CS4371 LAB 1

Dhruve Mistry

Due 2/3 with extension 2/5

Analysis on HTTP using wireshark and how packets are sent from machine to machine.
Using an existing wireshark file to find out malicious packets that a user received.
Using pyshark as well to understand a different way of collecting and capturing packets.
Understanding the CIA triad and pin pointing which one is violated during each example.

Mistry, Dhruve A
[Email address]

Task 1 – Install wireshark

First we are asked to find if any updates are needed for current applications/settings etc. by using,

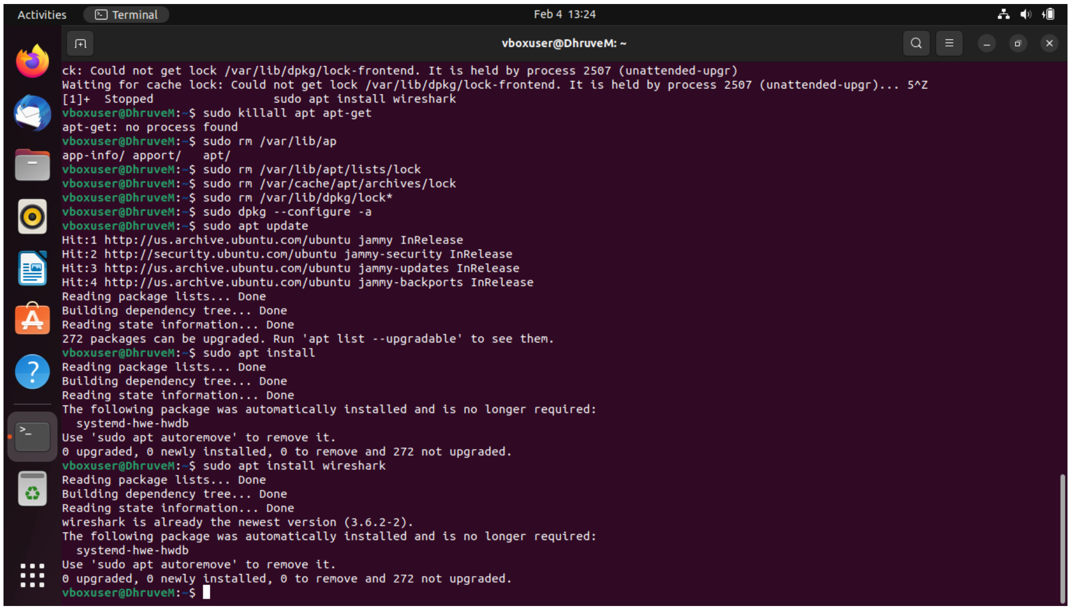
```
sudo apt update
```

If the terminal detects any updates, we are then required to install the updates by using,

```
sudo apt upgrade
```

After it's done, we have to install wireshark.

```
sudo apt install wireshark
```



```
ck: Could not get lock /var/lib/dpkg/lock-frontend. It is held by process 2507 (unattended-upgr)
Waiting for cache lock: Could not get lock /var/lib/dpkg/lock-frontend. It is held by process 2507 (unattended-upgr)... 5^Z
[1]+  Stopped                 sudo apt install wireshark
vboxuser@DhruveM:~$ sudo killall apt apt-get
apt-get: no process found
vboxuser@DhruveM:~$ sudo rm /var/lib/ap
app-info/ apport/   apt/
vboxuser@DhruveM:~$ sudo rm /var/lib/apt/lists/lock
vboxuser@DhruveM:~$ sudo rm /var/cache/apt/archives/lock
vboxuser@DhruveM:~$ sudo rm /var/lib/dpkg/lock*
vboxuser@DhruveM:~$ sudo dpkg --configure -a
vboxuser@DhruveM:~$ sudo apt update
Hit:1 http://us.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu jammy-backports InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
272 packages can be upgraded. Run 'apt list --upgradable' to see them.
vboxuser@DhruveM:~$ sudo apt install
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  systemd-hwe-hwdb
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 272 not upgraded.
vboxuser@DhruveM:~$ sudo apt install wireshark
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
wireshark is already the newest version (3.6.2-2).
The following package was automatically installed and is no longer required:
  systemd-hwe-hwdb
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 272 not upgraded.
vboxuser@DhruveM:~$
```

Note: if there's a problem with the apt running,

Try this first,

```
sudo killall apt apt-get
```

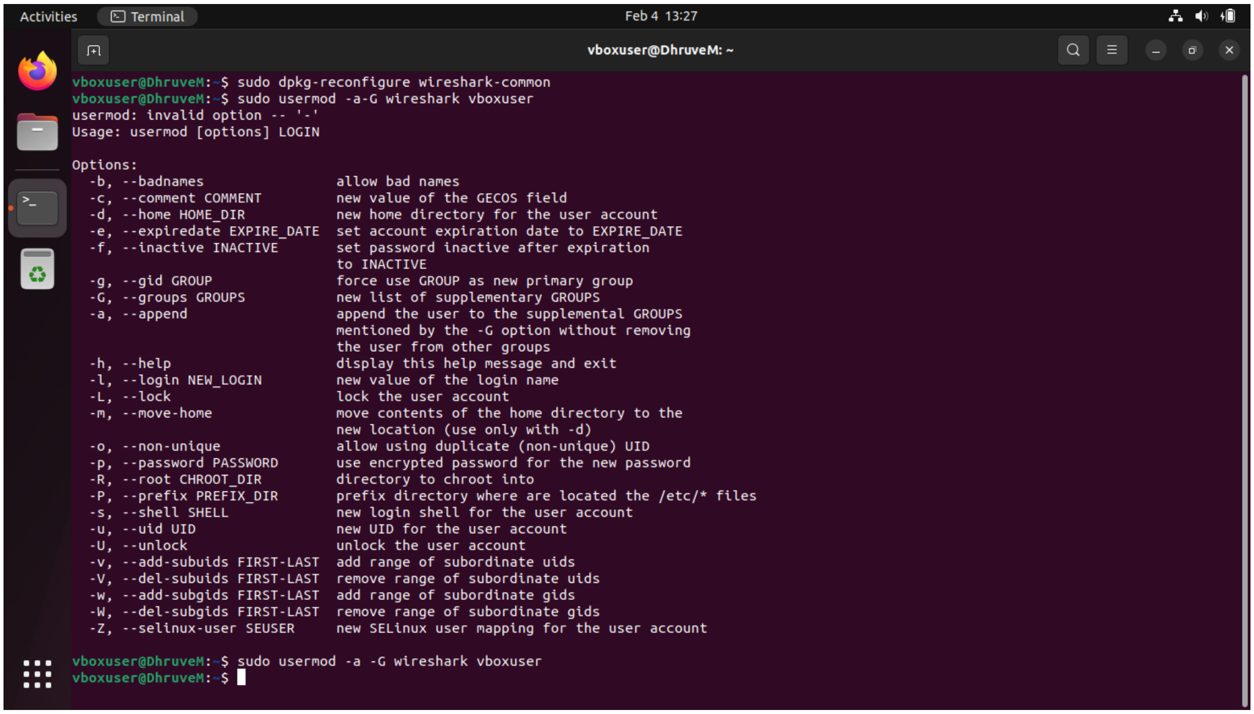
If that doesn't work, **(use at own risk, can break vm)**

```
sudo rm /var/lib/apt/lists/lock

sudo rm /var/cache/apt/archives/lock

sudo rm /var/lib/dpkg/lock*

sudo dpkg –configure -a
```



```
vboxuser@DhruveM:~$ sudo dpkg-reconfigure wireshark-common
vboxuser@DhruveM:~$ sudo usermod -a-G wireshark vboxuser
usermod: invalid option -- '-'
Usage: usermod [options] LOGIN

Options:
  -b, --badnames                allow bad names
  -c, --comment COMMENT         new value of the GECOS field
  -d, --home HOME_DIR           new home directory for the user account
  -e, --expiredate EXPIRE_DATE  set account expiration date to EXPIRE_DATE
  -f, --inactive INACTIVE       set password inactive after expiration
                                to INACTIVE
  -g, --gid GROUP               force use GROUP as new primary group
  -G, --groups GROUPS           new list of supplementary GROUPS
  -a, --append                  append the user to the supplemental GROUPS
                                mentioned by the -G option without removing
                                the user from other groups
  -h, --help                    display this help message and exit
  -l, --login NEW_LOGIN         new value of the login name
  -L, --lock                    lock the user account
  -m, --move-home               move contents of the home directory to the
                                new location (use only with -d)
  -o, --non-unique              allow using duplicate (non-unique) UID
  -p, --password PASSWORD       use encrypted password for the new password
  -R, --root CHROOT_DIR         directory to chroot into
  -P, --prefix PREFIX_DIR       prefix directory where are located the /etc/* files
  -s, --shell SHELL             new login shell for the user account
  -u, --uid UID                 new UID for the user account
  -U, --unlock                  unlock the user account
  -v, --add-subuids FIRST-LAST  add range of subordinate uids
  -V, --del-subuids FIRST-LAST  remove range of subordinate uids
  -w, --add-subgids FIRST-LAST  add range of subordinate gids
  -W, --del-subgids FIRST-LAST  remove range of subordinate gids
  -Z, --selinux-user SEUSER     new SELinux user mapping for the user account

vboxuser@DhruveM:~$ sudo usermod -a -G wireshark vboxuser
vboxuser@DhruveM:~$ 
```

Once wireshark is installed, you have to give it super user access, this way it has the ability to capture internet protocols.

Reboot the system using,

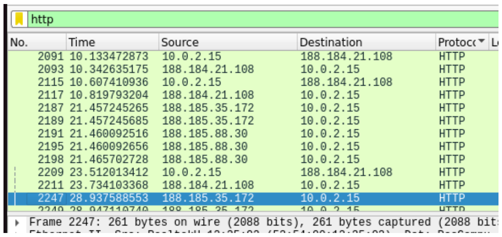
```
sudo reboot
```

After that, start capturing! *Shown in Task 2, page 3.*

Task 2 – Capturing HTTP

Disclaimer – Canvas would not detect as HTTP so I used a different website with HTTP, link: info.cern.ch

Once filtered with HTTP,

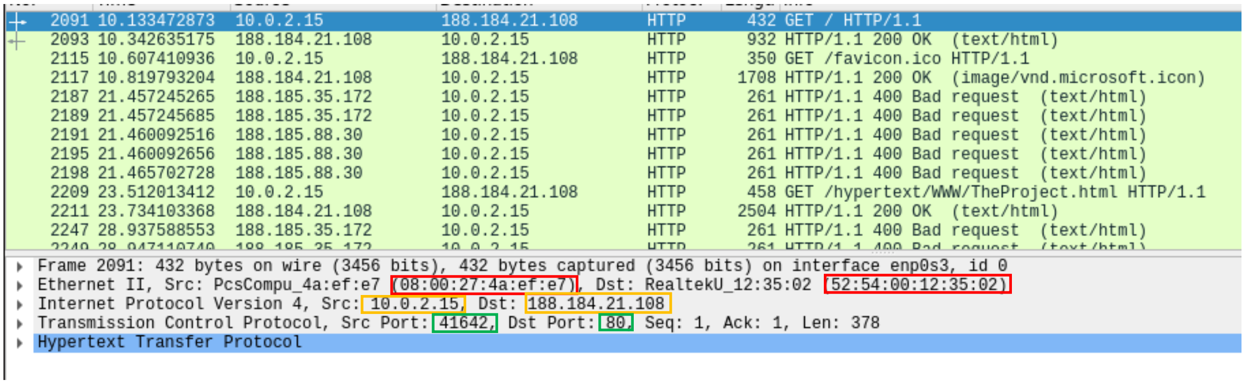


```
http
No.  Time         Source          Destination     Protocol
2091 10.133472873 10.0.2.15       188.184.21.108  HTTP
2093 10.342635175 188.184.21.108  10.0.2.15       HTTP
2115 10.607410936 10.0.2.15       188.184.21.108  HTTP
2117 10.819793204 188.184.21.108  10.0.2.15       HTTP
2187 21.457245265 188.185.35.172  10.0.2.15       HTTP
2189 21.457245685 188.185.35.172  10.0.2.15       HTTP
2191 21.460092516 188.185.88.30   10.0.2.15       HTTP
2195 21.460092656 188.185.88.30   10.0.2.15       HTTP
2198 21.465702728 188.185.88.30   10.0.2.15       HTTP
2209 23.512013412 10.0.2.15       188.184.21.108  HTTP
2211 23.734103368 188.184.21.108  10.0.2.15       HTTP
2247 28.937588553 188.185.35.172  10.0.2.15       HTTP
▸ Frame 2247: 261 bytes on wire (2088 bits), 261 bytes captured (2088 bit
```

You are able to find the source and destination MAC along with the IP. The first box on each line is the destination and second is the destination.

Legend:

- Red – MAC
- Orange – IP
- Green – TCP port

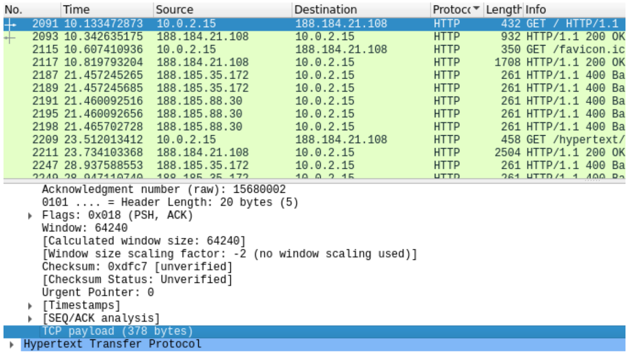


```
2091 10.133472873  10.0.2.15       188.184.21.108  HTTP   432 GET / HTTP/1.1
2093 10.342635175  188.184.21.108  10.0.2.15       HTTP   932 HTTP/1.1 200 OK  (text/html)
2115 10.607410936  10.0.2.15       188.184.21.108  HTTP   350 GET /favicon.ico HTTP/1.1
2117 10.819793204  188.184.21.108  10.0.2.15       HTTP  1708 HTTP/1.1 200 OK  (image/vnd.microsoft.icon)
2187 21.457245265  188.185.35.172  10.0.2.15       HTTP   261 HTTP/1.1 400 Bad request  (text/html)
2189 21.457245685  188.185.35.172  10.0.2.15       HTTP   261 HTTP/1.1 400 Bad request  (text/html)
2191 21.460092516  188.185.88.30   10.0.2.15       HTTP   261 HTTP/1.1 400 Bad request  (text/html)
2195 21.460092656  188.185.88.30   10.0.2.15       HTTP   261 HTTP/1.1 400 Bad request  (text/html)
2198 21.465702728  188.185.88.30   10.0.2.15       HTTP   261 HTTP/1.1 400 Bad request  (text/html)
2209 23.512013412  10.0.2.15       188.184.21.108  HTTP   458 GET /hypertext/WWW/TheProject.html HTTP/1.1
2211 23.734103368  188.184.21.108  10.0.2.15       HTTP  2504 HTTP/1.1 200 OK  (text/html)
2247 28.937588553  188.185.35.172  10.0.2.15       HTTP   261 HTTP/1.1 400 Bad request  (text/html)

▸ Frame 2091: 432 bytes on wire (3456 bits), 432 bytes captured (3456 bits) on interface enp0s3, id 0
▸ Ethernet II, Src: PcsCompu_4a:ef:e7 (08:00:27:4a:ef:e7), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)
▸ Internet Protocol Version 4, Src: 10.0.2.15, Dst: 188.184.21.108
▸ Transmission Control Protocol, Src Port: 41642, Dst Port: 80, Seq: 1, Ack: 1, Len: 378
▸ Hypertext Transfer Protocol
```

	MAC address	IP	TCP Port
Source	08:00:27:4a:ef:e7	10.0.2.15	41642
Destination	52:54:00:12:35:02	188.184.21.108	80

The TCP payload is 378 bytes.

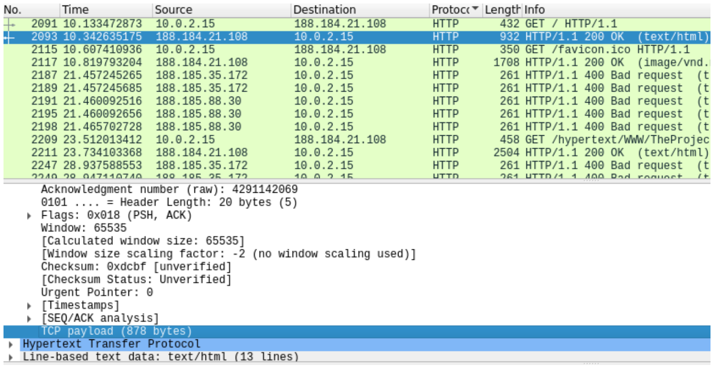


No.	Time	Source	Destination	Protocol	Length	Info
2091	10.133472873	10.0.2.15	188.184.21.108	HTTP	432	GET / HTTP/1.1
2093	10.342635175	188.184.21.108	10.0.2.15	HTTP	932	HTTP/1.1 200 OK
2115	10.607410936	10.0.2.15	188.184.21.108	HTTP	350	GET /favicon.ic
2117	10.819793204	188.184.21.108	10.0.2.15	HTTP	1708	HTTP/1.1 200 OK
2187	21.457245265	188.185.35.172	10.0.2.15	HTTP	261	HTTP/1.1 400 Ba
2189	21.457245685	188.185.35.172	10.0.2.15	HTTP	261	HTTP/1.1 400 Ba
2191	21.460092516	188.185.88.30	10.0.2.15	HTTP	261	HTTP/1.1 400 Ba
2195	21.460092656	188.185.88.30	10.0.2.15	HTTP	261	HTTP/1.1 400 Ba
2198	21.465702728	188.185.88.30	10.0.2.15	HTTP	261	HTTP/1.1 400 Ba
2209	23.512013412	10.0.2.15	188.184.21.108	HTTP	458	GET /hypertext/
2211	23.734103368	188.184.21.108	10.0.2.15	HTTP	2504	HTTP/1.1 200 OK
2247	28.937588553	188.185.35.172	10.0.2.15	HTTP	261	HTTP/1.1 400 Ba
2249	28.947110740	188.185.35.172	10.0.2.15	HTTP	261	HTTP/1.1 400 Ba

```
    Acknowledgment number (raw): 15680002
    0101 .... = Header Length: 20 bytes (5)
  ▸ Flags: 0x018 (PSH, ACK)
    Window: 64240
    [Calculated window size: 64240]
    [Window size scaling factor: -2 (no window scaling used)]
    Checksum: 0xdfc7 [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  ▸ [Timestamps]
  ▸ [SEQ/ACK analysis]
    TCP payload (378 bytes)
▸ Hypertext Transfer Protocol
```

To locate the return, you want to focus on the first line item that has the Destination IP as the Source. E.g., 188.184.21.108 is now the Source and my IP 10.0.2.15 is now the Destination.

The return TCP payload is 878 bytes.



No.	Time	Source	Destination	Protocol	Length	Info
2091	10.133472873	10.0.2.15	188.184.21.108	HTTP	432	GET / HTTP/1.1
2093	10.342635175	188.184.21.108	10.0.2.15	HTTP	932	HTTP/1.1 200 OK (text/html)
2115	10.607410936	10.0.2.15	188.184.21.108	HTTP	350	GET /favicon.ico HTTP/1.1
2117	10.819793204	188.184.21.108	10.0.2.15	HTTP	1708	HTTP/1.1 200 OK (image/vnd.
2187	21.457245265	188.185.35.172	10.0.2.15	HTTP	261	HTTP/1.1 400 Bad request (t
2189	21.457245685	188.185.35.172	10.0.2.15	HTTP	261	HTTP/1.1 400 Bad request (t
2191	21.460092516	188.185.88.30	10.0.2.15	HTTP	261	HTTP/1.1 400 Bad request (t
2195	21.460092656	188.185.88.30	10.0.2.15	HTTP	261	HTTP/1.1 400 Bad request (t
2198	21.465702728	188.185.88.30	10.0.2.15	HTTP	261	HTTP/1.1 400 Bad request (t
2209	23.512013412	10.0.2.15	188.184.21.108	HTTP	458	GET /hypertext/WWW/TheProjec
2211	23.734103368	188.184.21.108	10.0.2.15	HTTP	2504	HTTP/1.1 200 OK (text/html)
2247	28.937588553	188.185.35.172	10.0.2.15	HTTP	261	HTTP/1.1 400 Bad request (t
2249	28.947110740	188.185.35.172	10.0.2.15	HTTP	261	HTTP/1.1 400 Bad request (t

```
    Acknowledgment number (raw): 4291142069
    0101 .... = Header Length: 20 bytes (5)
  ▸ Flags: 0x018 (PSH, ACK)
    Window: 65535
    [Calculated window size: 65535]
    [Window size scaling factor: -2 (no window scaling used)]
    Checksum: 0xdcbf [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  ▸ [Timestamps]
  ▸ [SEQ/ACK analysis]
    TCP payload (878 bytes)
▸ Hypertext Transfer Protocol
▸ Line-based text data: text/html (13 lines)
```

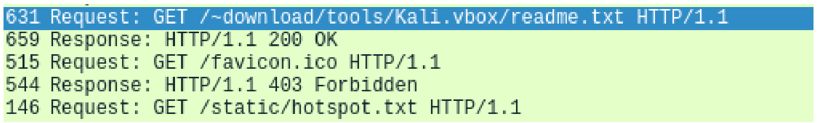
Task 3 - Open the .pcapng

In this task we are asked to open the hw1.q2.pcapng file provided and open in wireshark to see the sniffed packets when a user tried to download a txt file from a website.

Upon opening the file in wireshark, I was able to locate the IP address of the website is 192.168.0.86

I was unsuccessful on locating the URLs that the user browsed along with the contents of the text file.

I was however able to find the file names.



```
631 Request: GET /~download/tools/Kali.vbox/readme.txt HTTP/1.1
659 Response: HTTP/1.1 200 OK
515 Request: GET /favicon.ico HTTP/1.1
544 Response: HTTP/1.1 403 Forbidden
146 Request: GET /static/hotspot.txt HTTP/1.1
```

readme.txt and hotspot.txt

Task 4 – pyshark

First we are instructed on installing tshark,

```
sudo apt install tshark
```

Since this is a brand new OS you will need to do,

```
sudo apt install python3-pip
```

This is because the OS does not come with python.

Lastly, to install pyshark, you will need to do,

```
pip install pyshark
```

Open the python terminal by doing,

```
python3
```

You should see three arrows, “>>>”, this means python3 terminal opened successfully.

I’ve submitted an output.txt that shows the script of the commands ran as well as the output.

```
:~$ python3

>>>import pyshark

>>>capture = pyshark.LiveCapture(interface=’enp0s3’)

>>>capture.sniff(timeout=5)

>>>for pkt in capture:

...  print(pkt) // Since python is space sensitive, tab over on

                   this line

...
```

Check the output.txt for the rest.

Task 5 – CIA Triad

For each scenario, we have to choose **ONE** out of the CIA that each has violated and come up with a defense measure or detect the security violation.

a) John copies Mary’s homework.

Confidentiality. A way we can prevent this is by making the persons do the homework in the class this way it’ll show their own work. If that doesn’t work, compare previous work to determine if it’s their own style.

You can use a 2FA system to confirm this, or access control, who has the rights to hold on to the assignment. Mary should’ve kept in her hands all along to prevent data leak.

b) Paul crashes Linda’s system.

Availability.

c) Carol changes the amount of Angelo’s check from $100 to $1000.

Integrity. You can have the bank call the user to confirm the amount before it is cashed out. Like a 2FA system.

d) Gina forges Roger’s signature on a deed.

Integrity. You can combat this by needing more than just a signature to prove it’s that person. Like a 2FA system.

e) Rhonda deletes all web services from the university’s web servers.

Availability. You can failsafe RAID system to claim redundancy.

f) Henry spoofs Julie’s IP address to gain access to her computer.

Integrity. You can use a firewall to prevent the private IP being leaked.