Practical Computing

(UCS311)

Evaluation Assignment 2

Submitted By-

Name: Ananya Agarwal

Roll No: 102083036

Batch: 2CO14

Q1. Give technical commentary on each of the underlined text, be precise while you answer. We need crisp replies.

```
C:\Users\msing>nslookup
Default Server:  UnKnown
Address:  192.168.18.1

> server 8.8.8.8
Default Server:  dns.google
Address:  8.8.8.8

> set q=MX
> thapar.edu
Server:  dns.google
Address:  8.8.8.8

Non-authoritative answer:
thapar.edu      MX preference = 1, mail exchanger = aspmx.l.google.com
thapar.edu      MX preference = 5, mail exchanger = alt1.aspmx.l.google.com
thapar.edu      MX preference = 10, mail exchanger = alt4.aspmx.l.google.com
thapar.edu      MX preference = 5, mail exchanger = alt2.aspmx.l.google.com
thapar.edu      MX preference = 10, mail exchanger = alt3.aspmx.l.google.com
> set q=NS
> thapar.edu
Server:  dns.google
Address:  8.8.8.8

Non-authoritative answer:
thapar.edu      nameserver = dns2.easydns.net
thapar.edu      nameserver = dns1.easydns.com
thapar.edu      nameserver = dns3.easydns.ca
```

Sol 1:

- **nslookup**
  - It stands for **name server lookup.** It's main use is for **troubleshooting DNS** related problems. It is a network administration command-line tool available in many computer operating systems for querying the Domain Name System (DNS) to obtain domain name or IP address mapping or display DNS records such as the IP address of a system or the MX records of the domain.
  - It operates in 2 modes: **interactive or non-interactive mode**. In interactive mode, by invoking it **without arguments as in the given question**, the user issues parameter configurations or requests when presented with the **nslookup prompt (>).**
  - In non-interactive mode, i.e. when the first argument is a name or Internet address of the host being searched, parameters and the query are specified as command line arguments in the invocation of the program.
- **192.168.18.1**

  This is the **address of the default server** (UnKnown) since the server name is not mentioned explicitly.

- **server 8.8.8.8**

  We can **switch server** by typing server command. This means that instead of the default server, the server with address 8.8.8.8 is used.
- **dns.google**

  This means that **this time** the information is coming from the **google server** and not the default server**. dns.google** is the **DNS name** of that server.
- **Address: 8.8.8.8**

  This is the **google server's (dns.google) IP address**.

- **set q=MX**
    - set q=x specifies the type of records to be displayed, such as A, CNAME, MX, NS, PTR or SOA. This sets a **filter to only collect MX records** and related information from the DNS servers. If the MX record is not displayed, DNS is not configured properly.
    - When the internet service providers are failed, then these **mail exchangers** can be used as a backup.
- **Non-authoritative answer:**

  This means that the local DNS server was **unable** to **answer the query itself,** and instead had to contact one or more other name server address of the default DNS server and thus, this **DNS server** is **not directly responsible** for this domain name and this name is known to this server because it had previously resolved that name and has taken this information from it's cache record.
- **MX preference = 1 and MX preference = 5**
    - The "**MX preference**" specifies which **mail server** to use and in which order. The lower the number, the more preferred the mail server is.
    - If the preference for each mail server is same, you can use any of the given **"mail exchangers".**
    - **The lower the preference, the higher the priority**. Thus, the mail exchanger with the least value is going to get all the mails.
- **mail exchanger = aspmx.1.google.com and mail exchanger = alt1.aspmx.1.google.com**

  A **mail exchange record (MX record)** is a resource record or settings within the Domain Name System (DNS) that **redirects email to a specified mail server** which accepts email on behalf of a domain or users.

  Here, when the mail exchanger with the least value i.e. 1 gets failed, the mail exchanger with the second smallest value i.e. 5 is contacted. Thus, if the mail exchanger **aspmx.1.google.com gets failed** then**, alt1.aspmx.1.google.com** will be contacted**.**
- **set q= NS**

  set q=x specifies the type of records to be displayed, such as A, CNAME, MX, NS, PTR or SOA. This displays **DNS server responsible** for a **particular domain name.**

- **dns2.easydns.net**

  This is the **server name** which will be used instead of the google DNS server, in case it fails.

Q2. Look at the snippet given and answer

i) How this program would have been compiled, give command syntax?

ii) Certain portions in the snippet are highlighted, kindly comment on each one of these.

```
(gdb) b 15
Breakpoint 1 at 0x804845f: file mst.c, line 15.
(gdb) r
Starting program: /home/seed/mst

Breakpoint 1, main (argc=1, argv=0xbffff434, envp=0xbffff43c) at mst.c:16
16          foo(passme1,passme2);
(gdb) info frame
Stack level 0, frame at 0xbffff3a0:
 eip = 0x804845f in main (mst.c:16); saved eip 0xb7e384d3
 source language c.
 Arglist at 0xbffff398, args: argc=1, argv=0xbffff434, envp=0xbffff43c
 Locals at 0xbffff398, Previous frame's sp is 0xbffff3a0
 Saved registers:
  ebp at 0xbffff398, eip at 0xbffff39c
```

Sol 2:

i)

- A ‘C’ program can be compiled by using **gcc compiler**
- Option **–g** is used to **invoke the debugger** so that we can debug the given program**.**
- Option –**o** is used to **rename** the name of the executable file.

Syntax (for C program):

gcc –g filename.c –o name_of_executable_file

To compile the given program, use the command:

gcc -g mst.c –o mst

ii)

- **b 15**

  It inserts **breakpoint** at line number **15**. It helps to pause the program during execution when it starts to execute the function and helps to **debug** the program at that point. Multiple breakpoints can be inserted by executing the command wherever necessary. **b 15** command makes **the mst executable file pause** when the debugger starts to execute the main function.

- **0x804845f:**
  Address of the line number 15.
- **file mst.c, line 15**
  The **name of the file** is mst.c and the **breakpoin**t is initiated on line number **15**.
- **r**
  To stop the breakpoint and **run** the program i.e. r command runs the current executable file.
  The message that gets prompted on running (**gdb) r** is:
  The program being debugged has been started already.
  Start it from the beginning? (y or n)
- **argc=1**
  argc (**arg**ument **c**ount) stores the **number of the command line arguments** passed to the main function.
- **argv=0xbffff434**
  argv (argument **v**ector) is a vector of C strings i.e. it stores one-dimensional array of strings. So, **the passed arguments** will get **stored** in the **array argv** at the **base address 0xbffff434.**
- **envp=0xbffff43c**
  envp gives the program's environment. The *argv* mechanism is typically used to **pass command-line arguments** specific to the particular program being invoked. The environment, on the other hand, keeps track of information that is shared by many programs which here is **stored at the base address 0xbffff43c.**
- **info frame**
  - It displays **low-level** verbose description of the **selected stack frame including:**
    - The **address of the frame**
    - The **address of the next frame down** (called by this frame)
    - The address of the next frame up (caller of this frame)
    - The language in which the source code corresponding to this frame is written
    - The address of the frame's arguments
    - The address of the frame's local variables
    - The program counter saved in it (the address of execution in the caller frame)
    - Which registers were saved in the frame
- eip **= 0x804845f**
  eip or **instruction pointer register** stores **address for next instruction** to execute (also called **program counter**). So, at this moment, the next to execute is at "0x804845f", which is line **16** of the program **mst.c**.
- saved **eip 0xb7e384d3**
  saved eip "0xb7e384d3" is so called "return address", i.e. **the instruction to resume in caller stack frame after returning from this caller stack**. It is pushed into stack upon "CALL" instruction (save it for return).

- **Locals at 0xbffff398**

  It displays the **address of local variables.**

- **Previous frame's sp is 0xbffff3a0**

  This is where the **previous frame's stack pointer points to** (the caller frame), at the moment of calling.

- **ebp at 0xbffff398**

  ebp at 0xbffff398 that is the **address** where the ebp register of the caller´s stack frame is saved. This register is usually considered as **the starting address of the locals of this stack frame**. In another words, the operations of all local variables use ebp.

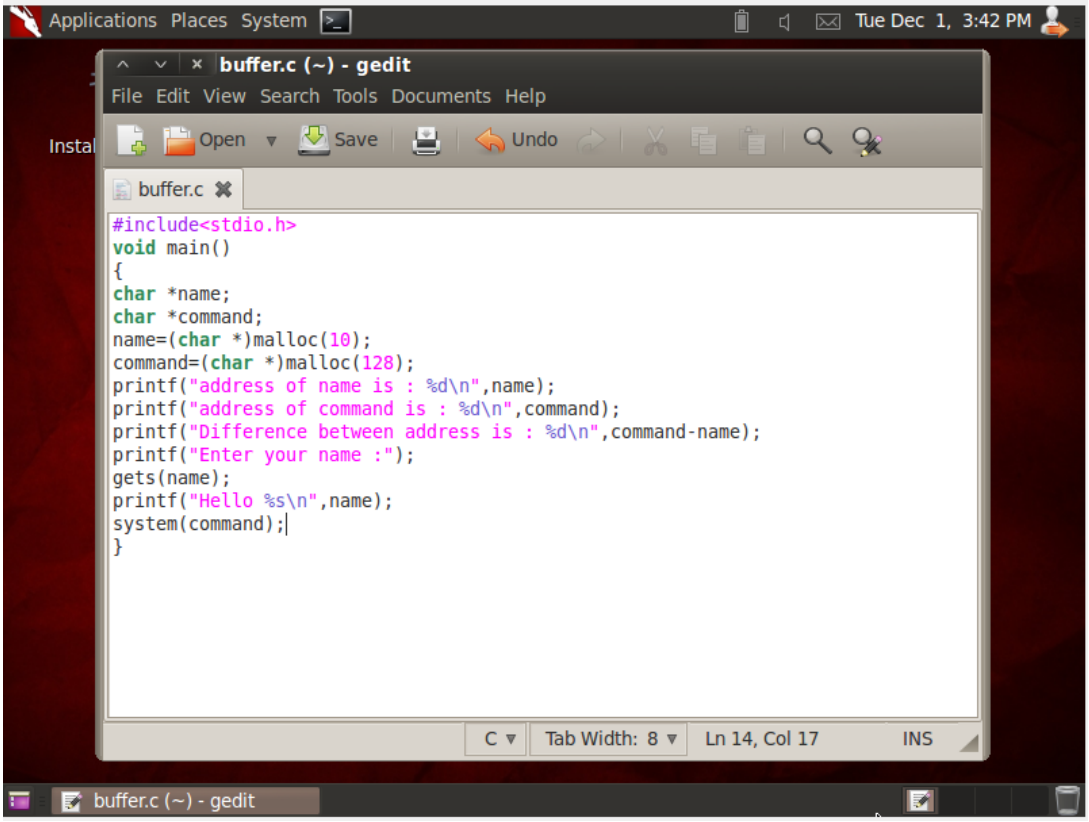
- **eip at 0xbffff39c**

  0xbffff39c will **overwrite the value of saved eip (0xb7e384d3)** so that when the function returns execution, it will continue from the value we stored there**.**

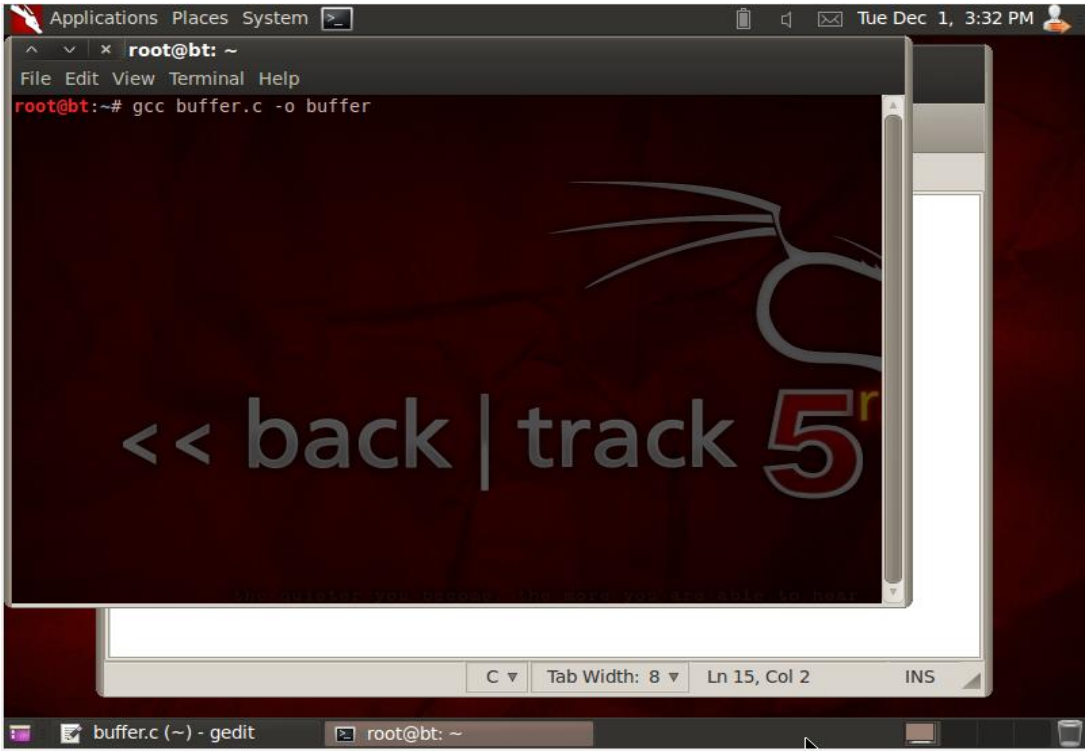
Q3. Write a program to perform buffer overflow attack.

<u>Sol 3:</u>

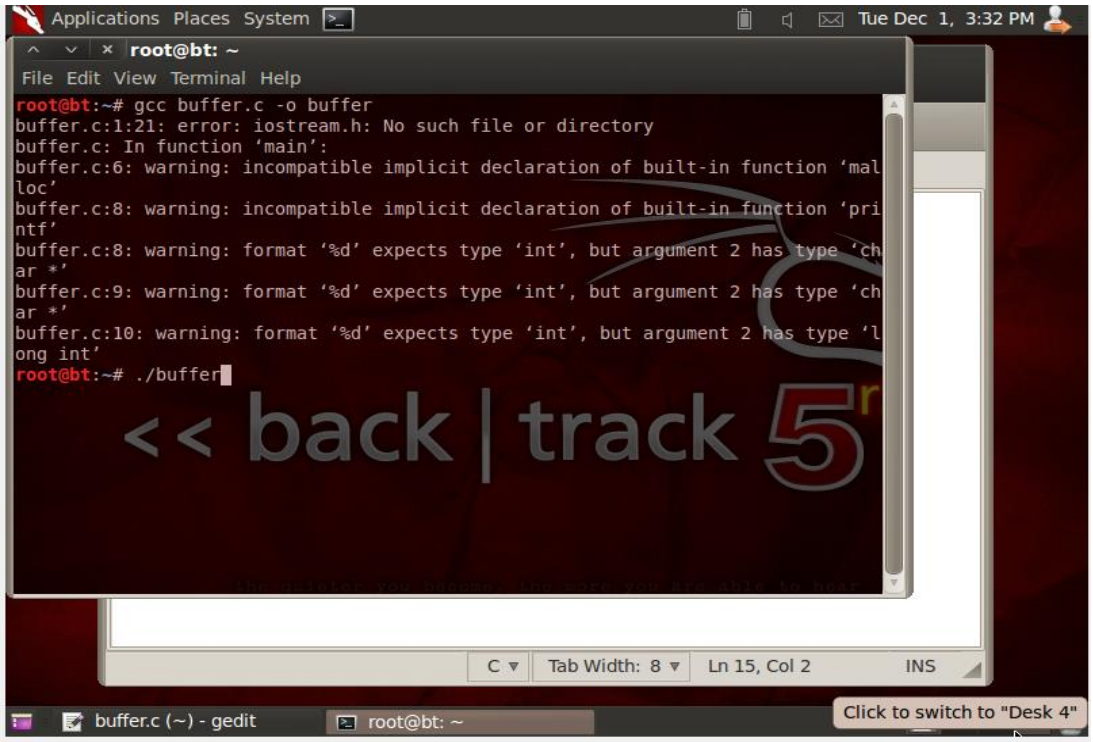
- Open the BackTrack Applications menu and then select Accessories->gedit text editor
- Type the following code to perform buffer overflow attack and then Save the code.



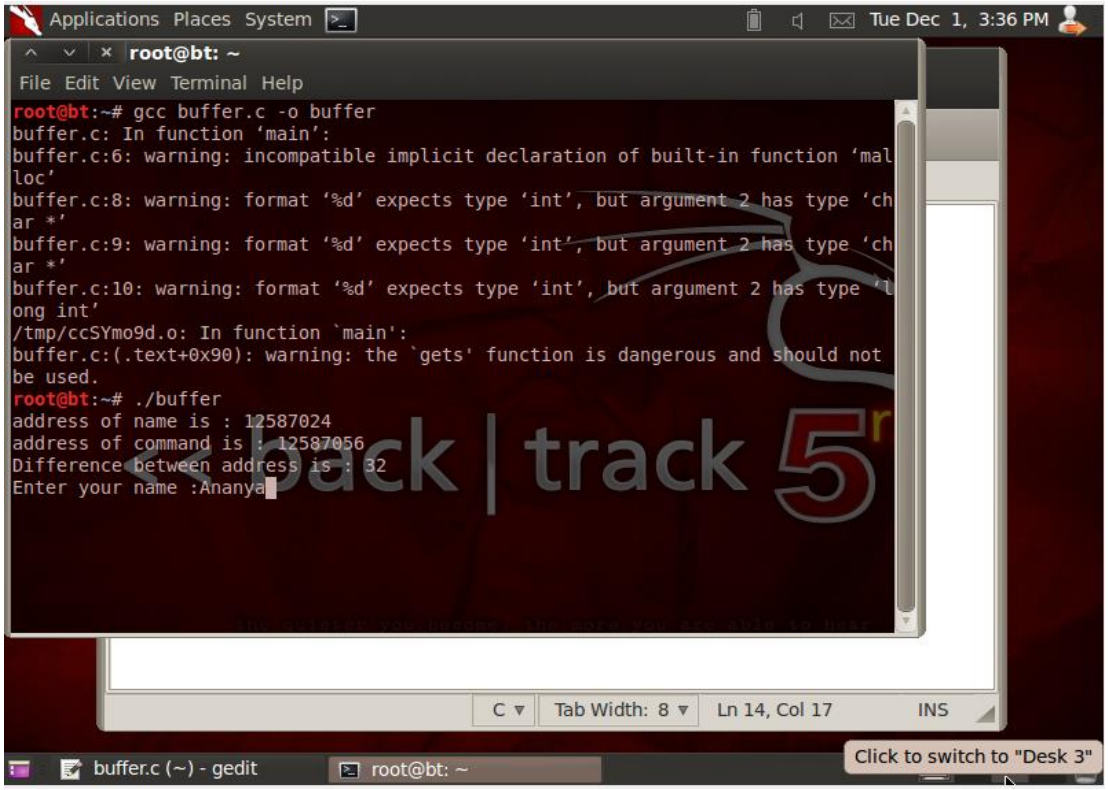
Launch the command terminal and compile the code by **gcc compiler**.



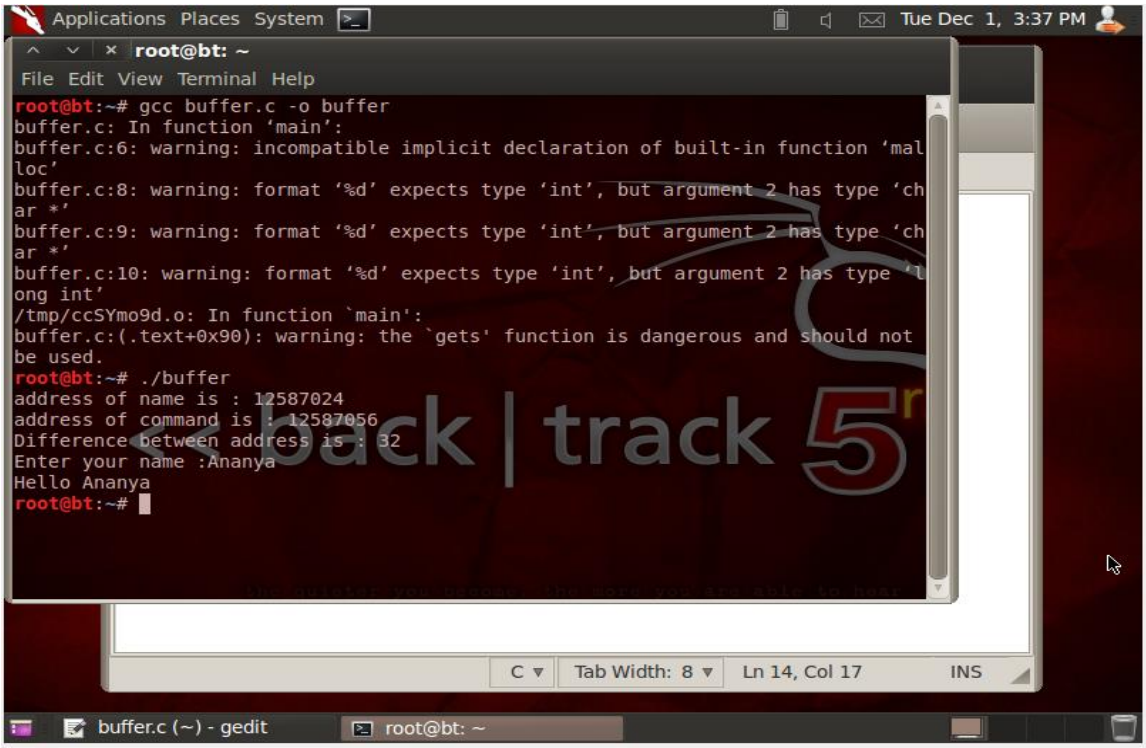
- Execute the program by typing **./buffer**.
- Ignore the warnings, if any.



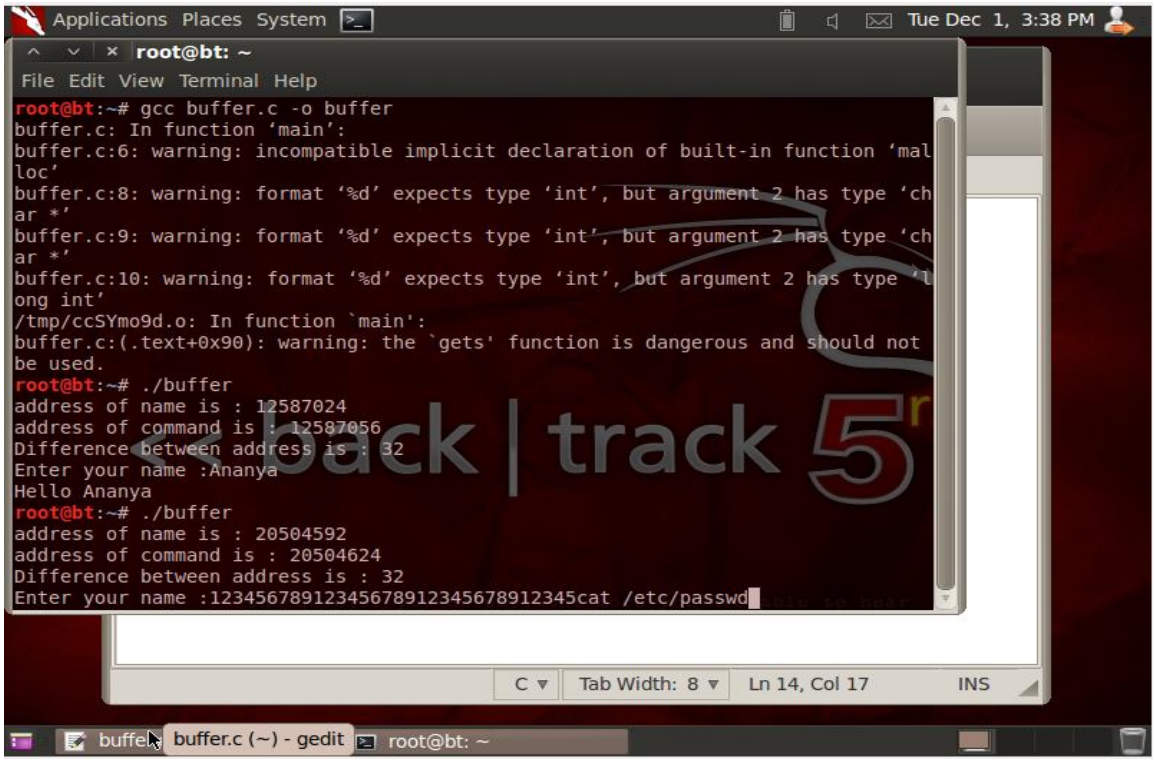
Just to check, before doing the buffer overflow attack, type any name (say Ananya) in the input field and press Enter.

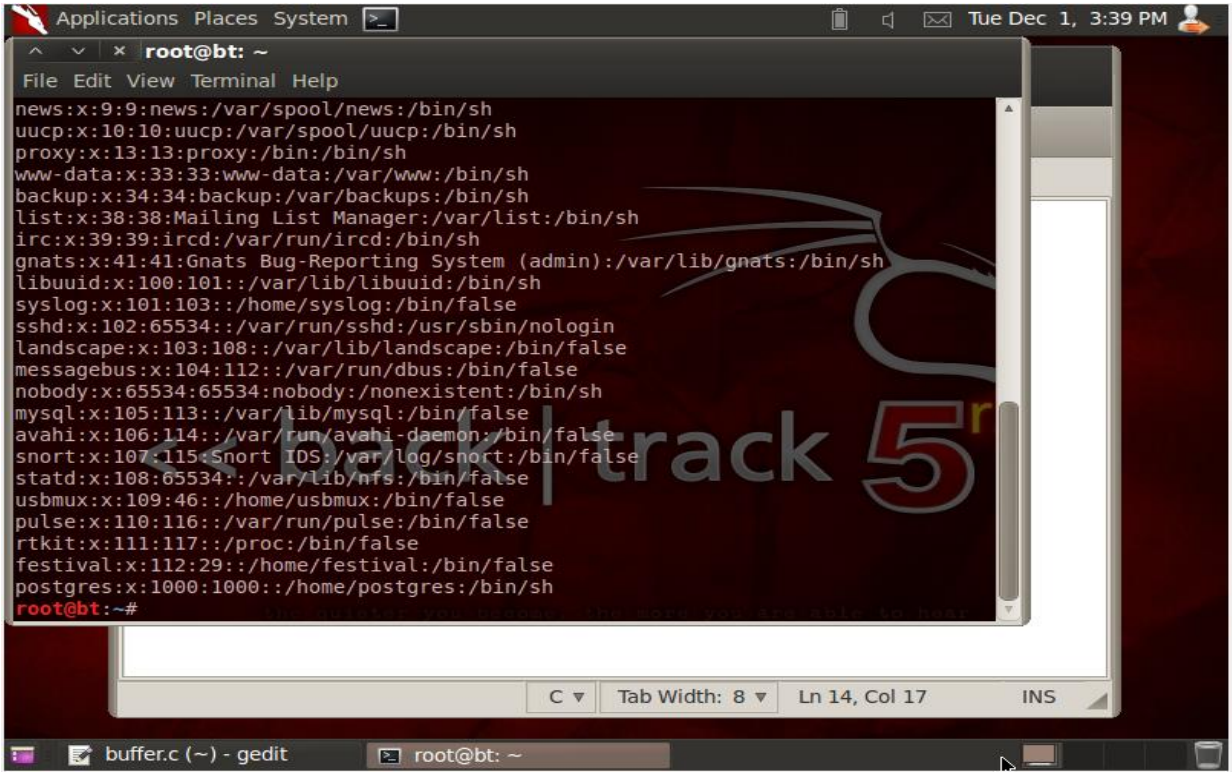


Hello Ananya should be printed.

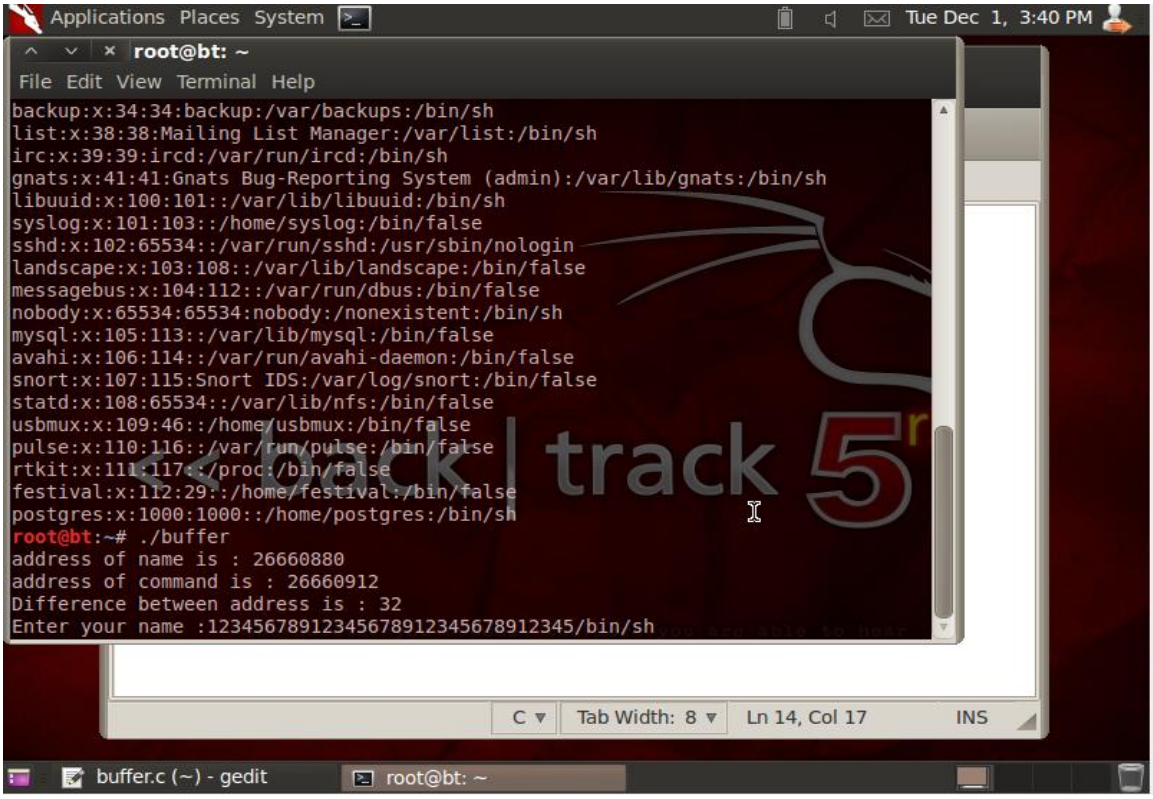


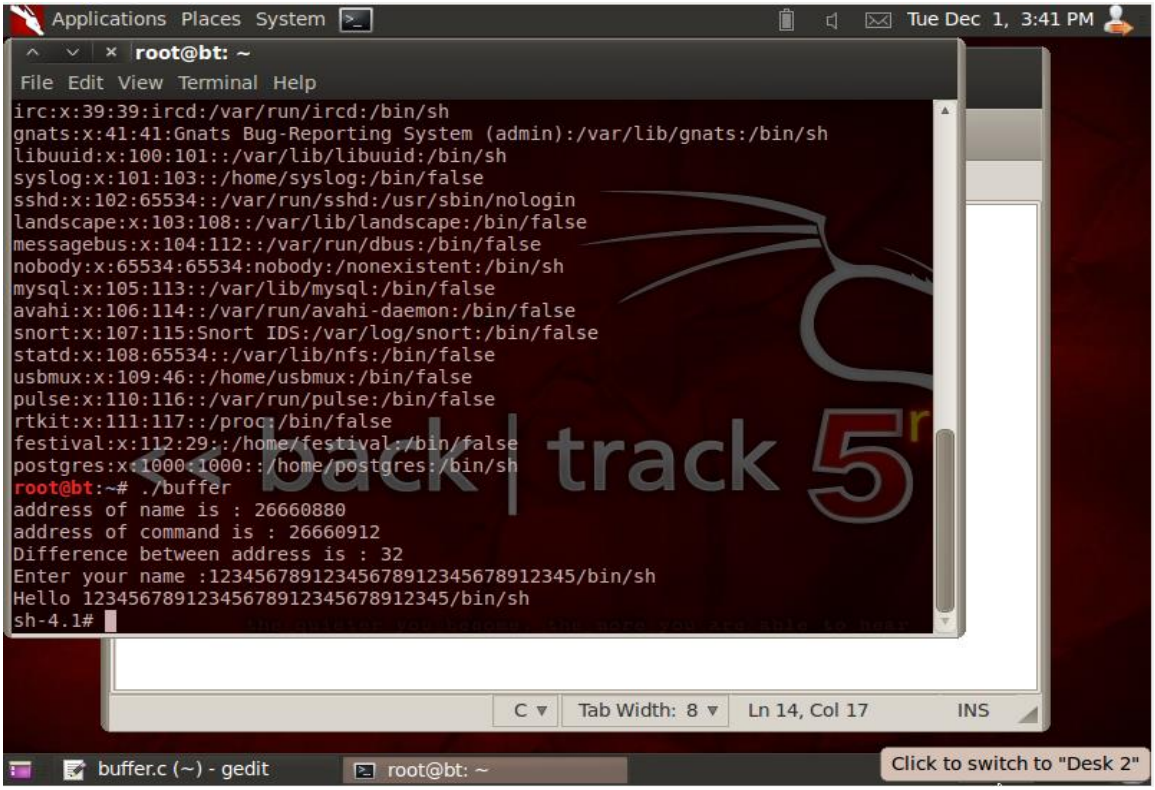
- Run **./buffer** again and execute the listed system commands for example: **1234567891234567891234567891234 5cat /etc/passwd** in the input field.
- **Writing data to the buffer (name), overruns the name boundary and overwrites the adjacent memory (command).**
- We can view the **contents** of the **/etc/passwd** file.



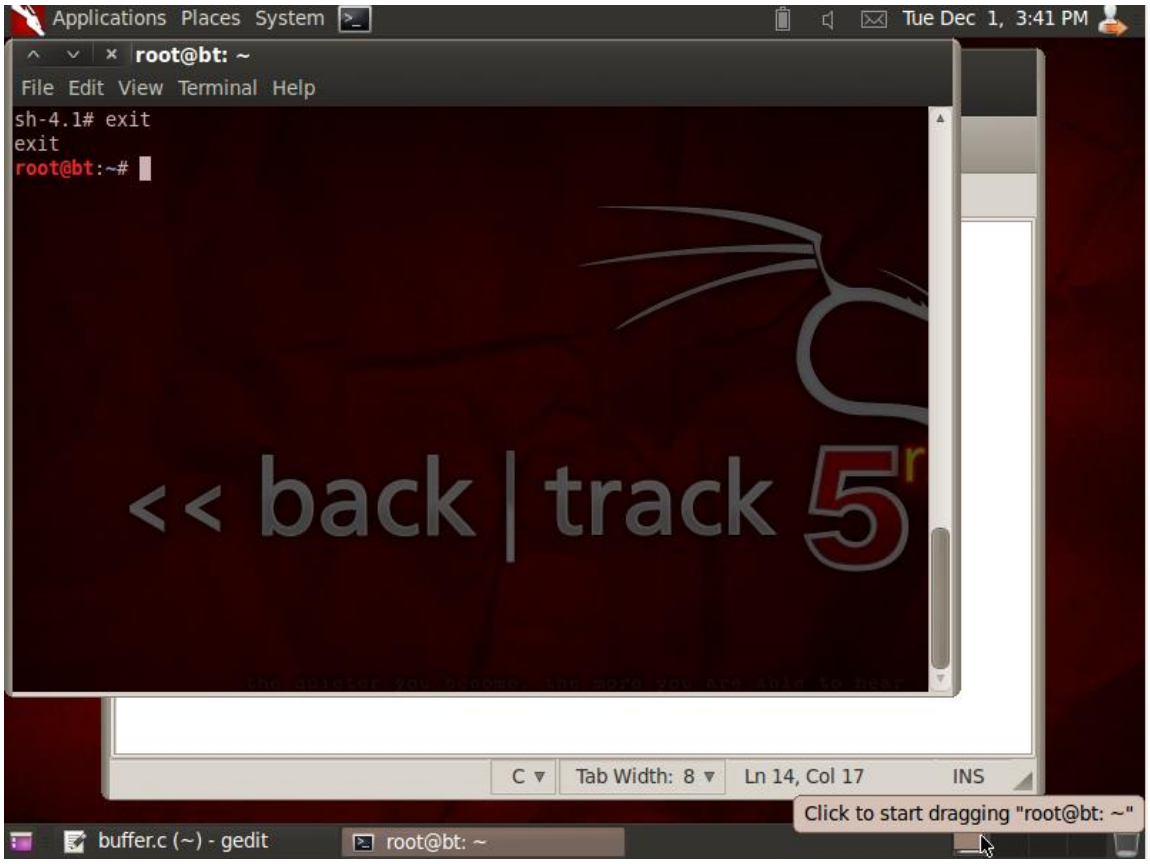


- Now, obtain the **command shell** : **sh-4.1** (here)
- Run the program again by **./buffer.c** by typing **1234567891234567891234567891234 5/bin/sh** in the input field.





Type exit in the shell Konsole or close the program.



Hence, we performed Buffer Overflow Attack by using a C program.

Q4. Give technical commentary about the 1 segment as highlighted in the captured frames, certain fields are highlighted in the segment 2, 3 and 4, give comments about each of the underlined fields.

1

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	Cisco251_af:f4:54	Broadcast	ARP	60	Who has 24.166.173.159? Tell 24.166.172.1
2	0.098594	Cisco251_af:f4:54	Broadcast	ARP	60	Who has 24.166.172.141? Tell 24.166.172.1
3	0.110617	Cisco251_af:f4:54	Broadcast	ARP	60	Who has 24.166.173.161? Tell 24.166.172.1
4	0.211791	Cisco251_af:f4:54	Broadcast	ARP	60	Who has 65.28.78.76? Tell 65.28.78.1
5	0.216744	Cisco251_af:f4:54	Broadcast	ARP	60	Who has 24.166.173.163? Tell 24.166.172.1
6	0.307909	Cisco251_af:f4:54	Broadcast	ARP	60	Who has 24.166.175.123? Tell 24.166.172.1
7	0.330433	Cisco251_af:f4:54	Broadcast	ARP	60	Who has 24.166.173.165? Tell 24.166.172.1
8	0.408556	Cisco251_af:f4:54	Broadcast	ARP	60	Who has 24.166.175.82? Tell 24.166.172.1
9	0.455104	Cisco251_af:f4:54	Broadcast	ARP	60	Who has 69.76.220.131? Tell 69.76.216.1
10	0.486666	Cisco251_af:f4:54	Broadcast	ARP	60	Who has 24.166.173.168? Tell 24.166.172.1

2

```
Source: Cisco251_af:f4:54 (00:07:0d:af:f4:54)
    Address: Cisco251_af:f4:54 (00:07:0d:af:f4:54)
    .... ..0. .... .... .... ....

Type: ARP (0x0806)

0000  ff ff ff ff ff ff 00 07  0d af f4 54 08 06 00 01   ........ ...T....
0010  08 00 06 04 00 01 00 07  0d af f4 54 18 a6 ac 01   ........ ...T....
0020  00 00 00 00 00 00 18 a6  af 7b 02 01 04 00 00 00   ........ .{......
0030  00 02 01 00 03 02 00 00  05 01 03 01               ........ ....
```

3

```
ff ff ff ff ff ff 00 07  0d af f4 54 08 06 00 01
08 00 06 04 00 01 00 07  0d af f4 54 18 a6 ac 01
00 00 00 00 00 00 18 a6  af 7b 02 01 04 00 00 00
00 02 01 00 03 02 00 00  05 01 03 01
```

4

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	145.254.160.237	65.208.228.223	TCP	62	3372 → 80 [SYN] Seq=0 Win=8760 Len=0 MSS=1460 SACK_PERM=1
2	0.911310	65.208.228.223	145.254.160.237	TCP	62	80 → 3372 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1380 SACK_PERM=1
3	0.911310	145.254.160.237	65.208.228.223	TCP	54	3372 → 80 [ACK] Seq=1 Ack=1 Win=9660 Len=0
4	0.911310	145.254.160.237	65.208.228.223	HTTP	533	GET /download.html HTTP/1.1

```
> Frame 4: 533 bytes on wire (4264 bits), 533 bytes captured (4264 bits)
> Ethernet II, Src: Xerox_00:00:00 (00:00:01:00:00:00), Dst: fe:ff:20:00:01:00 (fe:ff:20:00:01:00)
> Internet Protocol Version 4, Src: 145.254.160.237, Dst: 65.208.228.223
> Transmission Control Protocol, Src Port: 3372, Dst Port: 80, Seq: 1, Ack: 1, Len: 479
v Hypertext Transfer Protocol
  > GET /download.html HTTP/1.1\r\n
    Host: www.ethereal.com\r\n
    User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.6) Gecko/20040113\r\n
    Accept: text/xml,application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png,image/jpeg,image/gif;q=0.2,*/*;q=0.1\r\n
    Accept-Language: en-us,en;q=0.5\r\n
    Accept-Encoding: gzip,deflate\r\n
    Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7\r\n
    Keep-Alive: 300\r\n
    Connection: keep-alive\r\n
    Referer: http://www.ethereal.com/development.html\r\n
    \r\n
    [Full request URI: http://www.ethereal.com/download.html]
    [HTTP request 1/1]
    [Response in frame: 38]
```

Sol 4:

1)

- **No**: It displays the **packet number.**
- **Time**: It displays the **time the packet has spent** and not the arrival time of the packet. Thus, the starting rows have value of the time as 0.0+ and then keeps on increasing.
- **Cisco251_af:f4:54**
    - It displays the name of the **Ethernet company,** which here is **Cisco**.
    - The rest is the MAC **address of the source** (sender).
- **Destination:** ARP (Address Resolution Protocol) tells Ethernet to send a **broadcast** destined for the ARP protocols on all other machines on the network.
- **Protocol: ARP (Address Resolution Protocol**) is a communication protocol used for **discovering the link layer address**, such as a MAC address, associated with a given internet layer address, typically an IPv4 address.
- **Length:** It gives the information **of how big that packet is**.
- **Info:** The router broadcasts the ARP message "The machine with IP address **24.166.173.159,** tell your Ethernet address to the machine with the IP address **24.166.172.1** ".

2)

- Source: **Cisco251_af:f4:54 (00:07:0d:af:f4:54)**

  If we click on any single packet, it's information will be given such as:
    - The name of the **Ethernet company, which** in this case is **Cisco**.
    - The rest is the MAC **address of the source** (sender).
    - The address of the Ethernet provider will be in 48 bits wherein starting 24 bits will be the manufacturer's code and the next 24 bits are the address (MAC) of the Hardware device.

- …. ..0. …. …. …. …..
    - The LG bit (UL bit) is the second least significant bit.
    - **The U/L bit indicates whether the MAC address has been assigned by a local or universal administrator.**
    - **Universal addresses have the U/L bit set to 0 i.e. the MAC address must be globally unique.**
    - If the U/L bit is set to 1, the bits are locally administered i.e. the MAC address must be LOCALLY unique and the uniqueness does not need to extend beyond a router.
- The **Ethernet Type Field is 0806**, which means go to **ARP** instead of the default IP.

3)

- Since the Destination field is the Ethernet **broadcast** address **(ff ff ff ff ff ff).** All devices on the network will receive the ARP request.
- When we send the data which is in form of packets, **data is not received** at the receiver's end in the same sequence as it was send by the sender.
- Thus, we keep **the information of the headers in hexadecimal form** so that the **packets** can be **arranged sequentially** on the **destination end.**

4)

- **Source**: It is the **IP address of the source** (from where the packet came).
- **Destination**: It is the **IP address of the Destination** (where the packet is going).
- **TCP**: Here the Protocol or the set of rules that are followed by system for transmission of packets from source to destination is TCP i.e. **Transmission Control Protocol**. It **facilitates the exchange of messages between computing devices in a network.**
- **HTTP**: It is a **protocol** which allows the **fetching of resources, such as HTML documents.** It is the **foundation** of any **data exchange on the Web** and it is a client-server protocol, which means requests are initiated by the recipient, usually the Web browser.
- **Length:** It gives the information of **how big that packet is**.
- **Info**: It give all the information about that particular packet i.e.
  - **3372-> 80 [SYN]**
    - SYN **(synchronize)** is the first process of THREE-WAY HANDSHAKE or a **TCP 3-way handshake** in establishing communication between two systems over the **TCP/IP** protocol to make a connection between the server and client where client wants to establish a connection with server, **so it sends a segment with SYN (Synchronize Sequence Number) which informs server that client is likely to start communication** and with what sequence number it start the segment with receiver.
    - The **Destination port number** is 80.
    - The **source port number** is 3372.
  - **[SYN, ACK]**
    - It is the **intermediary process** of TCP 3-way handshake.
    - Server responds to the client request with SYN-ACK signal bits set. **Acknowledgement(ACK) signifies the response of segment it received and SYN signifies with what sequence number it is likely to start the segments with receiver** i.e. when a server receives a SYN request, it responds with a SYN-ACK (synchronize acknowledge) message.

- **[ACK]**
    - ACK**(acknowledge)** is the final process of TCP 3-way handshake where client **acknowledges the response of server** and they both establish a reliable connection with which they will start the actual data transfer.
    - It helps us to **confirm** to the other side that it has **received the SYN**.

- **HTTP/1.1**
  It is the latest version of Hypertext Transfer Protocol (HTTP) and the World Wide Web application protocol that runs on top of the Internet's TCP/IP suite of protocols**. It provides faster delivery of Web pages than the original HTTP and reduces Web traffic**.
  Instead of opening and closing a connection for each application request, HTTP 1.1 provides a persistent connection that allows multiple requests to be batched or pipelined to an output buffer.
- **GET**
  The underlying Transmission Control Protocol layer can put multiple requests (and responses to requests) into one TCP segment that gets forwarded to the Internet Protocol layer for packet transmission. Because the number of connection and disconnection requests for a **sequence of "get a file" requests is reduced, fewer packets need to flow across the Internet**. Since requests are pipelined, TCP segments are more efficient. The overall result is less Internet traffic and faster performance for the user.

❖ **Ethernet II:**
Ethernet II framing, where Intel and Xerox are the major participants in its design, defines the two-octet Ether Type field in an **Ethernet frame**, preceded by destination and source MAC addresses, that identifies an upper layer protocol encapsulated by the frame data.

❖ **Xerox_00:00:00** (00:00:01:00:00:00)
- The name of the **Ethernet company**, which in this case is **Xerox.**
- The rest is the **MAC address of the source** (sender).
- The address of the Ethernet provider will be in 48 bits wherein starting 24 bits will be the manufacturer's code and the next 24 bits are the address (MAC) of the Hardware device.

❖ **Dst: fe:ff:20:00:01:00** (fe:ff:20:00:01:00)
- The MAC address of the Destination's machine will be in 48 bits wherein starting 24 bits will be the manufacturer's code and the next 24 bits are the address (MAC) of the Hardware device.

❖ **3372:** It is the **TCP port number** of the **Source**.

❖ **80:** It is the **TCP port number** of the **Destination**.

- **User-Agent:**

  - It helps user in finding user agent string in packets using wireshark like whether which web browser or script is used for connecting to internet services.
  - The User-Agent here is **Microsoft's Windows XP (NT 5.1).**
  - **Rv:1.6 represents a model number** for this Window's device.

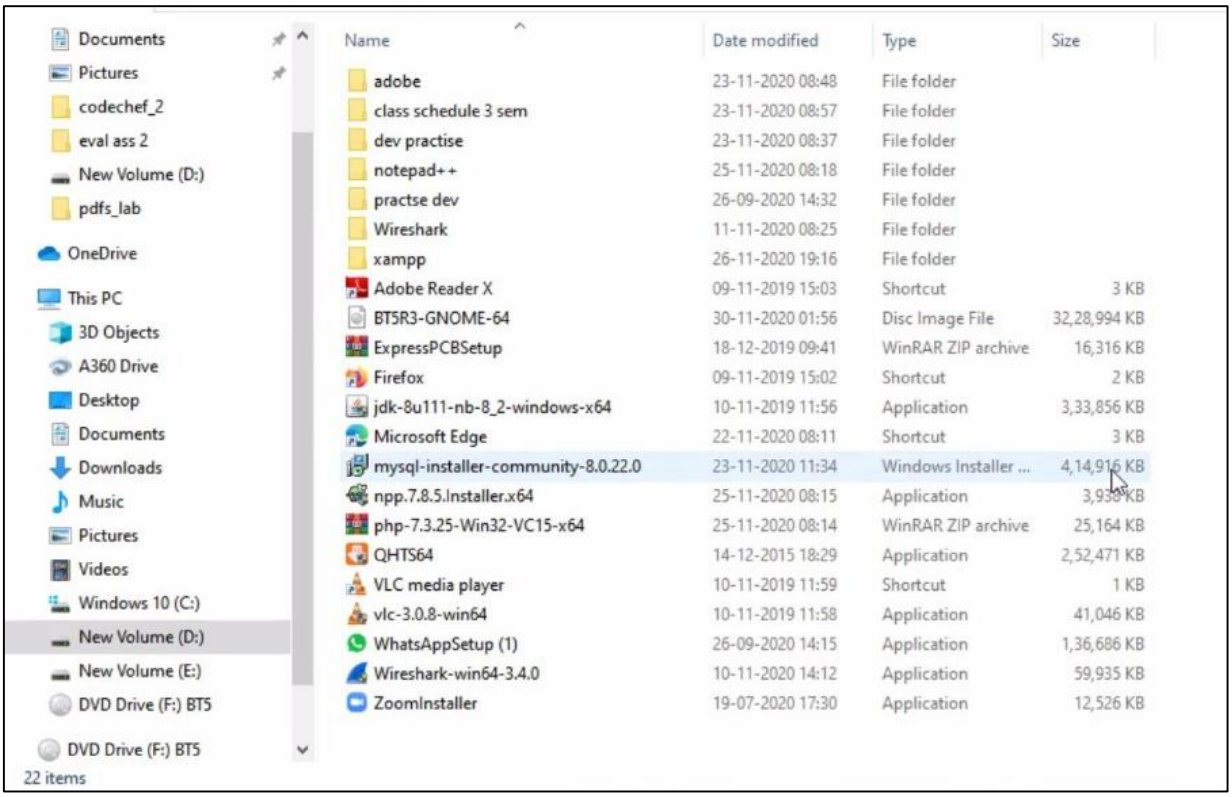
- **text/xml:**

  - This means that the media type is text that we are using in the **http request.**
  - eXtensible Markup Language (XML) is specified so as to get the http request in **XML format.**

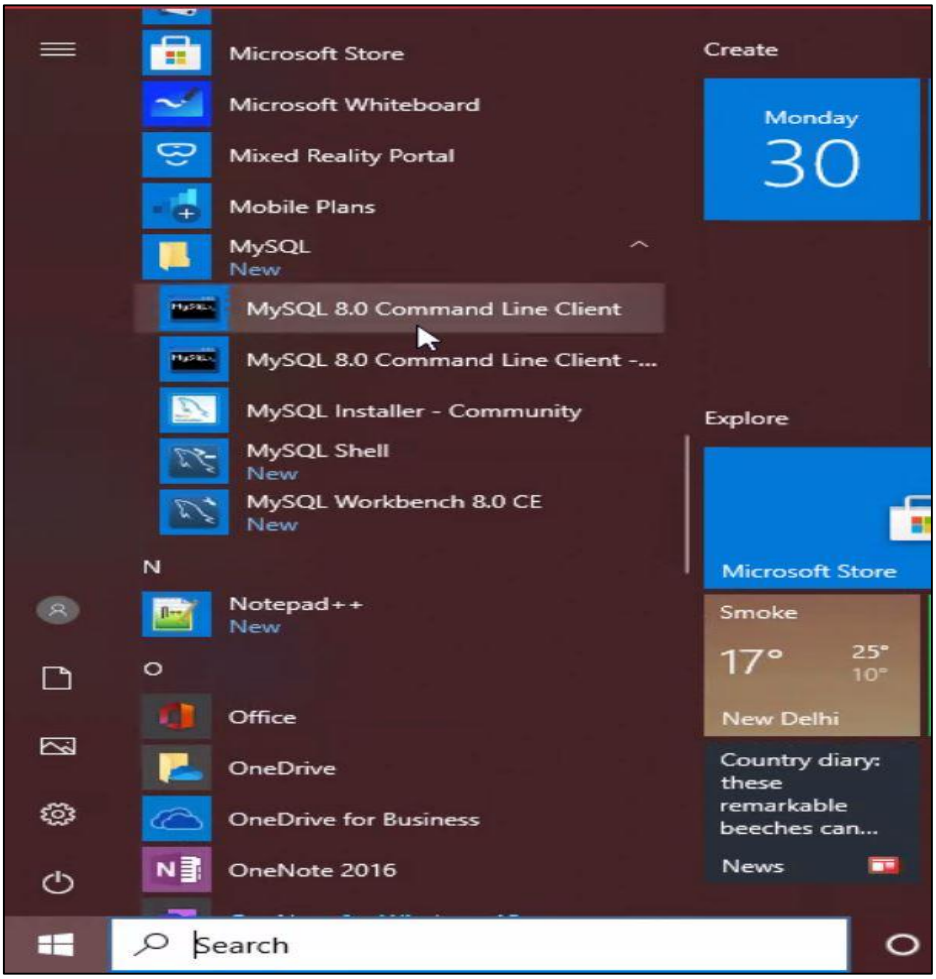
Q5. Let us assume that Charles is not satisfied with the salary she gets. She would like to increase her own salary using the SQL injection vulnerability. Please explain each and every step with screenshot.

<u>Sol 5</u>

- Install MYSQL on the system.



- MYSQL will be visible in the start menu. Click on the MYSQL 8.0 Command Line Client.



- Enter the root password which was used during the installation.

MySQL 8.0 Command Line Client

```
Enter password: ********
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 30
Server version: 8.0.22 MySQL Community Server - GPL

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

- Create a database, with the name **sql_inj**(say).

```
mysql> create database sql_inj;
Query OK, 1 row affected (1.14 sec)

mysql> use sql_inj;
Database changed
```

- Now, **create a table empl (say) with 5 attributes (i.e. columns)** for the database "sql_inj".

```
mysql> create table empl(ID int(6) NOT NULL auto_increment, NAME varchar(30) NOT NULL, PASSWORD varchar(30) NOT NULL, SALARY int(6) NOT NULL, Primary Key(ID));
Query OK, 0 rows affected, 2 warnings (8.58 sec)
```

- After a table is created, we can use "describe" to display the structure of the table.

```
mysql> describe empl;
+----------+-------------+------+-----+---------+----------------+
| Field    | Type        | Null | Key | Default | Extra          |
+----------+-------------+------+-----+---------+----------------+
| ID       | int         | NO   | PRI | NULL    | auto_increment |
| NAME     | varchar(30) | NO   |     | NULL    |                |
| PASSWORD | varchar(30) | NO   |     | NULL    |                |
| SALARY   | int         | NO   |     | NULL    |                |
+----------+-------------+------+-----+---------+----------------+
4 rows in set (0.86 sec)
```

- We can use the **INSERT INTO** statement to insert a new record into a table.
- Here, we insert a record with the required fields such as **ID, NAME, PASSWORD and SALARY into the "empl" table.**
- We do not specify a value of the ID column, as it will be automatically set by the database.
- If the condition is always True, then all the rows are affected by the SQL statement.

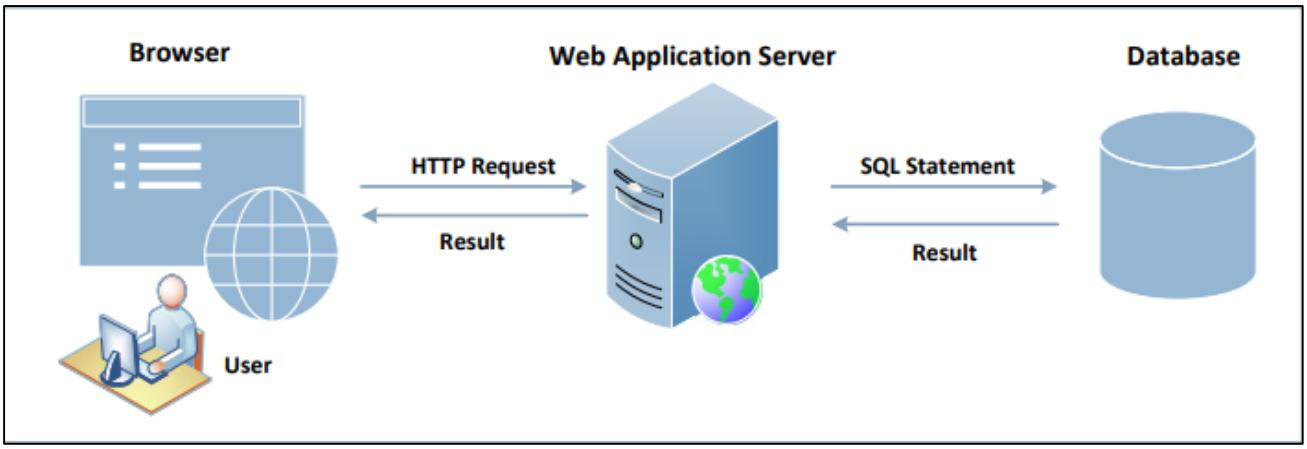
```
mysql> insert into empl(NAME, PASSWORD, SALARY) values ('Charles', 'ch34p', '35000');
Query OK, 1 row affected (1.19 sec)
```

```
mysql> insert into empl(NAME, PASSWORD, SALARY) values ('Ananya', 'anu87', '600000');
Query OK, 1 row affected (0.67 sec)
```

- **SELECT** retrieves information from a database.
- * asks the database for all its records, including all the columns.

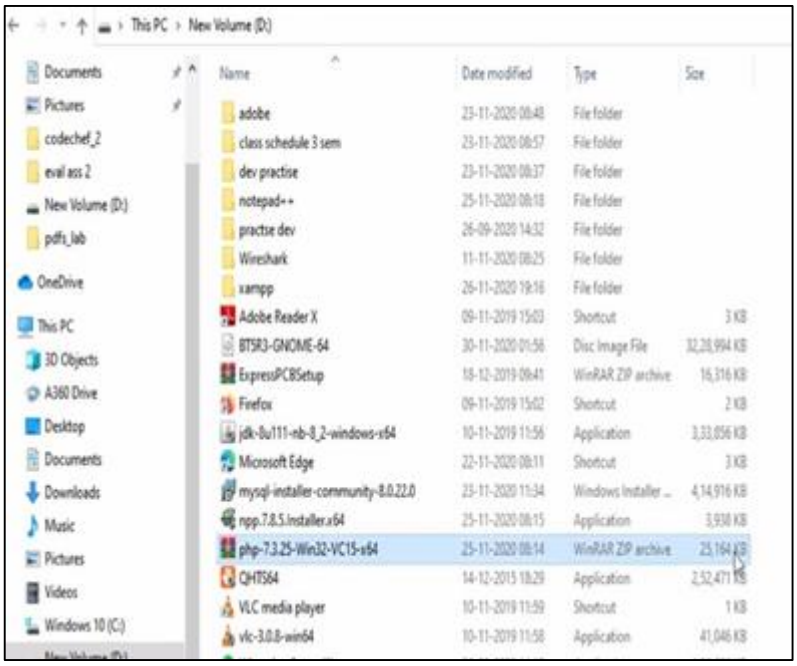
```
mysql> select * from empl;
+----+---------+----------+--------+
| ID | NAME    | PASSWORD | SALARY |
+----+---------+----------+--------+
|  1 | Charles | ch34p    |  35000 |
|  2 | Ananya  | anu87    | 600000 |
+----+---------+----------+--------+
2 rows in set (0.08 sec)
```

- **SQL Injection attacks :cause damage to the database**



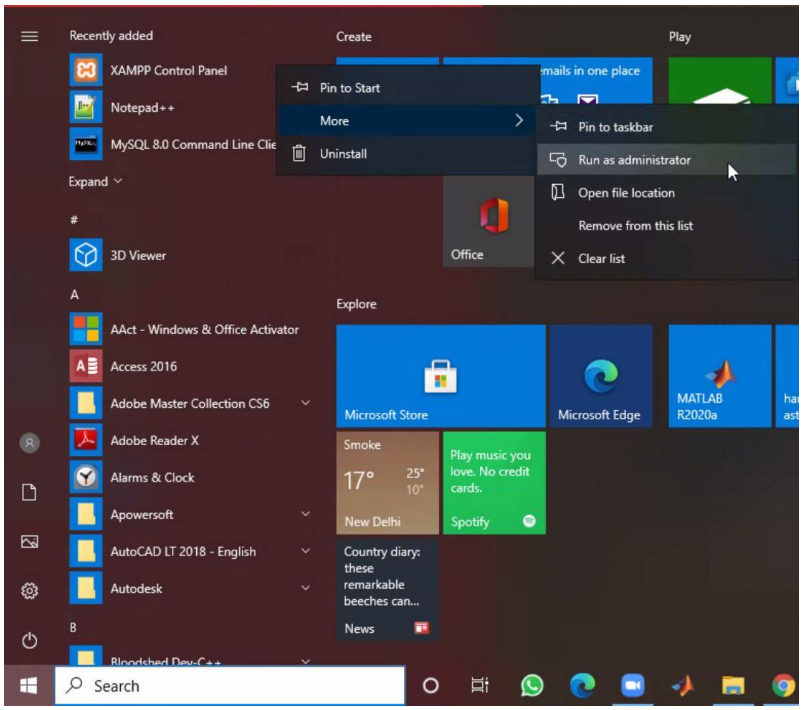
➢ As we notice in the figure, the users do not directly interact with the database but through a web server. If this channel is not implemented properly, malicious users can attack the database.

➢ **Install PHP** so as to create 2 PHP files and then we will be able to communicate with the server.

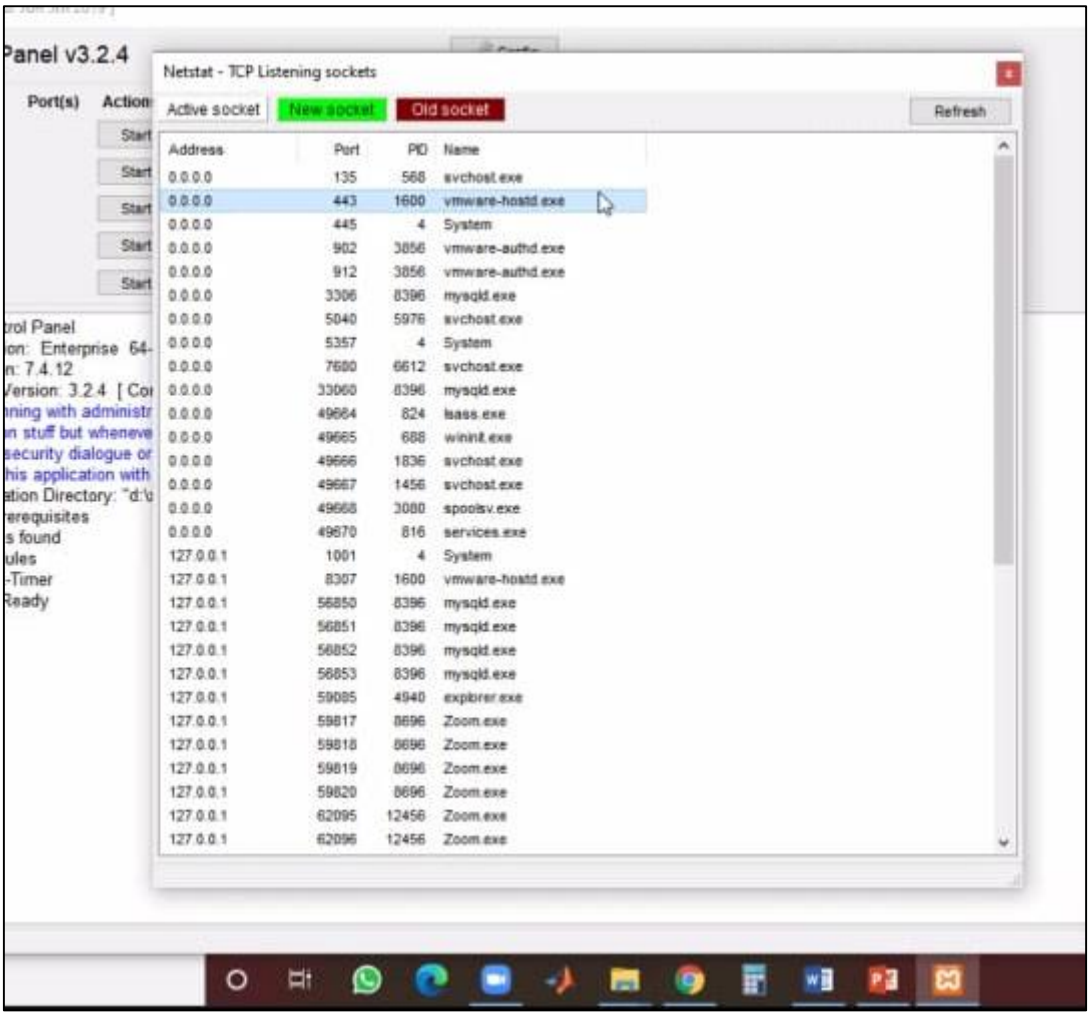


➢ The server here is **XAMPP**.

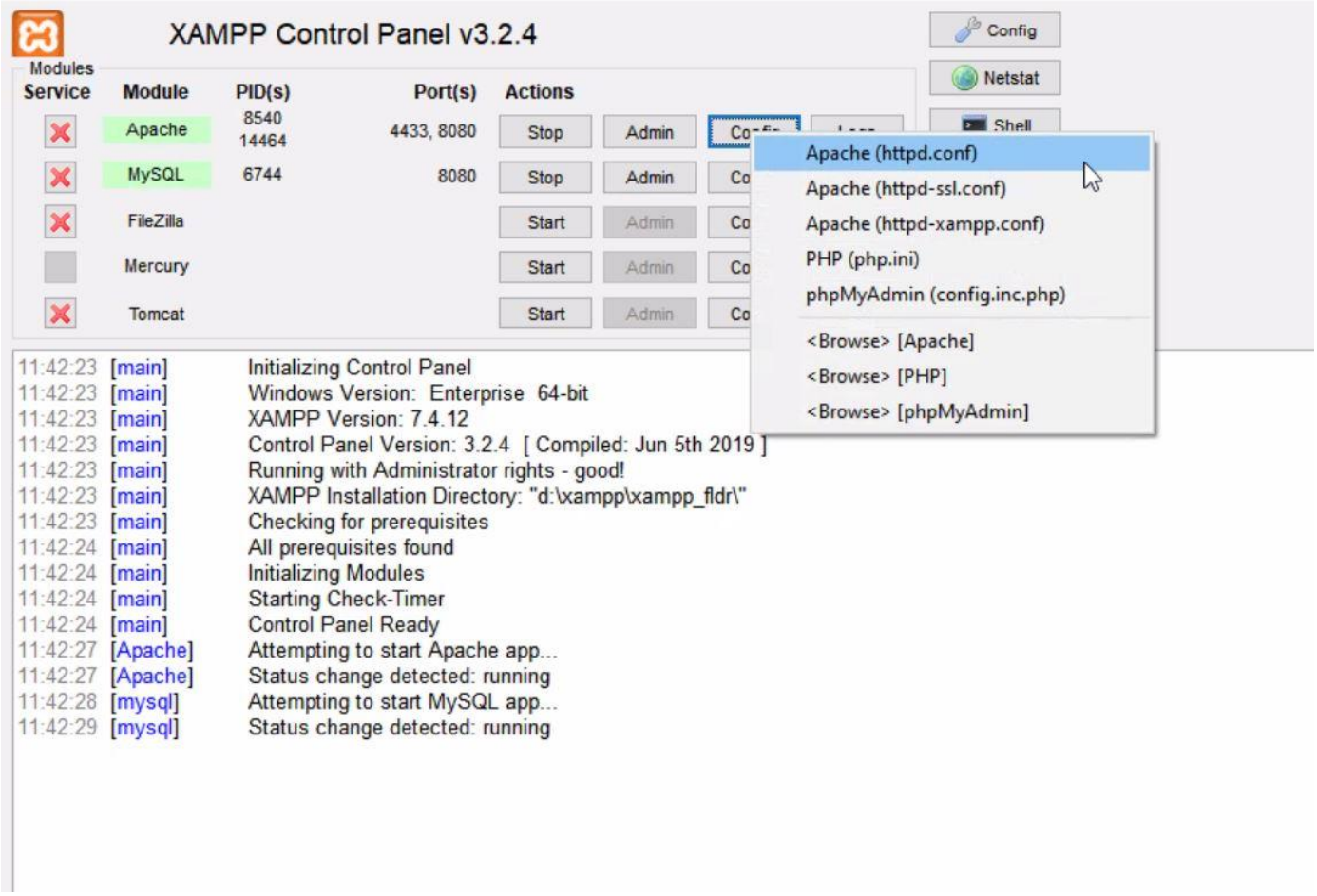
➢ Once the Xampp control panel is visible in the start menu, run it as an administrator.



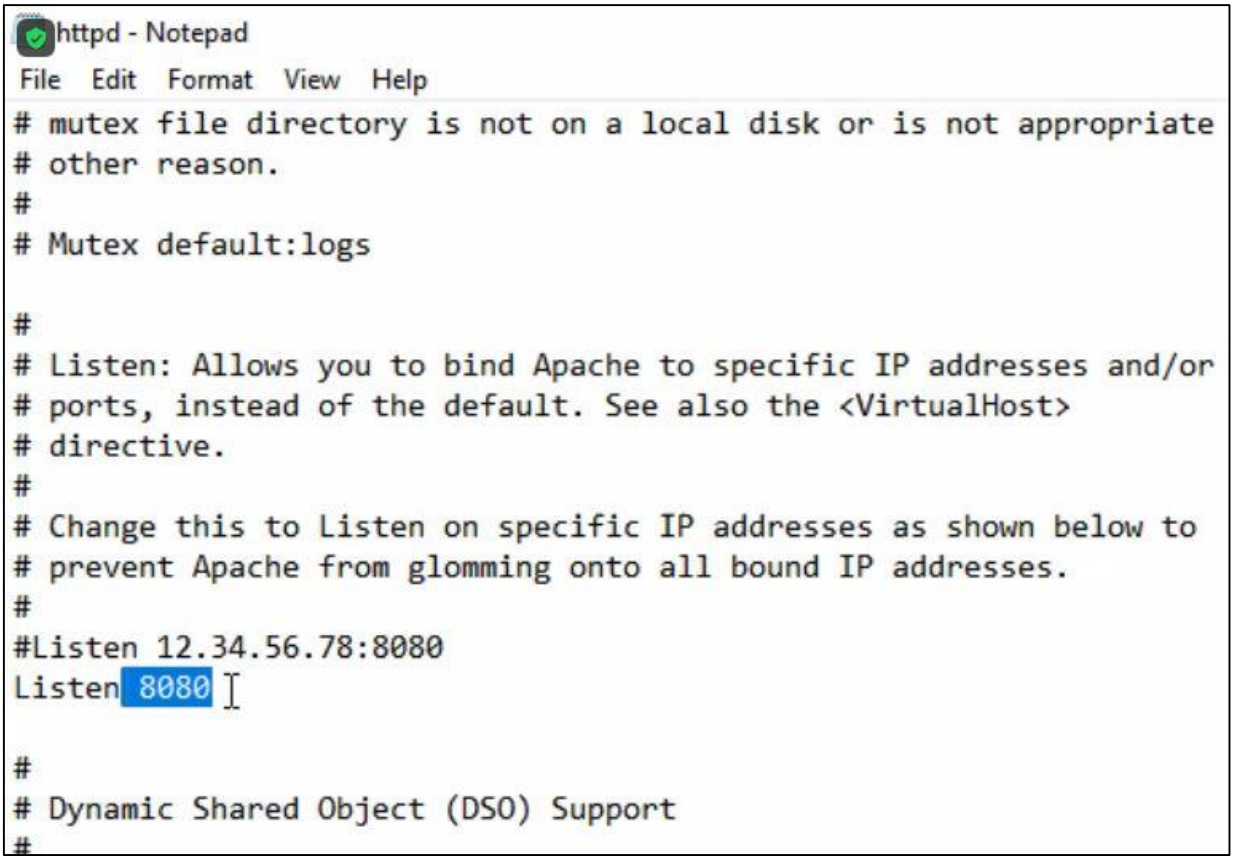
- It is possible that the by default port i.e. 80 and 443 given by Xampp are already in use by some other application. So, we have to change the port settings.
- Netstat can be used to check which port is used by which application.



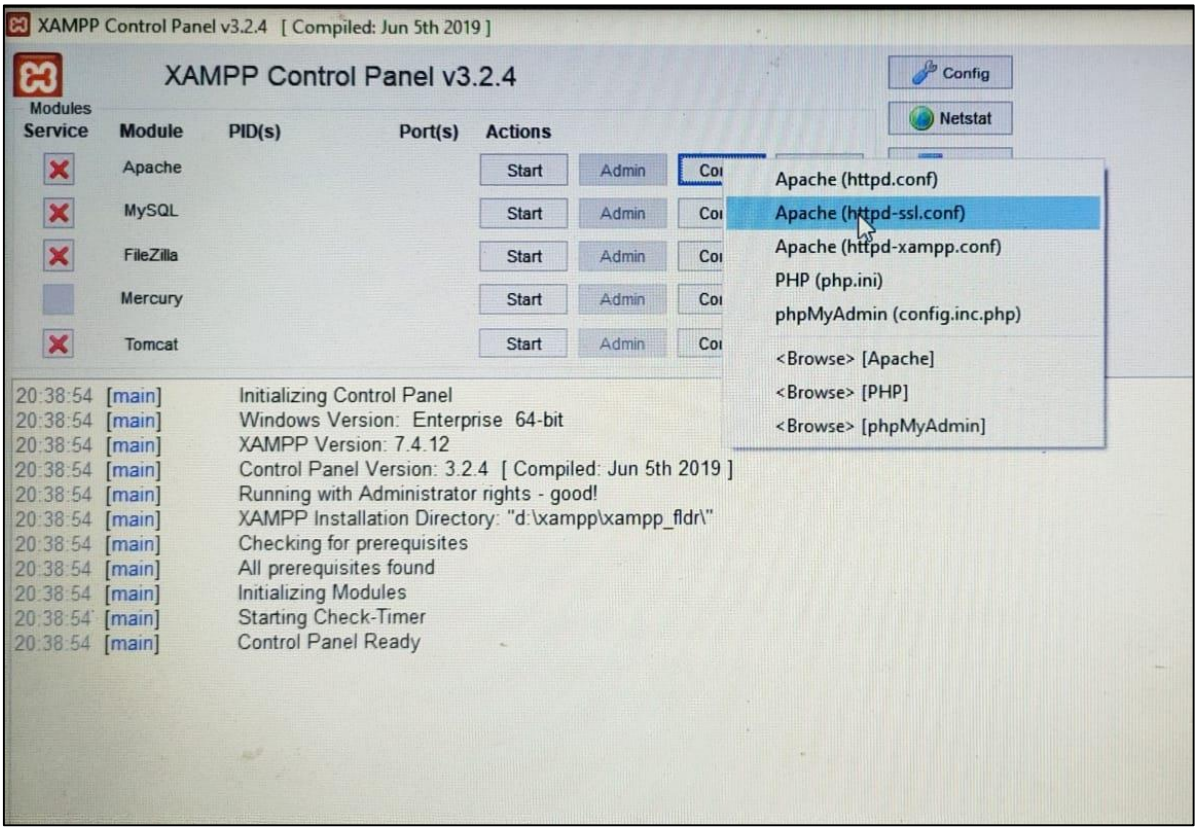
- These can be changed for both **Apache and MYSQL** by accessing their Config files and then changing the port number.

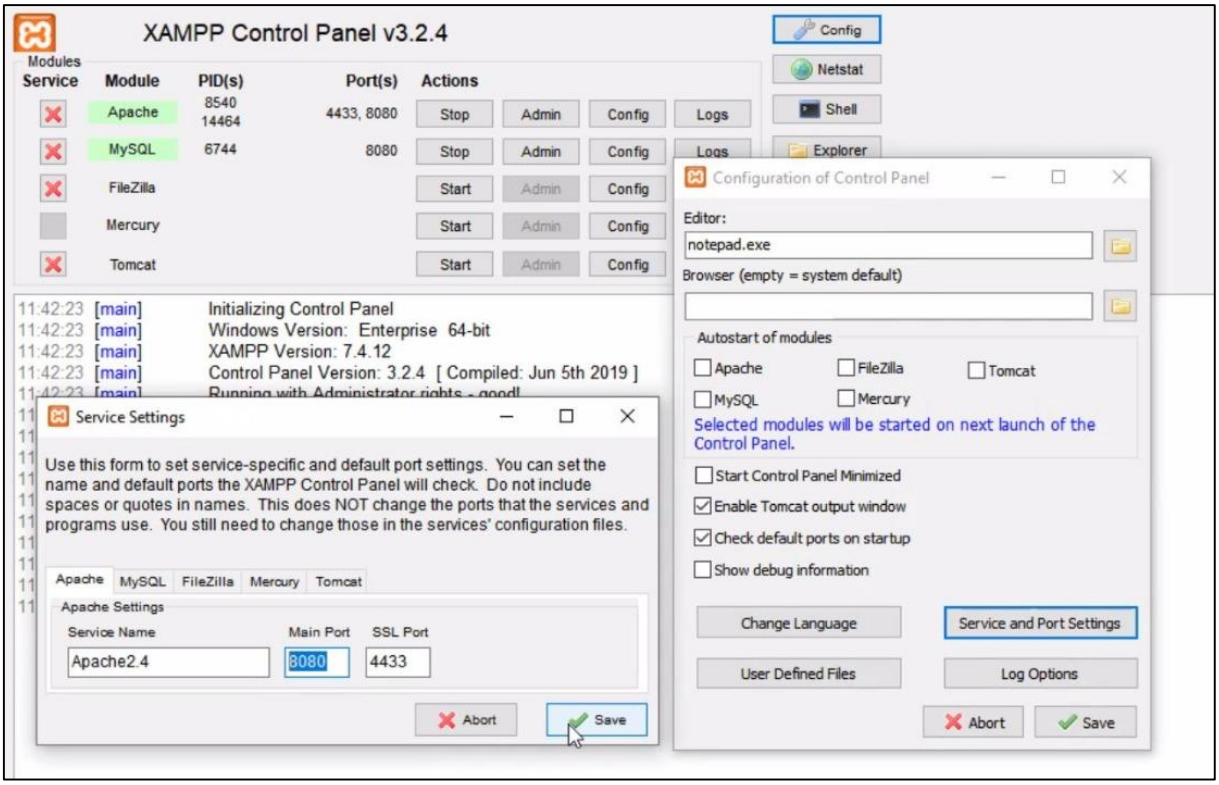


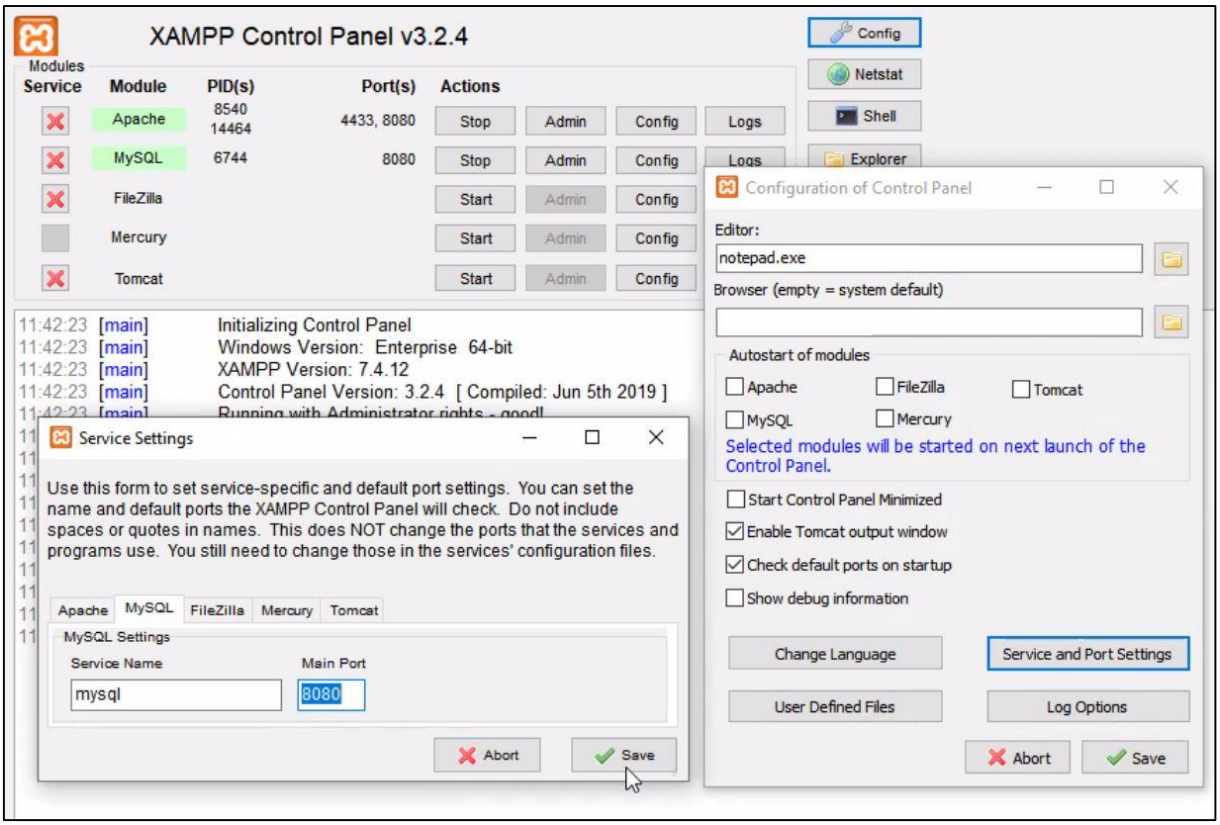
- The file httpd.config.



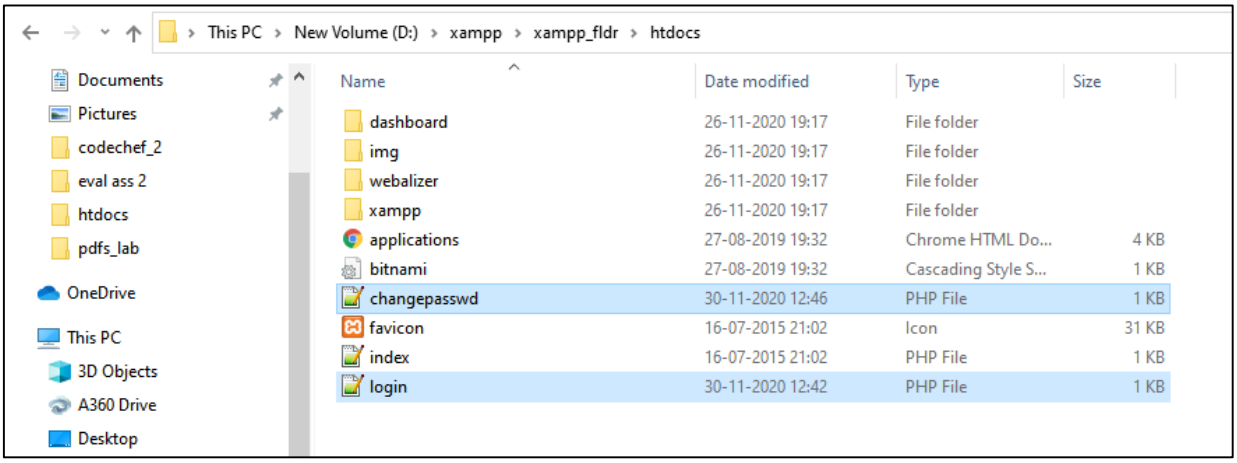
```
# mutex file directory is not on a local disk or is not appropriate
# other reason.
#
# Mutex default:logs

#
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, instead of the default. See also the <VirtualHost>
# directive.
#
# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses.
#
#Listen 12.34.56.78:8080
Listen 8080

#
# Dynamic Shared Object (DSO) Support
#
```



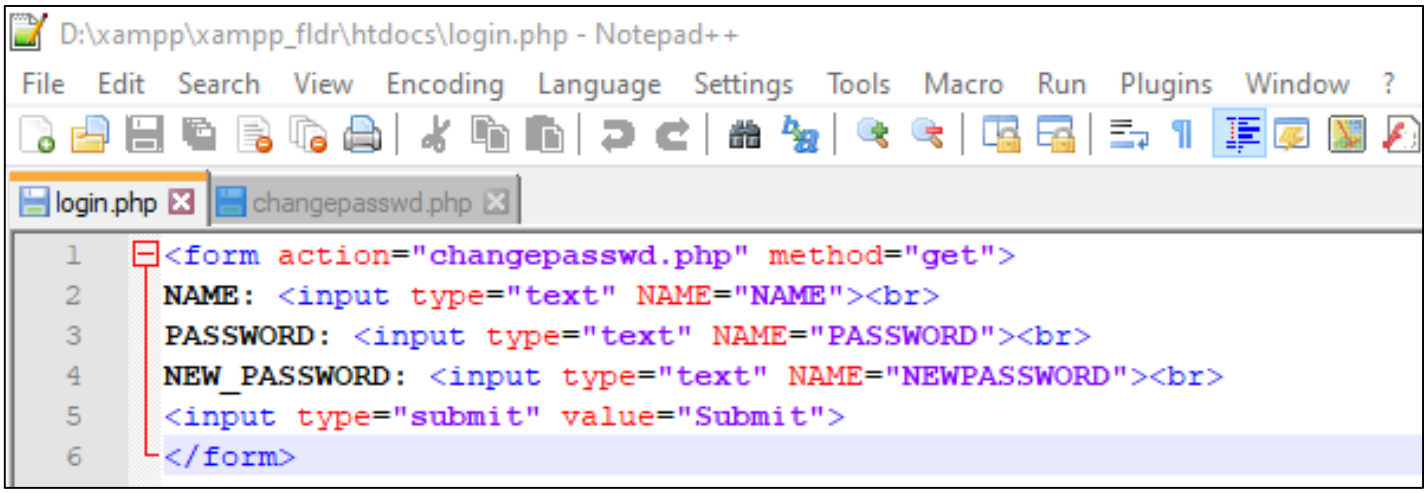

XAMPP Control Panel v3.2.4
Modules
Service
Module
PID(s)
Port(s)
Actions
Apache
8540
14464
4433, 8080
Stop
Admin
Config
Logs
MySQL
6744
8080
FileZilla
Start
Mercury
Tomcat
Config
Netstat
Shell
Explorer
11:42:23 [main] Initializing Control Panel
11:42:23 [main] Windows Version: Enterprise 64-bit
11:42:23 [main] XAMPP Version: 7.4.12
11:42:23 [main] Control Panel Version: 3.2.4 [Compiled: Jun 5th 2019]
Service Settings
Use this form to set service-specific and default port settings. You can set the name and default ports the XAMPP Control Panel will check. Do not include spaces or quotes in names. This does NOT change the ports that the services and programs use. You still need to change those in the services' configuration files.
Apache
MySQL
FileZilla
Mercury
Tomcat
Apache Settings
Service Name
Main Port
SSL Port
Apache2.4
8080
4433
Abort
Save
Configuration of Control Panel
Editor:
notepad.exe
Browser (empty = system default)
Autostart of modules
Selected modules will be started on next launch of the Control Panel.
Start Control Panel Minimized
Enable Tomcat output window
Check default ports on startup
Show debug information
Change Language
Service and Port Settings
User Defined Files
Log Options


XAMPP Control Panel v3.2.4
Modules
Service
Module
PID(s)
Port(s)
Actions
Apache
8540
14464
4433, 8080
Stop
Admin
Config
Logs
MySQL
6744
8080
FileZilla
Start
Mercury
Tomcat
Config
Netstat
Shell
Explorer
11:42:23 [main] Initializing Control Panel
11:42:23 [main] Windows Version: Enterprise 64-bit
11:42:23 [main] XAMPP Version: 7.4.12
11:42:23 [main] Control Panel Version: 3.2.4 [Compiled: Jun 5th 2019]
Service Settings
Use this form to set service-specific and default port settings. You can set the name and default ports the XAMPP Control Panel will check. Do not include spaces or quotes in names. This does NOT change the ports that the services and programs use. You still need to change those in the services' configuration files.
Apache
MySQL
FileZilla
Mercury
Tomcat
MySQL Settings
Service Name
Main Port
mysql
8080
Abort
Save
Configuration of Control Panel
Editor:
notepad.exe
Browser (empty = system default)
Autostart of modules
Selected modules will be started on next launch of the Control Panel.
Start Control Panel Minimized
Enable Tomcat output window
Check default ports on startup
Show debug information
Change Language
Service and Port Settings
User Defined Files
Log Options

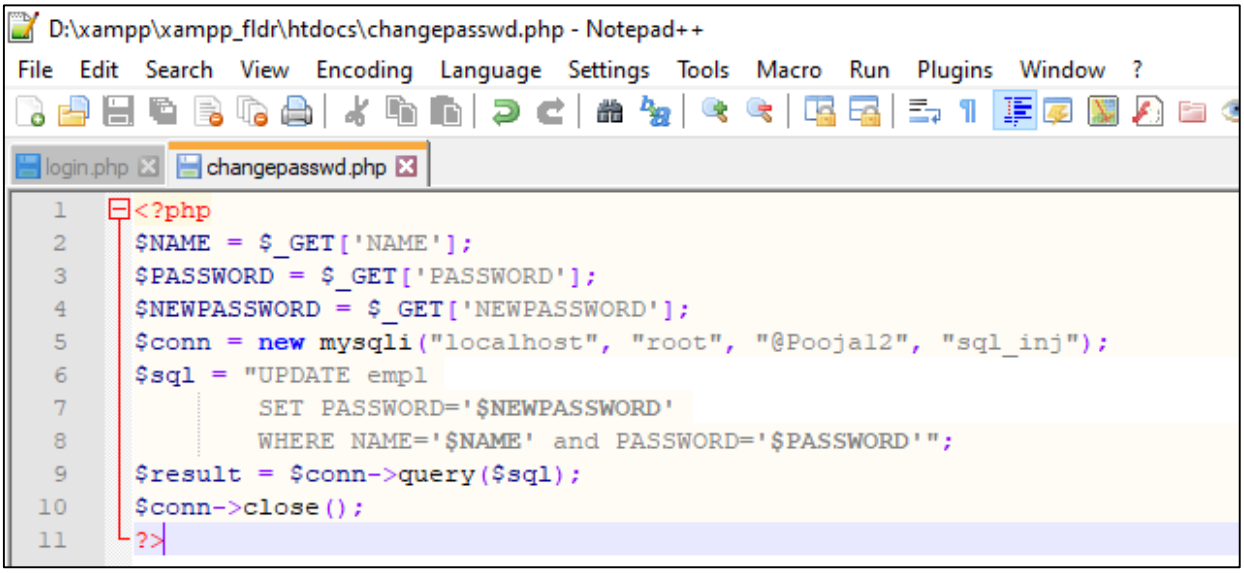
- Make 2 files: **login.php and changepasswd.php in the D:\xampp\xampp_fldr\htdocs**.



- The file login.php opened with Notepad++ text editor.
- Explanation of login.php:
  - The user will be asked to enter his\her credentials i.e**. Name, Password and New Password.**
  - When the user will press Submit button, the action is to make a HTTP GET request, because the method field in the HTML code specified the get type.



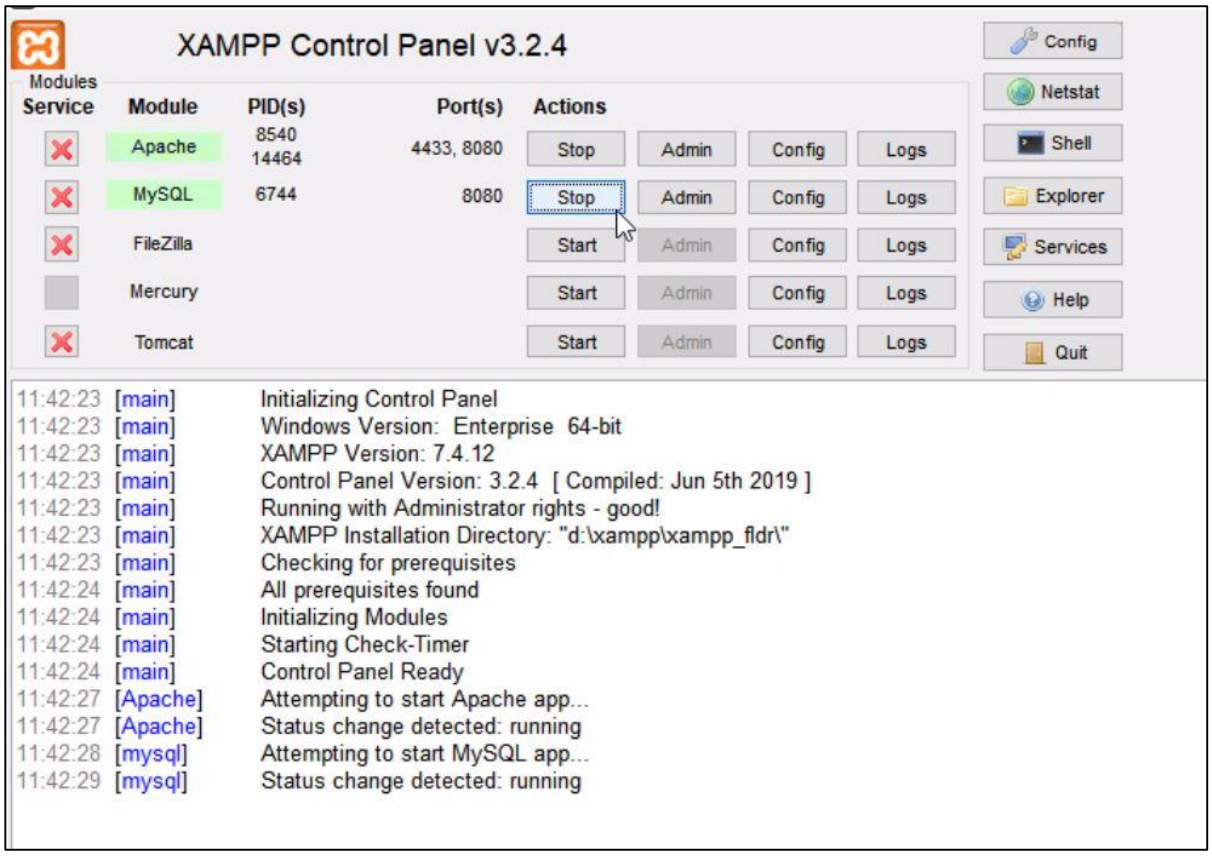
- File changepasswd.php.



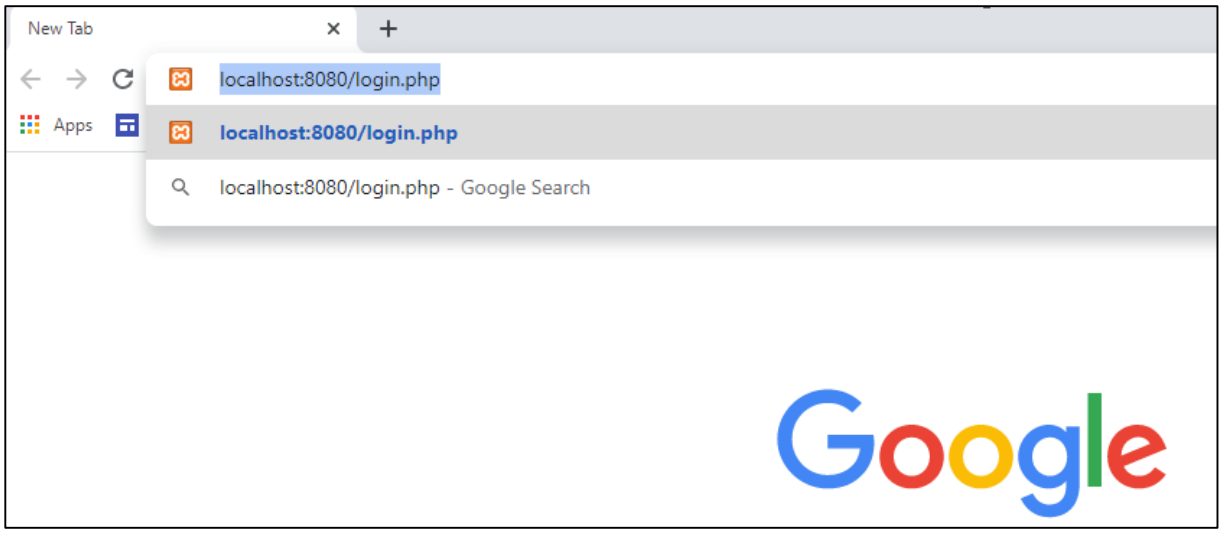
- Once this request reached the target PHP script, the parameters inside the HTTP request will be saved to an array $_GET. The following example shows a PHP script getting data from a GET request.
- PHP program connects to the database server before conducting query on database.
- The code shown below uses **new mysqli(…)** along with its **4 arguments** to connect to MySQL Database.
- These 4 arguments includes:

  - We have logged in by name local host as a database host.
  - Database user is root.
  - The database password i.e. @Pooja12.
  - The database name of the MYSQL Database i.e. sql_inj.
- Then, the connection is closed.

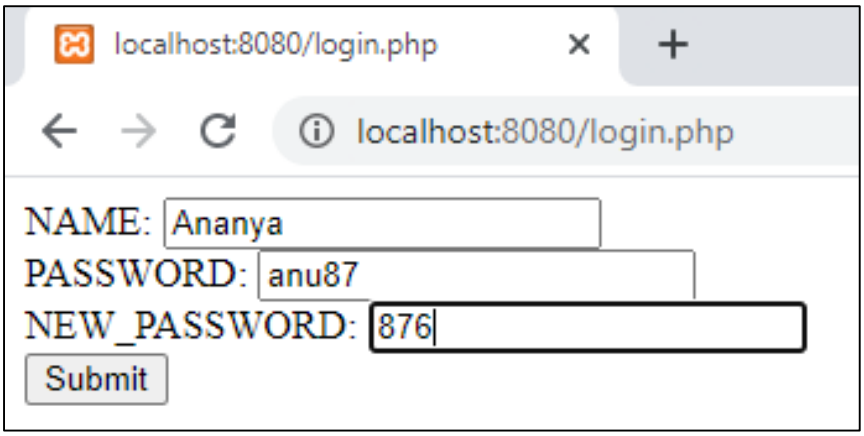
- Start Apache and MYSQL in the Xampp server and then minimize it.



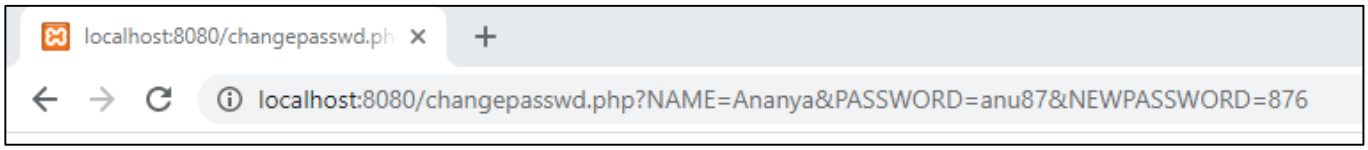
- Go to the browser and type localhost:8080/login.php where 8080 is the port number and login.php is the file name.



- Once the entries are done in this page, as soon as the Submit button is clicked, an HTTP request will be sent out with the data attached i.e. new page getdata1.php is opened and if the credentials on the login.php match, SALARY gets displayed.



Request generated is:

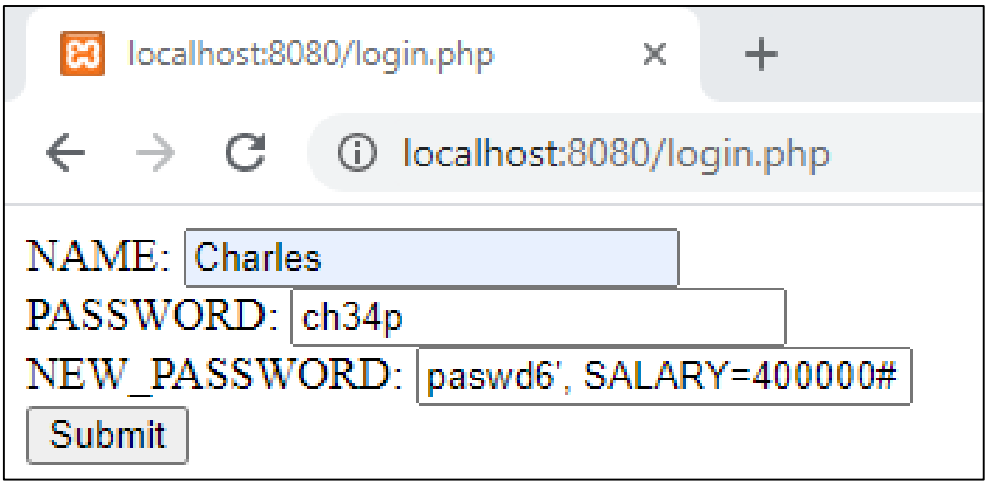


- SQL INJECTION ATTACK:

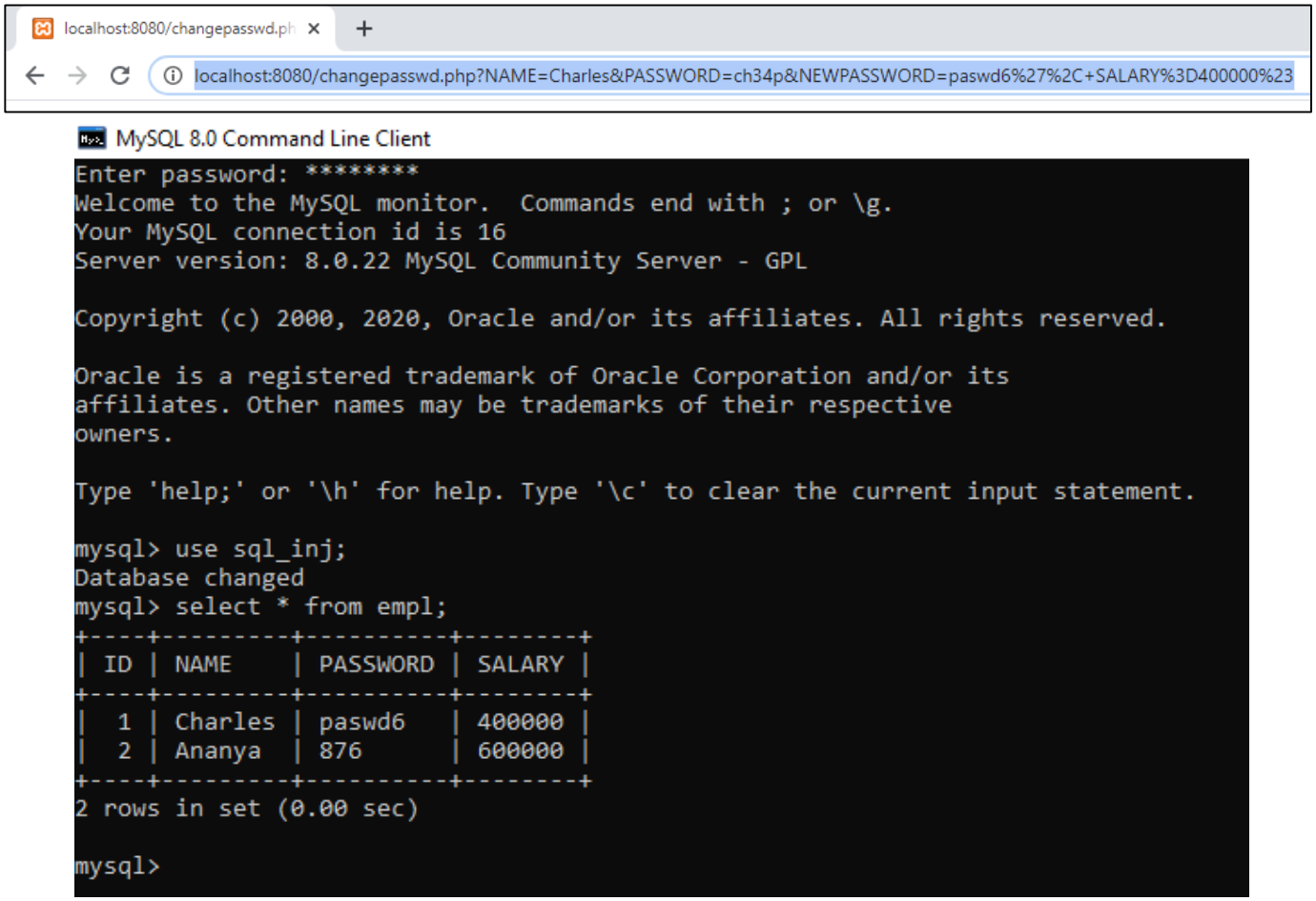
  - If the statement **is UPDATE or INSERT INTO**, we will have chance to change the database.
  - The form created for changing passwords asks users to fill in three pieces of information, Name, Password and New Password.
  - When Submit button is clicked, an HTTP POST request will be sent to the server-side script changepasswd.php, which uses an UPDATE statement to change the user's password.

- If Charles is not happy with her Salary she could manipulate the salary in the following way:
- She would type her own Name and old Password by keeping the fact in mind that: The text from the # character to the end of line is treated as a comment.
- The following will be typed into the "NewPassword" box:
- NEW_PASSWORD: paswd6', SALARY =400000#
- The SQL will now look as follows:
  "UPDATE empl
  SET PASSWORD='paswd6', SALARY=400000#'
  WHERE NAME='Charles' and PASSWORD='ch34p'";

- By typing the above string in “New Password” box, we get the UPDATE statement to set one more attribute for us. For example, here the salary attribute is set automatically.



Request generated is:



- The above statement will thus update Charles’ Salary as per her desire. This is security breach.

END