**Name :** Rajvardhan Reddy
**Reg No:** 180905093
**Sec :** B
**Roll No :** 19

# CN Lab – 3 : Packet Analysis with Wireshark

**P1)** Retrieve web pages using HTTP. Use Wireshark to capture packets for analysis. Learn about most common HTTP messages . Also capture response messages and analyze them. During the lab session, also examine and analyze some HTTP headers.

**HYPERTEXT TRANSFER PROTOCOL (HTTP) :** is an application layer protocol invented by CERN in the late 1990s. It sends data over the secure TCP channel and uses an RDT mechanism. In HTTP, the data is shared in plain text format and hence can't be relied on to transfer confidential information such as passwords and credit card numbers.

Although the HTTP is stateless, it can't maintain a state, and you can no way ensure that two similar requests are delivered through the same connection. Hence it is unfit for e-commerce websites. But the HTTP protocol has sessions, which saves cookies and cache in the client's device and make it accessible to them in the state which is left. HTTP connection works on port number 80

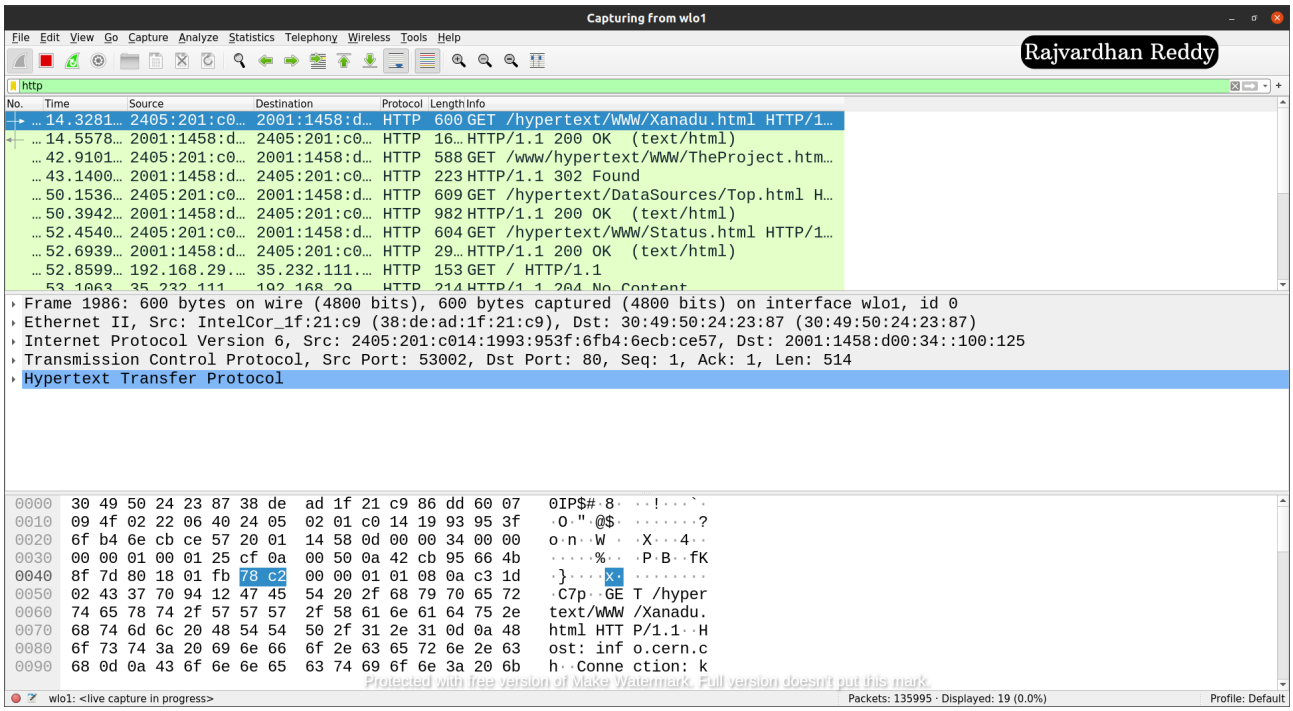
**Understanding :**

HTTP follows a classical client-server model, with a client opening a connection to make a request, then waiting until it receives a response. HTTP is a stateless protocol, meaning that the server does not keep any data (state) between two requests. Though often based on a TCP/IP layer, it can be used on any reliable transport layer, that is, a protocol that doesn't lose messages silently like UDP does.

**HTTP messages** are how data is exchanged between a server and a client. There are two types of messages: **requests** sent by the client to trigger an action on the server, and **responses**, the answer from the server.

**HTTP Headers**
1. Request headers contain more information about the resource to be fetched, or about the client requesting the resource.
2. Response headers hold additional information about the response, like its location or about the server providing it.
3. Representation headers contain information about the body of the resource, like its MIME type, or encoding/compression applied.
4. Payload headers contain representation-independent information about payload data, including content length and the encoding used for transport.

## Capturing HTTP Packets through WireShark :




**P2)** Use FTP to transfer some files, Use Wireshark to capture some packets. Show that FTP uses two separate connections: a control connection and a data-transfer connection. The data connection is opened and closed for each file transfer activity. Also show that FTP is an insecure file transfer protocol because the transaction is done in plaintext.

**FILE TRANSFER PROTOCOL(FTP) :** is an application layer protocol that uses TCP for the transport layer similar to HTTP protocol. This is used for transferring files. Two TCP connections are used in parallel and are reliable for sharing confidential information, as port number 21 can be used to establish a controlled connection. The file-sharing usually takes place on port number 20.

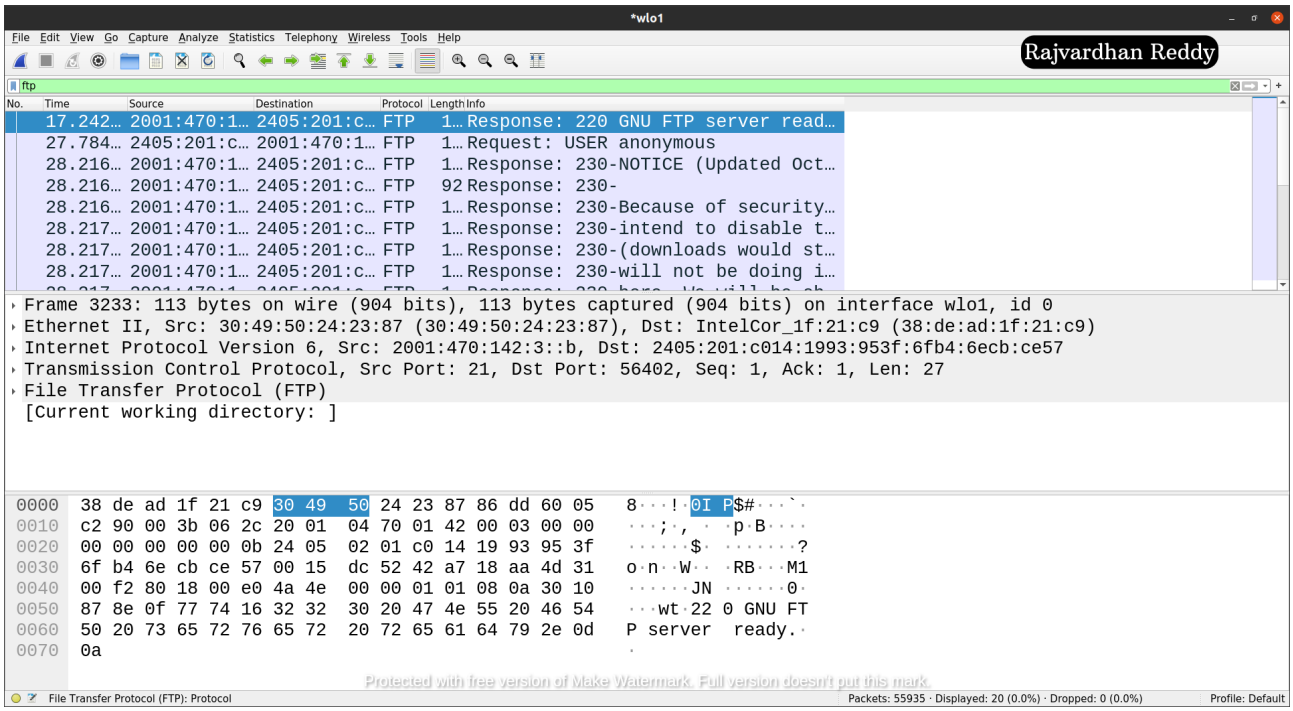
When an FTP session is started between a client and a server, the client initiates a control TCP connection with the server-side. The client sends control information over this. When the server receives this, it creates a data connection to the client-side. Only one file can be sent over one data connection. But the control connection remains active throughout the user session. As we know, HTTP is stateless, i.e. it does not have to keep track of any user state. But FTP needs to maintain a state about its user throughout the session.

## Establishing a FTP :

```
rajvardhan@rajvardhan-HP-Pavilion-Laptop-15-cc1xx:~$ ftp ftp.gnu.org
Connected to ftp.gnu.org.
220 GNU FTP server ready.
Name (ftp.gnu.org:rajvardhan): anonymous
230-NOTICE (Updated October 13 2017):
230-
230-Because of security concerns with plaintext protocols, we still
230-intend to disable the FTP protocol for downloads on this server
230-(downloads would still be available over HTTP and HTTPS), but we
230-will not be doing it on November 1, 2017, as previously announced
230-here. We will be sharing our reasons and offering a chance to
230-comment on this issue soon; watch this space for details.
230-
230-If you maintain scripts used to access ftp.gnu.org over FTP,
230-we strongly encourage you to change them to use HTTPS instead.
230-
230----
230-
230-Due to U.S. Export Regulations, all cryptographic software on this
230-site is subject to the following legal notice:
230-
230-    This site includes publicly available encryption source code
230-    which, together with object code resulting from the compiling of
230-    publicly available source code, may be exported from the United
230-    States under License Exception "TSU" pursuant to 15 C.F.R. Section
230-    740.13(e).
230-
230-This legal notice applies to cryptographic software only. Please see
230-the Bureau of Industry and Security (www.bxa.doc.gov) for more
230-information about current U.S. regulations.
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
```

```
ftp> ls
200 EPRT command successful. Consider using EPSV.
150 Here comes the directory listing.
lrwxrwxrwx    1 0        0               8 Aug 20  2004 CRYPTO.README -> .messag
e
-rw-r--r--    1 0        0           17864 Oct 23  2003 MISSING-FILES
-rw-r--r--    1 0        0            4178 Aug 13  2003 MISSING-FILES.README
-rw-r--r--    1 0        0            2991 Oct 03  2019 README
-rw-r--r--    1 0        0          405121 Oct 23  2003 before-2003-08-01.md5sum
s.asc
-rw-rw-r--    1 0        3003       270707 Jun 24 17:49 find.txt.gz
drwxrwxr-x  323 0        3003        12288 Jun 21 22:00 gnu
drwxrwxr-x    3 0        3003         4096 Mar 10  2011 gnu+linux-distros
-rw-rw-r--    1 0        3003       499493 Jun 24 17:49 ls-lrRt.txt.gz
drwxr-xr-x    3 0        0            4096 Apr 20  2005 mirrors
lrwxrwxrwx    1 0        0              11 Apr 15  2004 non-gnu -> gnu/non-gnu
drwxr-xr-x   92 0        0            4096 Nov 17  2020 old-gnu
lrwxrwxrwx    1 0        0               1 Aug 05  2003 pub -> .
drwxr-xr-x    2 0        0            4096 Nov 08  2007 savannah
drwxr-xr-x    2 0        0            4096 Aug 02  2003 third-party
drwxr-xr-x    2 0        0            4096 Apr 07  2009 tmp
-rw-rw-r--    1 0        3003       591270 Jun 24 17:49 tree.json.gz
drwxr-xr-x    2 0        0            4096 May 07  2013 video
-rw-r--r--    1 0        0            2830 Dec 18  2018 welcome.msg
226 Directory send OK.
ftp>
```

## Capturing FTP Packets through WireShark :




**P3)** Analyze the behavior of the DNS protocol. In addition to Wireshark [Several network utilities are available for finding some information stored in the DNS servers. Eg.dig utilities (which has replaced nslookup). Set Wireshark to capture the packets sent by this utility.]

**DOMAIN NAME SYSTEM(DNS) :** is a hostname to IP address translation service. DNS is a distributed database implemented in a hierarchy of name servers. It is an application layer protocol for message exchange between clients and servers.

The IP address identifies every host but remembering numbers is very difficult for people. The IP addresses are not static; therefore, a mapping must change the domain name to IP address. So DNS is used to convert the domain name of the websites to their numerical IP address.

It is tough to find out the IP address associated with a website because there are millions of websites, and with all those websites, we should be able to generate the IP address immediately, there should not be a lot of delay for that to happen organisation of the database is very important.

**DNS record :** Domain name, IP address, validity, the time to live, and all the information related to that domain name are saved in these records. These records are stored in a tree-like structure.

## Types of DNS Queries:

**1. Recursive query** - In a recursive query, a DNS client requires that a DNS server (typically a DNS recursive resolver) will respond to the client with either the requested resource record or an error message if the resolver can't find the record.

**2. Iterative query** - in this situation the DNS client will allow a DNS server to return the best answer it can. If the queried DNS server does not have a match for the query name, it will return a referral to a DNS server authoritative for a lower level of the domain namespace. The DNS client will then make a query to the referral address. This process continues with additional DNS servers down the query chain until either an error or timeout occurs.

**3. Non-recursive query** - typically this will occur when a DNS resolver client queries a DNS server for a record that it has access to either because it's authoritative for the record or the record exists inside of its cache. Typically, a DNS server will cache DNS records to prevent additional bandwidth consumption and load on upstream servers.

## Capturing DNS Packets through WireShark :