Computer Networks

Wireshark: HTTP Packet Analysis

- Wireshark is an open-source packet analyzer, which is used for education, analysis, software development, communication protocol development, and network troubleshooting.
- It is used to track the packets so that each one is filtered to meet our specific needs. It is commonly called as a sniffer, network protocol analyzer, and network analyzer
- While running Wireshark the machines network interface card is put in **promiscuous mode**

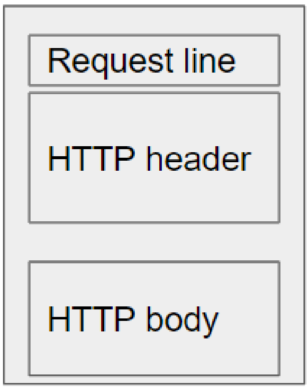
Uses of Wireshark

- t is used by network security engineers to examine security problems
- It allows the users to watch all the traffic being passed over the network.
- It is used by network engineers to troubleshoot network issues.
- It also helps to troubleshoot latency issues and malicious activities on your network.
- It can also analyze dropped packets.
- It has sort and filter options which makes ease to the user to view the data.
- It can also capture raw USB traffic.

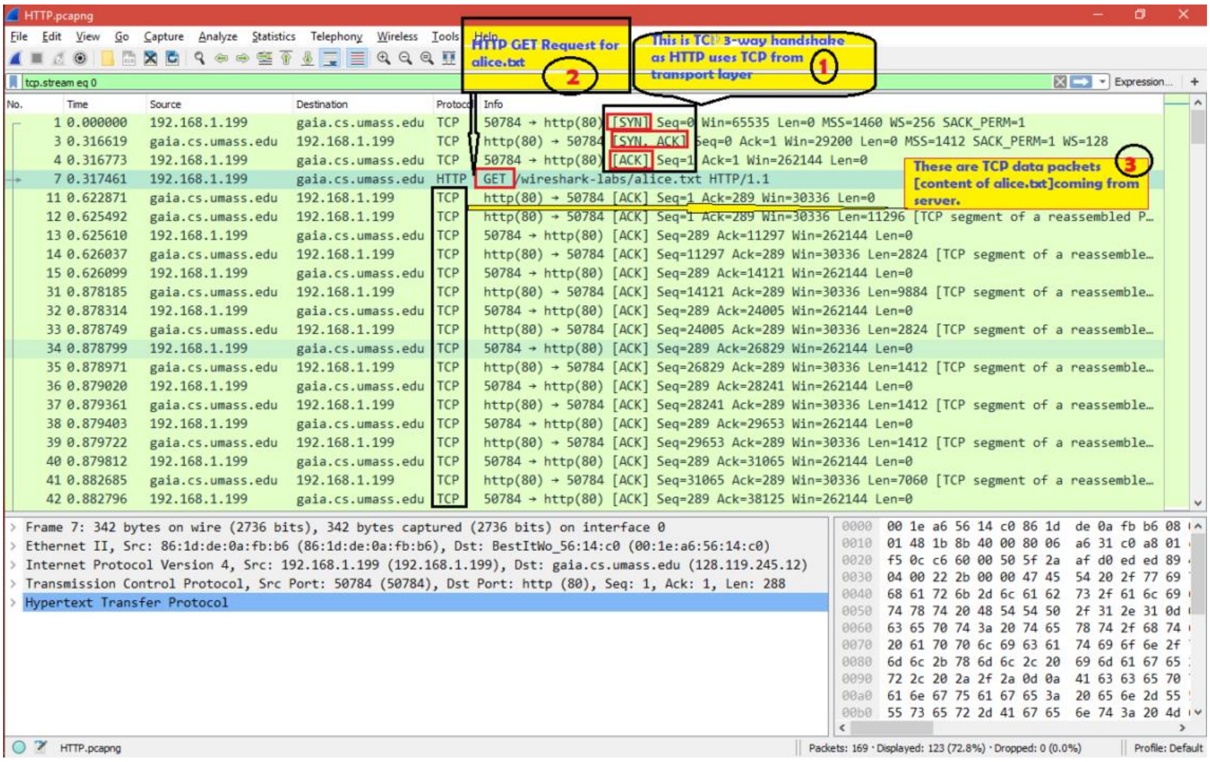
HTTP Packet Analysis

- **HTTP stands for** HyperText Transfer Protocol
- HTTP is an application layer protocol in ISO or TCP/IP model
- HTTP is used by the World Wide Web 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.
- HTTP uses several methods for communication for example GET, HEAD, POST, PUT, DELETE, CONNECT, OPTION and TRACE.
- HTTP uses port 80 and TCP as transport layer protocol

HTTP request



HTTP Packet Analysis on Wireshark


HTTP.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp.stream eq 0
Expression...

HTTP GET Request for alice.txt 2

This is TCP 3-way handshake as HTTP uses TCP from transport layer 1

These are TCP data packets [content of alice.txt]coming from server. 3

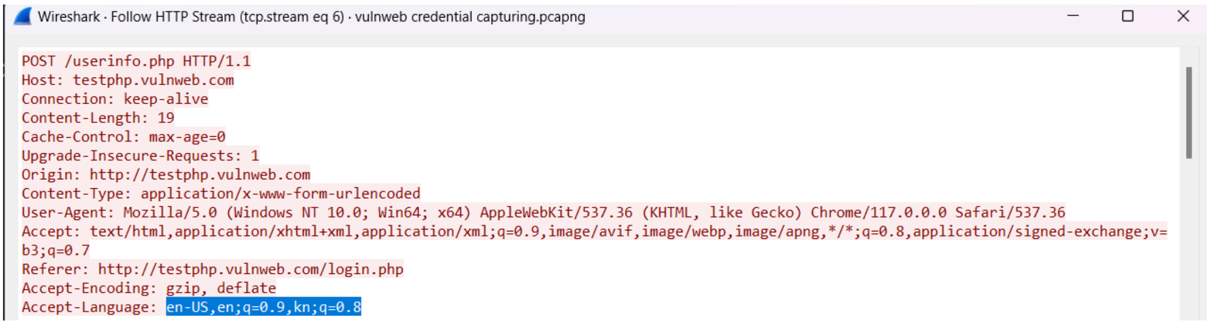
```
No.  Time      Source              Destination         Protocol Info
  1 0.000000   192.168.1.199       gaia.cs.umass.edu   TCP    50784 → http(80) [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
  3 0.316619   gaia.cs.umass.edu   192.168.1.199       TCP    http(80) → 50784 [SYN, ACK] Seq=0 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 reassemble…
 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 reassemble…
 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 reassemble…
 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 reassemble…
 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 reassemble…
 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 reassemble…
 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 reassemble…
 42 0.882796   192.168.1.199       gaia.cs.umass.edu   TCP    50784 → http(80) [ACK] Seq=289 Ack=38125 Win=262144 Len=0
```

```
> 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
```

```
0000  00 1e a6 56 14 c0 86 1d  de 0a fb b6 08
0010  01 48 1b 8b 40 00 80 06  a6 31 c0 a8 01
0020  f5 0c c6 60 00 50 5f 2a  af d0 ed ed 89
0030  04 00 22 2b 00 00 47 45  54 20 2f 77 69
0040  68 61 72 6b 2d 6c 61 62  73 2f 61 6c 69
0050  74 78 74 20 48 54 54 50  2f 31 2e 31 0d
0060  63 65 70 74 3a 20 74 65  78 74 2f 68 74
0070  20 61 70 70 6c 69 63 61  74 69 6f 6e 2f
0080  6d 6c 2b 78 6d 6c 2c 20  69 6d 61 67 65
0090  72 2c 20 2a 2f 2a 0d 0a  41 63 63 65 70
00a0  61 6e 67 75 61 67 65 3a  20 65 6e 2d 55
00b0  55 73 65 72 2d 41 67 65  6e 74 3a 20 4d
```

HTTP.pcapng | Packets: 169 · Displayed: 123 (72.8%) · Dropped: 0 (0.0%) | Profile: Default

- ➢ We can observe from the above analysis that http request is sent from 192.168.1.199 to gaia.cs.umass.edu
- ➢ It uses http GET method to access the resourse
- ➢ It uses TCP protocol for data transmission

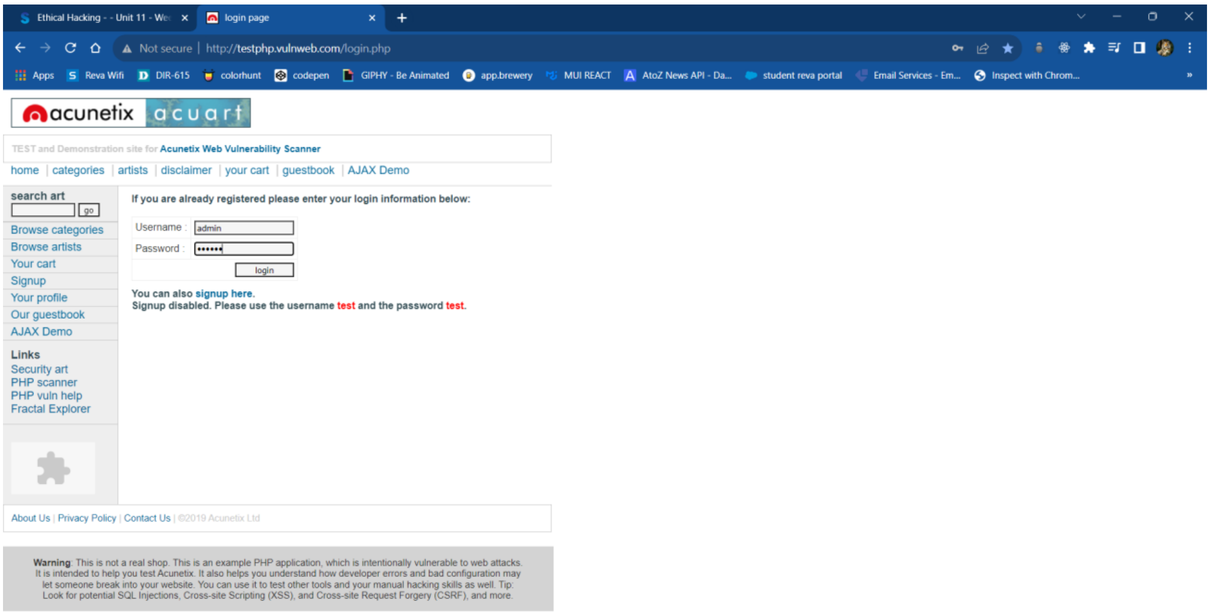

Wireshark · Follow HTTP Stream (tcp.stream eq 6) · vulnweb credential capturing.pcapng

```
POST /userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
Connection: keep-alive
Content-Length: 19
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://testphp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/117.0.0.0 Safari/537.36
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=0.7
Referer: http://testphp.vulnweb.com/login.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,kn;q=0.8
```

- http headers are used to pass additional information
- http header can include the information like user-agent, origin, host, connection, cache-control etc.
- the above image shows the header included within the packet in the wireshark tool

Experiments on HTTP using wireshark

Credential sniffing using wireshark and vulnweb:-



TEST and Demonstration site for Acunetix Web Vulnerability Scanner

home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo

search art
go

Browse categories
Browse artists
Your cart
Signup
Your profile
Our guestbook
AJAX Demo

Links
Security art
PHP scanner
PHP vuln help
Fractal Explorer

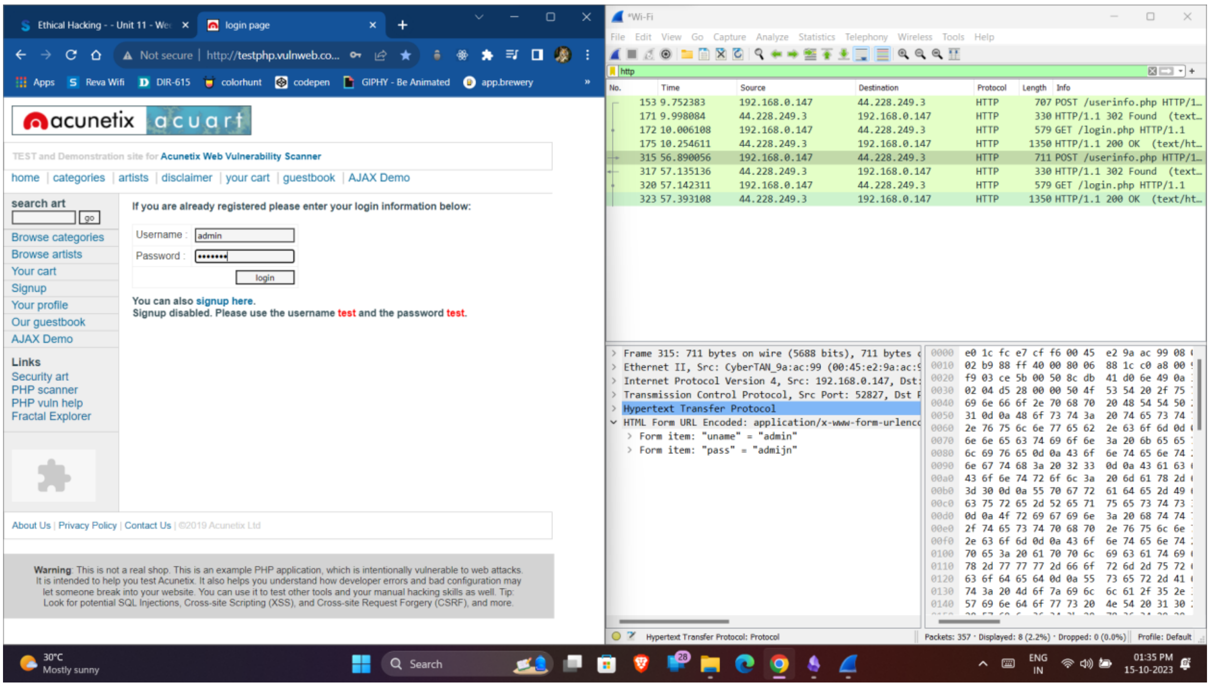
If you are already registered please enter your login information below:

Username : admin
Password : ••••••
login

You can also signup here.
Signup disabled. Please use the username test and the password test.

About Us | Privacy Policy | Contact Us | ©2019 Acunetix Ltd

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.



acunetix acuart

TEST and Demonstration site for Acunetix Web Vulnerability Scanner

home | categories | artists | disclaimer | your cart | guestbook | AJAX Demo

If you are already registered please enter your login information below:

Username : admin
Password : •••••••

login

You can also signup here.
Signup disabled. Please use the username test and the password test.

Warning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to web attacks. It is intended to help you test Acunetix. It also helps you understand how developer errors and bad configuration may let someone break into your website. You can use it to test other tools and your manual hacking skills as well. Tip: Look for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF), and more.

*Wi-Fi — filter: http

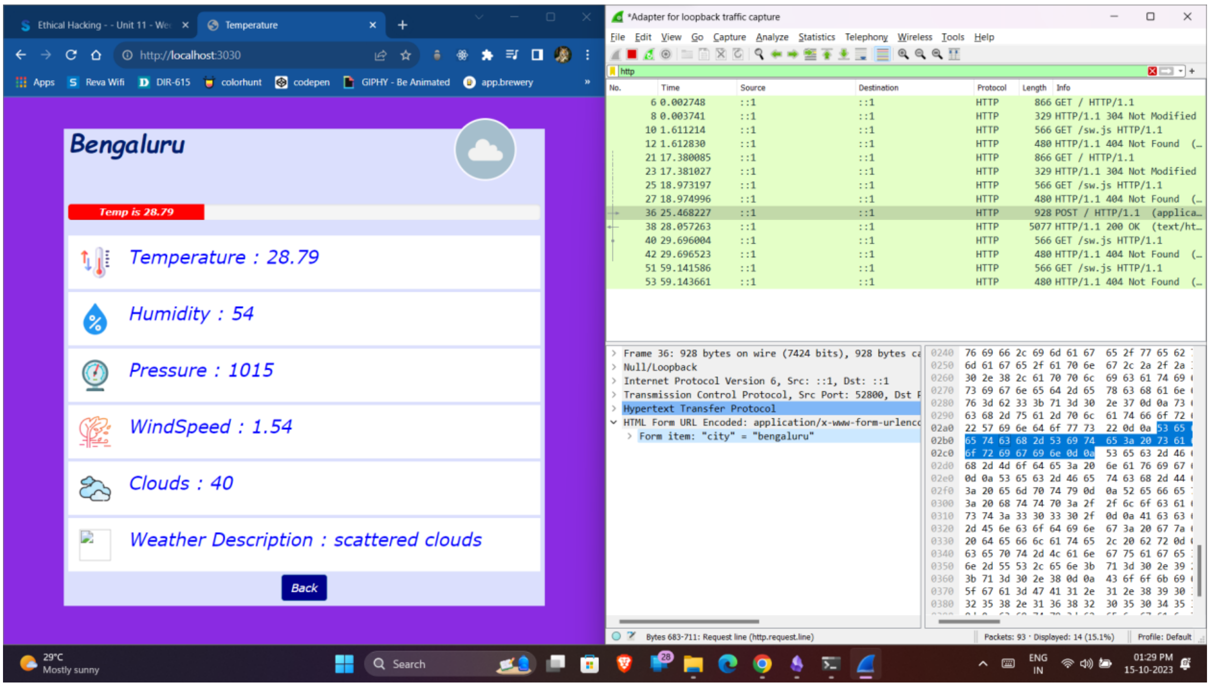
```
No.  Time       Source         Destination    Protocol Length Info
153  9.752383   192.168.0.147  44.228.249.3   HTTP     707    POST /userinfo.php HTTP/1…
171  9.998084   44.228.249.3   192.168.0.147  HTTP     330    HTTP/1.1 302 Found  (text…
172  10.006108  192.168.0.147  44.228.249.3   HTTP     579    GET /login.php HTTP/1.1
175  10.254611  44.228.249.3   192.168.0.147  HTTP     1350   HTTP/1.1 200 OK  (text/ht…
315  56.890056  192.168.0.147  44.228.249.3   HTTP     711    POST /userinfo.php HTTP/1…
317  57.135136  44.228.249.3   192.168.0.147  HTTP     330    HTTP/1.1 302 Found  (text…
320  57.142311  192.168.0.147  44.228.249.3   HTTP     579    GET /login.php HTTP/1.1
323  57.393108  44.228.249.3   192.168.0.147  HTTP     1350   HTTP/1.1 200 OK  (text/ht…
```

```
> Frame 315: 711 bytes on wire (5688 bits), 711 bytes c
> Ethernet II, Src: CyberTAN_9a:ac:99 (00:45:e2:9a:ac:9
> Internet Protocol Version 4, Src: 192.168.0.147, Dst:
> Transmission Control Protocol, Src Port: 52827, Dst P
> Hypertext Transfer Protocol
v HTML Form URL Encoded: application/x-www-form-urlenco
   > Form item: "uname" = "admin"
   > Form item: "pass" = "admijn"
```

Packets: 357 · Displayed: 8 (2.2%) · Dropped: 0 (0.0%)

- ➢ vulnweb is website designed with vulnerabilities for learning purpose
- ➢ it uses http for login
- ➢ using wireshark, we can capture the packets of vulnweb website
- ➢ we can look into http post method because login uses post method for sending data to the server
- ➢ we can observe that under HTML form URL encode we can clearly see the username and password because its unencrypted
- ➢ but most of the applications today uses encryption for data transmission

Capturing weather app data using wireshark



Bengaluru

Temp is 28.79

Temperature : 28.79

Humidity : 54

Pressure : 1015

WindSpeed : 1.54

Clouds : 40

Weather Description : scattered clouds

Back

*Adapter for loopback traffic capture — filter: http

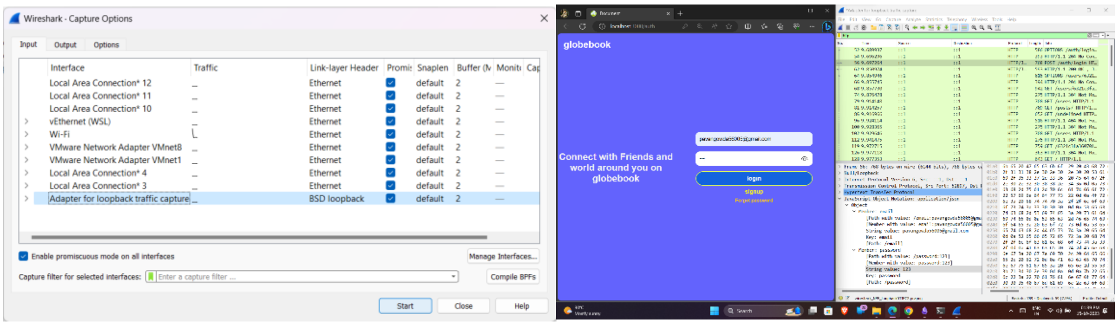
```
No.  Time       Source  Destination  Protocol Length Info
6    0.002748   ::1     ::1          HTTP     866    GET / HTTP/1.1
8    0.003741   ::1     ::1          HTTP     329    HTTP/1.1 304 Not Modified
10   1.611214   ::1     ::1          HTTP     566    GET /sw.js HTTP/1.1
12   1.612830   ::1     ::1          HTTP     480    HTTP/1.1 404 Not Found  (…
21   17.380085  ::1     ::1          HTTP     866    GET / HTTP/1.1
23   17.381027  ::1     ::1          HTTP     329    HTTP/1.1 304 Not Modified
25   18.973197  ::1     ::1          HTTP     566    GET /sw.js HTTP/1.1
27   18.974996  ::1     ::1          HTTP     480    HTTP/1.1 404 Not Found  (…
36   25.468227  ::1     ::1          HTTP     928    POST / HTTP/1.1  (applica…
38   28.057263  ::1     ::1          HTTP     5077   HTTP/1.1 200 OK  (text/ht…
40   29.696004  ::1     ::1          HTTP     566    GET /sw.js HTTP/1.1
42   29.696523  ::1     ::1          HTTP     480    HTTP/1.1 404 Not Found  (…
51   59.141586  ::1     ::1          HTTP     566    GET /sw.js HTTP/1.1
53   59.143661  ::1     ::1          HTTP     480    HTTP/1.1 404 Not Found  (…
```

```
> Frame 36: 928 bytes on wire (7424 bits), 928 bytes ca
> Null/Loopback
> Internet Protocol Version 6, Src: ::1, Dst: ::1
> Transmission Control Protocol, Src Port: 52800, Dst P
> Hypertext Transfer Protocol
v HTML Form URL Encoded: application/x-www-form-urlenco
   > Form item: "city" = "bengaluru"
```

Bytes 683-711: Request line (http.request.line)

Packets: 93 · Displayed: 14 (15.1%)

Capturing credentials of the localhost app



Wireshark - Capture Options

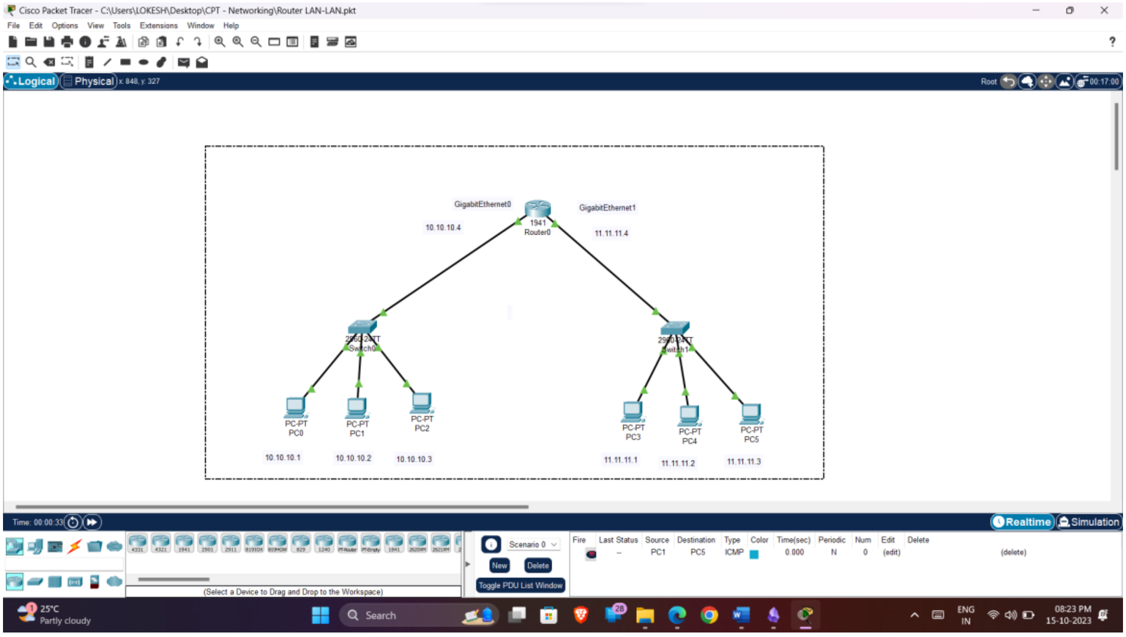
	Interface	Traffic	Link-layer Header	Promis	Snaplen	Buffer (M	Monito	Cap
	Local Area Connection* 12		Ethernet	☑	default	2	—	
	Local Area Connection* 11		Ethernet	☑	default	2	—	
	Local Area Connection* 10		Ethernet	☑	default	2	—	
>	vEthernet (WSL)		Ethernet	☑	default	2	—	
>	Wi-Fi		Ethernet	☑	default	2	—	
>	VMware Network Adapter VMnet8		Ethernet	☑	default	2	—	
>	VMware Network Adapter VMnet1		Ethernet	☑	default	2	—	
>	Local Area Connection* 4		Ethernet	☑	default	2	—	
>	Local Area Connection* 3		Ethernet	☑	default	2	—	
>	Adapter for loopback traffic capture		BSD loopback	☑	default	2	—	

☑ Enable promiscuous mode on all interfaces

Capture filter for selected interfaces: Enter a capture filter ...

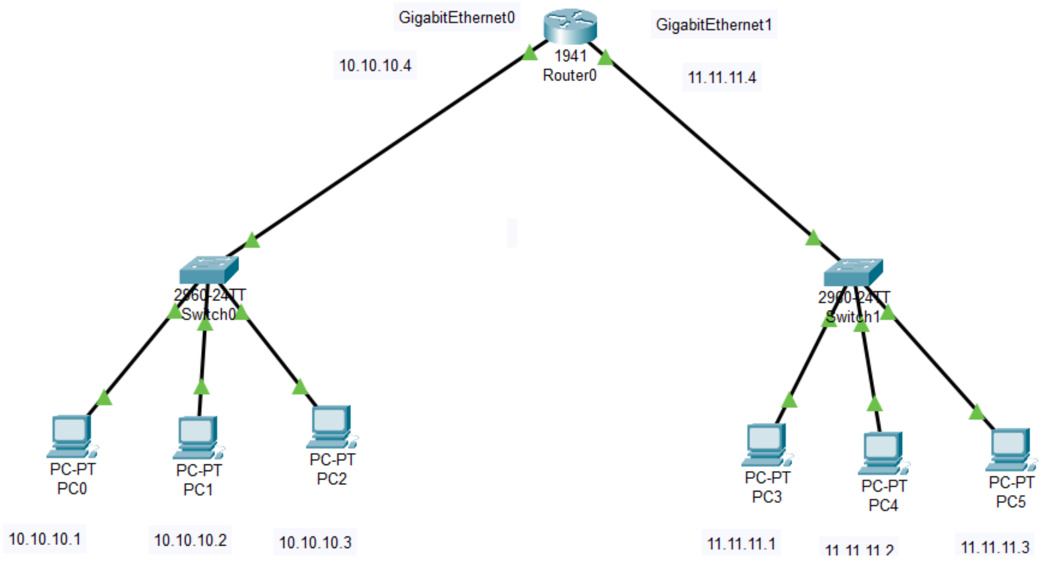
- ➢ The above application are built by own using nodejs and react library and they are running in the localhost environment
- ➢ First we need to start capturing the packets of the localhost environment by setting the interface as loopback as shown in the above figure
- ➢ Both applications uses http protocol for data transmission
- ➢ So we have captured both weather data and login credentials of the globebook app
- ➢ Weather app transfer data as form-url-encoded format
- ➢ Where as globebook sends the data in the form of JSON object to the server

Packet Tracer: Perform an Initial Switch and Router Configuration

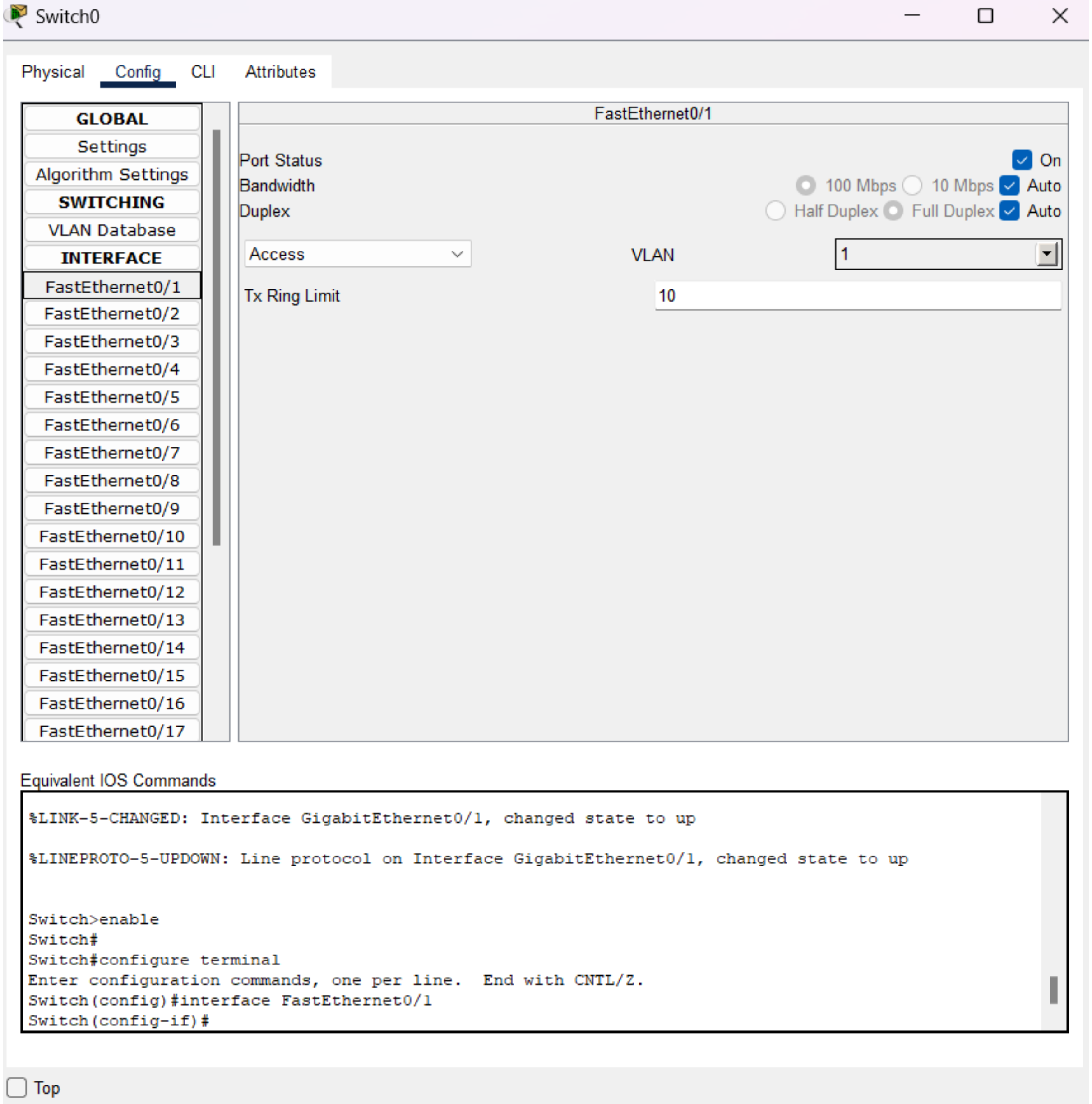


Cisco Packet Tracer - C:\Users\LOKESH\Desktop\CPT - Networking\Router LAN-LAN.pkt

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
●	--	PC1	PC5	ICMP	🟦	0.000	N	0	(edit)	(delete)



- First we need to install cisco packet tracer
- Once we installed packet tracer we have to login to access the resources
- We will trag and trop the components like computer, switch and router
- Once the components are placed in the project
- We have to connect them using Ethernet Straight-Through cable as we are connecting different devices
- Once the devices or nodes connected physically through a cable
- We need to connected them physically by mentioning some set of protocol rules
- First assign IP for each computers manually or even we can use DHCP(Dynamic host configuration procol) to do the same
- For making our network simple we are setting IP address manually
- We can now connect multiple computer using switch as a middleware
- Switch has a memory in it so it can directly communicate with the specific devices it doesn’t broadcast packet to all devices unlike Hub
- Switches configuration window are



FastEthernet0/1

Port Status ☑ On

Bandwidth ○ 100 Mbps ○ 10 Mbps ☑ Auto

Duplex ○ Half Duplex ○ Full Duplex ☑ Auto

Access VLAN 1

Tx Ring Limit 10

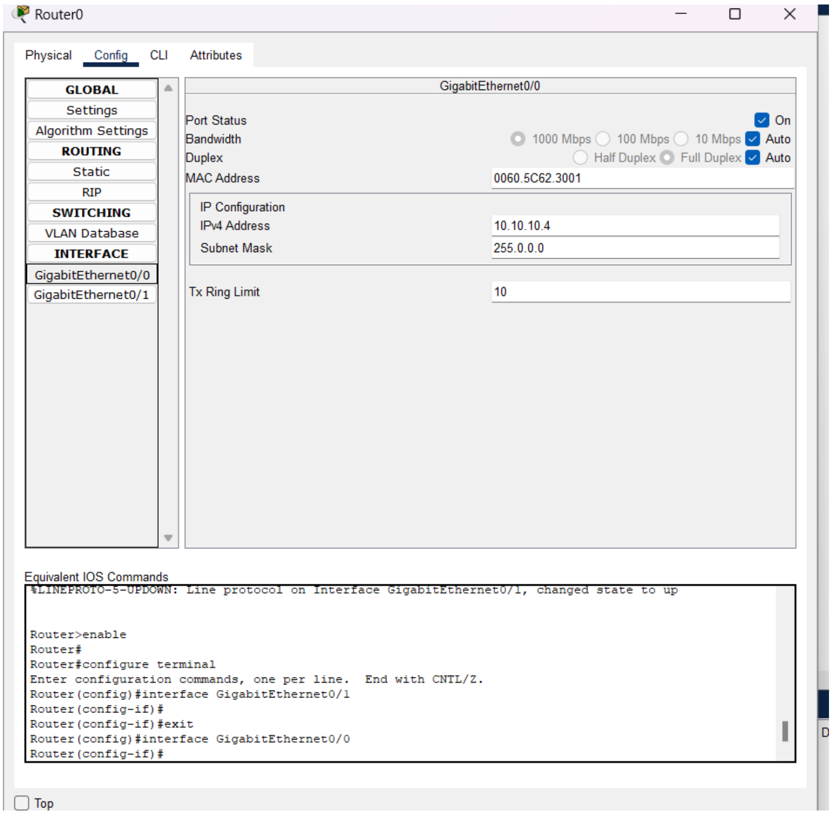
Equivalent IOS Commands

```
%LINK-5-CHANGED: Interface GigabitEthernet0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state to up


Switch>enable
Switch#
Switch#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Switch(config)#interface FastEthernet0/1
Switch(config-if)#
```

- In our network we have to LAN networks with base IP 10.0.0.0 and 11.11.11.0
- To connect different LAN we must use **router**
- Router establishes the connection using gateway and IP
- The router configurations are mentioned below



GigabitEthernet0/0

Port Status ☑ On

Bandwidth ● 1000 Mbps ○ 100 Mbps ○ 10 Mbps ☑ Auto

Duplex ○ Half Duplex ● Full Duplex ☑ Auto

MAC Address 0060.5C62.3001

IP Configuration

IPv4 Address 10.10.10.4

Subnet Mask 255.0.0.0

Tx Ring Limit 10

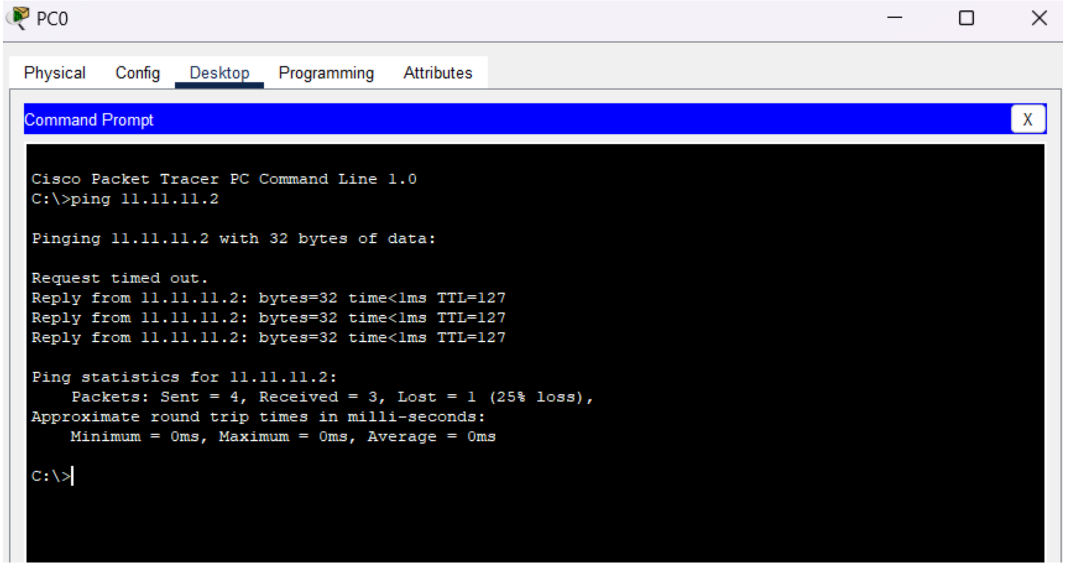
Equivalent IOS Commands

```
%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/1, changed state to up


Router>enable
Router#
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface GigabitEthernet0/1
Router(config-if)#
Router(config-if)#exit
Router(config)#interface GigabitEthernet0/0
Router(config-if)#
```

☐ Top

- ➢ Once the connection is established properly we can check whether to devices are communicating with each other or not using ping command



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 11.11.11.2

Pinging 11.11.11.2 with 32 bytes of data:

Request timed out.
Reply from 11.11.11.2: bytes=32 time<1ms TTL=127
Reply from 11.11.11.2: bytes=32 time<1ms TTL=127
Reply from 11.11.11.2: bytes=32 time<1ms TTL=127

Ping statistics for 11.11.11.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```