Name: SHREEYA CHATTERJI
Class: CO16

ASSIGNMENT 1

Q1. What are the different kinds of security attacks possible?

A: Cybersecurity attacks can take many forms, targeting various aspects of computer systems, networks, and data. Here are some of the different types of cyber attacks:

1. Malware Attacks:
    - Viruses: Programs that can replicate themselves and spread to other files and systems.
    - Worms: Self-replicating malware that can spread across networks.
    - Trojans: Malicious programs disguised as legitimate software.
    - Ransomware: Malware that encrypts data and demands a ransom for decryption.
    - Spyware: Software designed to gather information without the user's knowledge.
2. Phishing Attacks:
    - Phishing: Sending fraudulent emails or messages to trick users into revealing sensitive information.
    - Spear Phishing: Targeted phishing attacks that personalize the message based on the victim.
    - Whaling: Phishing attacks specifically targeting high-profile individuals or executives.
3. Denial of Service (DoS) Attacks:
    - DoS Attacks: Flooding a system or network with traffic to overwhelm and disrupt services.
    - Distributed DoS (DDoS) Attacks: Coordinated attacks using multiple compromised devices to amplify the impact.
4. Man-in-the-Middle (MitM) Attacks:
    - Intercepting and altering communication between two parties without their knowledge.
5. SQL Injection Attacks:
    - Exploiting vulnerabilities in database input to execute unauthorized SQL commands.
6. Cross-Site Scripting (XSS) Attacks:
    - Injecting malicious scripts into websites that are then executed by users' browsers.
7. Zero-Day Exploits:
    - Exploiting software vulnerabilities that are unknown to the vendor and have not been patched.
8. Drive-By Downloads:

- Infecting a user's system by visiting a compromised website, often without the user's knowledge.

9. Password Attacks:
    - Brute Force: Trying all possible combinations to guess a password.
    - Dictionary Attack: Using a list of commonly used passwords to guess a password.
    - Credential Stuffing: Using stolen usernames and passwords from one site to gain unauthorized access to other sites.
10. Social Engineering Attacks:
    - Manipulating individuals into revealing confidential information or performing actions that compromise security.
11. IoT (Internet of Things) Attacks:
    - Exploiting vulnerabilities in connected devices to gain access to networks or data.
12. Cryptojacking:
    - Illegally using someone else's computer to mine cryptocurrency.
13. Eavesdropping/Sniffing:
    - Unauthorized interception of network traffic to gather sensitive information.
14. Malvertising:
    - Distributing malware through online advertising networks.
15. Insider Threats:
    - Malicious actions by individuals with authorized access to systems and data.
16. Watering Hole Attacks:
    - Compromising a website frequented by the target audience to infect visitors with malware.
17. Fileless Attacks:
    - Exploiting vulnerabilities in software and applications without writing files to the system's disk.

These are just some examples of the various cyber attacks that can occur. Cybersecurity professionals work to prevent, detect, and respond to these threats to safeguard systems, networks, and data.

Q2. Difference between Worms , Trojans and Virus.

A:

Aspect	**Worms**	**Trojans**	**Viruses**

Name: SHREEYA CHATTERJI
Class: CO16

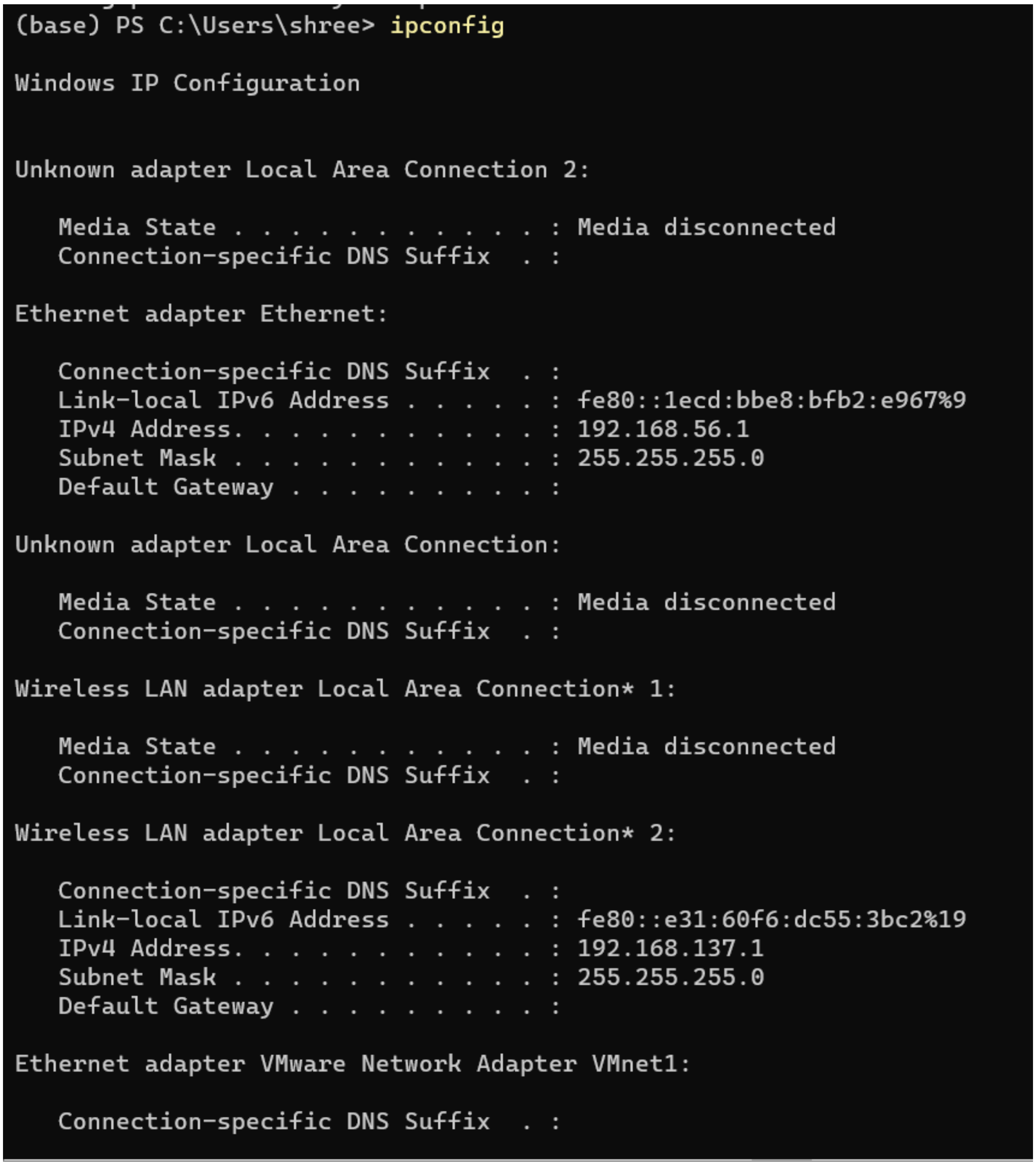
Propagation	Self-replicating, spread automatically	Rely on user interaction for execution	Attach to executable files, spread through user actions
Spread	Over networks, exploiting vulnerabilities	Often delivered through social engineering	Attach to files, require user interaction to spread
Purpose	Rapid spread, causing congestion	Carry various payloads (backdoors, keyloggers, etc.)	Carry various payloads, may corrupt data or execute other actions
Delivery	Exploits network vulnerabilities	Emails, malicious links, compromised software	Attach to files, delivered through user actions
Example	Conficker, Code Red, Blaster	Zeus, Emotet, Remote Access Trojans (RATs)	Melissa, ILOVEYOU, Sasser

Name: SHREEYA CHATTERJI
Class: CO16

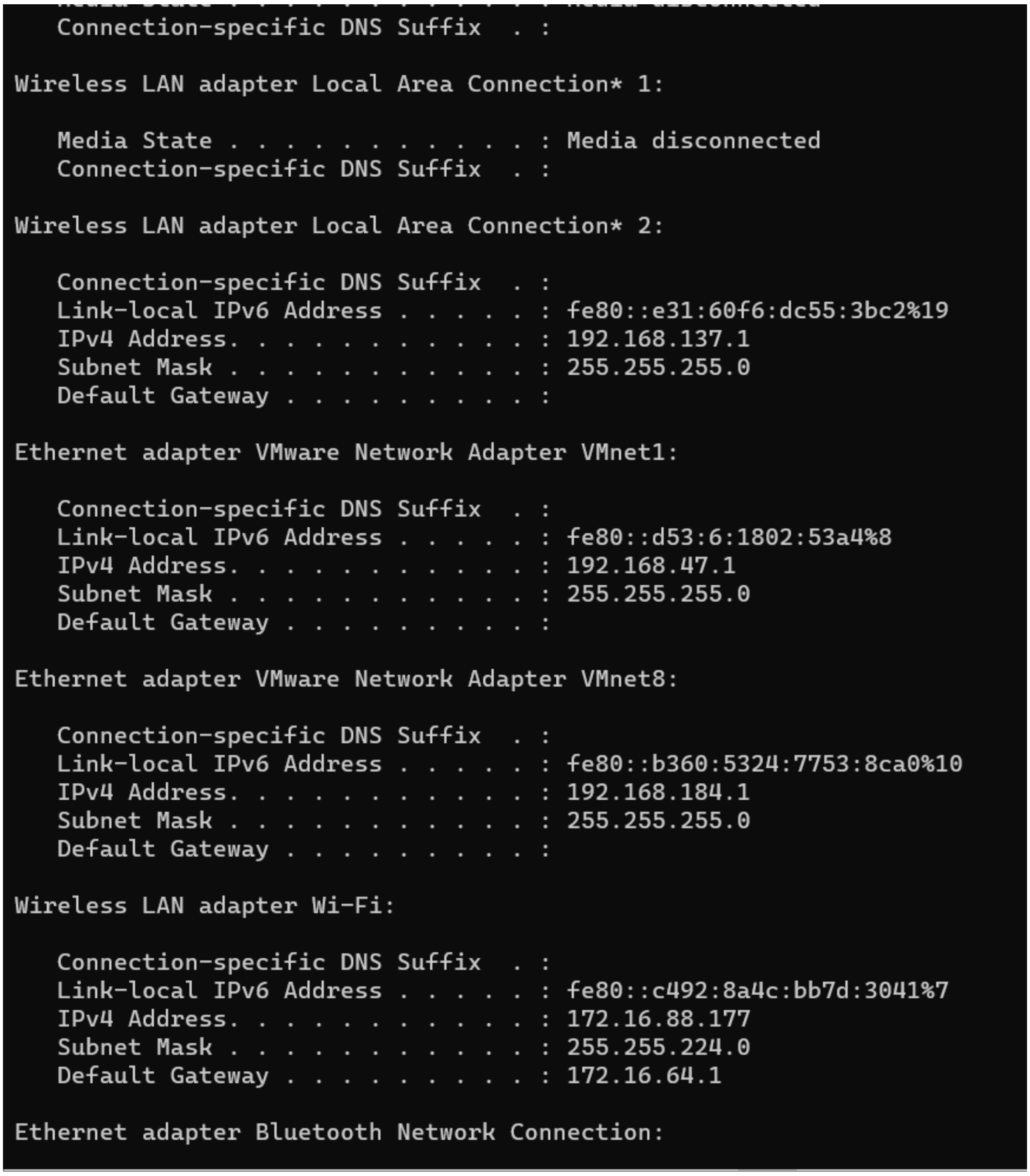
ASSIGNMENT 2

Q1. Find IP information assigned to your computer (IP address, DNS IP address, Gateway IP address)

Windows: ipconfig



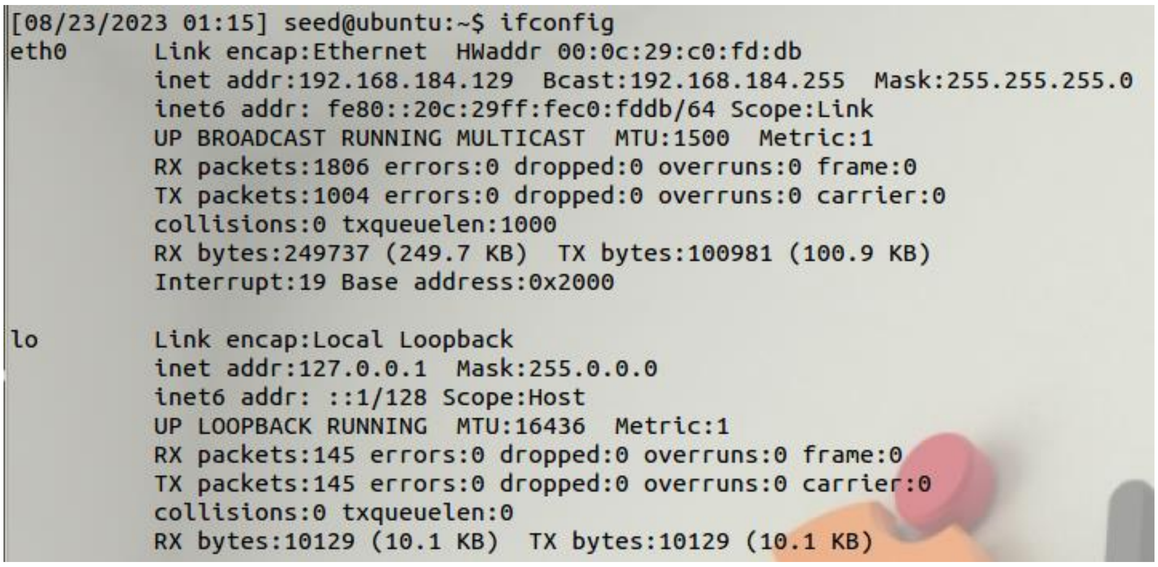
```
(base) PS C:\Users\shree> ipconfig

Windows IP Configuration


Unknown adapter Local Area Connection 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::1ecd:bbe8:bfb2:e967%9
   IPv4 Address. . . . . . . . . . . : 192.168.56.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Unknown adapter Local Area Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 2:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::e31:60f6:dc55:3bc2%19
   IPv4 Address. . . . . . . . . . . : 192.168.137.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Ethernet adapter VMware Network Adapter VMnet1:

   Connection-specific DNS Suffix  . :
```



```
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 2:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::e31:60f6:dc55:3bc2%19
   IPv4 Address. . . . . . . . . . . : 192.168.137.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Ethernet adapter VMware Network Adapter VMnet1:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::d53:6:1802:53a4%8
   IPv4 Address. . . . . . . . . . . : 192.168.47.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Ethernet adapter VMware Network Adapter VMnet8:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::b360:5324:7753:8ca0%10
   IPv4 Address. . . . . . . . . . . : 192.168.184.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::c492:8a4c:bb7d:3041%7
   IPv4 Address. . . . . . . . . . . : 172.16.88.177
   Subnet Mask . . . . . . . . . . . : 255.255.224.0
   Default Gateway . . . . . . . . . : 172.16.64.1

Ethernet adapter Bluetooth Network Connection:
```

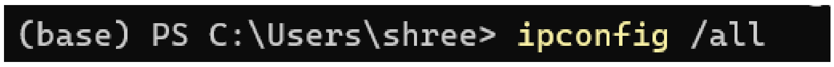
LINUX: ifconfig

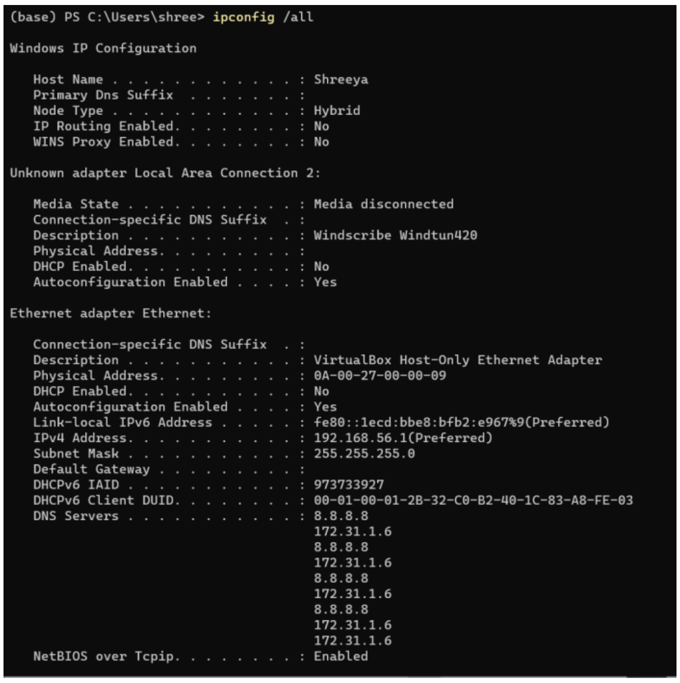
Name: SHREEYA CHATTERJI
Class: CO16


```
[08/23/2023 01:15] seed@ubuntu:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:c0:fd:db
          inet addr:192.168.184.129  Bcast:192.168.184.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fec0:fddb/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1806 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1004 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:249737 (249.7 KB)  TX bytes:100981 (100.9 KB)
          Interrupt:19 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:145 errors:0 dropped:0 overruns:0 frame:0
          TX packets:145 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:10129 (10.1 KB)  TX bytes:10129 (10.1 KB)
```

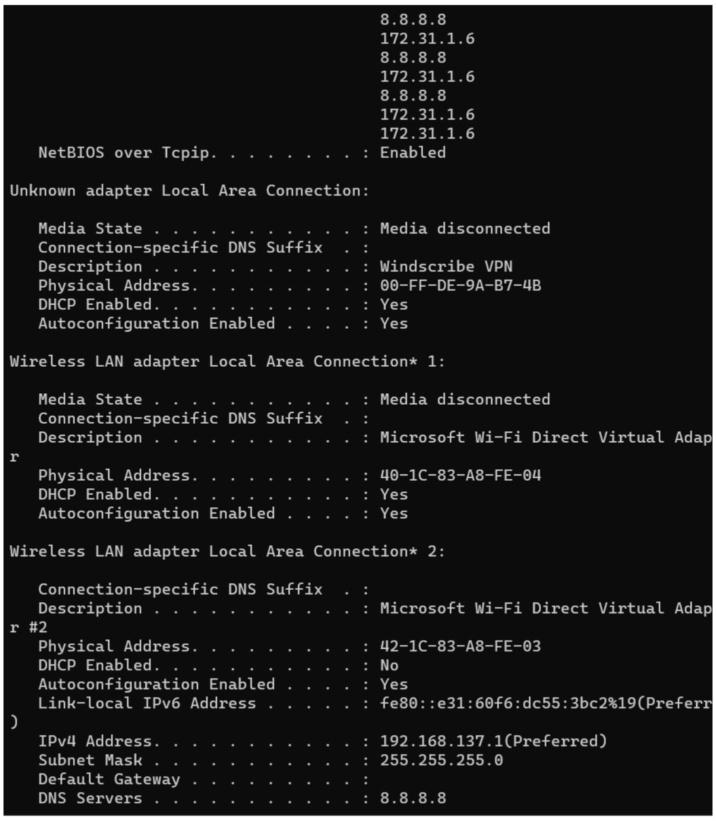
Q2. Check different interfaces connected to the network.

WINDOWS:


```
(base) PS C:\Users\shree> ipconfig /all
```

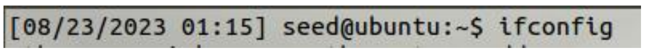

```
(base) PS C:\Users\shree> ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : Shreeya
   Primary Dns Suffix  . . . . . . . :
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No

Unknown adapter Local Area Connection 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Windscribe Windtun420
   Physical Address. . . . . . . . . :
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes

Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : VirtualBox Host-Only Ethernet Adapter
   Physical Address. . . . . . . . . : 0A-00-27-00-00-09
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::1ecd:bbe8:bfb2:e967%9(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.56.1(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
   DHCPv6 IAID . . . . . . . . . . . : 973733927
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-2B-32-C0-B2-40-1C-83-A8-FE-03
   DNS Servers . . . . . . . . . . . : 8.8.8.8
                                       172.31.1.6
                                       8.8.8.8
                                       172.31.1.6
                                       8.8.8.8
                                       172.31.1.6
                                       8.8.8.8
                                       172.31.1.6
                                       172.31.1.6
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

Name: SHREEYA CHATTERJI
Class: CO16

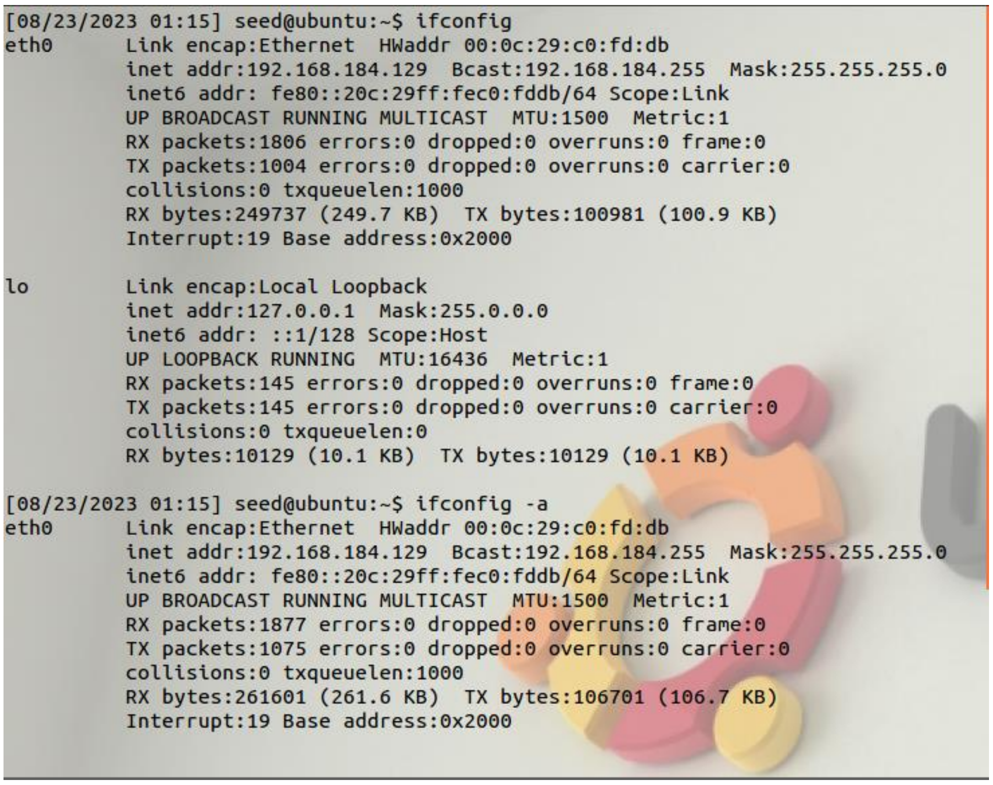


```
                                       8.8.8.8
                                       172.31.1.6
                                       8.8.8.8
                                       172.31.1.6
                                       8.8.8.8
                                       172.31.1.6
                                       172.31.1.6
   NetBIOS over Tcpip. . . . . . . . : Enabled

Unknown adapter Local Area Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Windscribe VPN
   Physical Address. . . . . . . . . : 00-FF-DE-9A-B7-4B
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Wireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual Adap
r
   Physical Address. . . . . . . . . : 40-1C-83-A8-FE-04
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Wireless LAN adapter Local Area Connection* 2:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual Adap
r #2
   Physical Address. . . . . . . . . : 42-1C-83-A8-FE-03
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::e31:60f6:dc55:3bc2%19(Preferr
)
   IPv4 Address. . . . . . . . . . . : 192.168.137.1(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
   DNS Servers . . . . . . . . . . . : 8.8.8.8
```

LINUX:



```
[08/23/2023 01:15] seed@ubuntu:~$ ifconfig
```



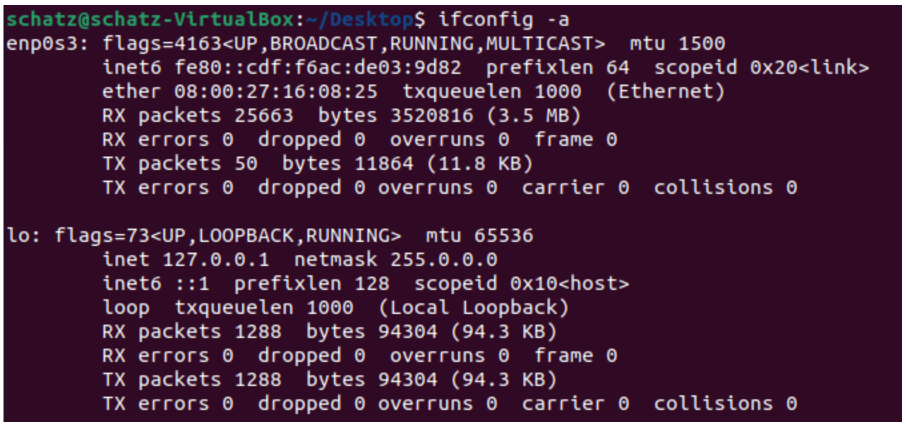
```
[08/23/2023 01:15] seed@ubuntu:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:c0:fd:db
          inet addr:192.168.184.129  Bcast:192.168.184.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fec0:fddb/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1806 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1004 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:249737 (249.7 KB)  TX bytes:100981 (100.9 KB)
          Interrupt:19 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:145 errors:0 dropped:0 overruns:0 frame:0
          TX packets:145 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:10129 (10.1 KB)  TX bytes:10129 (10.1 KB)

[08/23/2023 01:15] seed@ubuntu:~$ ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:0c:29:c0:fd:db
          inet addr:192.168.184.129  Bcast:192.168.184.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fec0:fddb/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1877 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1075 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:261601 (261.6 KB)  TX bytes:106701 (106.7 KB)
          Interrupt:19 Base address:0x2000
```

Name: SHREEYA CHATTERJI
Class: CO16

Q3. Alter the status of any interface

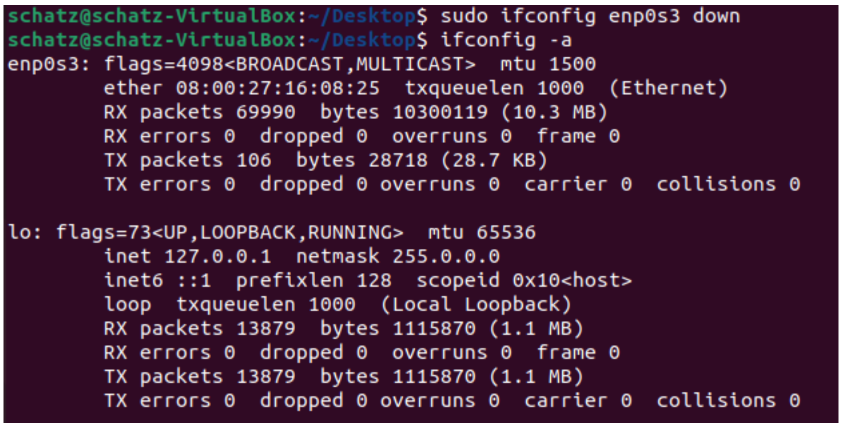
- **Set it to Offline mode**
- **Set it again to Online mode**

Before down:



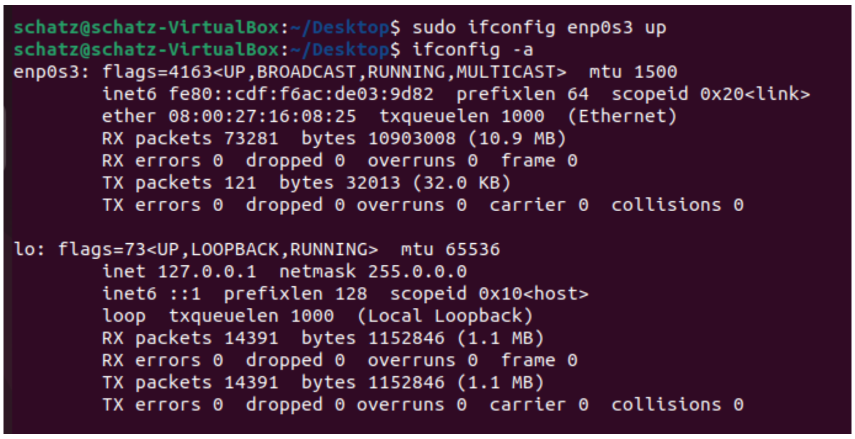
```
schatz@schatz-VirtualBox:~/Desktop$ ifconfig -a
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::cdf:f6ac:de03:9d82  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:16:08:25  txqueuelen 1000  (Ethernet)
        RX packets 25663  bytes 3520816 (3.5 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 50  bytes 11864 (11.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 1288  bytes 94304 (94.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1288  bytes 94304 (94.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

After running: **sudo ifconfig enp0s3 down**



```
schatz@schatz-VirtualBox:~/Desktop$ sudo ifconfig enp0s3 down
schatz@schatz-VirtualBox:~/Desktop$ ifconfig -a
enp0s3: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether 08:00:27:16:08:25  txqueuelen 1000  (Ethernet)
        RX packets 69990  bytes 10300119 (10.3 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 106  bytes 28718 (28.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 13879  bytes 1115870 (1.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13879  bytes 1115870 (1.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

RECONNECTING: **sudo ifconfig enp0s3 up**

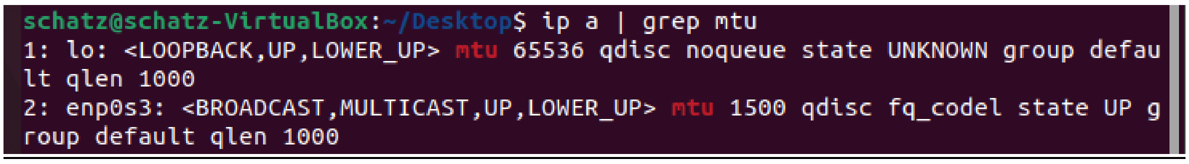


```
schatz@schatz-VirtualBox:~/Desktop$ sudo ifconfig enp0s3 up
schatz@schatz-VirtualBox:~/Desktop$ ifconfig -a
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::cdf:f6ac:de03:9d82  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:16:08:25  txqueuelen 1000  (Ethernet)
        RX packets 73281  bytes 10903008 (10.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 121  bytes 32013 (32.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 14391  bytes 1152846 (1.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 14391  bytes 1152846 (1.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Q4. Change the MTU size of a packet

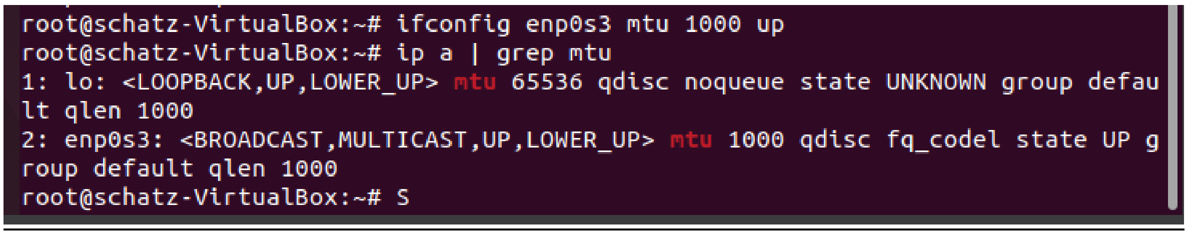
Checking the current MTU sizes: **ip a | grep mtu**

Name: SHREEYA CHATTERJI
Class: CO16



```
schatz@schatz-VirtualBox:~/Desktop$ ip a | grep mtu
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defau
lt qlen 1000
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP g
roup default qlen 1000
```

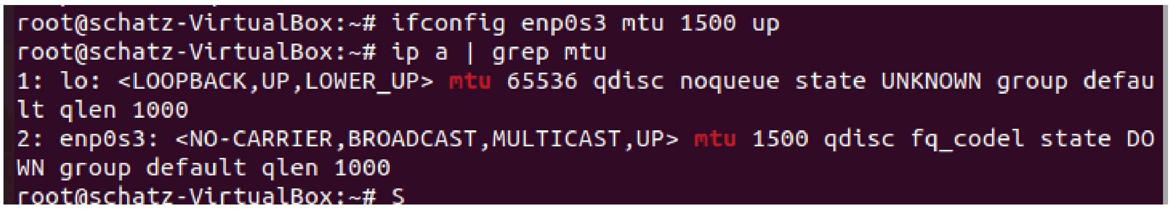
After running: **ifconfig enp0s3 mtu 1000 up**



```
root@schatz-VirtualBox:~# ifconfig enp0s3 mtu 1000 up
root@schatz-VirtualBox:~# ip a | grep mtu
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defau
lt qlen 1000
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1000 qdisc fq_codel state UP g
roup default qlen 1000
root@schatz-VirtualBox:~# S
```

The MTU of enp0s3 changes from 1500 to 1000.

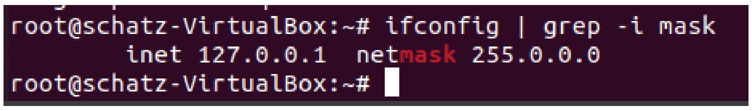
Now reset it to default:



```
root@schatz-VirtualBox:~# ifconfig enp0s3 mtu 1500 up
root@schatz-VirtualBox:~# ip a | grep mtu
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defau
lt qlen 1000
2: enp0s3: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc fq_codel state DO
WN group default qlen 1000
root@schatz-VirtualBox:~# S
```

Q5. Find the subnet mask

The subnet mask can be found out as: ifconfig | grep -i mask



```
root@schatz-VirtualBox:~# ifconfig | grep -i mask
        inet 127.0.0.1  netmask 255.0.0.0
root@schatz-VirtualBox:~#
```

Q6. Calculate number of host ids in your subnet

Subnet mask: 255.0.0.0

Binary Representation: 11111111.00000000.00000000.00000000

Number of zeroes in the binary representation: 24

Number of unique host addresses: 2^24

Actual number of hosts within this subnet= **2^24-2** (we subtract 2 to account for the network and broadcast addresses which are reserved)

Value= **16,777,214-** usable host ids

Q7. Find the network and broadcast address of the network you are connected to.

Subnet mask: 255.0.0.0

Name: SHREEYA CHATTERJI
Class: CO16

Binary Representation: 11111111.00000000.00000000.00000000

Flipped Binary Representation: 00000000.11111111.11111111.11111111

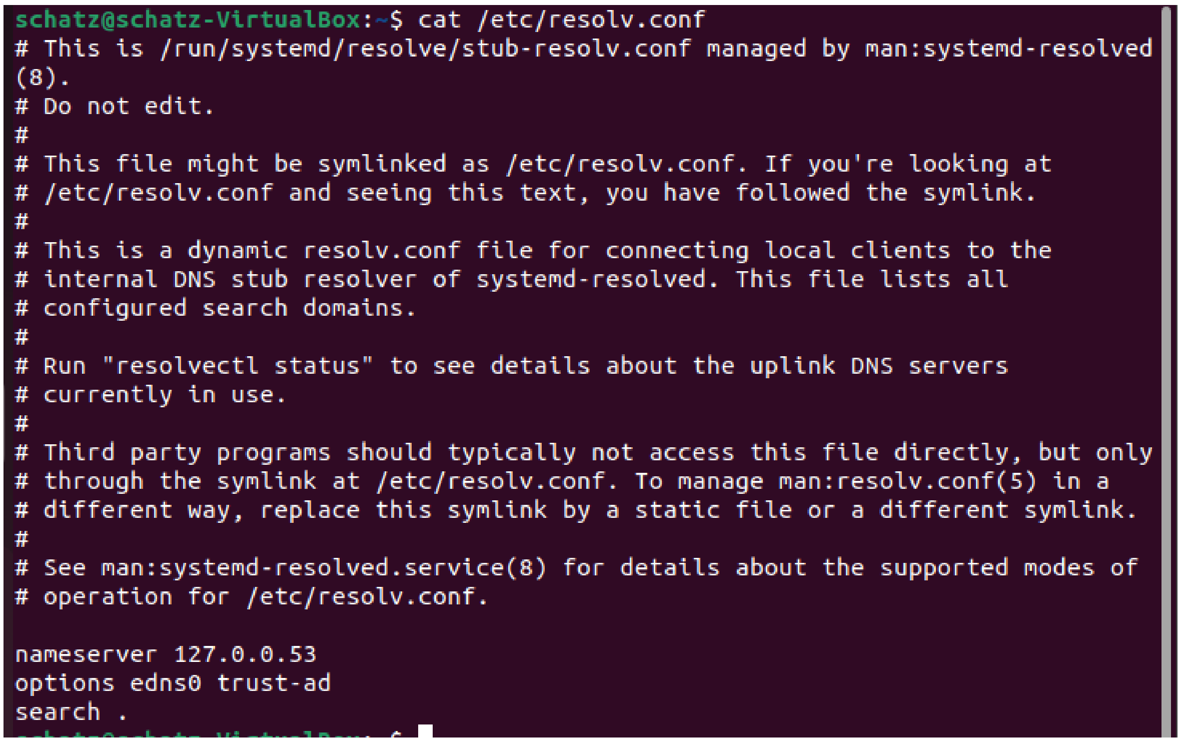
Flipped decimal: **0.255.255.255**

Thus **0.255.255.255** is the Broadcast address.

Q8.

a) Ping the DNS server and check the response.

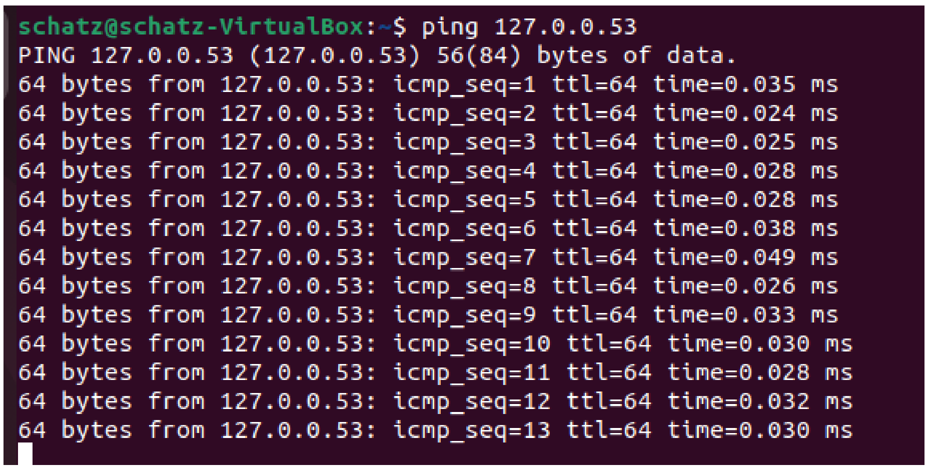
SERVER INFORMATION:



```
schatz@schatz-VirtualBox:~$ cat /etc/resolv.conf
# This is /run/systemd/resolve/stub-resolv.conf managed by man:systemd-resolved
(8).
# Do not edit.
#
# This file might be symlinked as /etc/resolv.conf. If you're looking at
# /etc/resolv.conf and seeing this text, you have followed the symlink.
#
# This is a dynamic resolv.conf file for connecting local clients to the
# internal DNS stub resolver of systemd-resolved. This file lists all
# configured search domains.
#
# Run "resolvectl status" to see details about the uplink DNS servers
# currently in use.
#
# Third party programs should typically not access this file directly, but only
# through the symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a
# different way, replace this symlink by a static file or a different symlink.
#
# See man:systemd-resolved.service(8) for details about the supported modes of
# operation for /etc/resolv.conf.

nameserver 127.0.0.53
options edns0 trust-ad
search .
```

Thus the DNS Server IP Address is 127.0.0.53

After Ping:

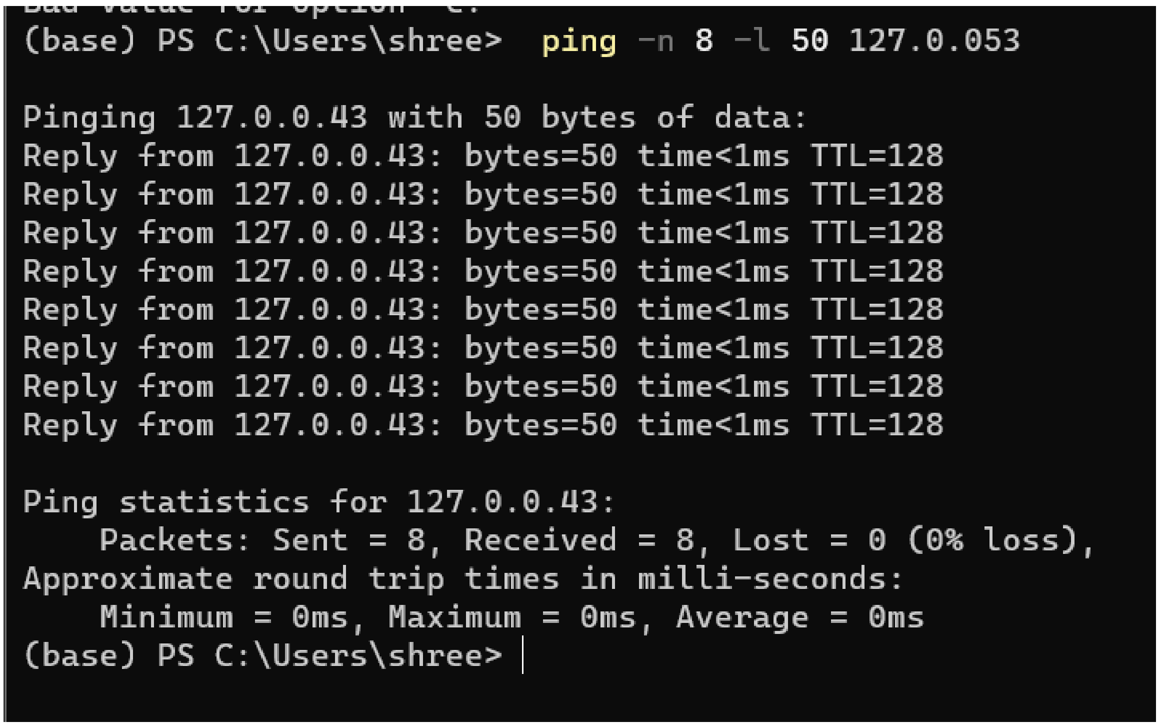


```
schatz@schatz-VirtualBox:~$ ping 127.0.0.53
PING 127.0.0.53 (127.0.0.53) 56(84) bytes of data.
64 bytes from 127.0.0.53: icmp_seq=1 ttl=64 time=0.035 ms
64 bytes from 127.0.0.53: icmp_seq=2 ttl=64 time=0.024 ms
64 bytes from 127.0.0.53: icmp_seq=3 ttl=64 time=0.025 ms
64 bytes from 127.0.0.53: icmp_seq=4 ttl=64 time=0.028 ms
64 bytes from 127.0.0.53: icmp_seq=5 ttl=64 time=0.028 ms
64 bytes from 127.0.0.53: icmp_seq=6 ttl=64 time=0.038 ms
64 bytes from 127.0.0.53: icmp_seq=7 ttl=64 time=0.049 ms
64 bytes from 127.0.0.53: icmp_seq=8 ttl=64 time=0.026 ms
64 bytes from 127.0.0.53: icmp_seq=9 ttl=64 time=0.033 ms
64 bytes from 127.0.0.53: icmp_seq=10 ttl=64 time=0.030 ms
64 bytes from 127.0.0.53: icmp_seq=11 ttl=64 time=0.028 ms
64 bytes from 127.0.0.53: icmp_seq=12 ttl=64 time=0.032 ms
64 bytes from 127.0.0.53: icmp_seq=13 ttl=64 time=0.030 ms
```

Name: SHREEYA CHATTERJI
Class: CO16

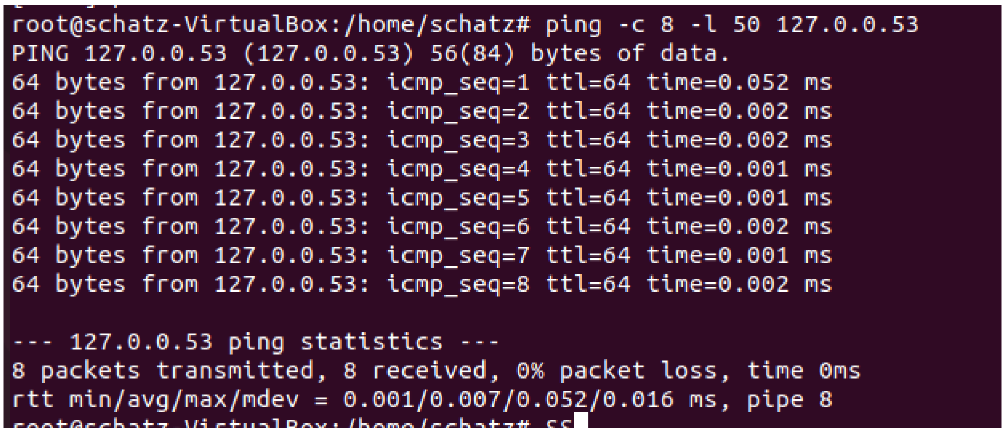
b) Ping 8 echo requests with each packet of size 50 to any domain name/IP address and check the response.

WINDOWS:



```
(base) PS C:\Users\shree>  ping -n 8 -l 50 127.0.053

Pinging 127.0.0.43 with 50 bytes of data:
Reply from 127.0.0.43: bytes=50 time<1ms TTL=128
Reply from 127.0.0.43: bytes=50 time<1ms TTL=128
Reply from 127.0.0.43: bytes=50 time<1ms TTL=128
Reply from 127.0.0.43: bytes=50 time<1ms TTL=128
Reply from 127.0.0.43: bytes=50 time<1ms TTL=128
Reply from 127.0.0.43: bytes=50 time<1ms TTL=128
Reply from 127.0.0.43: bytes=50 time<1ms TTL=128
Reply from 127.0.0.43: bytes=50 time<1ms TTL=128

Ping statistics for 127.0.0.43:
    Packets: Sent = 8, Received = 8, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
(base) PS C:\Users\shree>
```

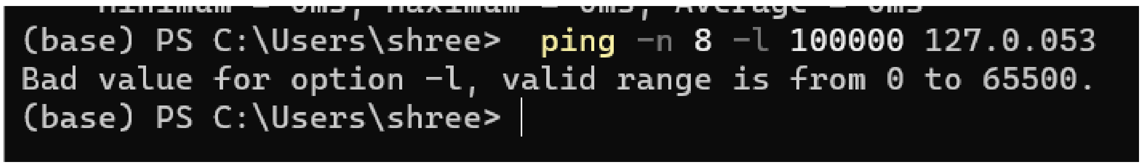
LINUX:



```
root@schatz-VirtualBox:/home/schatz# ping -c 8 -l 50 127.0.0.53
PING 127.0.0.53 (127.0.0.53) 56(84) bytes of data.
64 bytes from 127.0.0.53: icmp_seq=1 ttl=64 time=0.052 ms
64 bytes from 127.0.0.53: icmp_seq=2 ttl=64 time=0.002 ms
64 bytes from 127.0.0.53: icmp_seq=3 ttl=64 time=0.002 ms
64 bytes from 127.0.0.53: icmp_seq=4 ttl=64 time=0.001 ms
64 bytes from 127.0.0.53: icmp_seq=5 ttl=64 time=0.001 ms
64 bytes from 127.0.0.53: icmp_seq=6 ttl=64 time=0.002 ms
64 bytes from 127.0.0.53: icmp_seq=7 ttl=64 time=0.001 ms
64 bytes from 127.0.0.53: icmp_seq=8 ttl=64 time=0.002 ms

--- 127.0.0.53 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.001/0.007/0.052/0.016 ms, pipe 8
```

c) Ping 8 echo requests with each packet of size 100000 to any domain name/IP address and check the response

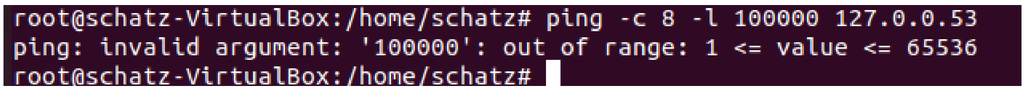
WINDOWS:



```
(base) PS C:\Users\shree>  ping -n 8 -l 100000 127.0.053
Bad value for option -l, valid range is from 0 to 65500.
(base) PS C:\Users\shree>
```

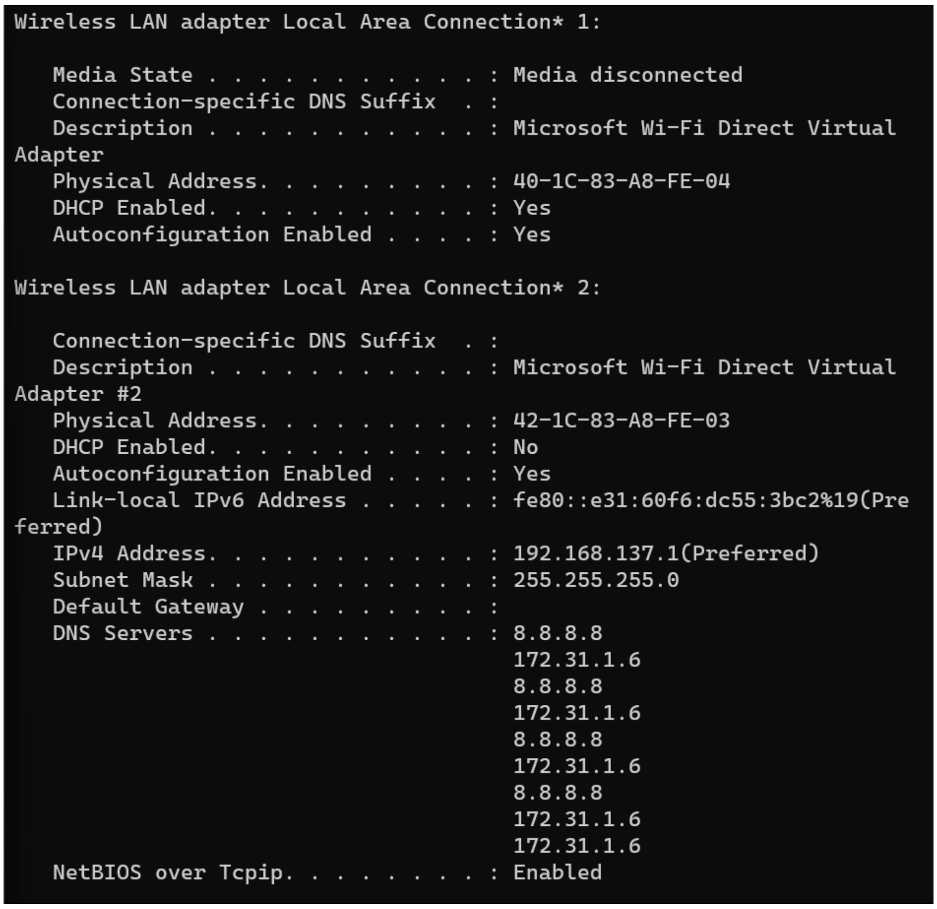
LINUX:

Name: SHREEYA CHATTERJI
Class: CO16



```
root@schatz-VirtualBox:/home/schatz# ping -c 8 -l 100000 127.0.0.53
ping: invalid argument: '100000': out of range: 1 <= value <= 65536
root@schatz-VirtualBox:/home/schatz#
```

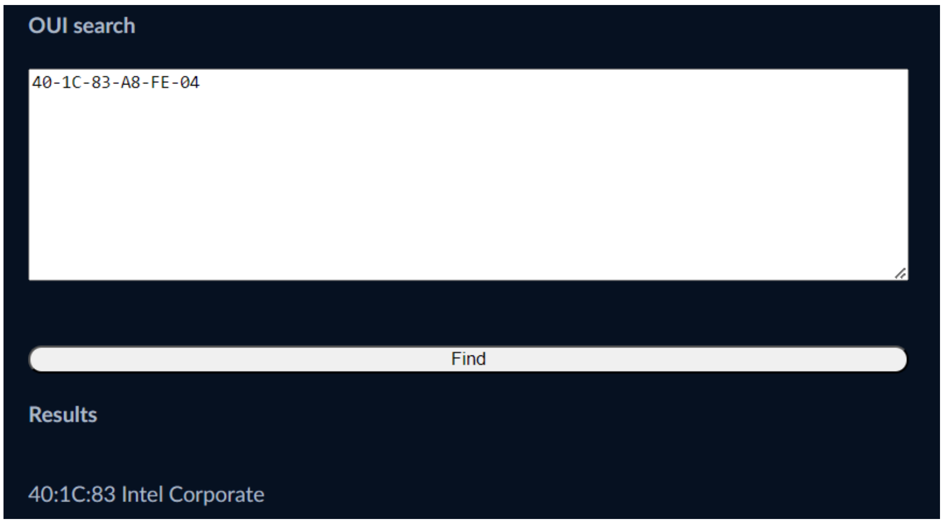
Q9. Find the MAC address of your device. Retrieve both the Manufacturer Id (Manufacturer as well) and the device ID.



```
Wireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual
Adapter
   Physical Address. . . . . . . . . : 40-1C-83-A8-FE-04
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Wireless LAN adapter Local Area Connection* 2:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual
Adapter #2
   Physical Address. . . . . . . . . : 42-1C-83-A8-FE-03
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::e31:60f6:dc55:3bc2%19(Pre
ferred)
   IPv4 Address. . . . . . . . . . . : 192.168.137.1(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
   DNS Servers . . . . . . . . . . . : 8.8.8.8
                                       172.31.1.6
                                       8.8.8.8
                                       172.31.1.6
                                       8.8.8.8
                                       172.31.1.6
                                       8.8.8.8
                                       172.31.1.6
                                       172.31.1.6
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

MAC Address: 40-1C-83-A8-FE-04

In OUI LOOKUP:



OUI search

```
40-1C-83-A8-FE-04
```

Results

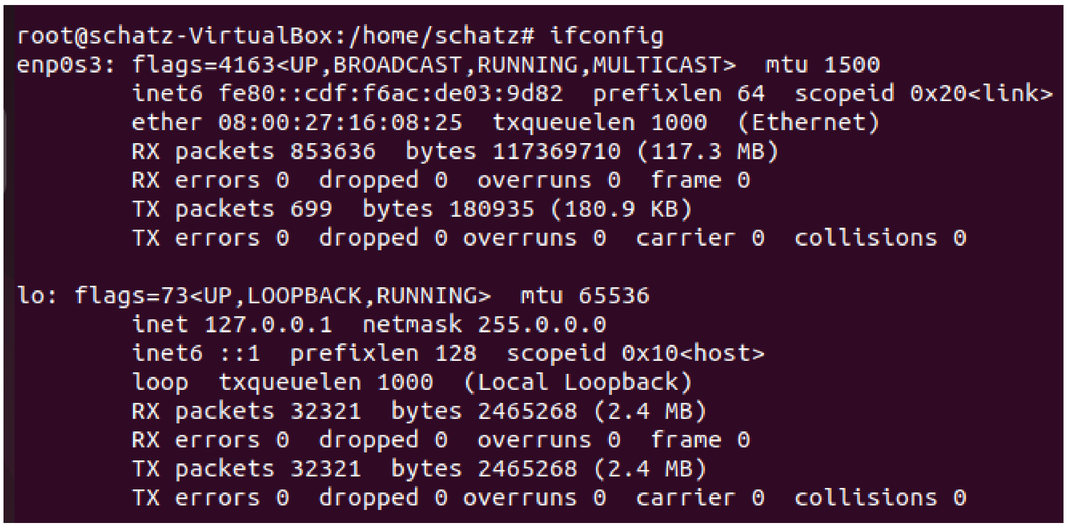
40:1C:83 Intel Corporate

Name: SHREEYA CHATTERJI
Class: CO16

Q10. What is the Private and Public IP address assigned for your current connection.

PRIVATE:

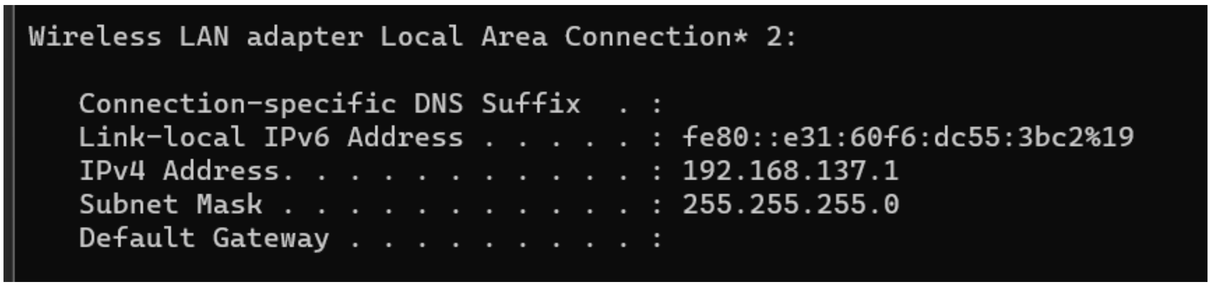
For the UBUNTU VM:



```
root@schatz-VirtualBox:/home/schatz# ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::cdf:f6ac:de03:9d82  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:16:08:25  txqueuelen 1000  (Ethernet)
        RX packets 853636  bytes 117369710 (117.3 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 699  bytes 180935 (180.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 32321  bytes 2465268 (2.4 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 32321  bytes 2465268 (2.4 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Then the private ip: 127.0.0.1

For the Host OS: WINDOWS:

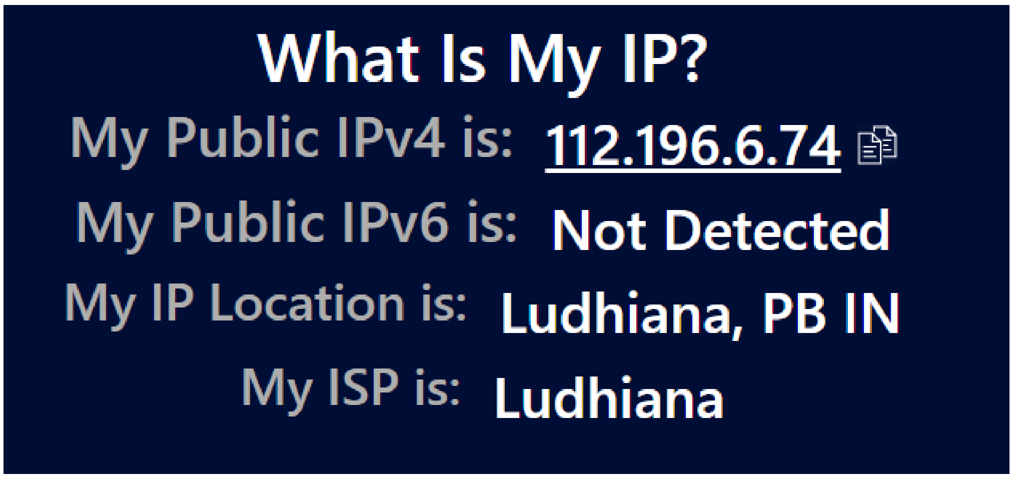


```
Wireless LAN adapter Local Area Connection* 2:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::e31:60f6:dc55:3bc2%19
   IPv4 Address. . . . . . . . . . . : 192.168.137.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
```

Then the private ip: 192.168.137.1

PUBLIC: 112.169.6.74

whatismyip.com



What Is My IP?

My Public IPv4 is: 112.196.6.74

My Public IPv6 is: Not Detected

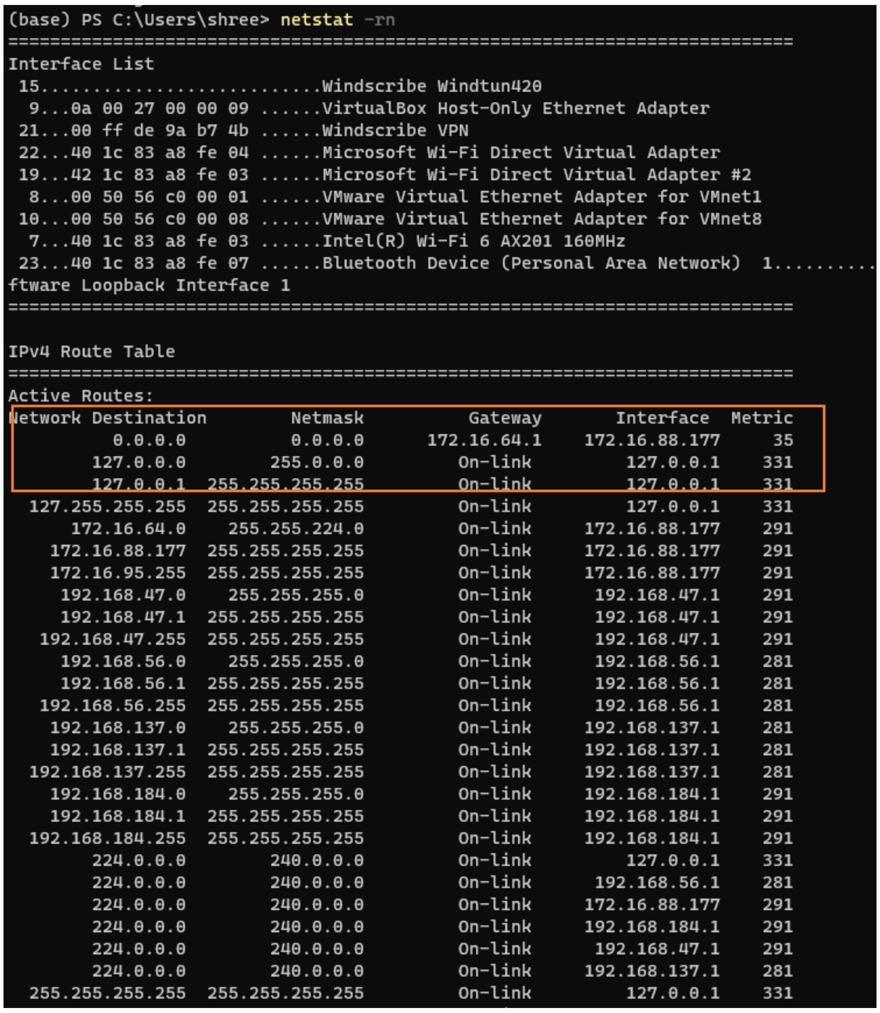
My IP Location is: Ludhiana, PB IN

My ISP is: Ludhiana

Name: SHREEYA CHATTERJI
Class: CO16

Q11. How can you redirect the traffic through a particular gateway?

Checking the current routes:

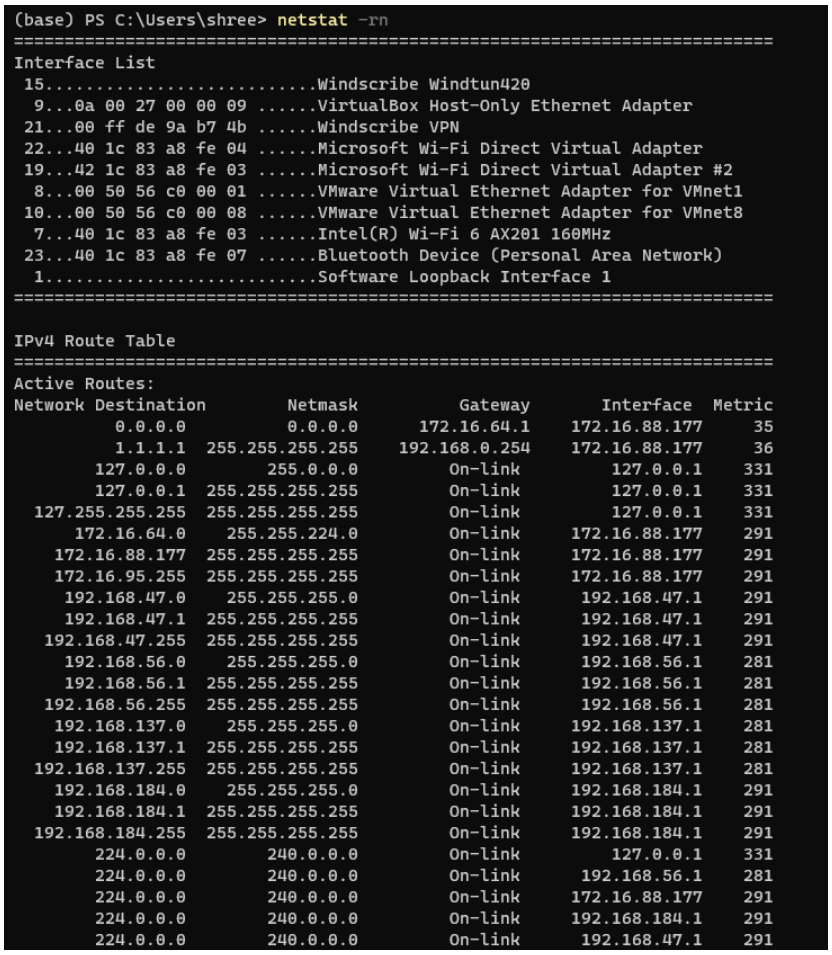


```
(base) PS C:\Users\shree> netstat -rn
===========================================================================
Interface List
 15...........................Windscribe Windtun420
  9...0a 00 27 00 00 09 ......VirtualBox Host-Only Ethernet Adapter
 21...00 ff de 9a b7 4b ......Windscribe VPN
 22...40 1c 83 a8 fe 04 ......Microsoft Wi-Fi Direct Virtual Adapter
 19...42 1c 83 a8 fe 03 ......Microsoft Wi-Fi Direct Virtual Adapter #2
  8...00 50 56 c0 00 01 ......VMware Virtual Ethernet Adapter for VMnet1
 10...00 50 56 c0 00 08 ......VMware Virtual Ethernet Adapter for VMnet8
  7...40 1c 83 a8 fe 03 ......Intel(R) Wi-Fi 6 AX201 160MHz
 23...40 1c 83 a8 fe 07 ......Bluetooth Device (Personal Area Network)  1.........
ftware Loopback Interface 1
===========================================================================

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0      172.16.64.1    172.16.88.177     35
        127.0.0.0        255.0.0.0         On-link         127.0.0.1    331
        127.0.0.1  255.255.255.255         On-link         127.0.0.1    331
  127.255.255.255  255.255.255.255         On-link         127.0.0.1    331
      172.16.64.0    255.255.224.0         On-link     172.16.88.177    291
    172.16.88.177  255.255.255.255         On-link     172.16.88.177    291
    172.16.95.255  255.255.255.255         On-link     172.16.88.177    291
     192.168.47.0    255.255.255.0         On-link      192.168.47.1    291
     192.168.47.1  255.255.255.255         On-link      192.168.47.1    291
   192.168.47.255  255.255.255.255         On-link      192.168.47.1    291
     192.168.56.0    255.255.255.0         On-link      192.168.56.1    281
     192.168.56.1  255.255.255.255         On-link      192.168.56.1    281
   192.168.56.255  255.255.255.255         On-link      192.168.56.1    281
    192.168.137.0    255.255.255.0         On-link     192.168.137.1    281
    192.168.137.1  255.255.255.255         On-link     192.168.137.1    281
  192.168.137.255  255.255.255.255         On-link     192.168.137.1    281
    192.168.184.0    255.255.255.0         On-link     192.168.184.1    291
    192.168.184.1  255.255.255.255         On-link     192.168.184.1    291
  192.168.184.255  255.255.255.255         On-link     192.168.184.1    291
        224.0.0.0        240.0.0.0         On-link         127.0.0.1    331
        224.0.0.0        240.0.0.0         On-link      192.168.56.1    281
        224.0.0.0        240.0.0.0         On-link     172.16.88.177    291
        224.0.0.0        240.0.0.0         On-link     192.168.184.1    291
        224.0.0.0        240.0.0.0         On-link      192.168.47.1    291
        224.0.0.0        240.0.0.0         On-link     192.168.137.1    281
  255.255.255.255  255.255.255.255         On-link         127.0.0.1    331
```

Here 1.1.1.1 does not exist

Adding a static route for example:

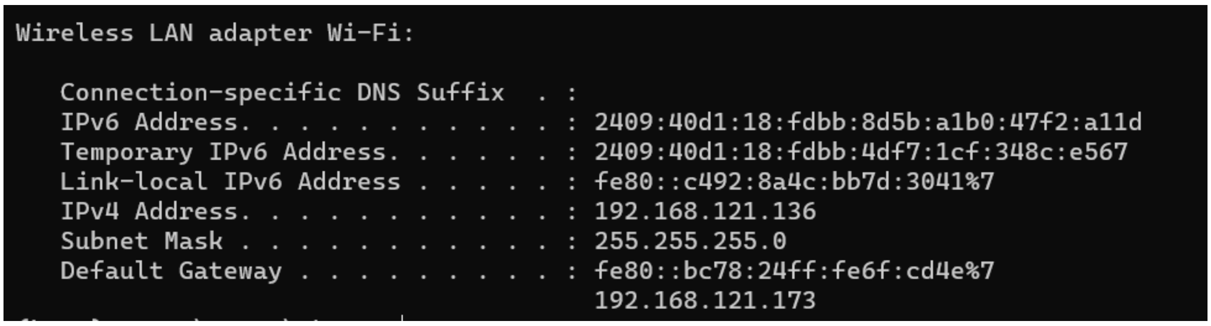
Name: SHREEYA CHATTERJI
Class: CO16



```
(base) PS C:\Users\shree> netstat -rn
===========================================================================
Interface List
 15...........................Windscribe Windtun420
  9...0a 00 27 00 00 09 ......VirtualBox Host-Only Ethernet Adapter
 21...00 ff de 9a b7 4b ......Windscribe VPN
 22...40 1c 83 a8 fe 04 ......Microsoft Wi-Fi Direct Virtual Adapter
 19...42 1c 83 a8 fe 03 ......Microsoft Wi-Fi Direct Virtual Adapter #2
  8...00 50 56 c0 00 01 ......VMware Virtual Ethernet Adapter for VMnet1
 10...00 50 56 c0 00 08 ......VMware Virtual Ethernet Adapter for VMnet8
  7...40 1c 83 a8 fe 03 ......Intel(R) Wi-Fi 6 AX201 160MHz
 23...40 1c 83 a8 fe 07 ......Bluetooth Device (Personal Area Network)
  1...........................Software Loopback Interface 1
===========================================================================

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0      172.16.64.1    172.16.88.177     35
          1.1.1.1  255.255.255.255    192.168.0.254    172.16.88.177     36
        127.0.0.0        255.0.0.0         On-link         127.0.0.1    331
        127.0.0.1  255.255.255.255         On-link         127.0.0.1    331
  127.255.255.255  255.255.255.255         On-link         127.0.0.1    331
      172.16.64.0    255.255.224.0         On-link     172.16.88.177    291
    172.16.88.177  255.255.255.255         On-link     172.16.88.177    291
    172.16.95.255  255.255.255.255         On-link     172.16.88.177    291
     192.168.47.0    255.255.255.0         On-link      192.168.47.1    291
     192.168.47.1  255.255.255.255         On-link      192.168.47.1    291
   192.168.47.255  255.255.255.255         On-link      192.168.47.1    291
     192.168.56.0    255.255.255.0         On-link      192.168.56.1    281
     192.168.56.1  255.255.255.255         On-link      192.168.56.1    281
   192.168.56.255  255.255.255.255         On-link      192.168.56.1    281
    192.168.137.0    255.255.255.0         On-link     192.168.137.1    281
    192.168.137.1  255.255.255.255         On-link     192.168.137.1    281
  192.168.137.255  255.255.255.255         On-link     192.168.137.1    281
    192.168.184.0    255.255.255.0         On-link     192.168.184.1    291
    192.168.184.1  255.255.255.255         On-link     192.168.184.1    291
  192.168.184.255  255.255.255.255         On-link     192.168.184.1    291
        224.0.0.0        240.0.0.0         On-link         127.0.0.1    331
        224.0.0.0        240.0.0.0         On-link      192.168.56.1    281
        224.0.0.0        240.0.0.0         On-link     172.16.88.177    291
        224.0.0.0        240.0.0.0         On-link     192.168.184.1    291
        224.0.0.0        240.0.0.0         On-link      192.168.47.1    291
```

Q12. Look for the address of the default gateway and firewall.

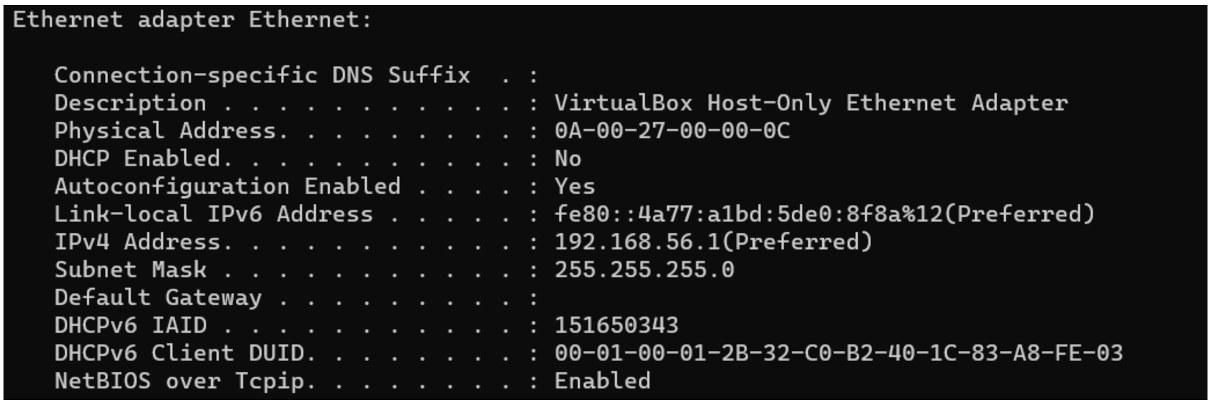
From **ipconfig**



```
Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : 2409:40d1:18:fdbb:8d5b:a1b0:47f2:a11d
   Temporary IPv6 Address. . . . . . : 2409:40d1:18:fdbb:4df7:1cf:348c:e567
   Link-local IPv6 Address . . . . . : fe80::c492:8a4c:bb7d:3041%7
   IPv4 Address. . . . . . . . . . . : 192.168.121.136
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::bc78:24ff:fe6f:cd4e%7
                                       192.168.121.173
```

Q13. Find the DHCP Server IP address (if enabled)

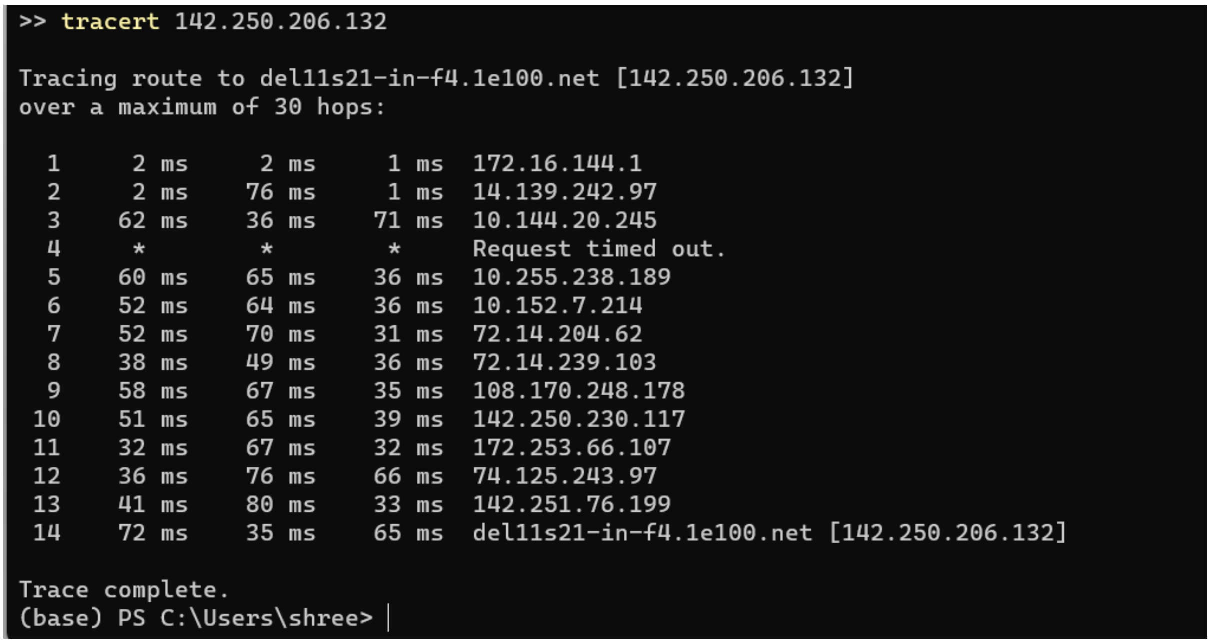
Code: **ipconfig /all**



```
Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : VirtualBox Host-Only Ethernet Adapter
   Physical Address. . . . . . . . . : 0A-00-27-00-00-0C
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::4a77:a1bd:5de0:8f8a%12(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.56.1(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
   DHCPv6 IAID . . . . . . . . . . . : 151650343
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-2B-32-C0-B2-40-1C-83-A8-FE-03
   NetBIOS over Tcpip. . . . . . . . : Enabled
```

Q14. Count number of hosts traveled by packet to reach the destination organization. What is the Ipaddress of second last host.

Command: **tracert <ip_address>**

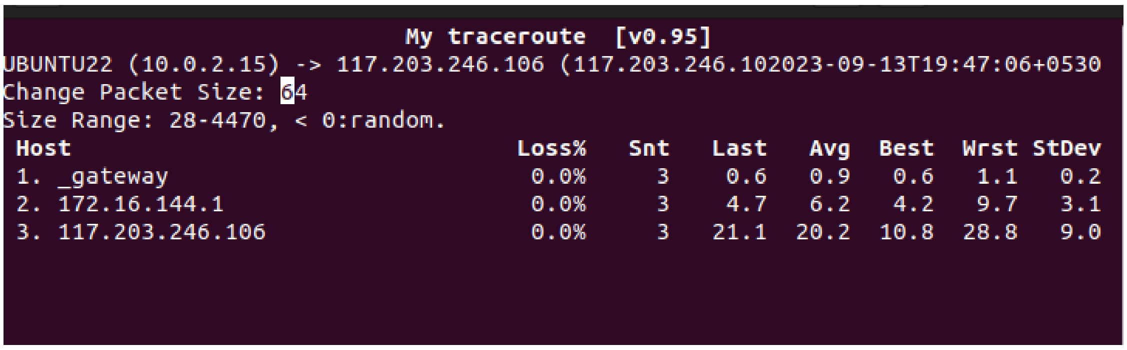


```
>> tracert 142.250.206.132

Tracing route to del11s21-in-f4.1e100.net [142.250.206.132]
over a maximum of 30 hops:

  1     2 ms     2 ms     1 ms  172.16.144.1
  2     2 ms    76 ms     1 ms  14.139.242.97
  3    62 ms    36 ms    71 ms  10.144.20.245
  4     *        *        *     Request timed out.
  5    60 ms    65 ms    36 ms  10.255.238.189
  6    52 ms    64 ms    36 ms  10.152.7.214
  7    52 ms    70 ms    31 ms  72.14.204.62
  8    38 ms    49 ms    36 ms  72.14.239.103
  9    58 ms    67 ms    35 ms  108.170.248.178
 10    51 ms    65 ms    39 ms  142.250.230.117
 11    32 ms    67 ms    32 ms  172.253.66.107
 12    36 ms    76 ms    66 ms  74.125.243.97
 13    41 ms    80 ms    33 ms  142.251.76.199
 14    72 ms    35 ms    65 ms  del11s21-in-f4.1e100.net [142.250.206.132]

Trace complete.
(base) PS C:\Users\shree> |
```

The IP address of the 2^nd^ last host is **142.251.76.199**

Q15. Generate a report of mtr command result (Use Ubuntu for this command) (Try to set different attributes for the same e.g. hops, number of pings etc.)

a)Generating the mtr report: Command: **mtr 117.203.246.106**

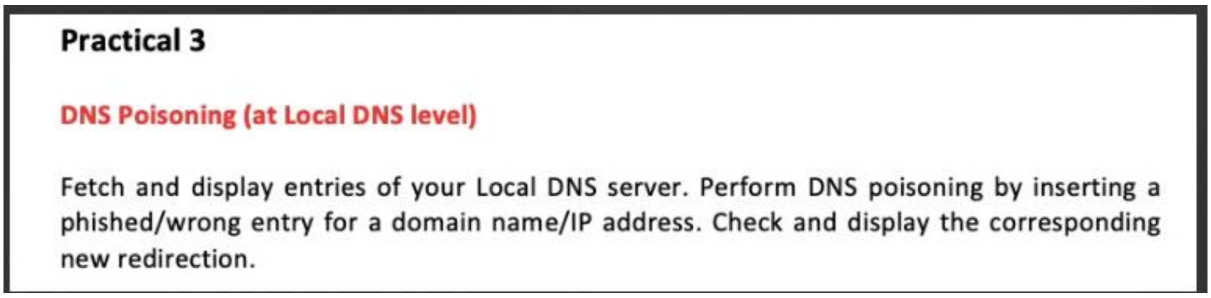


```
                         My traceroute  [v0.95]
UBUNTU22 (10.0.2.15) -> 117.203.246.106 (117.203.246.102023-09-13T19:47:06+0530
Change Packet Size: 64
Size Range: 28-4470, < 0:random.
 Host                                 Loss%   Snt   Last   Avg  Best  Wrst StDev
 1. _gateway                           0.0%     3    0.6   0.9   0.6   1.1   0.2
 2. 172.16.144.1                       0.0%     3    4.7   6.2   4.2   9.7   3.1
 3. 117.203.246.106                    0.0%     3   21.1  20.2  10.8  28.8   9.0
```

b)Generating the report with 10 hops and 5pings: Command: **mtr -h 10 -c 5 117.203.246.106**

Name: SHREEYA CHATTERJI
Class: CO16

LAB ASSIGNMENT 3



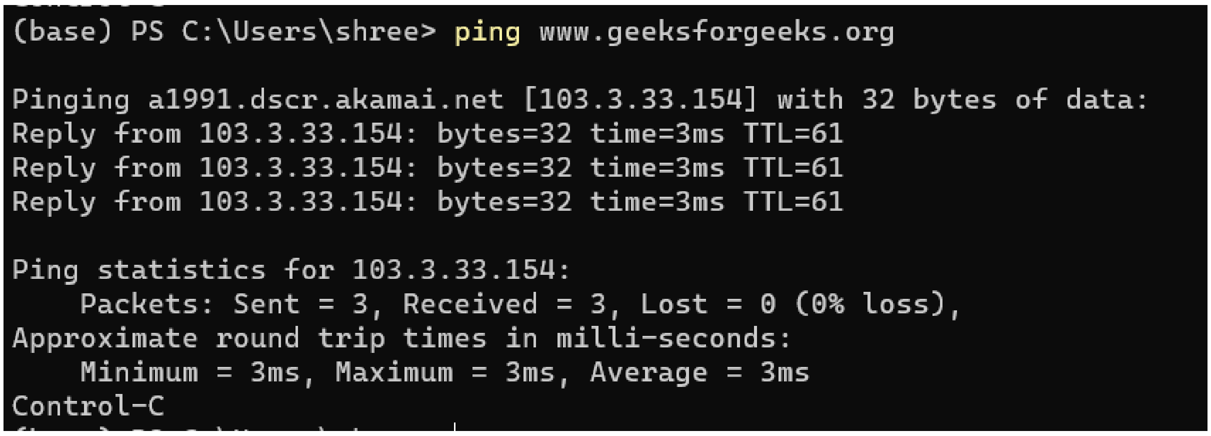
Practical 3

DNS Poisoning (at Local DNS level)

Fetch and display entries of your Local DNS server. Perform DNS poisoning by inserting a phished/wrong entry for a domain name/IP address. Check and display the corresponding new redirection.

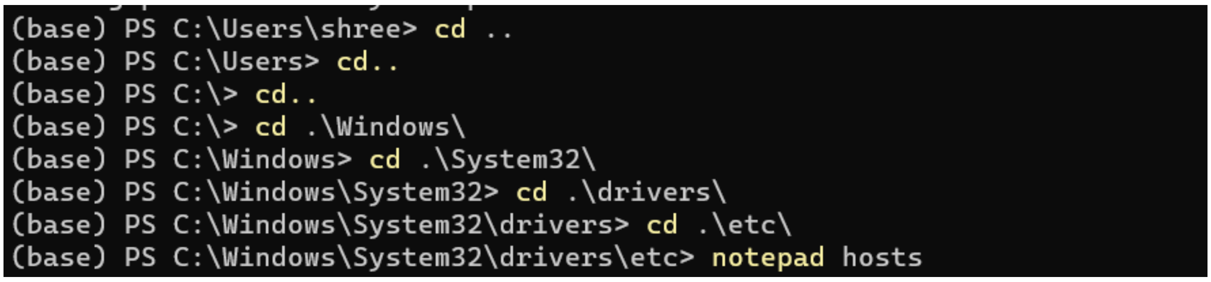
REDIRECTING WWW.PUNJABIUNIVERSITY.AC.IN TO THAPAR.EDU

Finding the IPADDRESS of the website TO which we want to redirect:



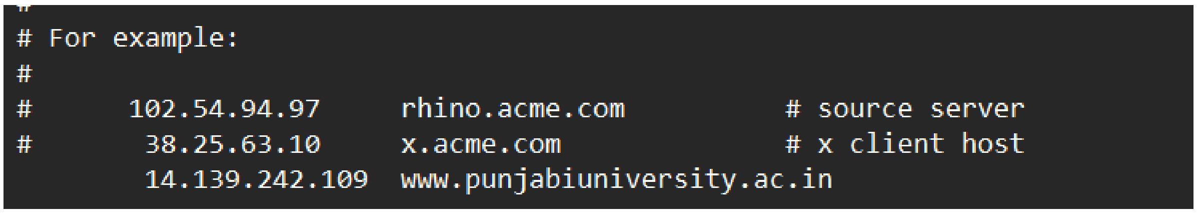
```
(base) PS C:\Users\shree> ping www.geeksforgeeks.org

Pinging a1991.dscr.akamai.net [103.3.33.154] with 32 bytes of data:
Reply from 103.3.33.154: bytes=32 time=3ms TTL=61
Reply from 103.3.33.154: bytes=32 time=3ms TTL=61
Reply from 103.3.33.154: bytes=32 time=3ms TTL=61

Ping statistics for 103.3.33.154:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 3ms, Average = 3ms
Control-C
```

Going to the hosts file with administrator access:



```
(base) PS C:\Users\shree> cd ..
(base) PS C:\Users> cd..
(base) PS C:\> cd..
(base) PS C:\> cd .\Windows\
(base) PS C:\Windows> cd .\System32\
(base) PS C:\Windows\System32> cd .\drivers\
(base) PS C:\Windows\System32\drivers> cd .\etc\
(base) PS C:\Windows\System32\drivers\etc> notepad hosts
```

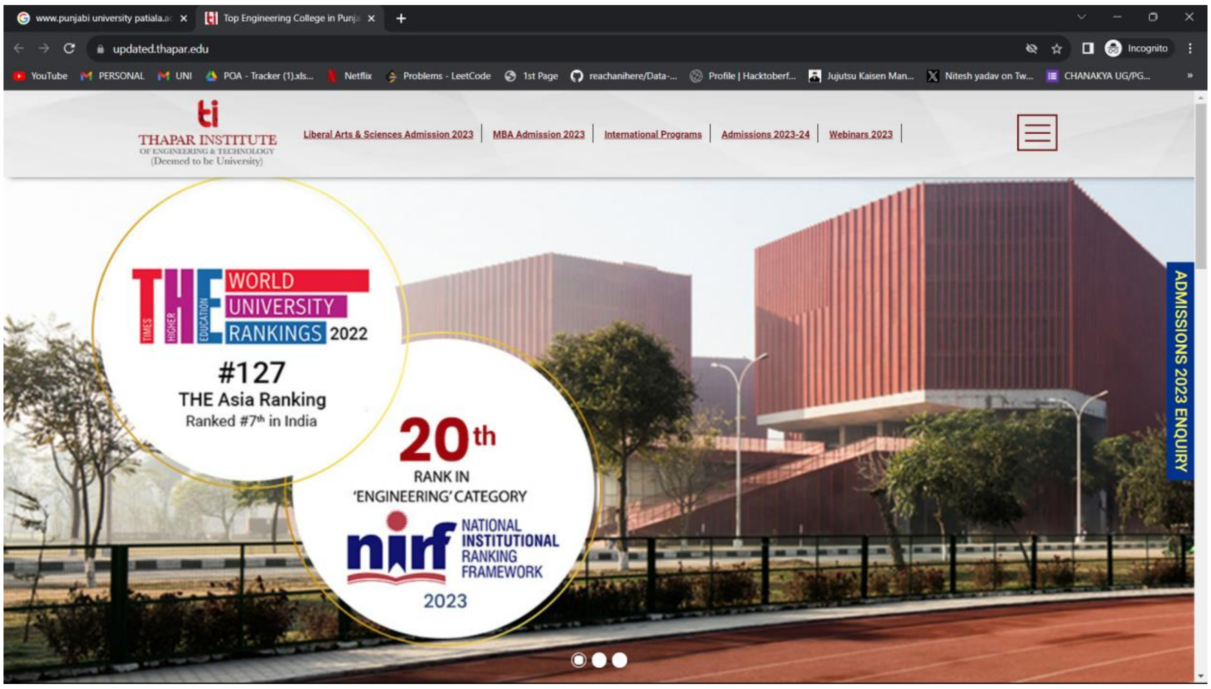
Change in the hosts file



```
# For example:
#
#      102.54.94.97     rhino.acme.com          # source server
#       38.25.63.10     x.acme.com              # x client host
        14.139.242.109  www.punjabiuniversity.ac.in
```

Looking at the website in action:

Name: SHREEYA CHATTERJI
Class: CO16



Name: SHREEYA CHATTERJI
Class: CO16

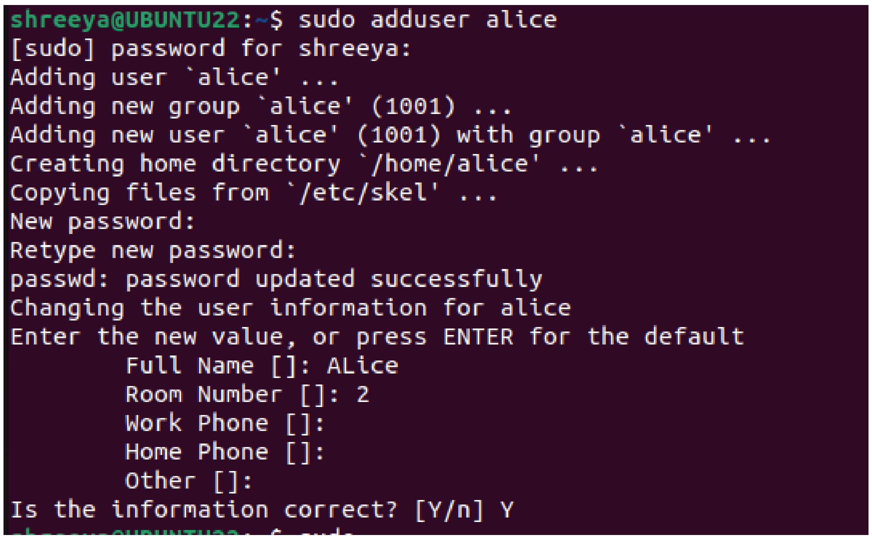
LAB ASSIGNMENT 4

CH 1: SET-UID Privileged Programs and Attacks on Them

Q1. Alice runs a Set-UID program that is owned by Bob. The program tries to read from /tmp/x, which is readable to Alice, but not to anybody else. Can this program successfully read from the file?

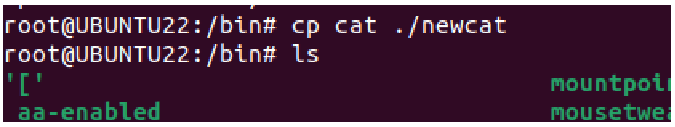
Ans.

1. Creating new user – Alice



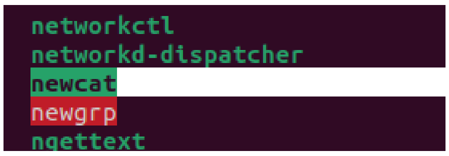
```
shreeya@UBUNTU22:~$ sudo adduser alice
[sudo] password for shreeya:
Adding user `alice' ...
Adding new group `alice' (1001) ...
Adding new user `alice' (1001) with group `alice' ...
Creating home directory `/home/alice' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for alice
Enter the new value, or press ENTER for the default
        Full Name []: ALice
        Room Number []: 2
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
```

2. Copying the cat file to new cat command file ie 'newcat' in this case.



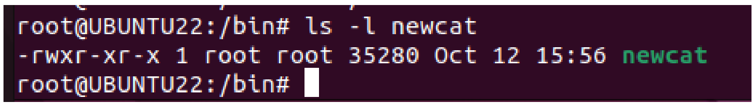
```
root@UBUNTU22:/bin# cp cat ./newcat
root@UBUNTU22:/bin# ls
'['                                        mountpoi
 aa-enabled                                mousetwe
```

3. The new file shows up as an output of ls



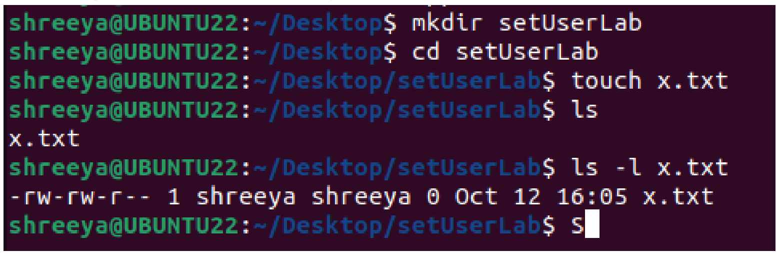
```
 networkctl
 networkd-dispatcher
 newcat
 newgrp
 ngettext
```

4. We see the owner status of the new file and we see that the owner is the root.



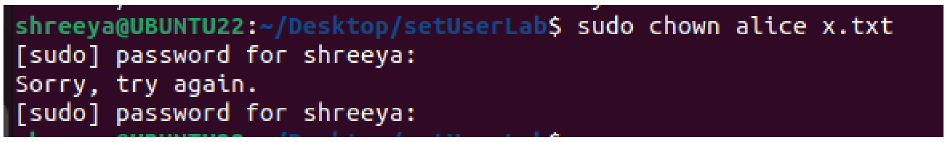
```
root@UBUNTU22:/bin# ls -l newcat
-rwxr-xr-x 1 root root 35280 Oct 12 15:56 newcat
root@UBUNTU22:/bin#
```

5. Creating a file in Bob(Shreeya) named x.txt. The current owner is Bob.



```
shreeya@UBUNTU22:~/Desktop$ mkdir setUserLab
shreeya@UBUNTU22:~/Desktop$ cd setUserLab
shreeya@UBUNTU22:~/Desktop/setUserLab$ touch x.txt
shreeya@UBUNTU22:~/Desktop/setUserLab$ ls
x.txt
shreeya@UBUNTU22:~/Desktop/setUserLab$ ls -l x.txt
-rw-rw-r-- 1 shreeya shreeya 0 Oct 12 16:05 x.txt
shreeya@UBUNTU22:~/Desktop/setUserLab$ S
```

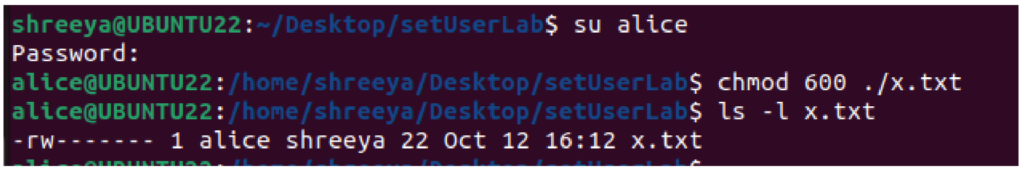
6. Changing the owner of the file to Alice.



```
shreeya@UBUNTU22:~/Desktop/setUserLab$ sudo chown alice x.txt
[sudo] password for shreeya:
Sorry, try again.
[sudo] password for shreeya:
```

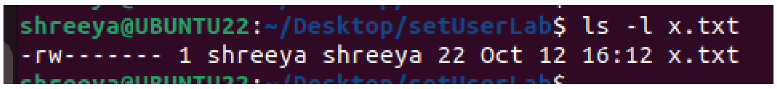
7. Making the file read only by alice.

Name: SHREEYA CHATTERJI
Class: CO16



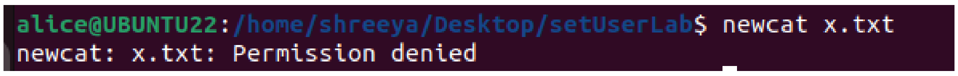
```
shreeya@UBUNTU22:~/Desktop/setUserLab$ su alice
Password:
alice@UBUNTU22:/home/shreeya/Desktop/setUserLab$ chmod 600 ./x.txt
alice@UBUNTU22:/home/shreeya/Desktop/setUserLab$ ls -l x.txt
-rw------- 1 alice shreeya 22 Oct 12 16:12 x.txt
```

8. The owner of newcat is Bob



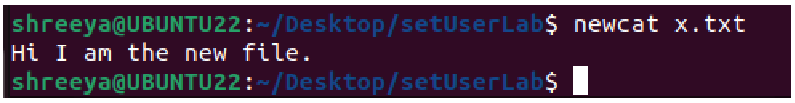
```
shreeya@UBUNTU22:~/Desktop/setUserLab$ ls -l x.txt
-rw------- 1 shreeya shreeya 22 Oct 12 16:12 x.txt
```

9. Alice cannot read x.txt read the file using mycat



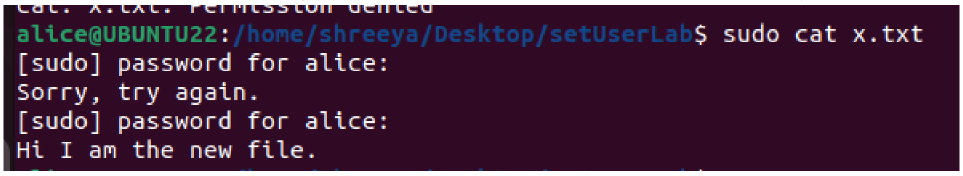
```
alice@UBUNTU22:/home/shreeya/Desktop/setUserLab$ newcat x.txt
newcat: x.txt: Permission denied
```

10. Bob can still read it



```
shreeya@UBUNTU22:~/Desktop/setUserLab$ newcat x.txt
Hi I am the new file.
shreeya@UBUNTU22:~/Desktop/setUserLab$
```

11. Alice can still read it using cat command as this command has root permissions.

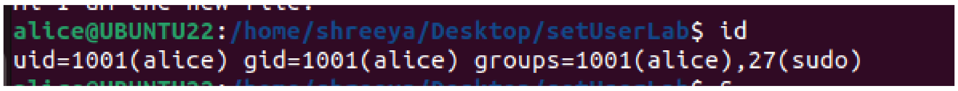


```
alice@UBUNTU22:/home/shreeya/Desktop/setUserLab$ sudo cat x.txt
[sudo] password for alice:
Sorry, try again.
[sudo] password for alice:
Hi I am the new file.
```

Q2. A process tries to open a file for read. The process's effective user ID is 1000, and real user ID is 2000. The file is readable to user ID 2000, but not to user ID 1000. Can this process successfully open the file?

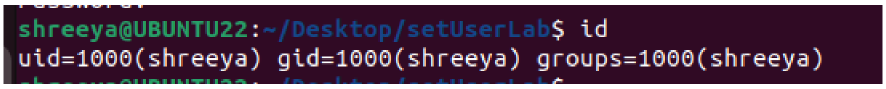
Ans.

1. ID for Alice



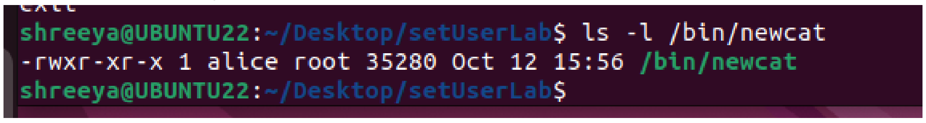
```
alice@UBUNTU22:/home/shreeya/Desktop/setUserLab$ id
uid=1001(alice) gid=1001(alice) groups=1001(alice),27(sudo)
```

2. ID for Shreeya



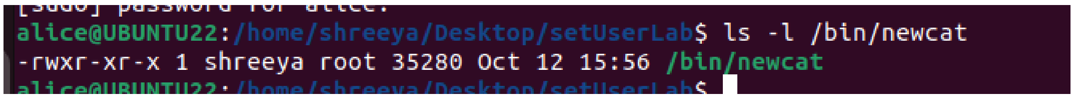
```
shreeya@UBUNTU22:~/Desktop/setUserLab$ id
uid=1000(shreeya) gid=1000(shreeya) groups=1000(shreeya)
```

3. Current ownership status of newcat



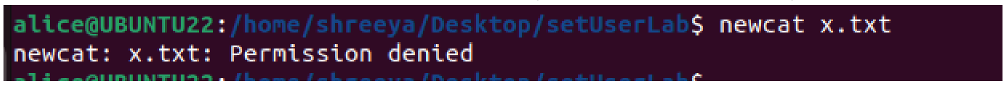
```
shreeya@UBUNTU22:~/Desktop/setUserLab$ ls -l /bin/newcat
-rwxr-xr-x 1 alice root 35280 Oct 12 15:56 /bin/newcat
shreeya@UBUNTU22:~/Desktop/setUserLab$
```

4. Changing the owner of newcat.



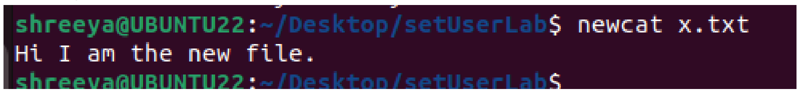
```
alice@UBUNTU22:/home/shreeya/Desktop/setUserLab$ ls -l /bin/newcat
-rwxr-xr-x 1 shreeya root 35280 Oct 12 15:56 /bin/newcat
```

5. Since owner of newcat is not Alice that is why it can not be used by Alice



```
alice@UBUNTU22:/home/shreeya/Desktop/setUserLab$ newcat x.txt
newcat: x.txt: Permission denied
```

6. Shreeya can still use it.



```
shreeya@UBUNTU22:~/Desktop/setUserLab$ newcat x.txt
Hi I am the new file.
shreeya@UBUNTU22:~/Desktop/setUserLab$
```

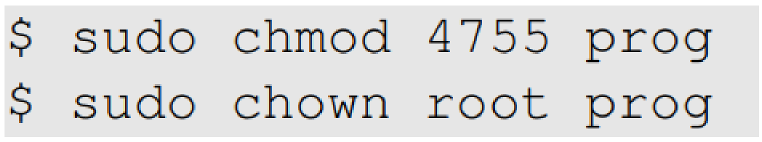
Q3. A root-owned Set-UID program allows a normal user to gain the root privilege while executing the program. What prevents the user from doing bad things using the privilege?

Name: SHREEYA CHATTERJI
Class: CO16

Ans. When a Set-UID program runs, it gains the privileges of the user who owns the executable (usually root) rather than the user who is executing the program. This can potentially allow normal users to perform actions with elevated privileges. However, there are several security measures in place to prevent abuse of this privilege:

- Minimal Privilege Principle: Set-UID programs should be designed to do the absolute minimum necessary to accomplish their task. Unnecessary privileges should not be elevated. This principle reduces the potential damage a user can do if they manage to exploit the program.

Q4. We are trying to turn a program prog owned by the seed user into a Set-UID program that is owned by root. Can running the following commands achieve the goal?



```
$ sudo chmod 4755 prog
$ sudo chown root prog
```

Ans. Yes, the commands you provided will set the Set-UID permission for the program prog and change its ownership to root, achieving the goal of making it a Set-UID program owned by root.

Here's what each command does:

1. **sudo chmod 4755 prog**: This command sets the permissions of the file prog to 4755. In octal notation, the 4 at the beginning represents the Set-UID permission, and 755 sets the read, write, and execute permissions for the owner, and read and execute permissions for others.

2. **sudo chown root prog**: This command changes the owner of the file prog to the root user.

Q5. The chown command automatically disables the Set-UID bit, when it changes the owner of a Set-UID program. Please explain why it does that.

Ans. When the 'chown' command is used to change the owner of a file, including the Set-UID program, the Set-UID program and Set-GID bits are automatically cleared to ensure security integrity. These are the reasons why this is important.

1. **Security Risk**: Allowing a non-privileged user to change the owner of a Set-UID program can be a significant security risk. If a regular user could change the owner of a Set-UID program to themselves and then execute it, they would effectively gain unauthorized access to the elevated privileges associated with the original owner (such as root). This could lead to exploitation and unauthorized access to sensitive system resources.
2. **Privilege Escalation:** Imagine a scenario where a vulnerable Set-UID program exists on a system, and a user discovers a way to exploit it. If the user could change the owner of this program to themselves without losing the Set-UID bit, they could execute arbitrary code with elevated privileges, which is a classic privilege escalation attack.

3. **Maintaining Security Boundaries:** The Set-UID bit is a powerful feature that allows users to execute programs with the permissions of the file owner. Clearing the Set-UID bit when the owner is changed helps maintain the security boundaries established by the system administrator. It ensures that only privileged users (typically, users with root access) can control which programs run with elevated privileges.

Q6. When we debug a program, we can change the program's internal variables during the execution. This can change a program's behavior. Can we use this technique to debug a Set-UID program and change its behavior? For example, if the program is supposed to open the /tmp/xyz file, we can modify the filename string, so the Set-UID program ends up opening /etc/passwd.

Ans.

Q7. Both system() and execve() can be used to execute external programs. Why is system() unsafe while execve() is safe?

Ans.

1. Command Injection Vulnerabilities:

- **system()**: **system()** passes the command string to the shell to be executed. For example, if you use **system("ls -l")**, it would internally execute **/bin/sh -c "ls -l"**. This makes it vulnerable to command injection attacks. If the command string is constructed from user input without proper validation, an attacker can inject malicious commands, leading to unintended and potentially harmful behavior.
- **execve()**: **execve()** does not use a shell to execute programs. It requires you to specify the program's path and arguments directly. For instance, **execve("/bin/ls", ["ls", "-l", NULL], NULL)** would execute **/bin/ls** directly, without involving a shell. Because there's no shell to interpret commands, there's no opportunity for command injection vulnerabilities if the arguments are properly sanitized.

2. Environment Variables:

- **system()**: **system()** uses the system's environment variables. If an attacker can manipulate the environment variables, they can potentially influence the behavior of the executed command.
- **execve()**: With **execve()**, you have complete control over the environment variables passed to the executed program. This means you can set up a clean and controlled environment, minimizing the risk of unintended behavior due to environment variable manipulation.

3. Return Values and Error Handling:

- **system()**: **system()** does not provide detailed error handling or the ability to capture the command's output or exit status. This can make it challenging to handle errors and respond appropriately to different outcomes.
- **execve()**: **execve()** provides detailed control over error handling. It allows you to handle errors directly, capture the program's output, and respond accordingly, enhancing the security of the overall system.

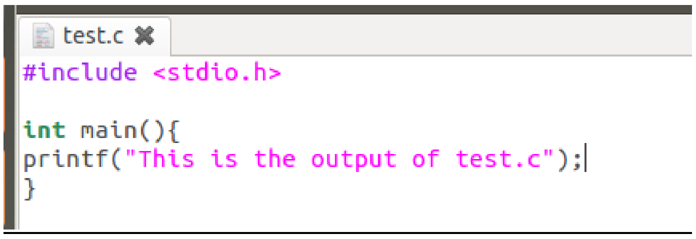
In summary, **system()** is less safe because it involves passing commands through a shell, making it susceptible to command injection attacks and environment variable manipulation. On the other hand, **execve()** is safer because it allows direct execution of programs without involving a shell, provides control over environment variables, and enables proper error handling, reducing the risk of security vulnerabilities. When executing external programs, it's generally recommended to use functions like **execve()** and sanitize inputs to prevent security threats.

Q8. When a program takes an input from users, we can redirect the input device, so the program can take the input from a file. For example, we can use prog < myfile to provide the data from myfile as input to the prog program. Now, if prog is a root-owned Set-UID program, can we use the following method to to get this privileged program to read from the /etc/shadow file?

prog < /etc/shadow

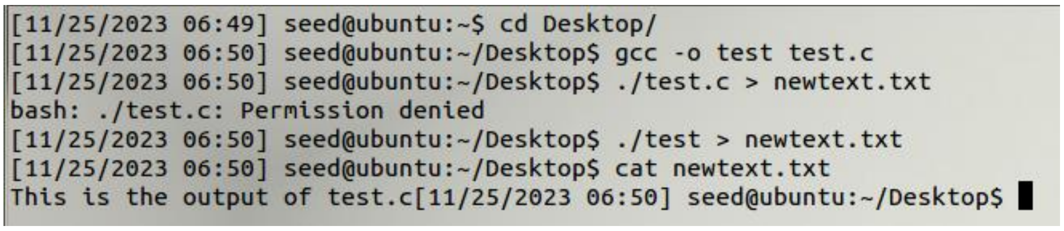
Ans:

a) **To demonstrate transferring the output of one file to a new file:**
TEST.C:



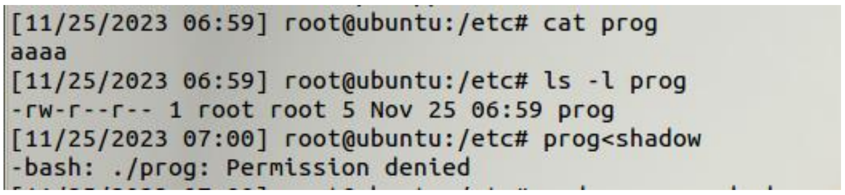
```
#include <stdio.h>

int main(){
printf("This is the output of test.c");
}
```

Output:



```
[11/25/2023 06:49] seed@ubuntu:~$ cd Desktop/
[11/25/2023 06:50] seed@ubuntu:~/Desktop$ gcc -o test test.c
[11/25/2023 06:50] seed@ubuntu:~/Desktop$ ./test.c > newtext.txt
bash: ./test.c: Permission denied
[11/25/2023 06:50] seed@ubuntu:~/Desktop$ ./test > newtext.txt
[11/25/2023 06:50] seed@ubuntu:~/Desktop$ cat newtext.txt
This is the output of test.c[11/25/2023 06:50] seed@ubuntu:~/Desktop$
```

b) **To do so with SetUID programs:**
The fact that prog is a Set-UID program with root privileges has nothing to do with the file that's input after. Since a normal user can not access /etc/shadow, they can not use it to redirect the input device. Trying to do so will cause a permission denied error



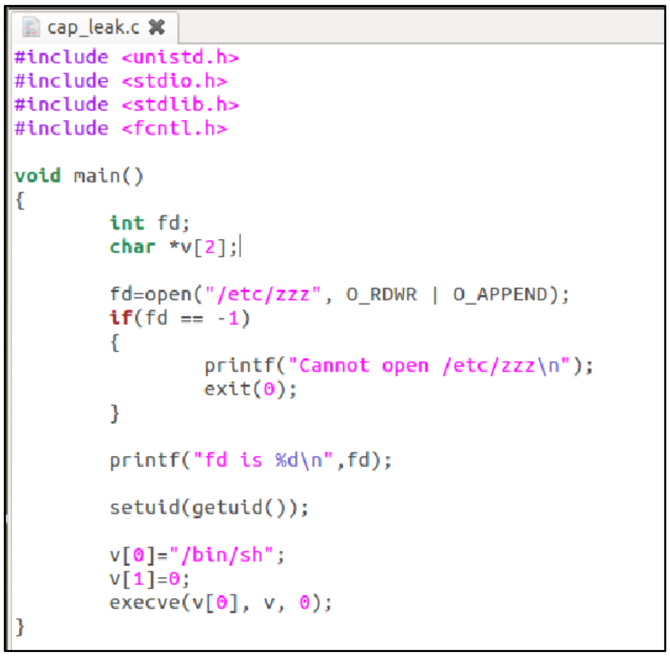
```
[11/25/2023 06:59] root@ubuntu:/etc# cat prog
aaaa
[11/25/2023 06:59] root@ubuntu:/etc# ls -l prog
-rw-r--r-- 1 root root 5 Nov 25 06:59 prog
[11/25/2023 07:00] root@ubuntu:/etc# prog<shadow
-bash: ./prog: Permission denied
```

As we can see that prog is root owned set UID program and all the operations are happening is SUDO mode still we cannot achieve what has been asked for and we get a permission denied error.

Q8. When a parent Set-UID process (effective user ID is root, and the real user ID is bob) creates a child process using fork(), the standard input, output, and error devices of the parent will be inherited by the child. If the child process drops its root privilege, it still retains the access right to these devices. This seems to be a capability leaking, similar to what we covered in Chapter 1.4.4. Can this pose any danger?

Ans: When a privileged process transitions to a non-privileged process, one of the common mistakes is capability leaking. The process may have gained some privileged capabilities when it was still privileged; when the privileges are downgraded, if the program does not clean up those capabilities, they may still be accessible by the non-privileged process. In other words, although the effective user ID of the process becomes non-privileged, the process is still privileged because it possesses privileged capabilities. And the access to privileged resources is obviously a danger.

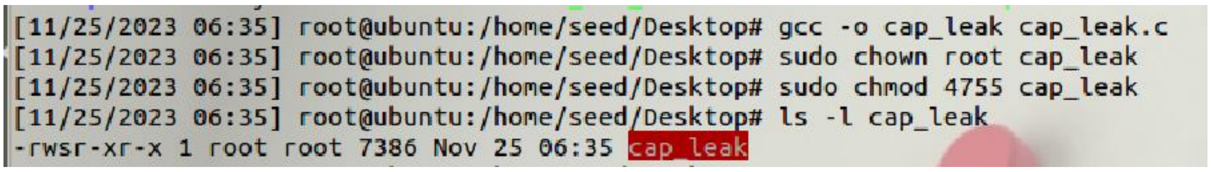
To demonstrate this we can run the following code:



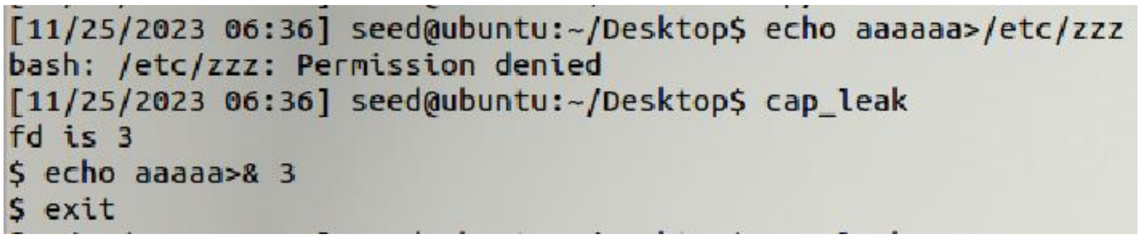
```
cap_leak.c
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>

void main()
{
        int fd;
        char *v[2];

        fd=open("/etc/zzz", O_RDWR | O_APPEND);
        if(fd == -1)
        {
                printf("Cannot open /etc/zzz\n");
                exit(0);
        }

        printf("fd is %d\n",fd);

        setuid(getuid());

        v[0]="/bin/sh";
        v[1]=0;
        execve(v[0], v, 0);
}
```

Since the above program forgets to close the file, the file descriptor is still valid and the process which does not have privileges is still capable of writing to the /etc/zzz. From the execution result, we can see that the file descriptor number is 3. We can easily write to /etc/zzz using the command " echo . . . >&3 " , where " &3 " means file descriptor 3. Before running the Set- UID program, we were not able to write to the protected fil e /etc/zzz, but after gaining the file descriptor via the Set-UID program, we can successfully modify it.

Execution result:



```
[11/25/2023 06:35] root@ubuntu:/home/seed/Desktop# gcc -o cap_leak cap_leak.c
[11/25/2023 06:35] root@ubuntu:/home/seed/Desktop# sudo chown root cap_leak
[11/25/2023 06:35] root@ubuntu:/home/seed/Desktop# sudo chmod 4755 cap_leak
[11/25/2023 06:35] root@ubuntu:/home/seed/Desktop# ls -l cap_leak
-rwsr-xr-x 1 root root 7386 Nov 25 06:35 cap_leak
```



```
[11/25/2023 06:36] seed@ubuntu:~/Desktop$ echo aaaaaa>/etc/zzz
bash: /etc/zzz: Permission denied
[11/25/2023 06:36] seed@ubuntu:~/Desktop$ cap_leak
fd is 3
$ echo aaaaa>& 3
$ exit
```

We see from the output that after running **"cap_leak"** the root access does not go away and we are able to run the command "**echo aaaa>& 3"** by making use of this leaked capability.

To avoid this we must always close the file descriptor as **close(fd)** to make sure that this does not happen.

Name: SHREEYA CHATTERJI
Class: CO16

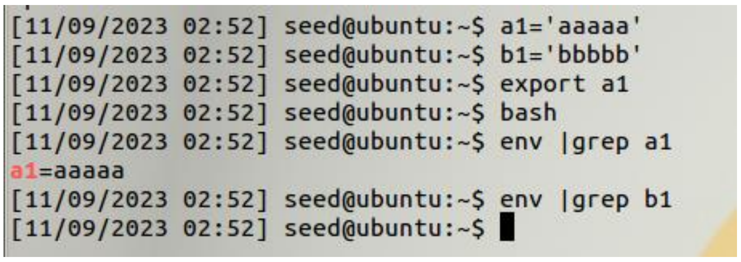
LAB ASSIGNMENT 5

Assignment - Environment Variables

1. What is the difference between environment variables and shell variables?
   Ans:
   Sure, here is the information in a table format:

	Shell Variables	Environment Variables
Definition	Shell variables are specific to the shell session in which they are defined	Environment variables are a type of shell variable that has been exported
Scope	They are used to keep track of data in the current shell session	They are visible not only in the shell session that created them, but also for any process (not just shells) that are started from that session
Inheritance	They are not inherited by child processes	They are available system-wide and can be accessed by any process or program running on the system
Naming Convention	Shell variables usually have names with lower-case letters	They often have names consisting of upper-case letters
Example	If you're running another application from the shell, that application will not inherit the shell variable	A typical case of this is the PATH environment variable, which may be set in the shell and later used by any program that wants to start programs without specifying a full path to them

   A screenshot of my terminal:



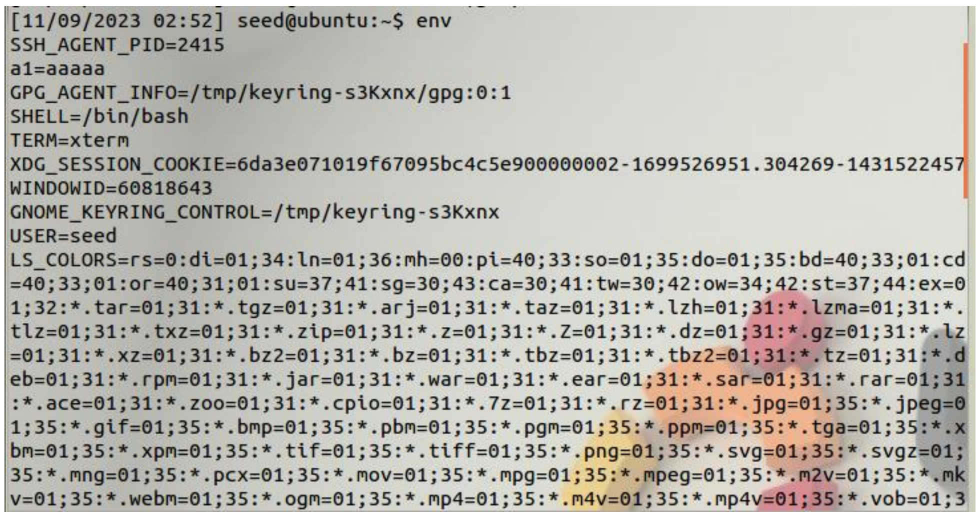
```
[11/09/2023 02:52] seed@ubuntu:~$ a1='aaaaa'
[11/09/2023 02:52] seed@ubuntu:~$ b1='bbbbb'
[11/09/2023 02:52] seed@ubuntu:~$ export a1
[11/09/2023 02:52] seed@ubuntu:~$ bash
[11/09/2023 02:52] seed@ubuntu:~$ env |grep a1
a1=aaaaa
[11/09/2023 02:52] seed@ubuntu:~$ env |grep b1
[11/09/2023 02:52] seed@ubuntu:~$ 
```

   In the above screenshot we can see that a1 is a global variable while b1 is a local variable. Inside the child process only **a1** is accessible as an env variable, **b1** shows no output.

2. Display the environment variables currently available to the shell. Display the contents of any 5 such variables. (env command)
   All the env variables in the system can be seen using the env cmd. We can check the content of it either using **echo $varname** or **env |grep varname**, here we have used the echo cmd.

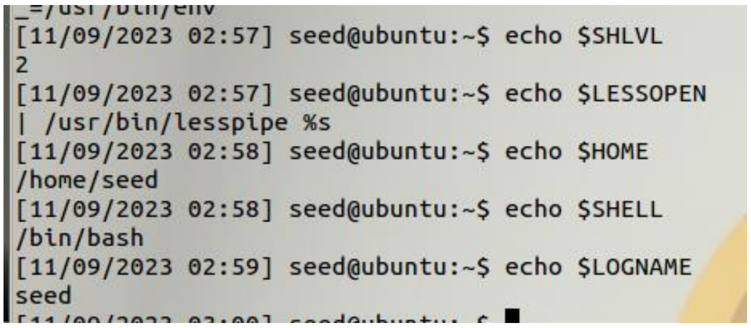
   Looking at all the environment variables: command: **env**

Name: SHREEYA CHATTERJI
Class: CO16



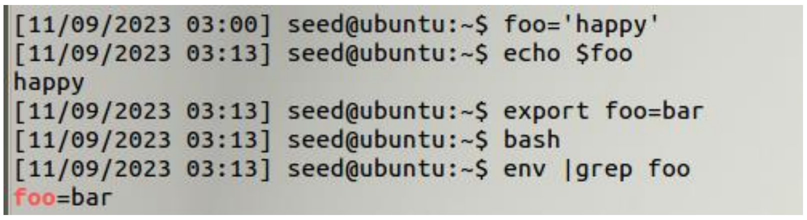
```
[11/09/2023 02:52] seed@ubuntu:~$ env
SSH_AGENT_PID=2415
a1=aaaaa
GPG_AGENT_INFO=/tmp/keyring-s3Kxnx/gpg:0:1
SHELL=/bin/bash
TERM=xterm
XDG_SESSION_COOKIE=6da3e071019f67095bc4c5e900000002-1699526951.304269-1431522457
WINDOWID=60818643
GNOME_KEYRING_CONTROL=/tmp/keyring-s3Kxnx
USER=seed
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd
=40;33;01:or=40;31;01:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=0
1;32:*.tar=01;31:*.tgz=01;31:*.arj=01;31:*.taz=01;31:*.lzh=01;31:*.lzma=01;31:*.
tlz=01;31:*.txz=01;31:*.zip=01;31:*.z=01;31:*.Z=01;31:*.dz=01;31:*.gz=01;31:*.lz
=01;31:*.xz=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.d
eb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31
:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.jpg=01;35:*.jpeg=0
1;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.x
bm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;
35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mk
v=01;35:*.webm=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;3
```

Printing any 5 environment variables:



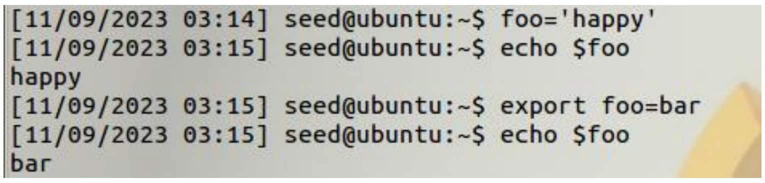
```
_=/usr/bin/env
[11/09/2023 02:57] seed@ubuntu:~$ echo $SHLVL
2
[11/09/2023 02:57] seed@ubuntu:~$ echo $LESSOPEN
| /usr/bin/lesspipe %s
[11/09/2023 02:58] seed@ubuntu:~$ echo $HOME
/home/seed
[11/09/2023 02:58] seed@ubuntu:~$ echo $SHELL
/bin/bash
[11/09/2023 02:59] seed@ubuntu:~$ echo $LOGNAME
seed
[11/09/2023 03:00] seed@ubuntu:~$ █
```

3. In Bash, if we run "export foo=bar", does it change the environment variable of the current process?
   Ans: Yes the value changes inn both the current process and child process.
   For child process:



```
[11/09/2023 03:00] seed@ubuntu:~$ foo='happy'
[11/09/2023 03:13] seed@ubuntu:~$ echo $foo
happy
[11/09/2023 03:13] seed@ubuntu:~$ export foo=bar
[11/09/2023 03:13] seed@ubuntu:~$ bash
[11/09/2023 03:13] seed@ubuntu:~$ env |grep foo
foo=bar
```

   For current process:



```
[11/09/2023 03:14] seed@ubuntu:~$ foo='happy'
[11/09/2023 03:15] seed@ubuntu:~$ echo $foo
happy
[11/09/2023 03:15] seed@ubuntu:~$ export foo=bar
[11/09/2023 03:15] seed@ubuntu:~$ echo $foo
bar
```

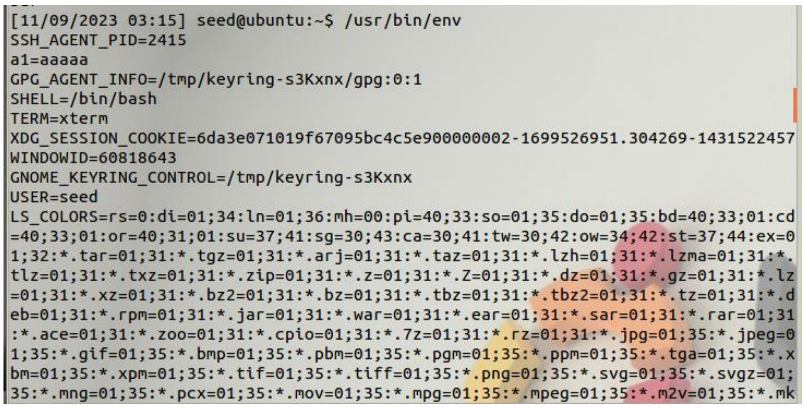
   Here we see that the initial value of foo was **happy** but once we do export it's value changes to **bar.**

Name: <u>SHREEYA CHATTERJI</u>
Class: <u>CO16</u>

4. The followings are two different ways to print out environment variables. Describe their differences:
   • $ /usr/bin/env

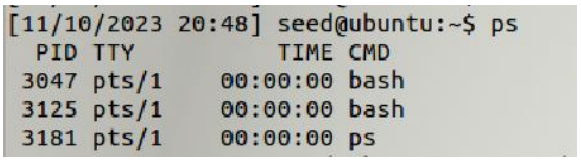
   • $ /usr/bin/strings /proc/$$/environ

   Ans: **$ /usr/bin/env** – this displays all the current environment variables. It prints the environment variables in the format **VARNAME=value** on separate lines. This command is versatile and can also be used to run a command with a modified environment. For example, **env VARNAME=value command** will run 'command' with an environment variable set to a specific value.



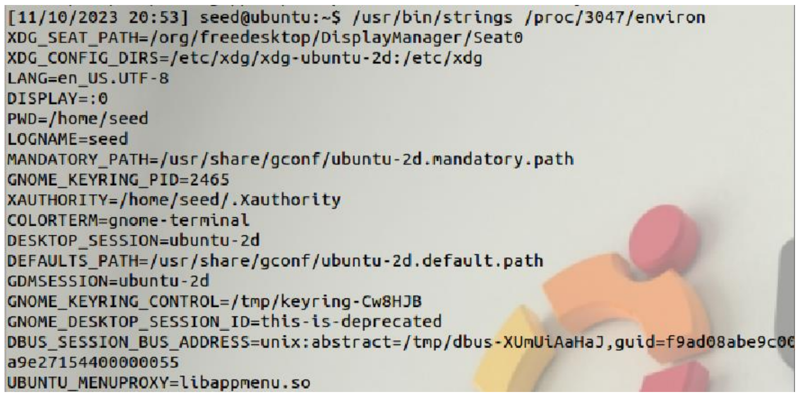
```
[11/09/2023 03:15] seed@ubuntu:~$ /usr/bin/env
SSH_AGENT_PID=2415
a1=aaaaa
GPG_AGENT_INFO=/tmp/keyring-s3Kxnx/gpg:0:1
SHELL=/bin/bash
TERM=xterm
XDG_SESSION_COOKIE=6da3e071019f67095bc4c5e900000002-1699526951.304269-1431522457
WINDOWID=60818643
GNOME_KEYRING_CONTROL=/tmp/keyring-s3Kxnx
USER=seed
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd
=40;33;01:or=40;31;01:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=0
1;32:*.tar=01;31:*.tgz=01;31:*.arj=01;31:*.taz=01;31:*.lzh=01;31:*.lzma=01;31:*.
tlz=01;31:*.txz=01;31:*.zip=01;31:*.z=01;31:*.Z=01;31:*.dz=01;31:*.gz=01;31:*.lz
=01;31:*.xz=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.d
eb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31
:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.jpg=01;35:*.jpeg=0
1;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.x
bm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;
35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mk
```

➢ **extracting some pids**



```
[11/10/2023 20:48] seed@ubuntu:~$ ps
  PID TTY          TIME CMD
 3047 pts/1    00:00:00 bash
 3125 pts/1    00:00:00 bash
 3181 pts/1    00:00:00 ps
```

$ /usr/bin/strings /proc/$$/environ- This method directly accesses the **/proc** filesystem, specifically the **environ** file for the current process (via the **$$** variable, which represents the PID of the current shell). The **strings** command extracts readable text from binary data, and in this case, it extracts the environment variables. This method directly reads the process environment, so it might include variables that are not part of the typical shell environment.



```
[11/10/2023 20:53] seed@ubuntu:~$ /usr/bin/strings /proc/3047/environ
XDG_SEAT_PATH=/org/freedesktop/DisplayManager/Seat0
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu-2d:/etc/xdg
LANG=en_US.UTF-8
DISPLAY=:0
PWD=/home/seed
LOGNAME=seed
MANDATORY_PATH=/usr/share/gconf/ubuntu-2d.mandatory.path
GNOME_KEYRING_PID=2465
XAUTHORITY=/home/seed/.Xauthority
COLORTERM=gnome-terminal
DESKTOP_SESSION=ubuntu-2d
DEFAULTS_PATH=/usr/share/gconf/ubuntu-2d.default.path
GDMSESSION=ubuntu-2d
GNOME_KEYRING_CONTROL=/tmp/keyring-Cw8HJB
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
DBUS_SESSION_BUS_ADDRESS=unix:abstract=/tmp/dbus-XUmUiAaHaJ,guid=f9ad08abe9c0
a9e27154400000055
UBUNTU_MENUPROXY=libappmenu.so
```

Name: SHREEYA CHATTERJI
Class: CO16

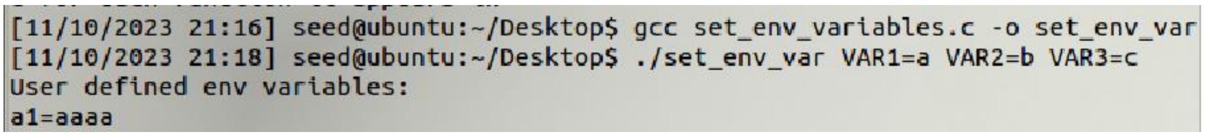
5. Write a program to pass user defined environment variables using the help of command line arguments and display the same.

The program:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

// Declare extern char **environ to access environment variables
extern char **environ;
int main(int argc, char *argv[]) {
    // Check if there are user-defined environment variables
    if (argc < 2) {
        printf("Usage: %s VAR1=value1 VAR2=value2 ...\n", argv[0]);
        return 1;
    }

    // Iterate through the provided variables
    for (int i = 1; i < argc; ++i) {
        // Split each variable into name and value
        char *variable = argv[i];
        char *equals_pos = strchr(variable, '=');

        if (equals_pos != NULL) {
            // Extract name and value
            *equals_pos = '\0';  // Null-terminate at '=' to separate name and value
            char *name = variable;
            char *value = equals_pos + 1;

            // Set the environment variable
            if (setenv(name, value, 1) != 0) {
                fprintf(stderr, "Failed to set environment variable: %s\n", name);
                return 1;
            }
        } else {
            fprintf(stderr, "Invalid format for environment variable: %s\n", variable);
            return 1;
        }
    }

    // Display the environment variables
    printf("User-defined environment variables:\n");
    char **env;
    for (env = environ; *env != NULL; ++env) {
        printf("%s\n", *env);
    }

    return 0;
}
```

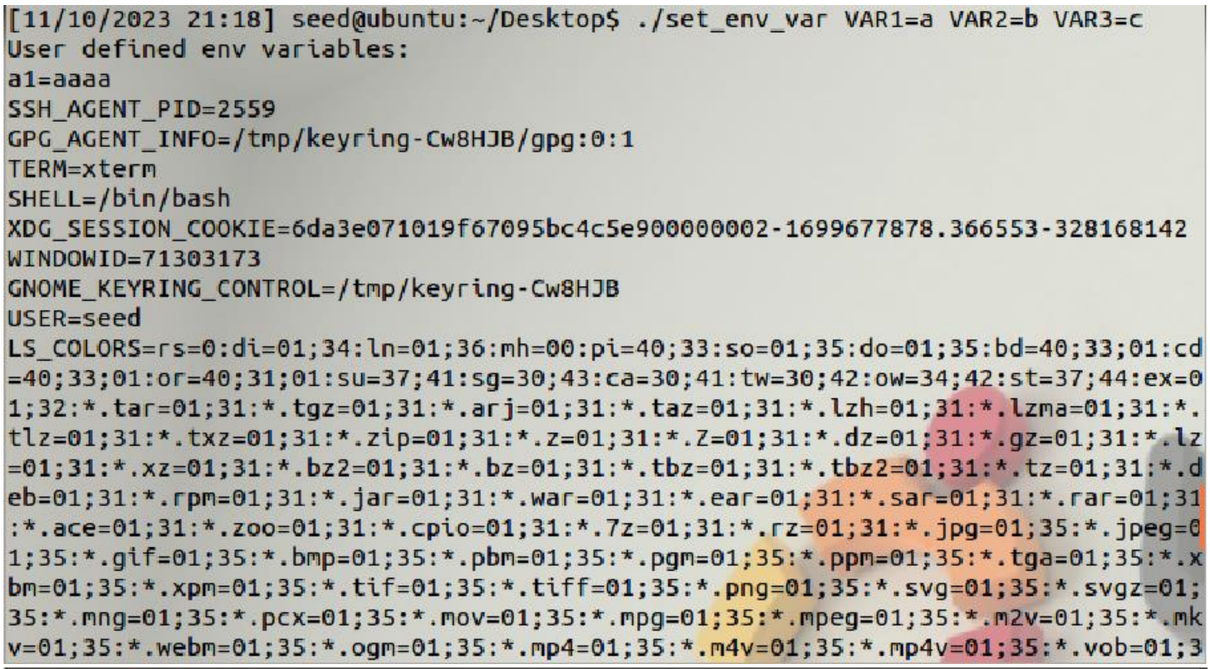
Name: SHREEYA CHATTERJI
Class: CO16

Running the code:

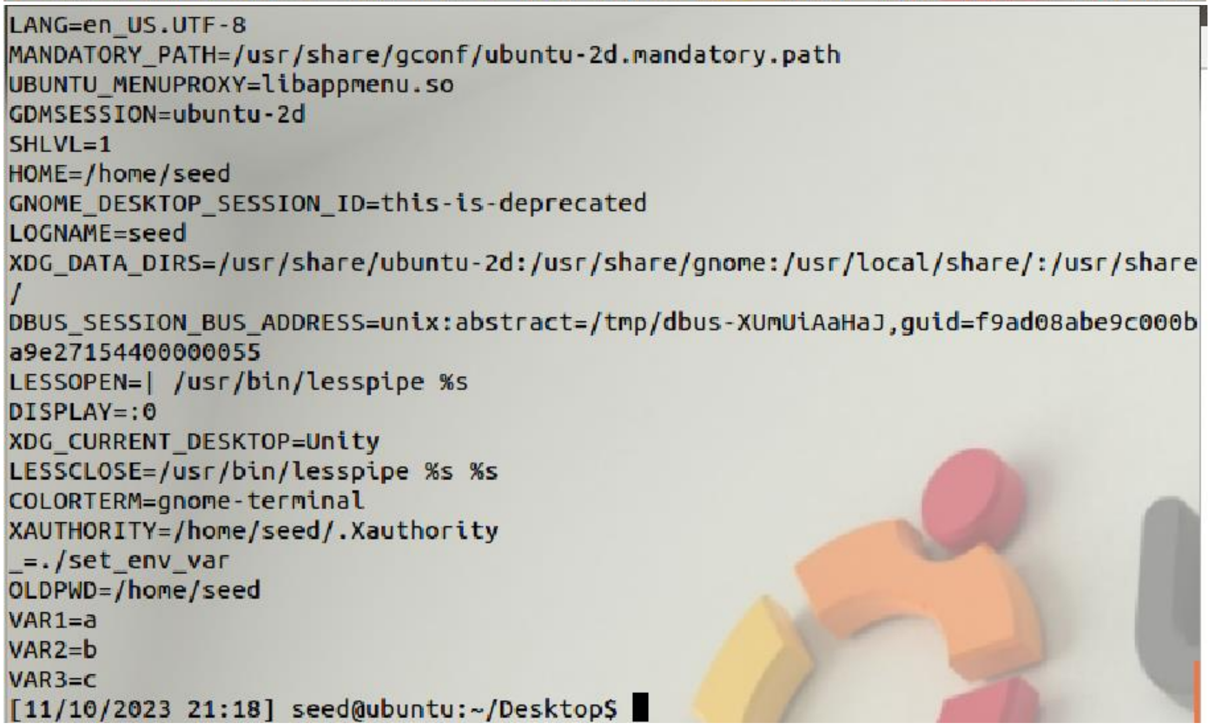


```
[11/10/2023 21:16] seed@ubuntu:~/Desktop$ gcc set_env_variables.c -o set_env_var
[11/10/2023 21:18] seed@ubuntu:~/Desktop$ ./set_env_var VAR1=a VAR2=b VAR3=c
User defined env variables:
a1=aaaa
```

OUTPUT:
It prints all the current env variables along with the new env variables.



```
[11/10/2023 21:18] seed@ubuntu:~/Desktop$ ./set_env_var VAR1=a VAR2=b VAR3=c
User defined env variables:
a1=aaaa
SSH_AGENT_PID=2559
GPG_AGENT_INFO=/tmp/keyring-Cw8HJB/gpg:0:1
TERM=xterm
SHELL=/bin/bash
XDG_SESSION_COOKIE=6da3e071019f67095bc4c5e900000002-1699677878.366553-328168142
WINDOWID=71303173
GNOME_KEYRING_CONTROL=/tmp/keyring-Cw8HJB
USER=seed
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd
=40;33;01:or=40;31;01:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=0
1;32:*.tar=01;31:*.tgz=01;31:*.arj=01;31:*.taz=01;31:*.lzh=01;31:*.lzma=01;31:*.
tlz=01;31:*.txz=01;31:*.zip=01;31:*.z=01;31:*.Z=01;31:*.dz=01;31:*.gz=01;31:*.lz
=01;31:*.xz=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.d
eb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31
:*.ace=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;31:*.jpg=01;35:*.jpeg=0
1;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.x
bm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;
35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mk
v=01;35:*.webm=01;35:*.ogm=01;35:*.mp4=01;35:*.m4v=01;35:*.mp4v=01;35:*.vob=01;3
```



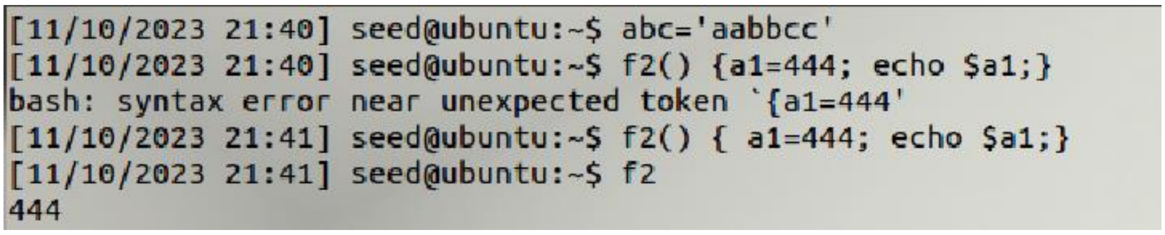
```
LANG=en_US.UTF-8
MANDATORY_PATH=/usr/share/gconf/ubuntu-2d.mandatory.path
UBUNTU_MENUPROXY=libappmenu.so
GDMSESSION=ubuntu-2d
SHLVL=1
HOME=/home/seed
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
LOGNAME=seed
XDG_DATA_DIRS=/usr/share/ubuntu-2d:/usr/share/gnome:/usr/local/share/:/usr/share
/
DBUS_SESSION_BUS_ADDRESS=unix:abstract=/tmp/dbus-XUmUiAaHaJ,guid=f9ad08abe9c000b
a9e27154400000055
LESSOPEN=| /usr/bin/lesspipe %s
DISPLAY=:0
XDG_CURRENT_DESKTOP=Unity
LESSCLOSE=/usr/bin/lesspipe %s %s
COLORTERM=gnome-terminal
XAUTHORITY=/home/seed/.Xauthority
_=./set_env_var
OLDPWD=/home/seed
VAR1=a
VAR2=b
VAR3=c
[11/10/2023 21:18] seed@ubuntu:~/Desktop$ █
```

As we can see VAR1,VAR2,VAR3 are the new variables declared as environment variables on running the program.

Name: SHREEYA CHATTERJI
Class: CO16

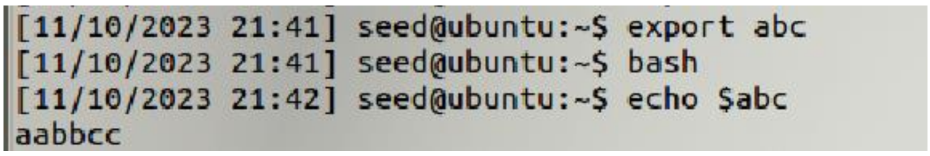
6. Experiment passing both variables and functions as environment variables to the child process.

$ Creating variables and functions



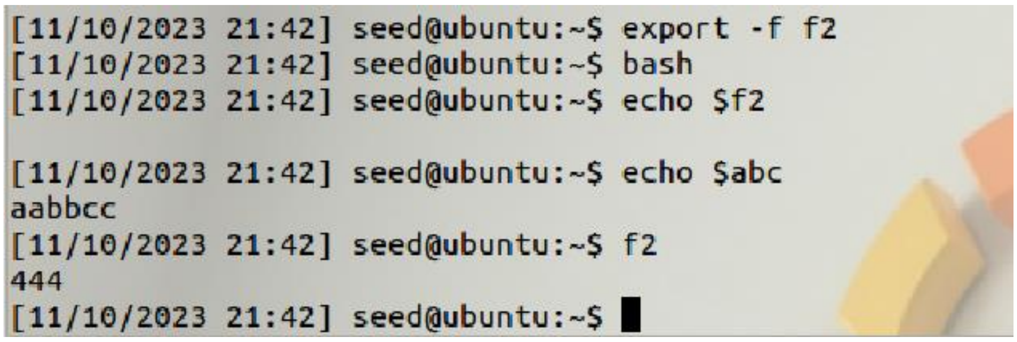
```
[11/10/2023 21:40] seed@ubuntu:~$ abc='aabbcc'
[11/10/2023 21:40] seed@ubuntu:~$ f2() {a1=444; echo $a1;}
bash: syntax error near unexpected token `{a1=444'
[11/10/2023 21:41] seed@ubuntu:~$ f2() { a1=444; echo $a1;}
[11/10/2023 21:41] seed@ubuntu:~$ f2
444
```

$Passing variable as env variable



```
[11/10/2023 21:41] seed@ubuntu:~$ export abc
[11/10/2023 21:41] seed@ubuntu:~$ bash
[11/10/2023 21:42] seed@ubuntu:~$ echo $abc
aabbcc
```

$Passing function as an environment function

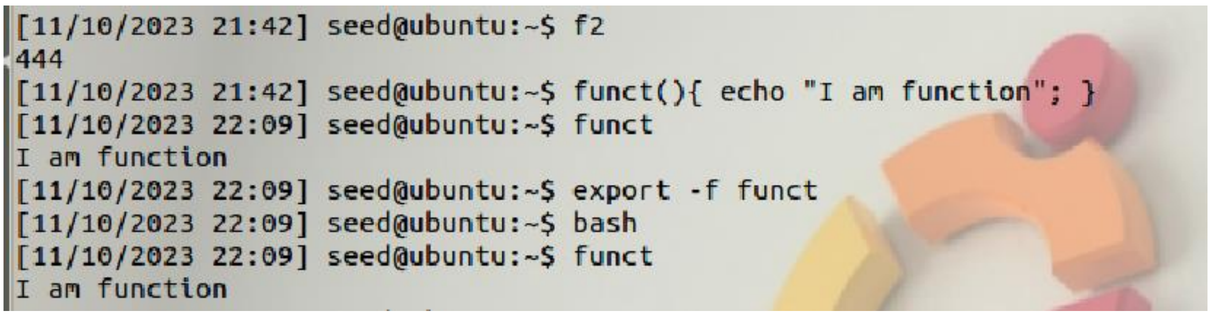


```
[11/10/2023 21:42] seed@ubuntu:~$ export -f f2
[11/10/2023 21:42] seed@ubuntu:~$ bash
[11/10/2023 21:42] seed@ubuntu:~$ echo $f2

[11/10/2023 21:42] seed@ubuntu:~$ echo $abc
aabbcc
[11/10/2023 21:42] seed@ubuntu:~$ f2
444
[11/10/2023 21:42] seed@ubuntu:~$ █
```

As we notice both the function and the work run as environment variable in the child process initiated by **“bash”** command.

7. Define a function inside an environment variable. When executing a child process how can we invoke that function by passing that as an environment variable to child process.

Ans: Same as above. Let’s show another example

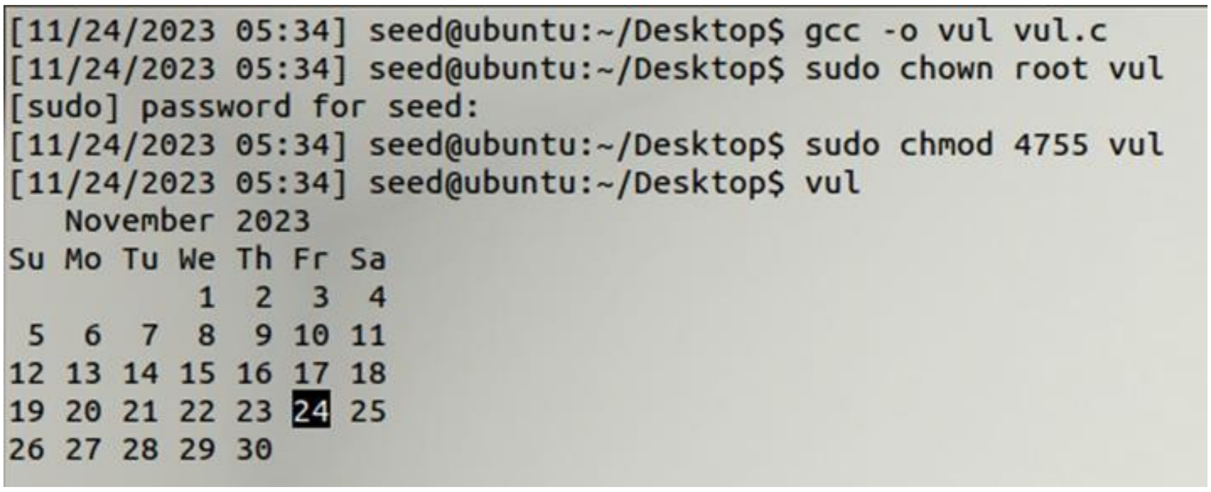


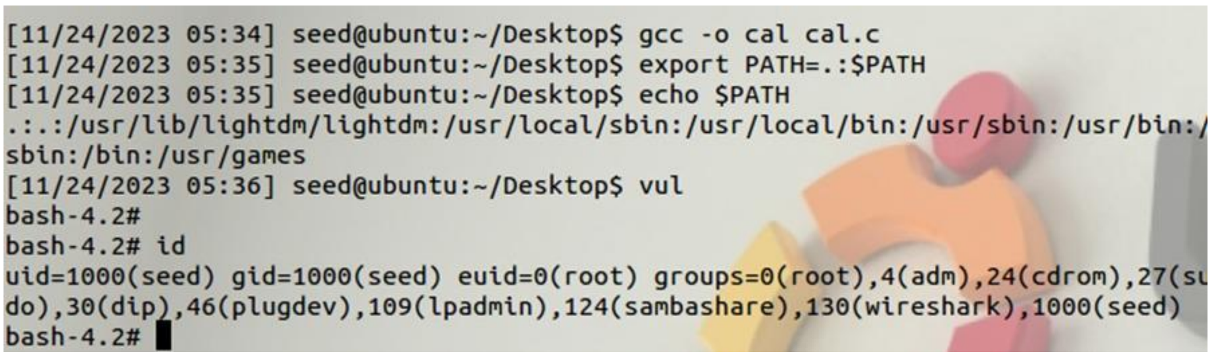
```
[11/10/2023 21:42] seed@ubuntu:~$ f2
444
[11/10/2023 21:42] seed@ubuntu:~$ funct(){ echo "I am function"; }
[11/10/2023 22:09] seed@ubuntu:~$ funct
I am function
[11/10/2023 22:09] seed@ubuntu:~$ export -f funct
[11/10/2023 22:09] seed@ubuntu:~$ bash
[11/10/2023 22:09] seed@ubuntu:~$ funct
I am function
```

8. Demonstrate the redirection of an inbuilt command or an application (e.g. ls, cat, cal or any command of your choice) to some user defined executable by manipulating the required environment variable.

ANS:

Name: SHREEYA CHATTERJI
Class: CO16

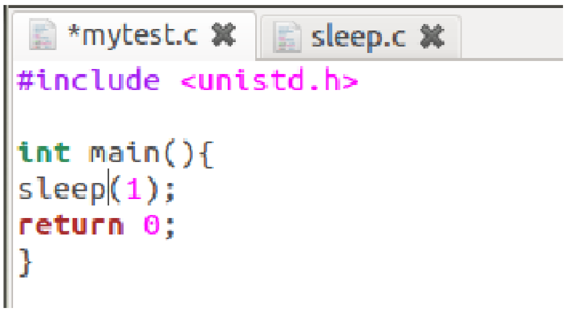

```
[11/24/2023 05:34] seed@ubuntu:~/Desktop$ gcc -o vul vul.c
[11/24/2023 05:34] seed@ubuntu:~/Desktop$ sudo chown root vul
[sudo] password for seed:
[11/24/2023 05:34] seed@ubuntu:~/Desktop$ sudo chmod 4755 vul
[11/24/2023 05:34] seed@ubuntu:~/Desktop$ vul
   November 2023
Su Mo Tu We Th Fr Sa
          1  2  3  4
 5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30
```


```
[11/24/2023 05:34] seed@ubuntu:~/Desktop$ gcc -o cal cal.c
[11/24/2023 05:35] seed@ubuntu:~/Desktop$ export PATH=.:$PATH
[11/24/2023 05:35] seed@ubuntu:~/Desktop$ echo $PATH
.:.:/usr/lib/lightdm/lightdm:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/
sbin:/bin:/usr/games
[11/24/2023 05:36] seed@ubuntu:~/Desktop$ vul
bash-4.2#
bash-4.2# id
uid=1000(seed) gid=1000(seed) euid=0(root) groups=0(root),4(adm),24(cdrom),27(su
do),30(dip),46(plugdev),109(lpadmin),124(sambashare),130(wireshark),1000(seed)
bash-4.2#
```

9. Write a program to demonstrate the attacks through the use of Dynamic Linker. Also, show the effect if we set the same program as a SetUID program

ANS:

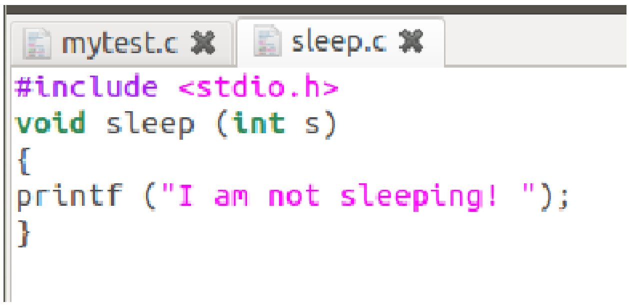
Creating a file that is using a SETUID program:



```c
#include <unistd.h>

int main(){
sleep(1);
return 0;
}
```

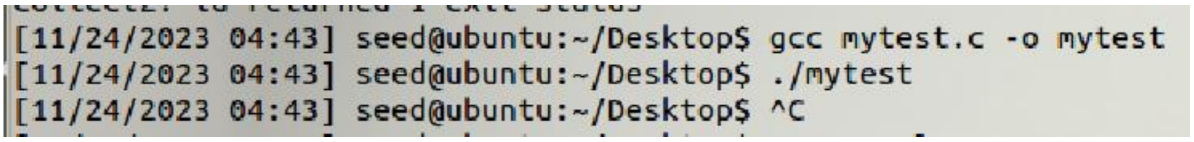
Creating a file to escape the setUID and run something else:

Name: SHREEYA CHATTERJI
Class: CO16



```
#include <stdio.h>
void sleep (int s)
{
printf ("I am not sleeping! ");
}
```

Running mytest.c without changing the sleep(1) function:

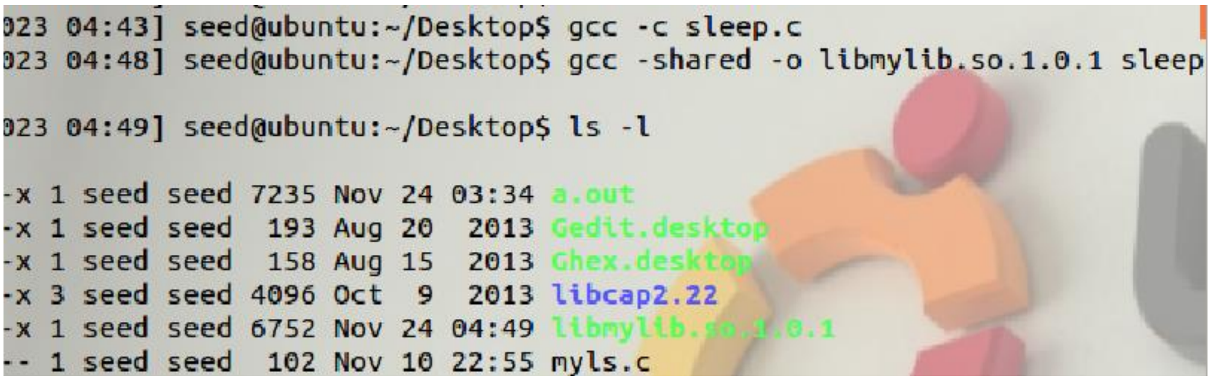


```
[11/24/2023 04:43] seed@ubuntu:~/Desktop$ gcc mytest.c -o mytest
[11/24/2023 04:43] seed@ubuntu:~/Desktop$ ./mytest
[11/24/2023 04:43] seed@ubuntu:~/Desktop$ ^C
```

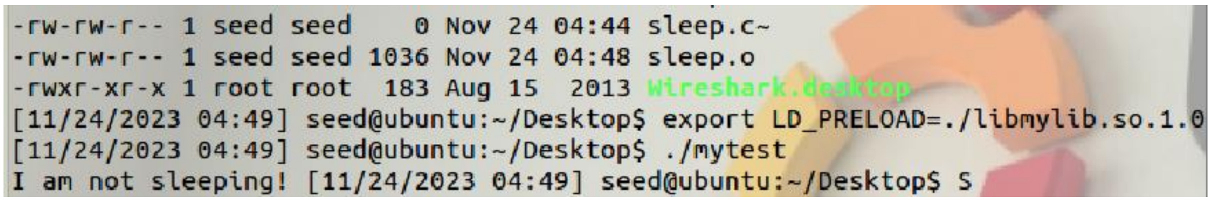
As we can see the sleep function is working and to escape the sleep I use ctrl+C.

Next we make sleep.c that we created act like the sleep function.

To do this we create sleep.o then using LD_PRELOAD variable we make sleep use the sleep.c file that we just created instead of the actual SetUID sleep.

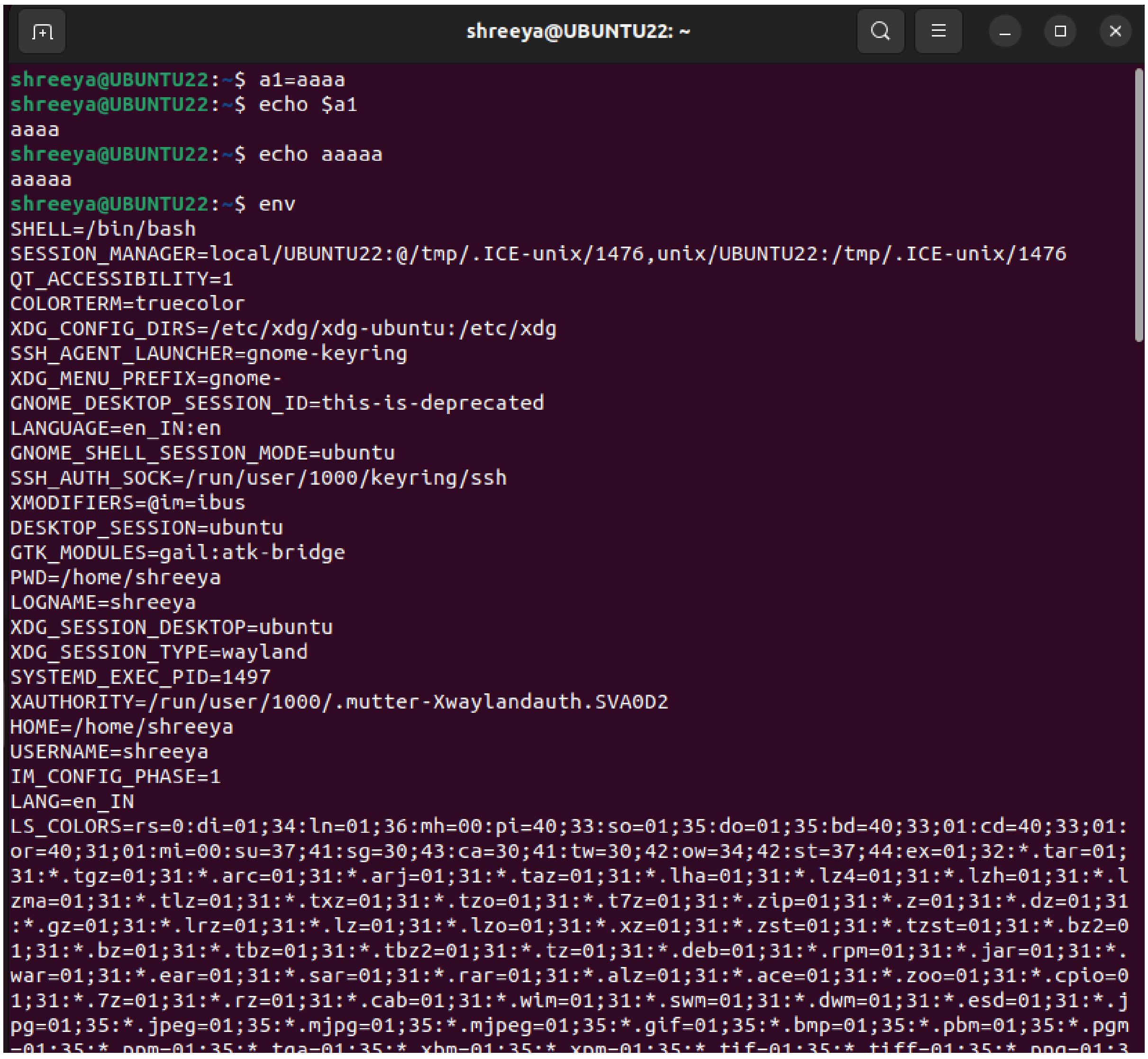


```
023 04:43] seed@ubuntu:~/Desktop$ gcc -c sleep.c
023 04:48] seed@ubuntu:~/Desktop$ gcc -shared -o libmylib.so.1.0.1 sleep

023 04:49] seed@ubuntu:~/Desktop$ ls -l

-x 1 seed seed 7235 Nov 24 03:34 a.out
-x 1 seed seed  193 Aug 20  2013 Gedit.desktop
-x 1 seed seed  158 Aug 15  2013 Ghex.desktop
-x 3 seed seed 4096 Oct  9  2013 libcap2.22
-x 1 seed seed 6752 Nov 24 04:49 libmylib.so.1.0.1
-- 1 seed seed  102 Nov 10 22:55 myls.c
```



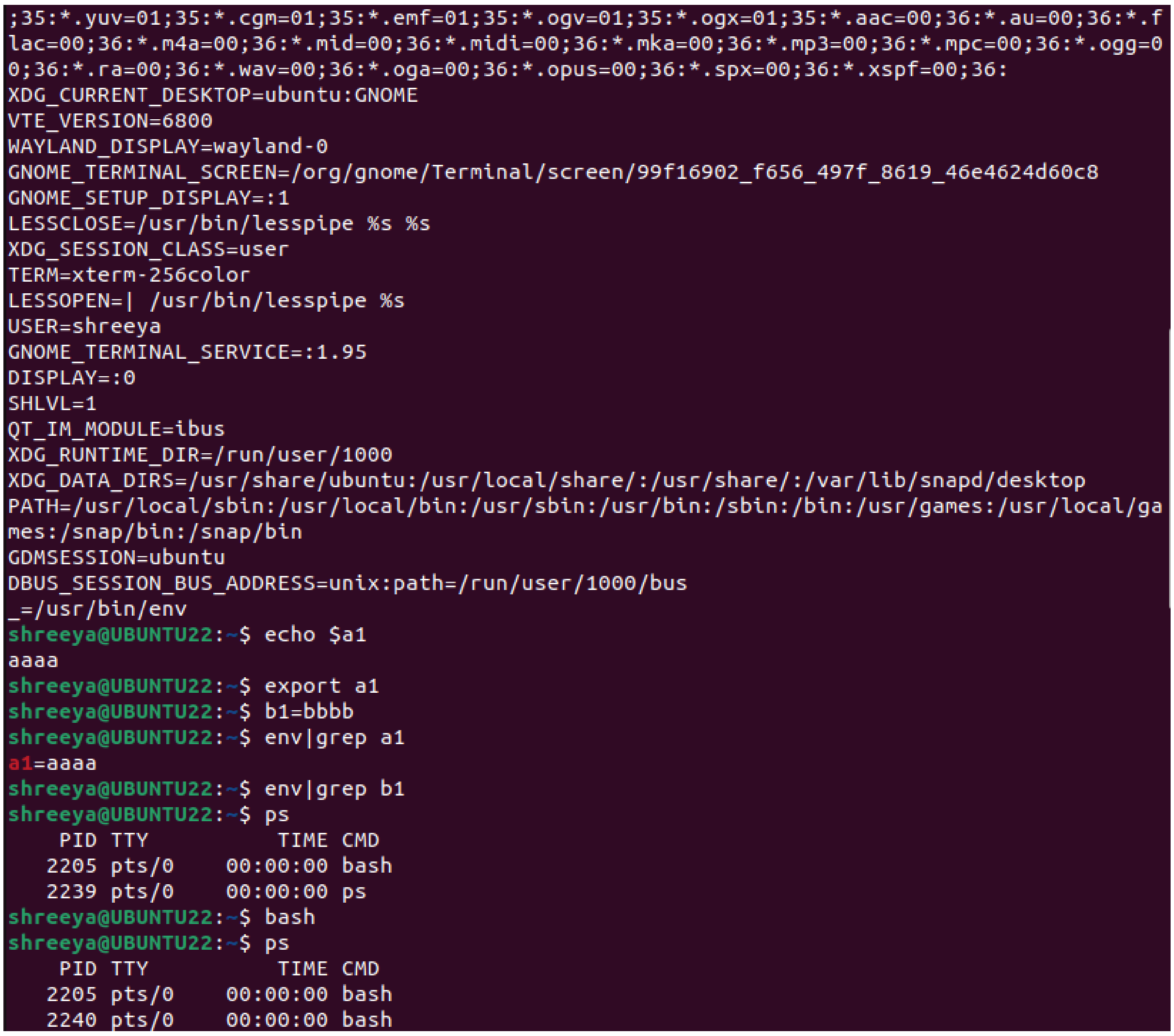
```
-rw-rw-r-- 1 seed seed    0 Nov 24 04:44 sleep.c~
-rw-rw-r-- 1 seed seed 1036 Nov 24 04:48 sleep.o
-rwxr-xr-x 1 root root  183 Aug 15  2013 Wireshark.desktop
[11/24/2023 04:49] seed@ubuntu:~/Desktop$ export LD_PRELOAD=./libmylib.so.1.0
[11/24/2023 04:49] seed@ubuntu:~/Desktop$ ./mytest
I am not sleeping! [11/24/2023 04:49] seed@ubuntu:~/Desktop$ S
```

Name: SHREEYA CHATTERJI
Class: CO16



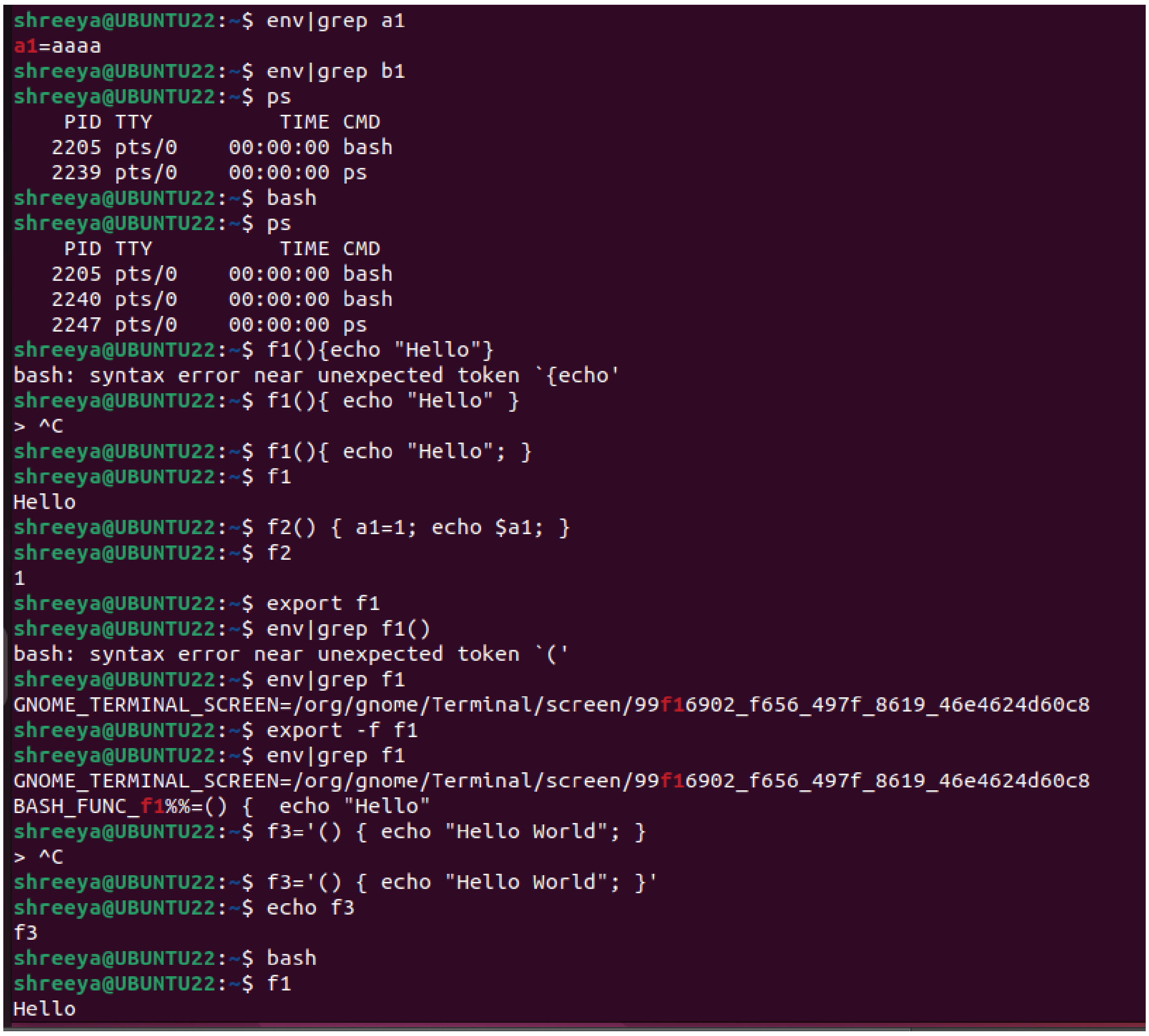
```
shreeya@UBUNTU22:~$ a1=aaaa
shreeya@UBUNTU22:~$ echo $a1
aaaa
shreeya@UBUNTU22:~$ echo aaaaa
aaaaa
shreeya@UBUNTU22:~$ env
SHELL=/bin/bash
SESSION_MANAGER=local/UBUNTU22:@/tmp/.ICE-unix/1476,unix/UBUNTU22:/tmp/.ICE-unix/1476
QT_ACCESSIBILITY=1
COLORTERM=truecolor
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
SSH_AGENT_LAUNCHER=gnome-keyring
XDG_MENU_PREFIX=gnome-
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
LANGUAGE=en_IN:en
GNOME_SHELL_SESSION_MODE=ubuntu
SSH_AUTH_SOCK=/run/user/1000/keyring/ssh
XMODIFIERS=@im=ibus
DESKTOP_SESSION=ubuntu
GTK_MODULES=gail:atk-bridge
PWD=/home/shreeya
LOGNAME=shreeya
XDG_SESSION_DESKTOP=ubuntu
XDG_SESSION_TYPE=wayland
SYSTEMD_EXEC_PID=1497
XAUTHORITY=/run/user/1000/.mutter-Xwaylandauth.SVA0D2
HOME=/home/shreeya
USERNAME=shreeya
IM_CONFIG_PHASE=1
LANG=en_IN
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:
or=40;31;01:mi=00:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;
31:*.tgz=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.l
zma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31
:*.gz=01;31:*.lrz=01;31:*.lz=01;31:*.lzo=01;31:*.xz=01;31:*.zst=01;31:*.tzst=01;31:*.bz2=0
1;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.
war=01;31:*.ear=01;31:*.sar=01;31:*.rar=01;31:*.alz=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=0
1;31:*.7z=01;31:*.rz=01;31:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.j
pg=01;35:*.jpeg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm
=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35:*.png=01;3
```

Name: SHREEYA CHATTERJI
Class: CO16



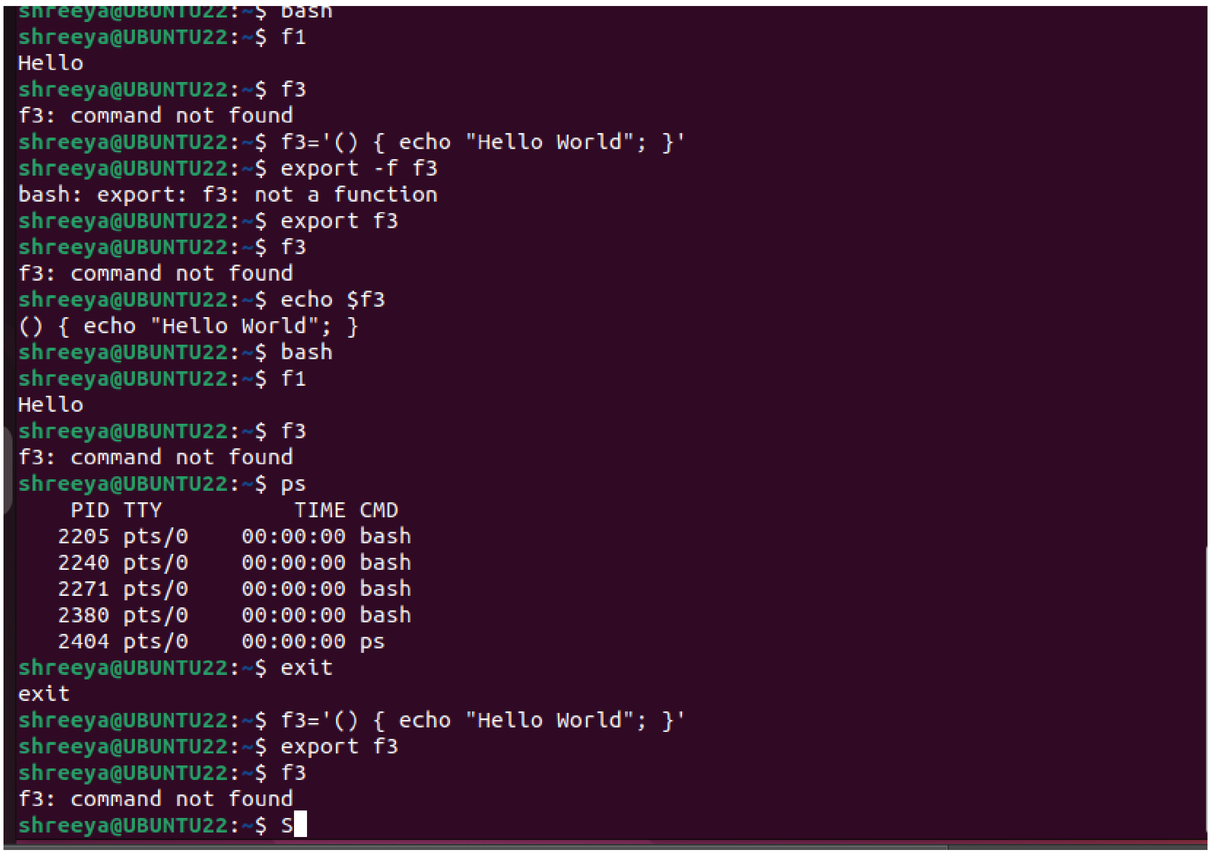
```
;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.f
lac=00;36:*.m4a=00;36:*.mid=00;36:*.midi=00;36:*.mka=00;36:*.mp3=00;36:*.mpc=00;36:*.ogg=0
0;36:*.ra=00;36:*.wav=00;36:*.oga=00;36:*.opus=00;36:*.spx=00;36:*.xspf=00;36:
XDG_CURRENT_DESKTOP=ubuntu:GNOME
VTE_VERSION=6800
WAYLAND_DISPLAY=wayland-0
GNOME_TERMINAL_SCREEN=/org/gnome/Terminal/screen/99f16902_f656_497f_8619_46e4624d60c8
GNOME_SETUP_DISPLAY=:1
LESSCLOSE=/usr/bin/lesspipe %s %s
XDG_SESSION_CLASS=user
TERM=xterm-256color
LESSOPEN=| /usr/bin/lesspipe %s
USER=shreeya
GNOME_TERMINAL_SERVICE=:1.95
DISPLAY=:0
SHLVL=1
QT_IM_MODULE=ibus
XDG_RUNTIME_DIR=/run/user/1000
XDG_DATA_DIRS=/usr/share/ubuntu:/usr/local/share/:/usr/share/:/var/lib/snapd/desktop
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/ga
mes:/snap/bin:/snap/bin
GDMSESSION=ubuntu
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
_=/usr/bin/env
shreeya@UBUNTU22:~$ echo $a1
aaaa
shreeya@UBUNTU22:~$ export a1
shreeya@UBUNTU22:~$ b1=bbbb
shreeya@UBUNTU22:~$ env|grep a1
a1=aaaa
shreeya@UBUNTU22:~$ env|grep b1
shreeya@UBUNTU22:~$ ps
    PID TTY          TIME CMD
   2205 pts/0    00:00:00 bash
   2239 pts/0    00:00:00 ps
shreeya@UBUNTU22:~$ bash
shreeya@UBUNTU22:~$ ps
    PID TTY          TIME CMD
   2205 pts/0    00:00:00 bash
   2240 pts/0    00:00:00 bash
```

Name: SHREEYA CHATTERJI
Class: CO16

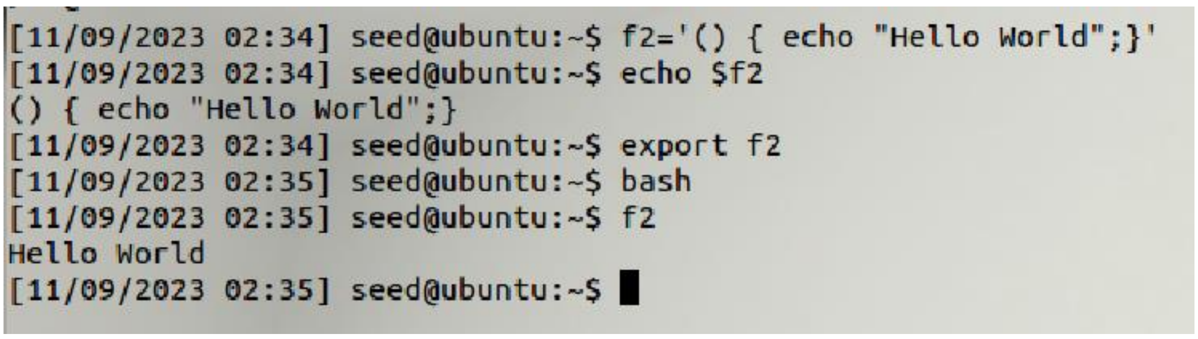


```
shreeya@UBUNTU22:~$ env|grep a1
a1=aaaa
shreeya@UBUNTU22:~$ env|grep b1
shreeya@UBUNTU22:~$ ps
    PID TTY          TIME CMD
   2205 pts/0    00:00:00 bash
   2239 pts/0    00:00:00 ps
shreeya@UBUNTU22:~$ bash
shreeya@UBUNTU22:~$ ps
    PID TTY          TIME CMD
   2205 pts/0    00:00:00 bash
   2240 pts/0    00:00:00 bash
   2247 pts/0    00:00:00 ps
shreeya@UBUNTU22:~$ f1(){echo "Hello"}
bash: syntax error near unexpected token `{echo'
shreeya@UBUNTU22:~$ f1(){ echo "Hello" }
> ^C
shreeya@UBUNTU22:~$ f1(){ echo "Hello"; }
shreeya@UBUNTU22:~$ f1
Hello
shreeya@UBUNTU22:~$ f2() { a1=1; echo $a1; }
shreeya@UBUNTU22:~$ f2
1
shreeya@UBUNTU22:~$ export f1
shreeya@UBUNTU22:~$ env|grep f1()
bash: syntax error near unexpected token `('
shreeya@UBUNTU22:~$ env|grep f1
GNOME_TERMINAL_SCREEN=/org/gnome/Terminal/screen/99f16902_f656_497f_8619_46e4624d60c8
shreeya@UBUNTU22:~$ export -f f1
shreeya@UBUNTU22:~$ env|grep f1
GNOME_TERMINAL_SCREEN=/org/gnome/Terminal/screen/99f16902_f656_497f_8619_46e4624d60c8
BASH_FUNC_f1%%=() {  echo "Hello"
shreeya@UBUNTU22:~$ f3='() { echo "Hello World"; }
> ^C
shreeya@UBUNTU22:~$ f3='() { echo "Hello World"; }'
shreeya@UBUNTU22:~$ echo f3
f3
shreeya@UBUNTU22:~$ bash
shreeya@UBUNTU22:~$ f1
Hello
```

Name: SHREEYA CHATTERJI
Class: CO16



```
shreeya@UBUNTU22:~$ bash
shreeya@UBUNTU22:~$ f1
Hello
shreeya@UBUNTU22:~$ f3
f3: command not found
shreeya@UBUNTU22:~$ f3='() { echo "Hello World"; }'
shreeya@UBUNTU22:~$ export -f f3
bash: export: f3: not a function
shreeya@UBUNTU22:~$ export f3
shreeya@UBUNTU22:~$ f3
f3: command not found
shreeya@UBUNTU22:~$ echo $f3
() { echo "Hello World"; }
shreeya@UBUNTU22:~$ bash
shreeya@UBUNTU22:~$ f1
Hello
shreeya@UBUNTU22:~$ f3
f3: command not found
shreeya@UBUNTU22:~$ ps
    PID TTY          TIME CMD
   2205 pts/0    00:00:00 bash
   2240 pts/0    00:00:00 bash
   2271 pts/0    00:00:00 bash
   2380 pts/0    00:00:00 bash
   2404 pts/0    00:00:00 ps
shreeya@UBUNTU22:~$ exit
exit
shreeya@UBUNTU22:~$ f3='() { echo "Hello World"; }'
shreeya@UBUNTU22:~$ export f3
shreeya@UBUNTU22:~$ f3
f3: command not found
shreeya@UBUNTU22:~$ S
```



```
[11/09/2023 02:34] seed@ubuntu:~$ f2='() { echo "Hello World";}'
[11/09/2023 02:34] seed@ubuntu:~$ echo $f2
() { echo "Hello World";}
[11/09/2023 02:34] seed@ubuntu:~$ export f2
[11/09/2023 02:35] seed@ubuntu:~$ bash
[11/09/2023 02:35] seed@ubuntu:~$ f2
Hello World
[11/09/2023 02:35] seed@ubuntu:~$ 
```