NATIONAL UNIVERSITY OF COMPUTER & EMERGING SCIENCE
Computer Network Lab (CL3001)
Lab Session 06

Objective:

- Implementation & understanding of HTTP/HTTPS.
- Network traffic analysis of HTTP/S protocol headers, cookies using Wireshark

HTTP/HTTPS

1. Hypertext Transfer Protocol (HTTP):

Hypertext Transfer Protocol (HTTP) is a protocol used in networking. When you type any web address in your web browser, your browser acts as a client, and the computer having the requested information acts as a server. When client requests for any information from the server, it uses HTTP protocol to do so. The server responds back to the client after the request completes. The response comes in the form of web page which you see just after typing the web address and press "Enter".

2. Hypertext Transfer Protocol Secure (HTTPS):

Hypertext Transfer Protocol Secure (HTTPS) is a combination of two different protocols. It is more secure way to access the web. It is combination of Hypertext Transfer Protocol (HTTPS) and SSL/TLS protocol. It is more secure way to sending request to server from a client, also the communication is purely encrypted which means no one can know what you are looking for. This kind of communication is used for accessing those websites where security is required. Banking websites, payment gateway, emails (Gmail offers HTTPS by default in Chrome browser), and corporate sector websites are some great examples where HTTPS protocols are used.
For HTTPS connection, public key trusted and signed certificate is required for the server. Thesecertificates come either free or it costs few dollars depends on the signing authority. There is one other method for distributing certificates. Site admin creates certificates and loads in the browser of users. Now when user requests information to the web server, his identity can be verified easily.

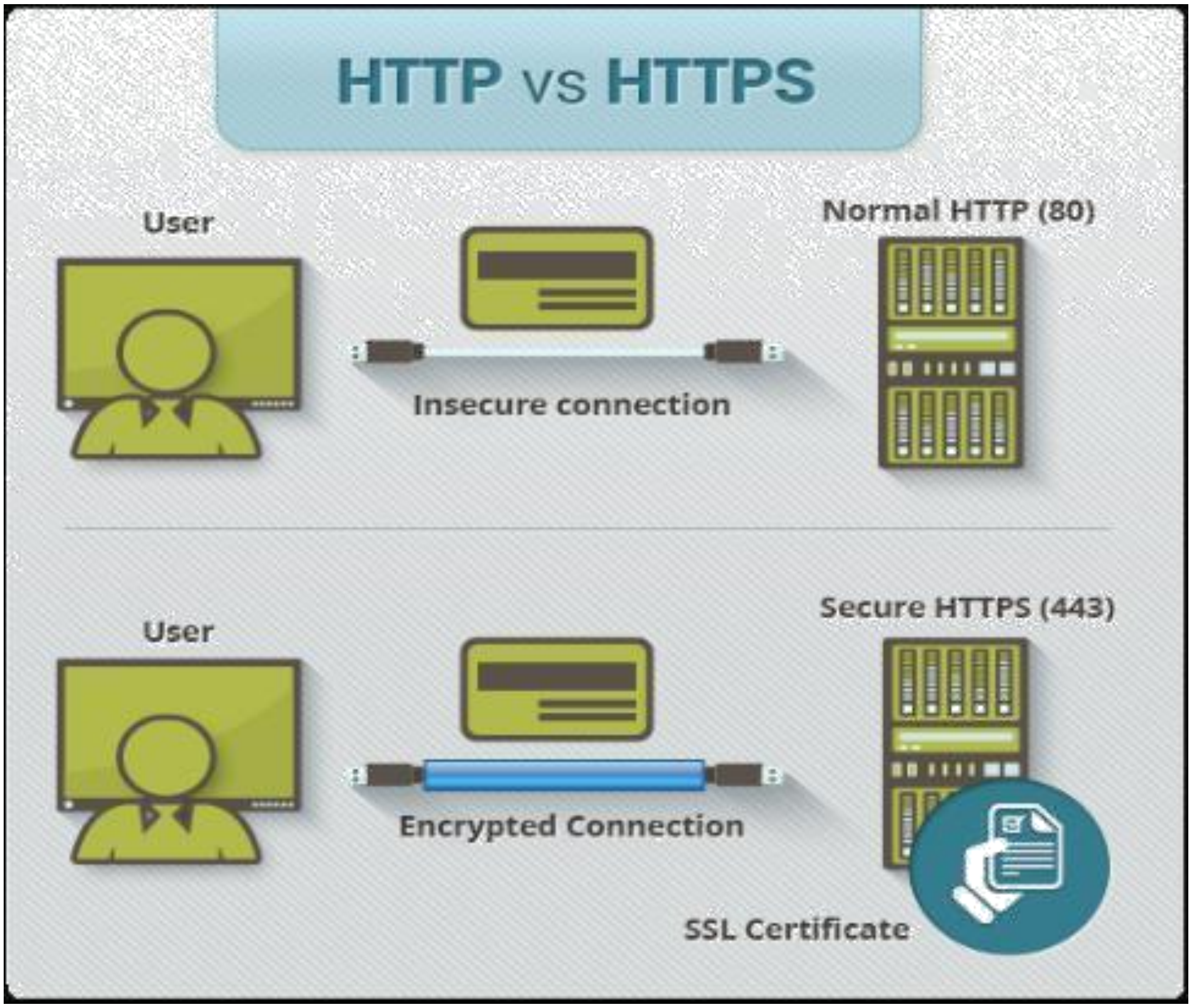


Fig-1: HTTP & HTTPS difference

3. HTTP & HTTPS Differences:

Here are some major difference between HTTP & HTTPS

HTTP	HTTPS
URL begins with “http://”	URL begins with “https://”
It uses port 80 for communication	It uses port 443 for communication
Unsecured	Secured
Operates at Application Layer	Operates at Transport Layer
No encryption	Encryption is present
No certificates required	Certificates required

4a. Client Error:

The 4xx class of status code is intended for cases in which the client seems to have erred. Exceptwhen responding to a HEAD request, the server should include an entity containing an explanation of the error situation, and whether it is a temporary or permanent condition. These status codes are applicable to any request method. User agents should display any included entity to the user.

400 Bad Request:

The server cannot or will not process the request due to something that is perceived tobe a client error (e.g., malformed request syntax, invalid request message framing, or deceptive request routing).

401 Unauthorized (RFC 7235):

Similar to 403 Forbidden, but specifically for use when authentication is required and hasfailed or has not yet been provided. The response must include a WWW -Authenticate header field containing a challenge applicable to the requested resource. See Basic accessauthentication and Digest access authentication.

403 Forbidden:

The request was a valid request, but the server is refusing to respond to it. Unlike a 401 unauthorized response, authenticating will make no difference**.**

404 Not Found:

The requested resource could not be found but may be available again in the future.Subsequent requests by the client are permissible.

408 Request Timeout:

The server timed out waiting for the request. According to HTTP specifications: "The client did not produce a request within the time that the server was prepared to wait. Theclient MAY repeat the request without modifications at any later time."

4b. Server Error:

The server failed to fulfill an apparently valid request.
Response status codes beginning with the digit "5" indicate cases in which the server is aware that it has encountered an error or is otherwise incapable of performing the request. Except when responding to a HEAD request, the server should include an entity containing an explanation of the error situation, and indicate whether it is a temporary or permanent condition. Likewise, user agents should display any included entity to the user. These response codes are applicable to any request method.

500 Internal Server Error:

A generic error message, given when an unexpected condition was encountered andno more specific message is suitable.

501 Not Implemented:

The server either does not recognize the request method, or it lacks the ability to fulfil the request. Usually this implies future availability (e.g., a new feature of a web-service API).

502 Bad Gateway:

The server was acting as a gateway or proxy and received an invalid response from the upstream server.

503 Service Unavailable:

The server is currently unavailable (because it is overloaded or down for maintenance). Generally, this is a temporary state.

5. Implementation:

Design the given topology shown in figure 2. Assign IP address to PC using DHCP through router as done in pervious lab.

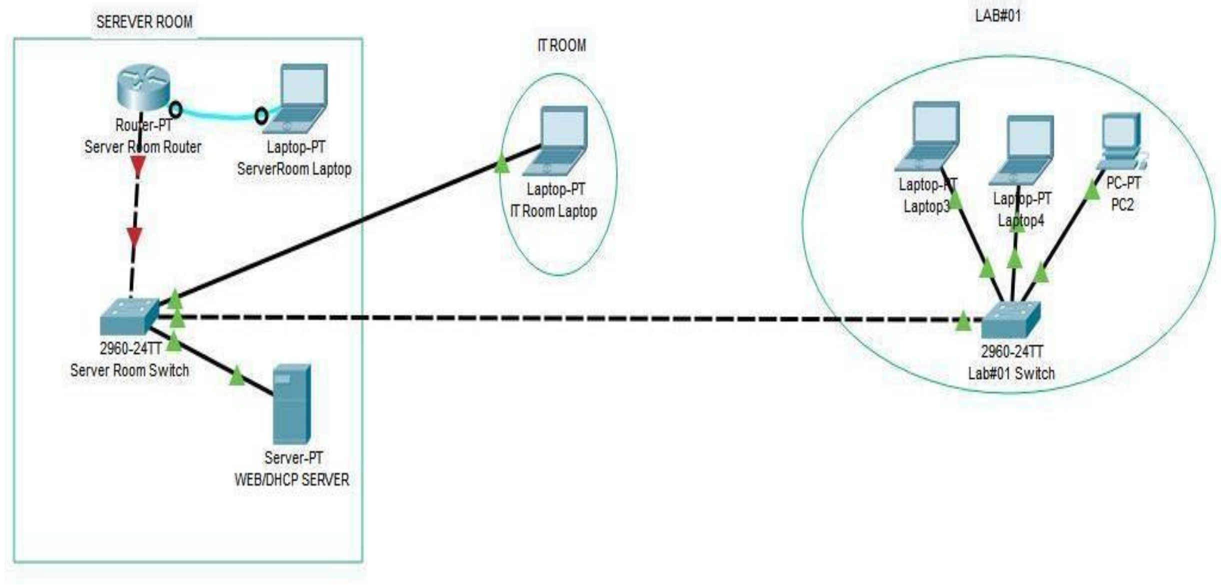
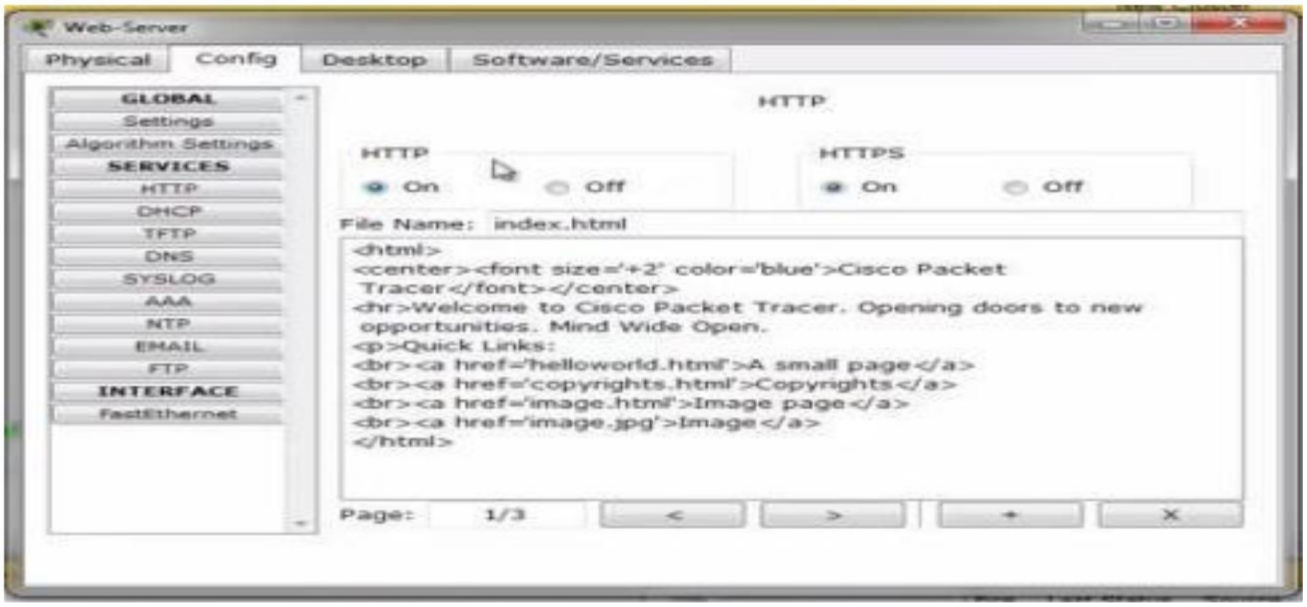


Fig-2: Lab 5 network topology

The above topology configured as “one server room”, “one IT room: and “Lab#01 environment having three systems”. On our server we have enabled web services as well as DNS services. Click on the web server, go to config --->services—HTTP
Here you can see HTTP & HTTPS services are on.



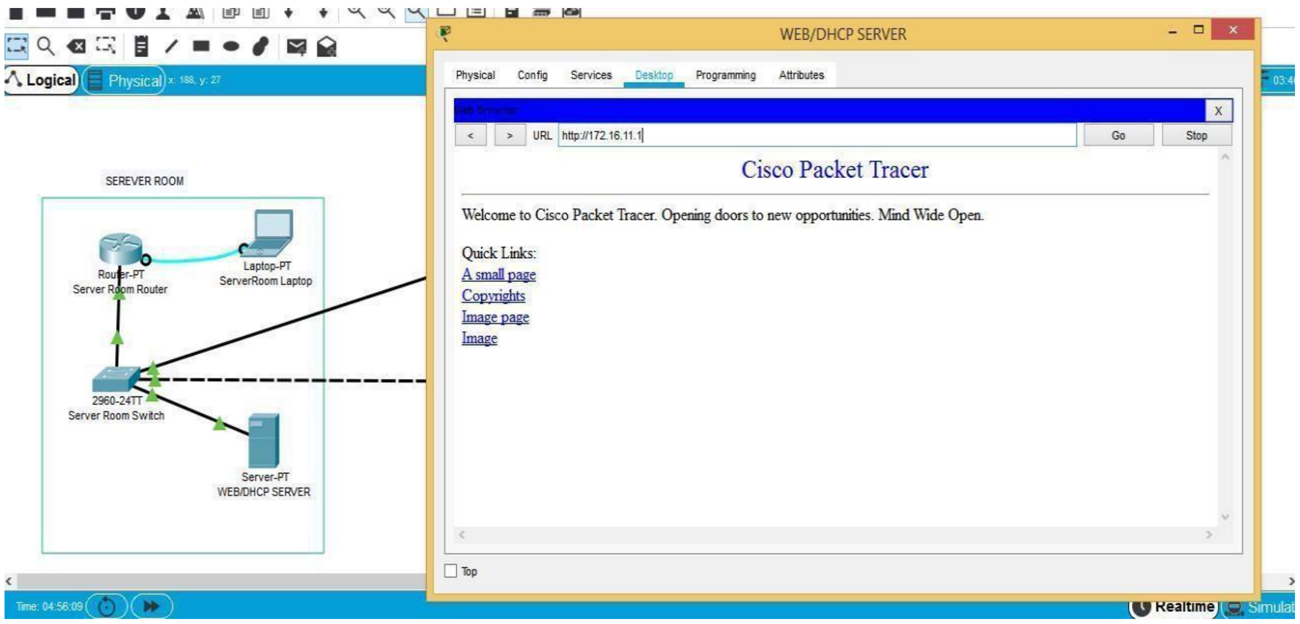
HTTP: [✓ On] Off
HTTPS: [✓ On] Off
File Name: index.html

```
<html>
<center><font size='+2' color='blue'>Cisco Packet
 Tracer</font></center>
<hr>Welcome to Cisco Packet Tracer. Opening doors to new
 opportunities. Mind Wide Open.
<p>Quick Links:
<br><a href='helloworld.html'>A small page</a>
<br><a href='copyrights.html'>Copyrights</a>
<br><a href='image.html'>Image page</a>
<br><a href='image.jpg'>Image</a>
</html>
```

Page: 1/3

Fig-3: HTTP services on server interface in Packet Tracer

Now click on PC0 and go to Desktop -> Web Browser. Now type web-server IP which you have assign or the website name which you have store in the DNS server record.



URL http://172.16.11.1

Cisco Packet Tracer

Welcome to Cisco Packet Tracer. Opening doors to new opportunities. Mind Wide Open.

Quick Links:
A small page
Copyrights
Image page
Image

Fig-4: HTTP services on server interface in Packet Tracer

To note the http header format information, go to simulation mode edit filters and clickon http check box then click on capture/forward button.

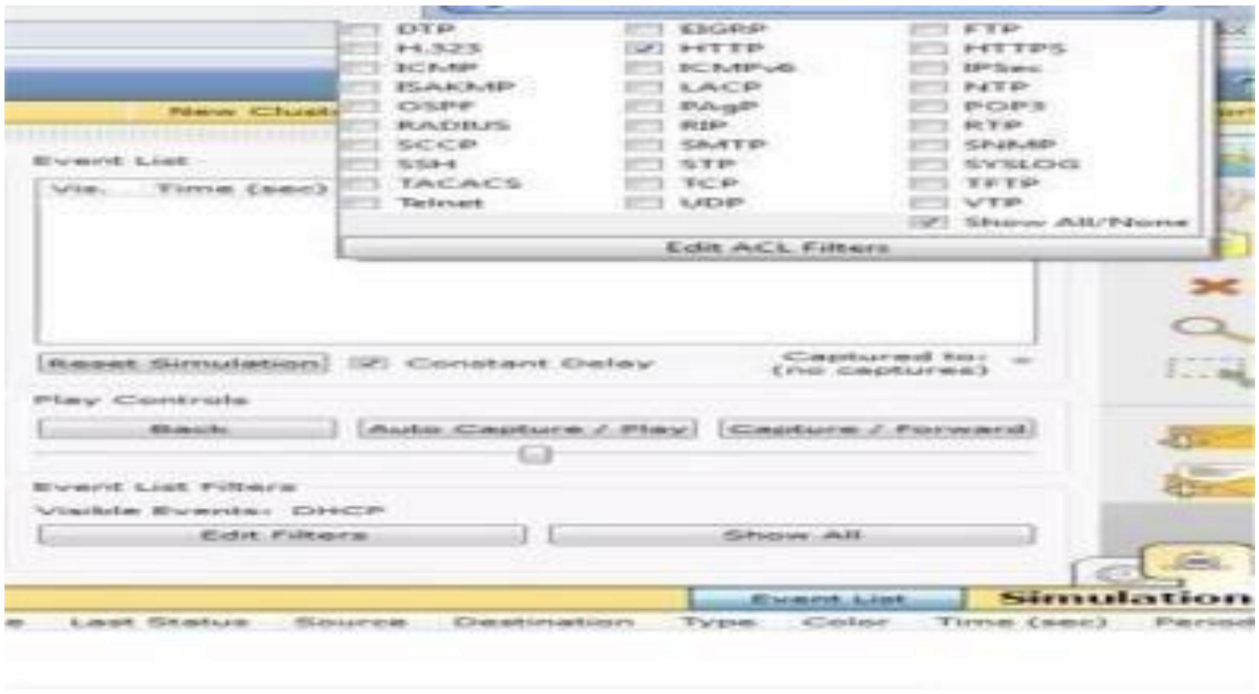
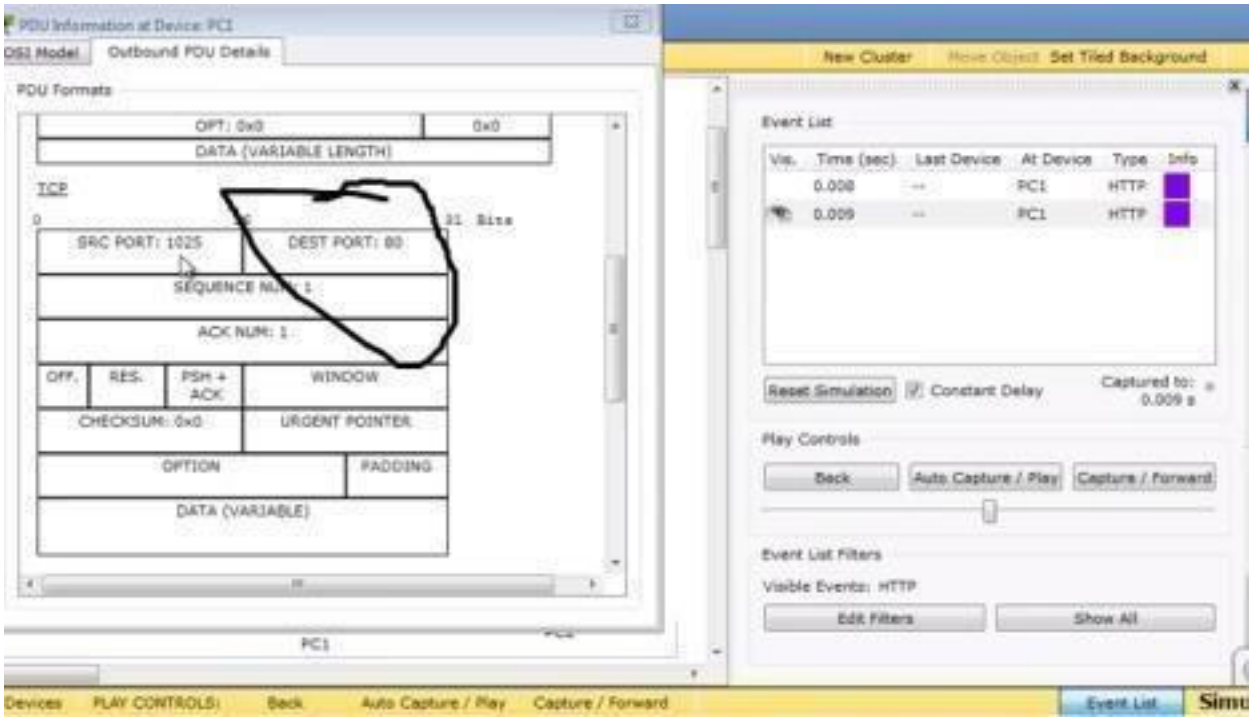


Fig-5: Packet Tracer Simulation Mode Interface

Now click on the http packet, you can note that the destination port is 80.



OPT: 0x0 | 0x0
DATA (VARIABLE LENGTH)

TCP

0 | 16 | 31 Bits

SRC PORT: 1025 | DEST PORT: 80
SEQUENCE NUM: 1
ACK NUM: 1
OFF. | RES. | PSH + ACK | WINDOW
CHECKSUM: 0x0 | URGENT POINTER
OPTION | PADDING
DATA (VARIABLE)

Event List

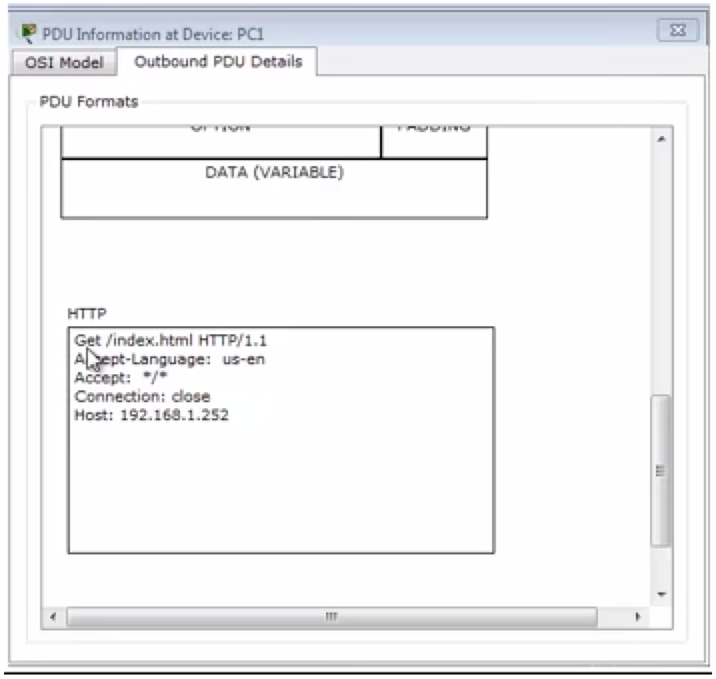
Vis.	Time (sec)	Last Device	At Device	Type	Info
	0.008	--	PC1	HTTP	🟪
	0.009	--	PC1	HTTP	🟪

Captured to: 0.009 s

Visible Events: HTTP

Fig-6: HTTP PDU in Packet Tracer

Now scroll the Outbound PDU Details, you can see the http protocol information.



OPTION | PADDING
DATA (VARIABLE)

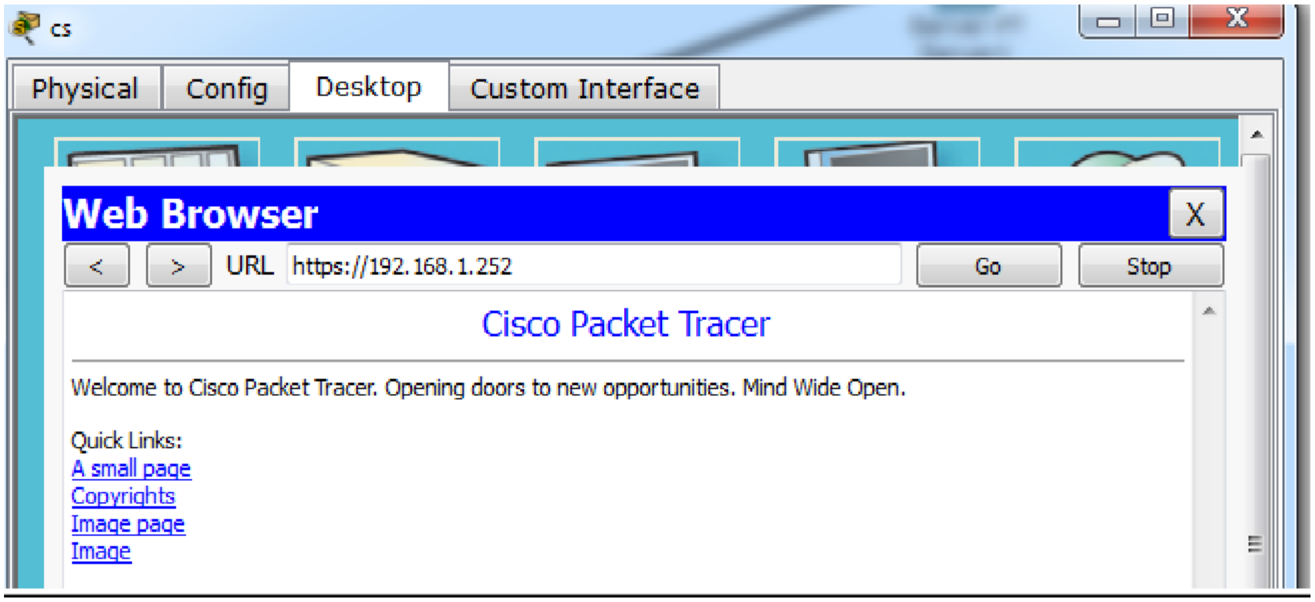
HTTP

```
Get /index.html HTTP/1.1
Accept-Language:  us-en
Accept:  */*
Connection: close
Host: 192.168.1.252
```

Fig-7: HTTP details in PDU

For HTTPS:

Now click on PC and go to Desktop ---->Web Browser. Now type web-server IP 192.168.1.252



URL https://192.168.1.252

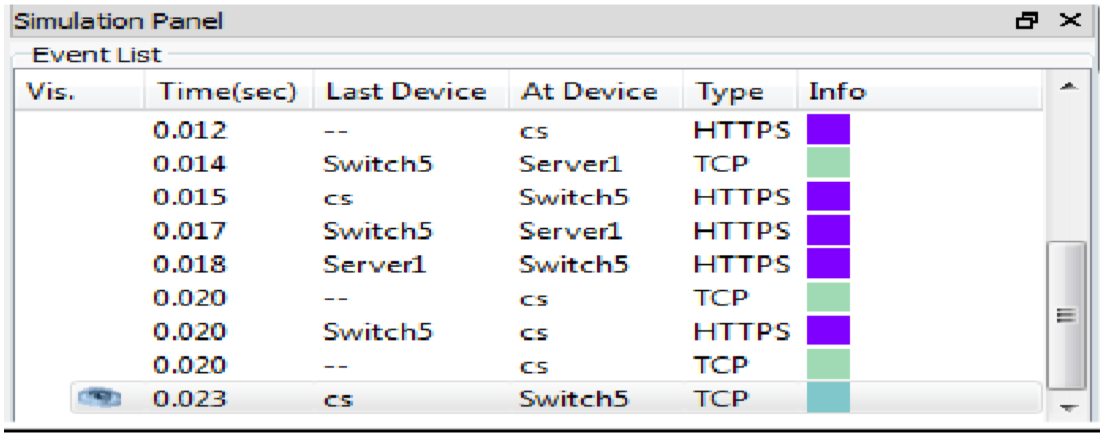
Cisco Packet Tracer

Welcome to Cisco Packet Tracer. Opening doors to new opportunities. Mind Wide Open.

Quick Links:
A small page
Copyrights
Image page
Image

Fig-8: Web page using HTTPS

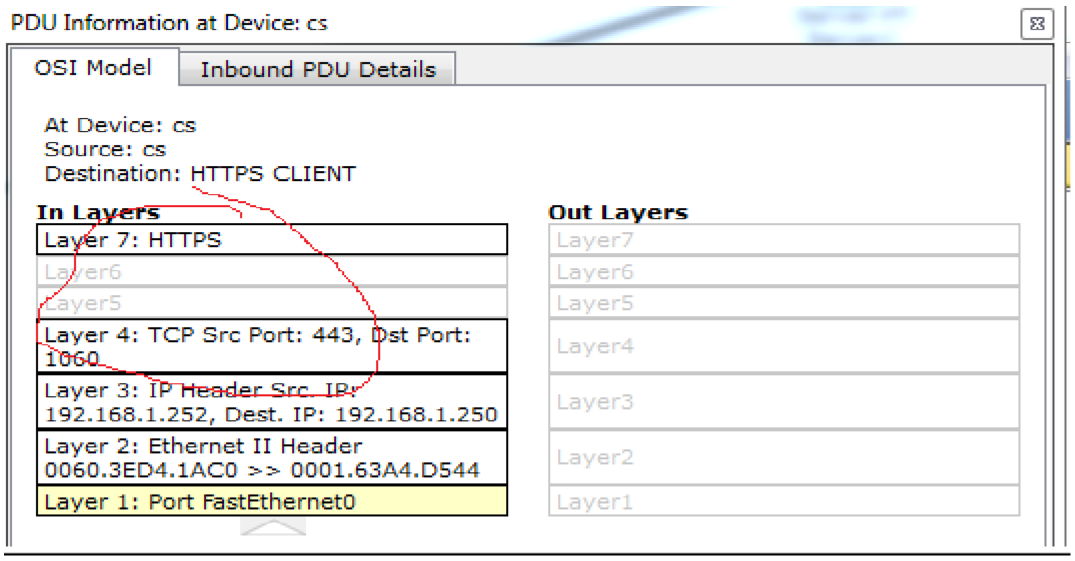
Now to note the https header format information go to simulation mode ---> editfilters and click on https check box then click on capture/forward button.



Vis.	Time(sec)	Last Device	At Device	Type	Info
	0.012	--	cs	HTTPS	🟪
	0.014	Switch5	Server1	TCP	🟩
	0.015	cs	Switch5	HTTPS	🟪
	0.017	Switch5	Server1	HTTPS	🟪
	0.018	Server1	Switch5	HTTPS	🟪
	0.020	--	cs	TCP	🟩
	0.020	Switch5	cs	HTTPS	🟪
	0.020	--	cs	TCP	🟩
	0.023	cs	Switch5	TCP	🟩

Fig-9: Packets flow in simulation

Now click on the https packet, you can note that the destination port is 443.


PDU Information at Device: cs

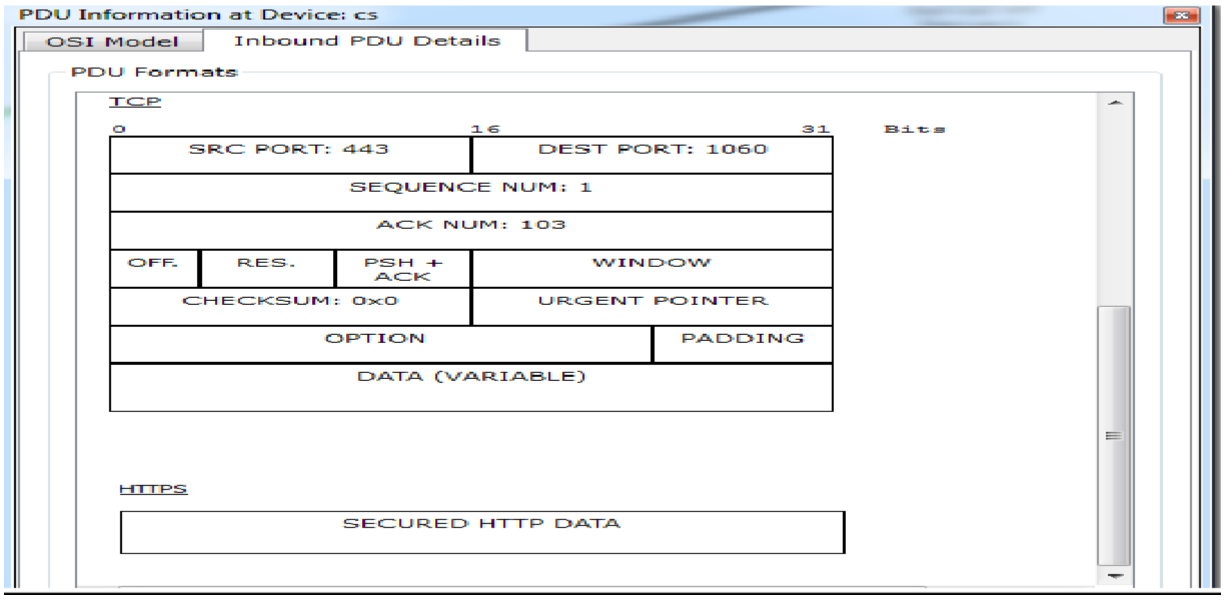
OSI Model | Inbound PDU Details

At Device: cs
Source: cs
Destination: HTTPS CLIENT

In Layers	Out Layers
Layer 7: HTTPS	Layer7
Layer6	Layer6
Layer5	Layer5
Layer 4: TCP Src Port: 443, Dst Port: 1060	Layer4
Layer 3: IP Header Src. IP: 192.168.1.252, Dest. IP: 192.168.1.250	Layer3
Layer 2: Ethernet II Header 0060.3ED4.1AC0 >> 0001.63A4.D544	Layer2
Layer 1: Port FastEthernet0	Layer1

Fig-10: Packet information

Now scroll the Outbound PDU Details, you can see the https PDU.


PDU Information at Device: cs

OSI Model | Inbound PDU Details

PDU Formats

TCP

0 16 31 Bits

SRC PORT: 443		DEST PORT: 1060	
SEQUENCE NUM: 1			
ACK NUM: 103			
OFF.	RES.	PSH + ACK	WINDOW
CHECKSUM: 0x0		URGENT POINTER	
OPTION			PADDING
DATA (VARIABLE)			

HTTPS

SECURED HTTP DATA

Fig-11: HTTPs PDU details

6. Lab Exercise:

Q1) In caching, what is the difference between the age header and expires?
Q2) What are the four groupings of HTTP headers?
Q3) Implement the below topology using web server.

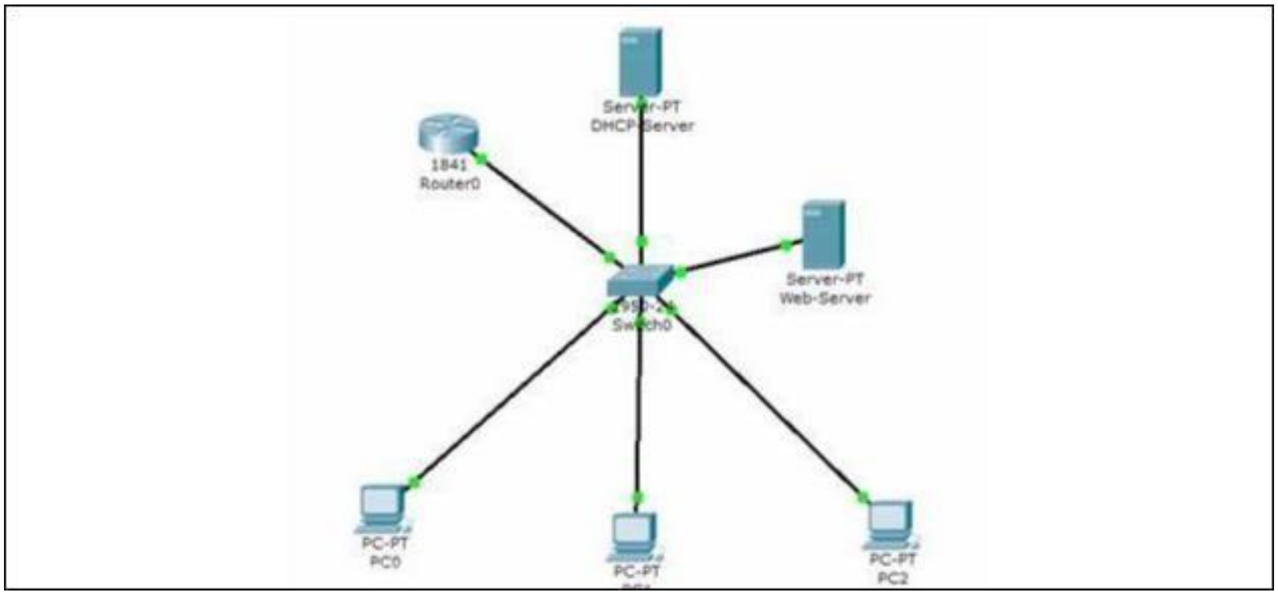


Fig-12: Exercise Topology

Wireshark: HTTP

We're now ready to use Wireshark to investigate protocols in operation. In this lab, we'll explore several aspects of the HTTP protocol: the basic GET/response interaction, HTTP messageformats, retrieving large HTML files, retrieving HTML files with embedded objects, and HTTP authentication and security.

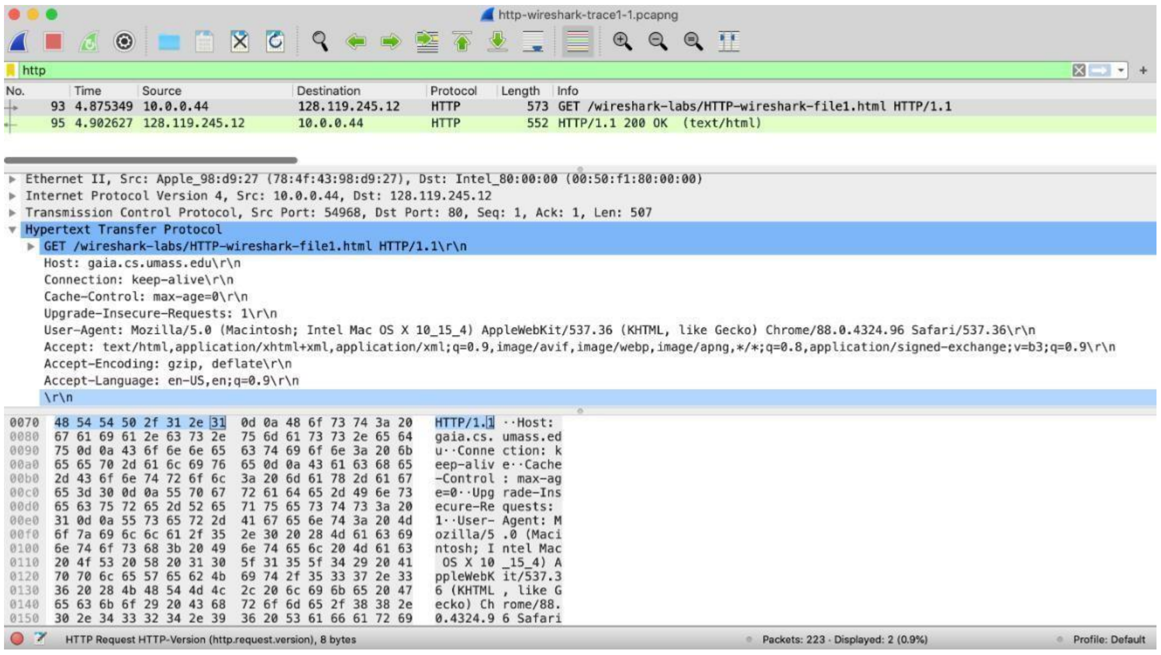
7. The Basic HTTP GET/response interaction:

Let's begin our exploration of HTTP by downloading a very simple HTML file - one that is very short and contains no embedded objects.

Do the following:

1. Start up your web browser.
2. Start up the Wireshark packet sniffer, as described in the introductory lab (but don't yet begin packet capture). Enter "http" (just the letters, not the quotation marks) in the display-filter- specification window, so that only captured HTTP messages will be displayed later in the packet-listing window. (We're only interested in the HTTP protocol here, and don't want to see the clutter of all captured packets).
3. Wait a bit more than one minute (we'll see why shortly), and then beginWireshark packet capture.
4. Enter the following to your browser http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html. Your browser should display the very simple, one-line HTML file.
5. Stop Wireshark packet capture.

Your Wireshark window should look like the window shown in Figure 13. If you are unable to runWireshark on a live network connection, you can download a packet trace that was created when the steps above were followed.



```
No.    Time     Source          Destination     Protocol  Length  Info
   93  4.875349 10.0.0.44       128.119.245.12  HTTP         573  GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1
   95  4.902627 128.119.245.12  10.0.0.44       HTTP         552  HTTP/1.1 200 OK  (text/html)
```

```
▶ Ethernet II, Src: Apple_98:d9:27 (78:4f:43:98:d9:27), Dst: Intel_80:00:00 (00:50:f1:80:00:00)
▶ Internet Protocol Version 4, Src: 10.0.0.44, Dst: 128.119.245.12
▶ Transmission Control Protocol, Src Port: 54968, Dst Port: 80, Seq: 1, Ack: 1, Len: 507
▼ Hypertext Transfer Protocol
  ▶ GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1\r\n
    Host: gaia.cs.umass.edu\r\n
    Connection: keep-alive\r\n
    Cache-Control: max-age=0\r\n
    Upgrade-Insecure-Requests: 1\r\n
    User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.96 Safari/537.36\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9\r\n
    Accept-Encoding: gzip, deflate\r\n
    Accept-Language: en-US,en;q=0.9\r\n
    \r\n
```

```
0070  48 54 54 50 2f 31 2e 31  0d 0a 48 6f 73 74 3a 20   HTTP/1.1 ··Host: 
0080  67 61 69 61 2e 63 73 2e  75 6d 61 73 73 2e 65 64   gaia.cs. umass.ed
0090  75 0d 0a 43 6f 6e 6e 65  63 74 69 6f 6e 3a 20 6b   u··Conne ction: k
00a0  65 65 70 2d 61 6c 69 76  65 0d 0a 43 61 63 68 65   eep-aliv e··Cache
00b0  2d 43 6f 6e 74 72 6f 6c  3a 20 6d 61 78 2d 61 67   -Control : max-ag
00c0  65 3d 30 0d 0a 55 70 67  72 61 64 65 2d 49 6e 73   e=0··Upg rade-Ins
00d0  65 63 75 72 65 2d 52 65  71 75 65 73 74 73 3a 20   ecure-Re quests: 
00e0  31 0d 0a 55 73 65 72 2d  41 67 65 6e 74 3a 20 4d   1··User- Agent: M
00f0  6f 7a 69 6c 6c 61 2f 35  2e 30 20 28 4d 61 63 69   ozilla/5 .0 (Maci
0100  6e 74 6f 73 68 3b 20 49  6e 74 65 6c 20 4d 61 63   ntosh; I ntel Mac
0110  20 4f 53 20 58 20 31 30  5f 31 35 5f 34 29 20 41    OS X 10 _15_4) A
0120  70 70 6c 65 57 65 62 4b  69 74 2f 35 33 37 2e 33   ppleWebK it/537.3
0130  36 20 28 4b 48 54 4d 4c  2c 20 6c 69 6b 65 20 47   6 (KHTML , like G
0140  65 63 6b 6f 29 20 43 68  72 6f 6d 65 2f 38 38 2e   ecko) Ch rome/88.
0150  30 2e 34 33 32 34 2e 39  36 20 53 61 66 61 72 69   0.4324.9 6 Safari
```

```
HTTP Request HTTP-Version (http.request.version), 8 bytes     Packets: 223 · Displayed: 2 (0.9%)     Profile: Default
```

Fig-13: Wireshark Display after http://gaia.cs.umass.edu/wireshark-labs/ HTTP-wireshark-file1.htmlhas been retrieved by our browser

The example in above figure shows in the packet-listing window that two HTTP messages werecaptured: the GET message (from your browser to the gaia.cs.umass.edu web server) and theresponse message from the server to your browser. The packet-contents window shows details ofthe selected message (in this case the HTTP OK message, which is highlighted in the packet-listing window). Recall that since the HTTP message was carried inside a TCP segment, whichwas carried inside an IP datagram, which was carried within an Ethernet frame, Wiresharkdisplays the Frame, Ethernet, IP, and TCP packet information as well. We want to minimize the amount of non-HTTP data displayed (we're interested in HTTP here, and will be investigating these other protocols is later labs), so make sure the boxes at the far left of the Frame, Ethernet,IP and TCP information have a plus sign or a right-pointing triangle (which means there ishidden, non-displayed information), and the HTTP line has a minus sign or a down-pointing triangle (which means that all information about the HTTP message is displayed).

By looking at the information in the HTTP GET and response messages, answer the following questions. When answering the following questions, you should print out the GET and response message (see the introductory Wireshark lab for an explanation of how to do this) and indicate where in the message you've found the information that answers the following questions. When you hand in your assignment, annotate the output so that it's clear where in the output you're getting the information for your answer (e.g., for our classes, we ask that students markup paper copies with a pen, or annotate electronic copies with text in a colored font).

What are cookies and headers?

HTTP headers are used to pass additional information with HTTP response or HTTP requests. A cookie is an HTTP request header i.e. used in the requests sent by the user to the server

Cookies are strings of data that a web server sends to the browser. When a browser requests an object from the same domain in the future, the browser will send the same string of data back to the origin server.
The data is sent from the web server in the form of an HTTP header called "Set-Cookie". The browser sends the cookie back to the server in an HTTP header called "Cookie". Here is an example of what an HTTP cookie transaction might look like:
HTTP response from the web server:
[...]
Set-Cookie: first.lastname HTTP GET from the client:
[...]
Cookie: first.lastname
In the sample transaction, the web server told the client to create the cookie "first.lastname". Thenext time the client requests an object from this domain it sends the cookie with the request. Thisillustrates how a web server might be able to recall certain information such as user logins.

Types of Cookies

There are two different types of cookies - session cookies and persistent cookies. If a cookie doesnot contain an expiration date, it is considered a session cookie. Session cookies are stored in memory and never written to disk. When the browser closes, the cookie is permanently lost from this point on. If the cookie contains an expiration date, it is considered a persistent cookie. On the date specified in the expiration, the cookie will be removed from the disk.

There are several different fields a cookie can contain, separated by semicolons. The definitionsare:
expires
expires="Wdy, DD-Mon-YYYY HH:MM:SS GMT"Determines when the cookie is to be deleted.

Path

path=/
Determines what path to return the cookie on. In this example, the cookie will be sentwhen going to the root path in a domain.

Domain

domain=whatever.domain.com
Specifies what domain the cookie is used for. If this does not match the domain currently beingbrowsed to, it is considered to be a "3rd Party cookie" and will be rejected by the browser. Thisprevents one domain setting a cookie for a different domain.

Cookies are passed as HTTP headers, both in the request (client -> server), and in the response(server -> client)
Major groups of HTTP headers include General, Request, Response and Entity Headers.

Answer these questions

1. Is your browser running HTTP version 1.0 or 1.1? What version of HTTP is theserver running?
2. What languages (if any) does your browser indicate that it can accept to the server?
3. What is the IP address of your computer? Of the gaia.cs.umass.edu server?
4. What is the status code returned from the server to your browser?
5. When was the HTML file that you are retrieving last modified at the server?
6. How many bytes of content are being returned to your browser?
7. By inspecting the raw data in the packet content window, do you see any headers within the data that are not displayed in the packet-listing window? If so, name one.

In your answer to question 5 above, you might have been surprised to find that the document youjust retrieved was last modified within a minute before you downloaded the document. That’s because (for this file), the gaia.cs.umass.edu server is setting the file’s last-modified time to be the current time and is doing so once per minute. Thus, if you wait a minute between accesses, the file will appear to have been recently modified, and hence your browser will download a “new” copy of the document.

HTTP Authentication

Finally, let’s try visiting a web site that is password-protected and examine the sequence ofHTTP message exchanged for such a site. The URL http://gaia.cs.umass.edu/wireshark- labs/protected_pages/HTTP-wireshark-file5.html is password protected. The username is “Wireshark-students” (without the quotes), and the password is “network” (again, without the quotes). So, let’s access this “secure” password-protected site. Do the following:
Make sure your browser’s cache is cleared, as discussed above, and close your browser,then, start up your browser, then start up the Wireshark packet sniffer.
Enter the following URL into your browser http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wireshark-file5.html

Type the requested user name and password into the pop up box.
Stop Wireshark packet capture and enter “http” in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window.
Note: If you are unable to run Wireshark on a live network connection, you can use the http-ethereal-trace-5 packet trace to answer the questions below; see footnote 2. This trace file was gathered while performing the steps above on one of the author’s computers.)
Now let’s examine the Wireshark output. You might want to first read up on HTTP authentication by reviewing the easy-to-read material on “HTTP Access Authentication Framework” at http://frontier.userland.com/stories/storyReader$2159

Answer the following questions

1. What is the server’s response (status code and phrase) in response to the initial HTTPGET message from your browser?
2. When your browser’s sends the HTTP GET message for the second time, what new field isincluded in the HTTP GET message?

The username (wireshark-students) and password (network) that you entered are encoded inthe string of characters (d2lyZXNoYXJrLXN0dWRlbnRzOm5ldHdvcms=) following the
“Authorization: Basic” header in the client’s HTTP GET message. While it may appear that yourusername and password are encrypted, they are simply encoded in a format known as Base64 format. The username and password are not encrypted! To see this, go to http://www.motobit.com/util/base64-decoder-encoder.asp and enter the base64-encoded stringd2lyZXNoYXJrLXN0dWRlbnRz and decode. Voila! You have translated from Base64 encodingto ASCII encoding, and thus should see your username! To view the password, enter the remainder of the string Om5ldHdvcms= and press decode. Since anyone can download a tool like Wireshark and sniff packets (not just their own) passing by their network adaptor, and anyone can translate from Base64 to ASCII (you just did it!), it should be clear to you that simple passwords on WWW sites are not secure unless additional measures are taken.