs.umass.edu TCP 50784 → http(80) [ACK] Seq=289 Ack=26829 Win=262144 Len=0
- 35 0.878971 gaia.cs.umass.edu → 192.168.1.199 TCP http(80) → 50784 [ACK] Seq=26829 Ack=289 Win=30336 Len=1412 [TCP segment of a reassembled...]
- 36 0.879020 192.168.1.199 → gaia.cs.umass.edu TCP 50784 → http(80) [ACK] Seq=289 Ack=28241 Win=262144 Len=0
- 37 0.879361 gaia.cs.umass.edu → 192.168.1.199 TCP http(80) → 50784 [ACK] Seq=28241 Ack=289 Win=30336 Len=1412 [TCP segment of a reassembled...]
- 38 0.879403 192.168.1.199 → gaia.cs.umass.edu TCP 50784 → http(80) [ACK] Seq=289 Ack=29653 Win=262144 Len=0
- 39 0.879722 gaia.cs.umass.edu → 192.168.1.199 TCP http(80) → 50784 [ACK] Seq=29653 Ack=289 Win=30336 Len=1412 [TCP segment of a reassembled...]
- 40 0.879812 192.168.1.199 → gaia.cs.umass.edu TCP 50784 → http(80) [ACK] Seq=289 Ack=31065 Win=262144 Len=0
- 41 0.882685 gaia.cs.umass.edu → 192.168.1.199 TCP http(80) → 50784 [ACK] Seq=31065 Ack=289 Win=30336 Len=7060 [TCP segment of a reassembled...]
- 42 0.882796 192.168.1.199 → gaia.cs.umass.edu TCP 50784 → http(80) [ACK] Seq=289 Ack=38125 Win=262144 Len=0

The packet details for packet 7 (HTTP GET) are shown below:

```
> Frame 7: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface 0
> Ethernet II, Src: 86:1d:de:0a:fb:b6 (86:1d:de:0a:fb:b6), Dst: BestItWo_56:14:c0 (00:1e:a6:56:14:c0)
> Internet Protocol Version 4, Src: 192.168.1.199 (192.168.1.199), Dst: gaia.cs.umass.edu (128.119.245.12)
> Transmission Control Protocol, Src Port: 50784 (50784), Dst Port: http (80), Seq: 1, Ack: 1, Len: 288
> Hypertext Transfer Protocol
```

The packet bytes are displayed in hexadecimal and ASCII format on the right side of the window.



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.pcapng

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

tcp.stream eq 0

No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.1.199	gaia.cs.umass.edu	TCP	50784 → http(80) [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=
3	0.316619	gaia.cs.umass.edu	192.168.1.199	TCP	http(80) → 50784 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 M
4	0.316773	192.168.1.199	gaia.cs.umass.edu	TCP	50784 → http(80) [ACK] Seq=1 Ack=1 Win=262144 Len=0
7	0.317461	192.168.1.199	gaia.cs.umass.edu	HTTP	GET /wireshark-labs/alice.txt HTTP/1.1

> Frame 7: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface 0

> Ethernet II, Src: 86:1d:de:0a:fb:b6 (86:1d:de:0a:fb:b6), Dst: BestItWo\_56:14:c0 (00:1e:a6:56:14:c0)

> Internet Protocol Version 4, Src: 192.168.1.199 (192.168.1.199), Dst: gaia.cs.umass.edu (128.119.245.12)

> Transmission Control Protocol, Src Port: 50784 (50784), Dst Port: http (80), Seq: 1, Ack: 1, Len: 288

▼ Hypertext Transfer Protocol

▼ GET /wireshark-labs/alice.txt HTTP/1.1\r\n

> [Expert Info (Chat/Sequence): GET /wireshark-labs/alice.txt HTTP/1.1\r\n]

Request Method: GET 1

Request URI: /wireshark-labs/alice.txt 2

Request Version: HTTP/1.1 3

Accept: text/html, application/xhtml+xml, image/jxr, \*/\*\r\n 4

<Accept: text/html, application/xhtml+xml, image/jxr, \*/\*\r\n>

Accept-Language: en-US\r\n 5

<Accept-Language: en-US\r\n>

User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko\r\n 6

<User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; Trident/7.0; rv:11.0) like Gecko\r\n>

Accept-Encoding: gzip, deflate\r\n 7

<Accept-Encoding: gzip, deflate\r\n>

Host: gaia.cs.umass.edu\r\n 8

<Host: gaia.cs.umass.edu\r\n>

Connection: Keep-Alive\r\n 9

<Connection: Keep-Alive\r\n>

\r\n

[Full request URI: http://gaia.cs.umass.edu/wireshark-labs/alice.txt]

<Request: True>

[HTTP request 1/1]

[Response in frame: 152]

This is the complete URL. This is not present in HTTP GET packet.

HTTP Request Method (http.request.method), 3 bytes

Packets: 169 · Display



## 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.

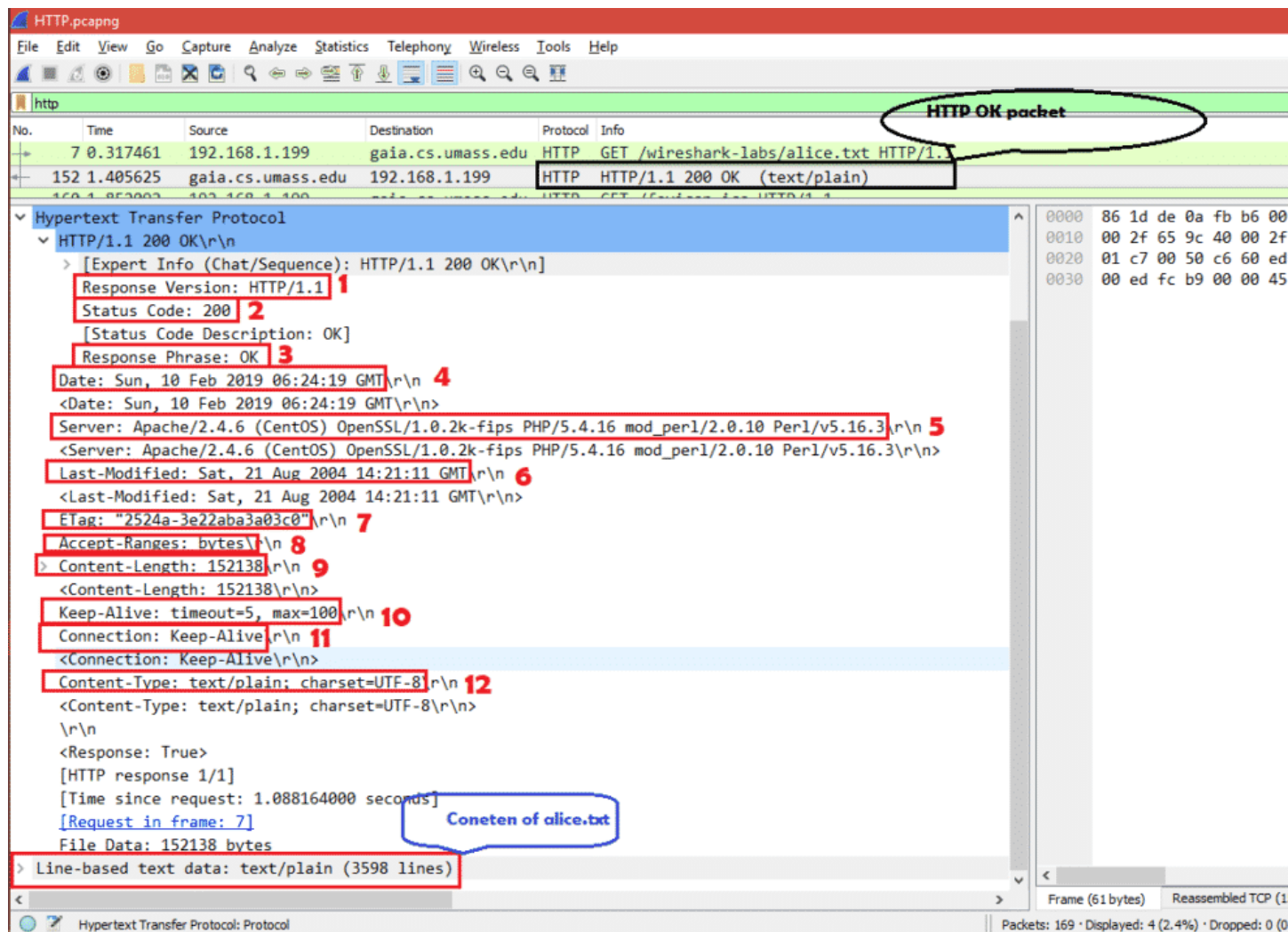
**9. Content-Length: 152138** ==> This is the total length of the alice.txt in bytes.

**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.

### 1.1.1.7 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 `https://[website name]`. This means all communication between you and server are encrypted.

### **STUDENT TASK-01**

Students should make the scenario exactly given in the lab objectives. Implement the following configuration Email Server STMP, DNS DHCP and Webserver . Students shall implement the security techniques used to secure the routers. Submit a final report after doing the mentioned task physically.



