

# Programming Assignment 01 - Using command-line utilities for network debugging

## Assignment Report - Saksham Singh (2022434)

- CSE232 Computer Networks Assignment

### 1. **ifconfig** command

**1.a Learn to use the ifconfig command, and figure out the IP address of your network interface. Put a screenshot.**

- The **ifconfig** command displays all the network interfaces of the device and allows us to configure their parameters
- using **ifconfig** on device outputs the following:

```
Lecture ifconfig
1 atta lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
        options=1203<RXCSUM,TXCSUM,TXSTATUS,SW_TIMESTAMP>
        inet 127.0.0.1 netmask 0xffff00000
        inet6 ::1 prefixlen 128
        inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
        nd6 options=201<PERFORMNUD,DAD>
gif0: flags=8010<POINTOPOINT,MULTICAST> mtu 1280
stf0: flags=0<> mtu 1280
anpi0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=400<CHANNEL_IO>
        ether 1a:9d:22:e1:77:e3
        media: none
        status: inactive
anpi1: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=400<CHANNEL_IO>
        ether 1a:9d:22:e1:77:e4
        media: none
        status: inactive
en3: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=400<CHANNEL_IO>
        ether 1a:9d:22:e1:77:c3
        nd6 options=201<PERFORMNUD,DAD>
        media: none
        status: inactive
en4: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=400<CHANNEL_IO>
        ether 1a:9d:22:e1:77:c4
        nd6 options=201<PERFORMNUD,DAD>
        media: none
        status: inactive
en1: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
        options=460<TS04,TS06,CHANNEL_IO>
        ether 36:00:c7:60:87:40
        media: autosel <full-duplex>
        status: inactive
en2: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
        options=460<TS04,TS06,CHANNEL_IO>
        ether 36:00:c7:60:87:44
        media: autosel <full-duplex>
        status: inactive
```

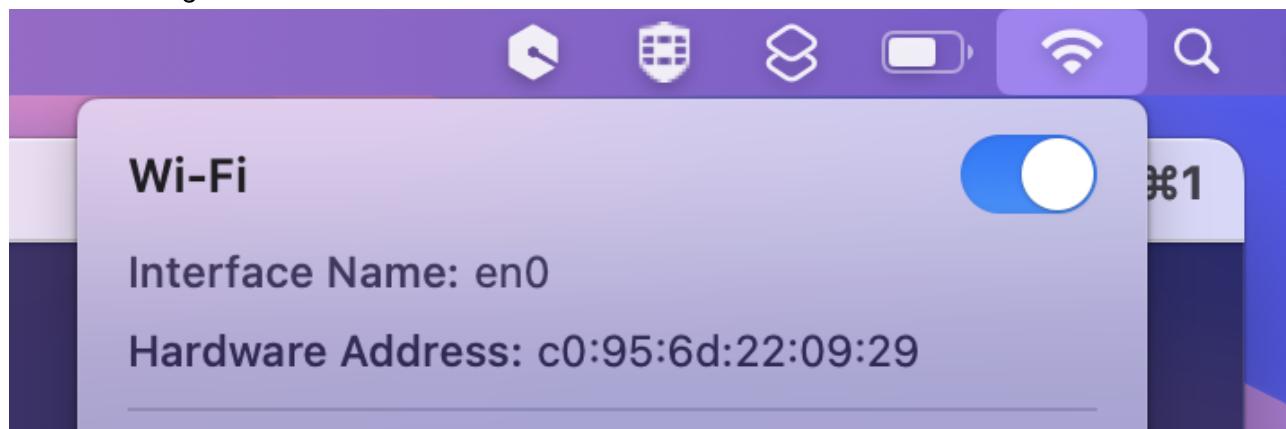
```

status: inactive
Yester ap1: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=6460<TS04,TS06,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
        ether aa:dd:ff:3e:3f:ff
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect (none)
        status: inactive
OMORRO en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=6460<TS04,TS06,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
        ether a6:08:01:1a:2b:6d
        inet6 fe80::6e:8668:9414:72af%en0 prefixlen 64 secured scopeid 0xb
        inet 192.168.42.154 netmask 0xfffffe000 broadcast 192.168.63.255
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect
        status: active
4 PM awdl0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=6460<TS04,TS06,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
        ether ae:7a:eb:75:34:64
        inet6 fe80::ac7a:ebff:fe75:3464%awdl0 prefixlen 64 scopeid 0xd
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect
        status: active
31° llw0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=400<CHANNEL_IO>
        ether ae:7a:eb:75:34:64
        inet6 fe80::ac7a:ebff:fe75:3464%llw0 prefixlen 64 scopeid 0xe
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect (none)
utun0: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1500
        inet6 fe80::5bf3:c049:a730:bfd3%utun0 prefixlen 64 scopeid 0xf
        nd6 options=201<PERFORMNUD,DAD>
utun1: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1380
        inet6 fe80::435a:8f00:dcb1:e2bd%utun1 prefixlen 64 scopeid 0x10
        nd6 options=201<PERFORMNUD,DAD>
utun2: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 2000
        inet6 fe80::cd65:15f:191d:7e9f%utun2 prefixlen 64 scopeid 0x11
        nd6 options=201<PERFORMNUD,DAD>
utun3: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1000
        inet6 fe80::ce81:b1c:bd2c:69e%utun3 prefixlen 64 scopeid 0x12
        nd6 options=201<PERFORMNUD,DAD>
utun4: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1380
        inet6 fe80::8cbc:99e4:8a76:22dd%utun4 prefixlen 64 scopeid 0x13
        nd6 options=201<PERFORMNUD,DAD>
utun5: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1380
        inet6 fe80::eef4:dd8f:117a:ec5a%utun5 prefixlen 64 scopeid 0x14
        nd6 options=201<PERFORMNUD,DAD>

```

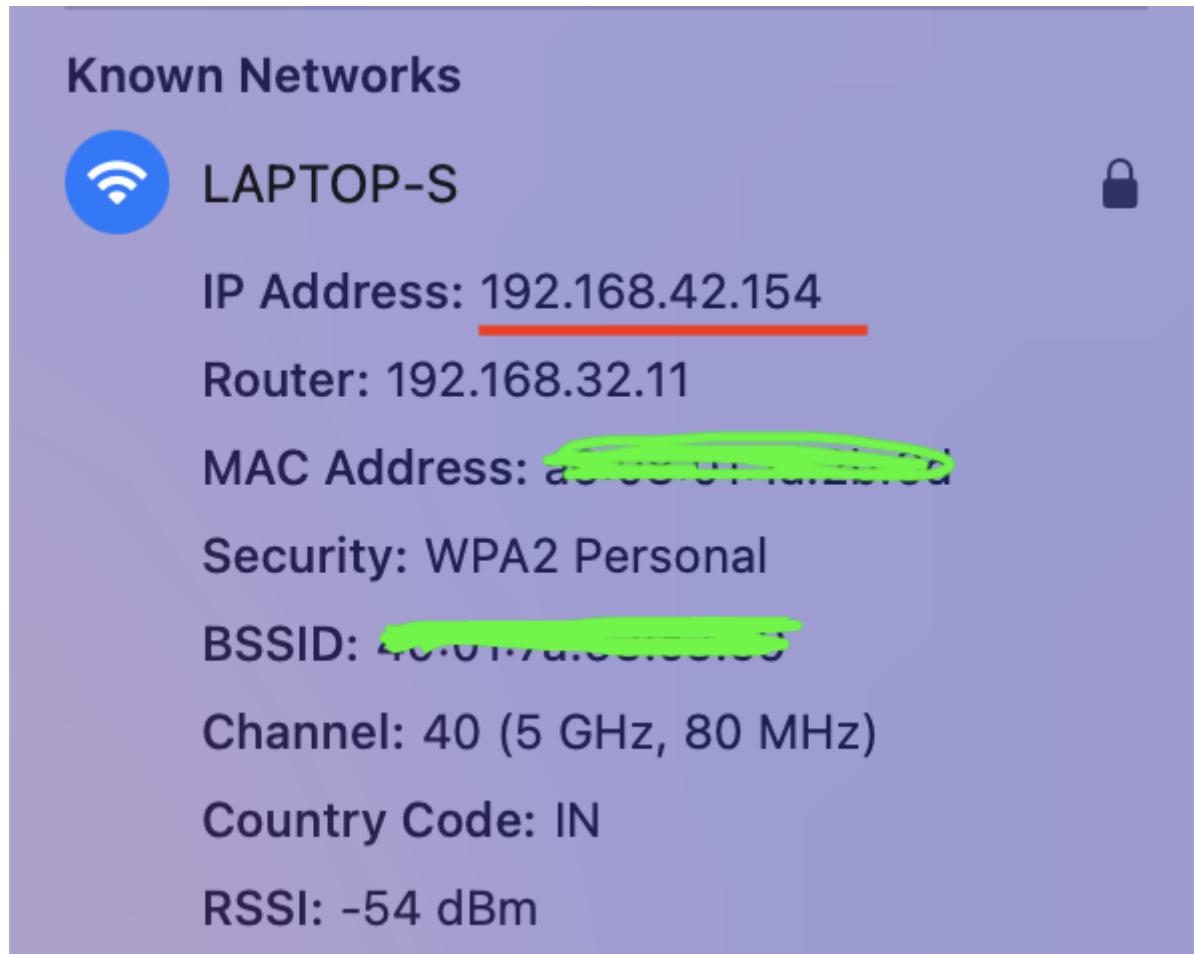
these all are the network interfaces on my device

- **en0** is the primary WiFi interface of the device, which can be cross-checked from macbook's own internal settings like follows:



```
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
      options=6460<TS04,TS06,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
      ether a6:08:01:1a:2b:6d
      inet6 fe80::6e:8668:9414:72af%en0 prefixlen 64 secured scopeid 0xb
      inet 192.168.42.154 netmask 0xfffffe000 broadcast 192.168.63.255
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect
        status: active
```

The red-rectangle of the above image shows the **ipv4** address of the **en0** network interface, which is **192.168.42.154**. (note that the hardware address don't match, as by default macOS randomises the mac address for every WiFi network.) This again can be cross-checked by internal setting of macbook as follows:



- some other interfaces that we can see are **lo0** which is the loopback interface, **en1** which is the Thunderbolt 1 interface, **en2** which is the Thunderbolt 2 interface, **en3** which is the Ethernet interface, **bridge0** which is the thunderbolt bridge, and many more.

**1.b Go to the webpage <https://www.whatismyip.com> and find out what IP is shown for your machine. Are they identical or different? Why?**

Below is the public IP address of my device as shown on the website <https://www.whatismyip.com>:

## What Is My IP?

My Public IPv4: [103.25.231.125](https://www.whatismyip.com/ip/103.25.231.125) 

My Public IPv6: Not Detected

My IP Location: Noida, UP IN 

My ISP: Indraprastha Institute of Information Technology Delhi 

As we can see, the IP address - **103.25.231.125** is different from the IP address of the **en0** network interface - **192.168.42.154**. This is because the **en0** network interface is a private IP address, which is assigned by the router to the device on the local network. The public IP address is the address of the router, which is assigned by the ISP. The router then assigns private IP addresses to the devices on the local network. The public IP address is used to identify the device on the internet, while the private IP address is used to identify the device on the local network. This translation is explained by a process called **NAT - Network Address Translation**.

## 2. **ifconfig** continued

### 2.a Change the IP address of your network interface using the command line. Put a screenshot that shows the change. Revert to the original IP address.

- To change the IP address of any interface, it can be done simply using

```
sudo ifconfig <interface_name> <new_ip_addr>
```

where **<interface\_name>** is the name of the interface whose IP address is to be changed, and **<new\_ip\_addr>** is the new IP address to be assigned to the interface.

- so for my device, changing to a random IP address, let's say - **192.168.100.100**, I have to do

```
sudo ifconfig en0 192.168.100.100
```

which can be verified by using `ifconfig en0` command

```
>> ifconfig en0 192.168.100.100
ifconfig: ioctl (SIOCDIFADDR): permission denied
>> sudo ifconfig en0 192.168.100.100
Password:
>> ifconfig en0
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
      options=6460<TS04,TS06,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
      ether a6:08:01:1a:2b:6d
      inet6 fe80::6e:8668:9414:72af%en0 prefixlen 64 secured scopeid 0xb
      inet 192.168.100.100 netmask 0xffffffff00 broadcast 192.168.100.255
      nd6 options=201<PERFORMNUD,DAD>
      media: autoselect
      status: active
```

- to revert back to the original IP address, I can simply do

```
sudo ifconfig en0 192.168.42.154
```

```
>> sudo ifconfig en0 192.168.100.100
Password:
>> ifconfig en0
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
      options=6460<TS04,TS06,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
      ether a6:08:01:1a:2b:6d
      inet6 fe80::6e:8668:9414:72af%en0 prefixlen 64 secured scopeid 0xb
      inet 192.168.100.100 netmask 0xffffffff00 broadcast 192.168.100.255
      nd6 options=201<PERFORMNUD,DAD>
      media: autoselect
      status: active
>> sudo ifconfig en0 192.168.42.154
>> ifconfig en0
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
      options=6460<TS04,TS06,CHANNEL_IO,PARTIAL_CSUM,ZEROINVERT_CSUM>
      ether a6:08:01:1a:2b:6d
      inet6 fe80::6e:8668:9414:72af%en0 prefixlen 64 secured scopeid 0xb
      inet 192.168.42.154 netmask 0xffffffff00 broadcast 192.168.42.255
      nd6 options=201<PERFORMNUD,DAD>
      media: autoselect
      status: active
```

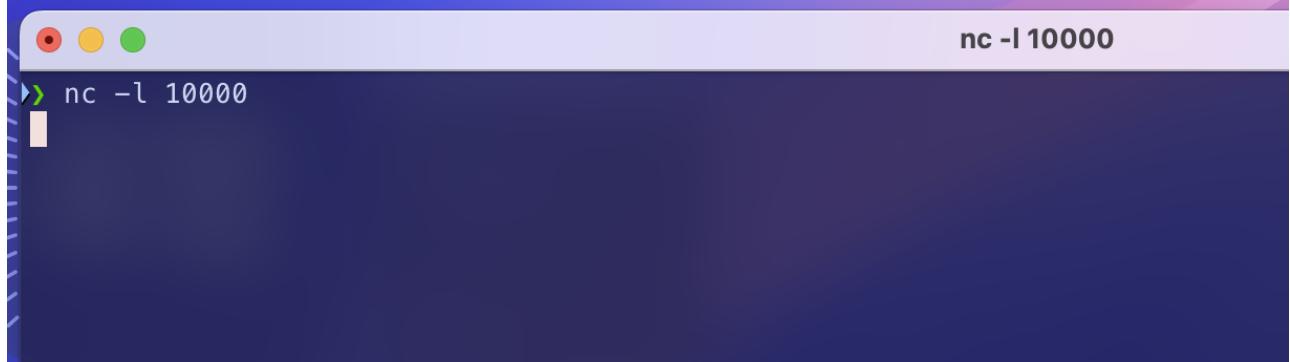
### 3. `netcat` command

**3.a Use “`netcat`” to set up a TCP client/server connection between your VM and host machine. If you are not using a VM, you can set up the connection with `localhost`. Put a screenshot.**

- using `netcat` command involves running the `nc` command in two terminals, one acting as the server and the other as the client.
- since I don't have access to a VM, I will be setting up the connection between my host machine and `localhost` by running the server on my host machine and the client on `localhost`, by using two different terminals.
- the server can be started on the host machine by running the command

```
nc -l <port_num>
```

where `<port_num>` is the port number on which the server will listen for incoming connections.

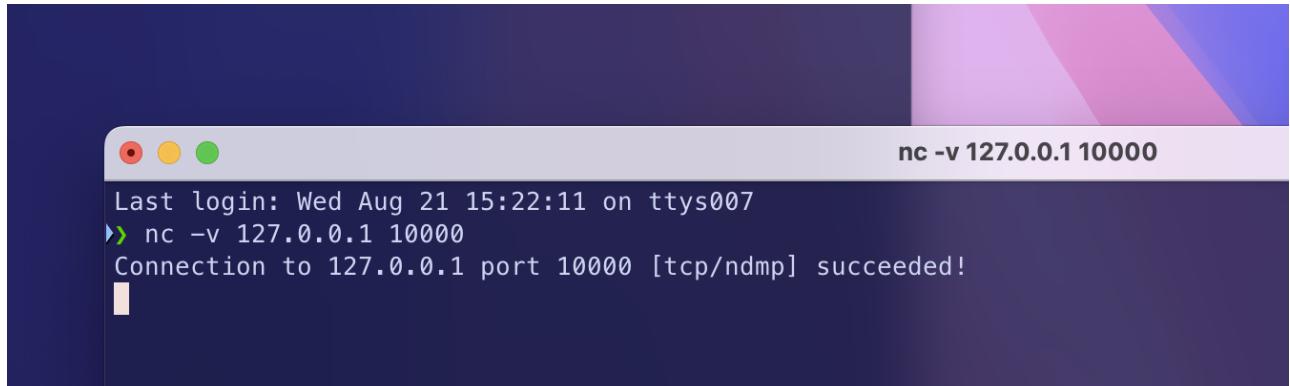


A terminal window with a dark blue background and light blue header bar. The title bar says "nc -l 10000". The command "nc -l 10000" is typed in the terminal. A small white square icon is visible on the left edge of the window.

- the client can be started on the localhost by running the command

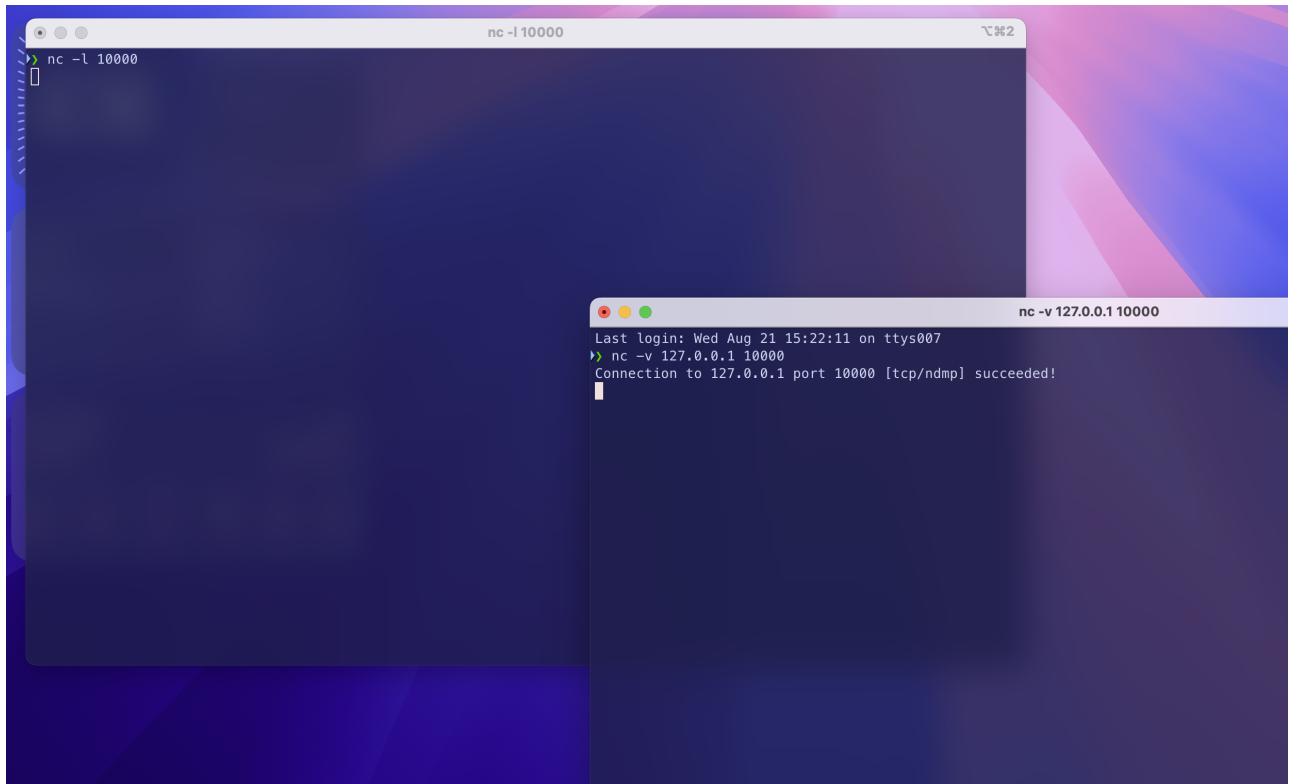
```
nc -v <host_ip> <port_num>
```

where `<host_ip>` is the IP address of the host machine, and `<port_num>` is the port number on which the server is listening, here the `host_ip` will be `127.0.0.1` (localhost).

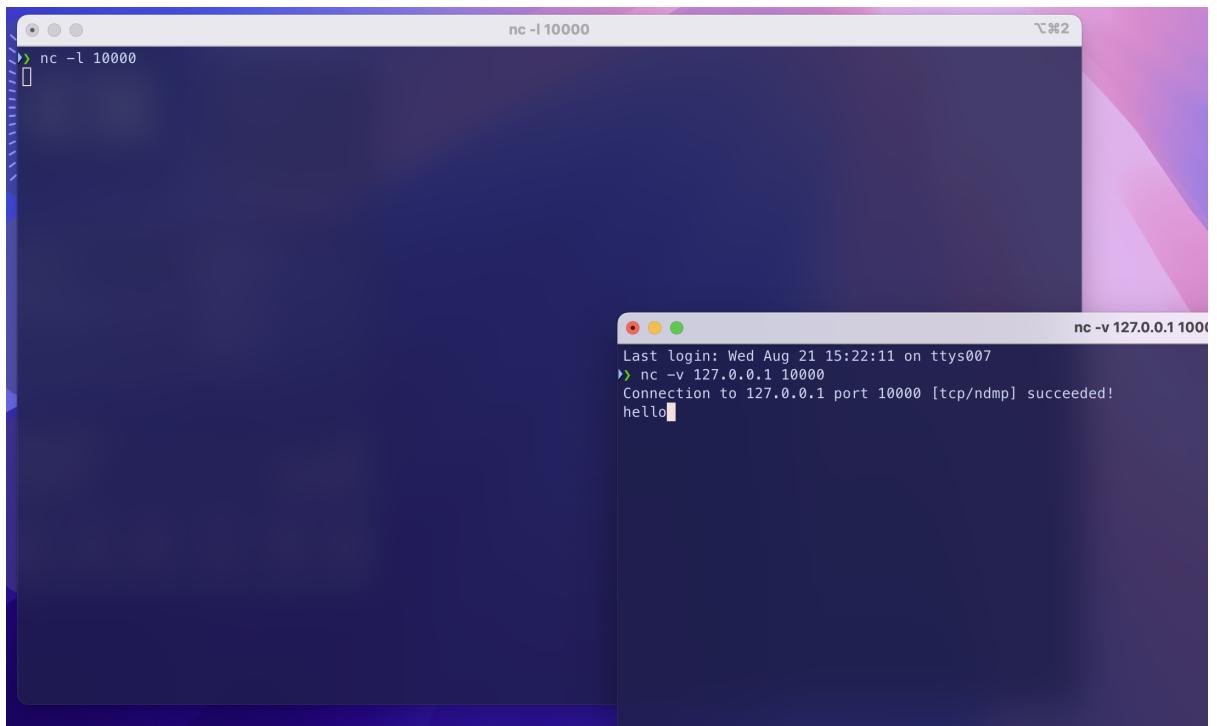


A terminal window with a dark blue background and light blue header bar. The title bar says "nc -v 127.0.0.1 10000". The command "nc -v 127.0.0.1 10000" is typed in the terminal. The output shows "Last login: Wed Aug 21 15:22:11 on ttys007" and "Connection to 127.0.0.1 port 10000 [tcp/ndmp] succeeded!". A small white square icon is visible on the left edge of the window.

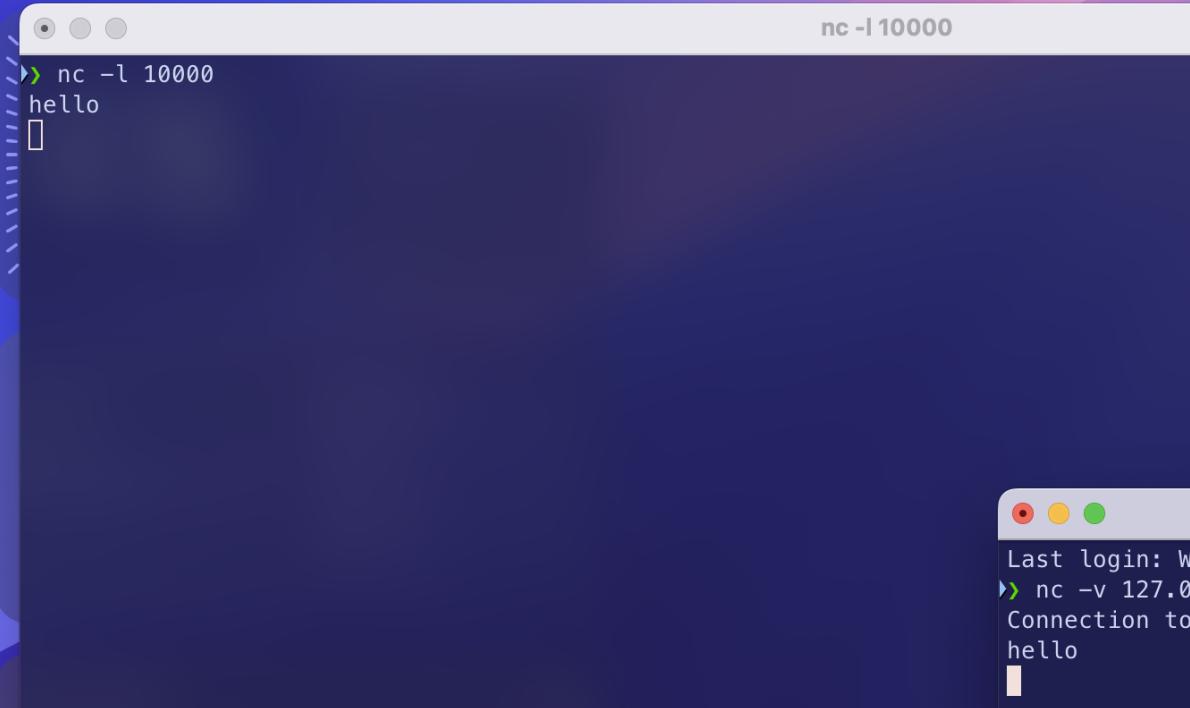
- the connection is established between the server and the client, and the client can send messages to the server, which will be displayed on the server terminal.



- **sending a message from the client to the server**



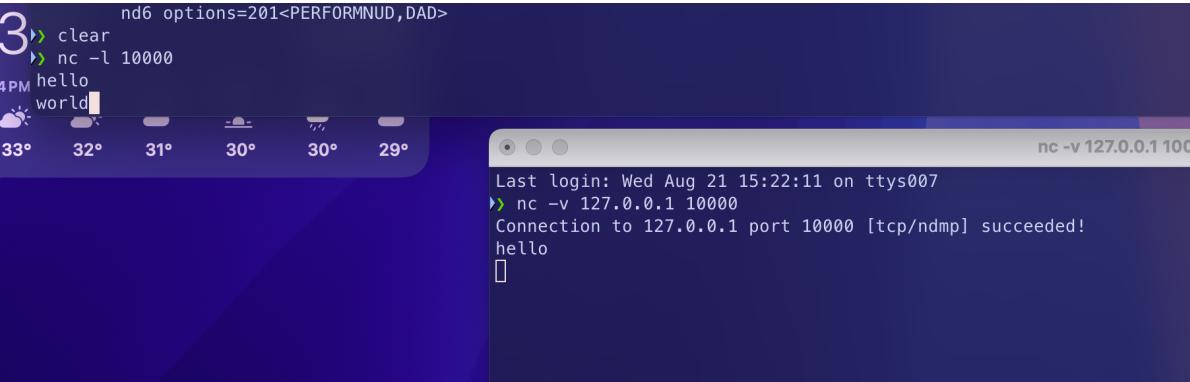
- **receiving the message on the server**



```
nc -l 10000
hello
```

```
Last login: Wed Aug 21 15:22:11 on ttys007
nc -v 127.0.0.1 10000
Connection to 127.0.0.1 port 10000 [tcp/ndmp] succeeded!
hello
```

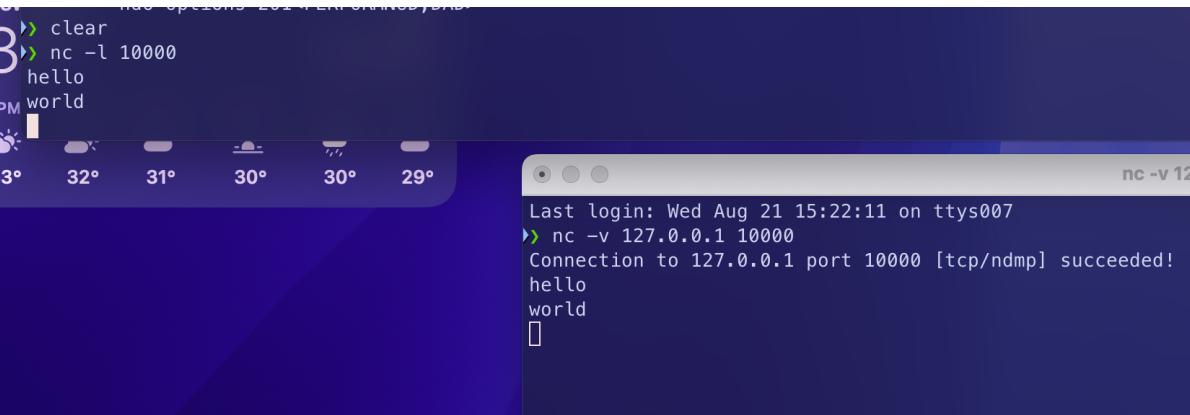
- sending a message from the server to the client



```
nd6 options=201<PERFORMNUD,DAD>
3 clear
4PM hello
world
```

```
Last login: Wed Aug 21 15:22:11 on ttys007
nc -v 127.0.0.1 10000
Connection to 127.0.0.1 port 10000 [tcp/ndmp] succeeded!
hello
world
```

- receiving the message on the client



```
nd6 options=201<PERFORMNUD,DAD>
3 clear
4PM nc -l 10000
hello
world
```

```
Last login: Wed Aug 21 15:22:11 on ttys007
nc -v 127.0.0.1 10000
Connection to 127.0.0.1 port 10000 [tcp/ndmp] succeeded!
hello
world
```

- the connection can be closed by pressing **Ctrl+C** on the server or client terminal.

```
saksham@Sakshams-MacBook-Air-5:~
```

```
inet6 fe80::ac7a:ebff:fe75:3464%llw0 prefixlen 64 scopeid 0xe
nd6 options=201<PERFORMNUD,DAD>
media: autoselect (none)
utun0: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1500
inet6 fe80::5bf3:c049:a730:bf3%utun0 prefixlen 64 scopeid 0xf
nd6 options=201<PERFORMNUD,DAD>
utun1: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1380
inet6 fe80::435a:8f00:dcb1:e2bd%utun1 prefixlen 64 scopeid 0x10
nd6 options=201<PERFORMNUD,DAD>
utun2: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 2000
inet6 fe80::cd65:15f:191d:7e9f%utun2 prefixlen 64 scopeid 0x11
nd6 options=201<PERFORMNUD,DAD>
utun3: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1000
inet6 fe80::ce81:b1c:bd2c:69e%utun3 prefixlen 64 scopeid 0x12
nd6 options=201<PERFORMNUD,DAD>
utun4: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1380
inet6 fe80::8cbc:99e4:8a76:22dd%utun4 prefixlen 64 scopeid 0x13
nd6 options=201<PERFORMNUD,DAD>
utun5: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1380
inet6 fe80::eef4:dd8f:117a:ec5a%utun5 prefixlen 64 scopeid 0x14
nd6 options=201<PERFORMNUD,DAD>
```

```
4> clear
4> nc -l 10000
4> hello
4> world
```

```
saksham@Sakshams-MacBook-Air-5:~
```

```
Last login: Wed Aug 21 15:22:11 on ttys007
4> nc -v 127.0.0.1 10000
Connection to 127.0.0.1 port 10000 [tcp/ndmp] succeeded!
hello
world
^C
```

### 3.b Determine the state of this TCP connection(s) at the client node. Put a screenshot.

- the state of the TCP connection can be determined by using the option **-z** with the **nc** command, which will scan the port for listening services. Using this command won't establish a connection, but it will show the state of the connection.

```
nc -zv <host_ip> <port_num>
```

where **<host\_ip>** is the IP address of the host machine, and **<port\_num>** is the port number on which the server is listening.

```
saksham@Sakshams-MacBook-Air-5:~
```

```
4> nc -l 6900
```

```
saksham@Sakshams-MacBook-Air-5:~
```

```
4> nc -zv 127.0.0.1 6900
Connection to 127.0.0.1 port 6900 [tcp/*] succeeded!
```

the message on client terminal shows that the connection is established and the port is open.

- if no listening service found, it will show an error message, which can be because the server is not running or the port number is incorrect.

The image shows two terminal windows side-by-side. The left window has the title 'nc -l 6969' and contains the command 'nc -l 6900' followed by 'nc -l 6969'. The right window has the title 'saksham@Sakshams-MacBook-Air-5:' and contains the command 'nc -vz 127.0.0.1 6900', which succeeds, and 'nc -vz 127.0.0.1 4200', which fails with 'Connection refused'.

#### 4. nslookup command

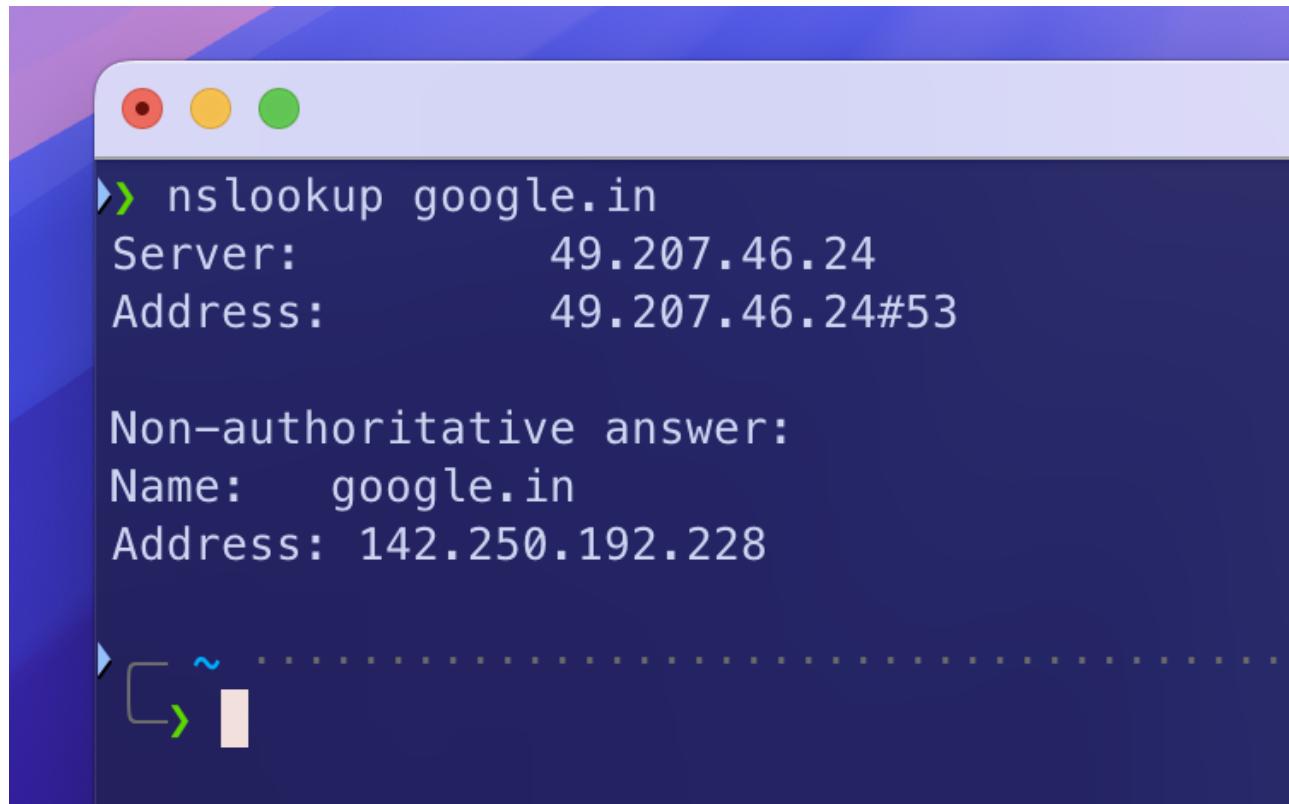
4.a Get an authoritative result for "google.in" using nslookup. Put a screenshot. Explain how you did it.

- **nslookup** is a command-line tool used to query the Domain Name System (DNS) to obtain domain name or IP address mapping, or other DNS records.
- generally, the **nslookup** command is used as follows:

```
nslookup <domain_name>
```

where **<domain\_name>** is the domain name for which the DNS records are to be queried. It returns the closest DNS server and its socket address, and returns the non-authoritative IP address of the domain entered. This non-authoritative IP address is the IP address of the DNS server that resolved

the domain name, not necessarily the IP address of the domain itself.



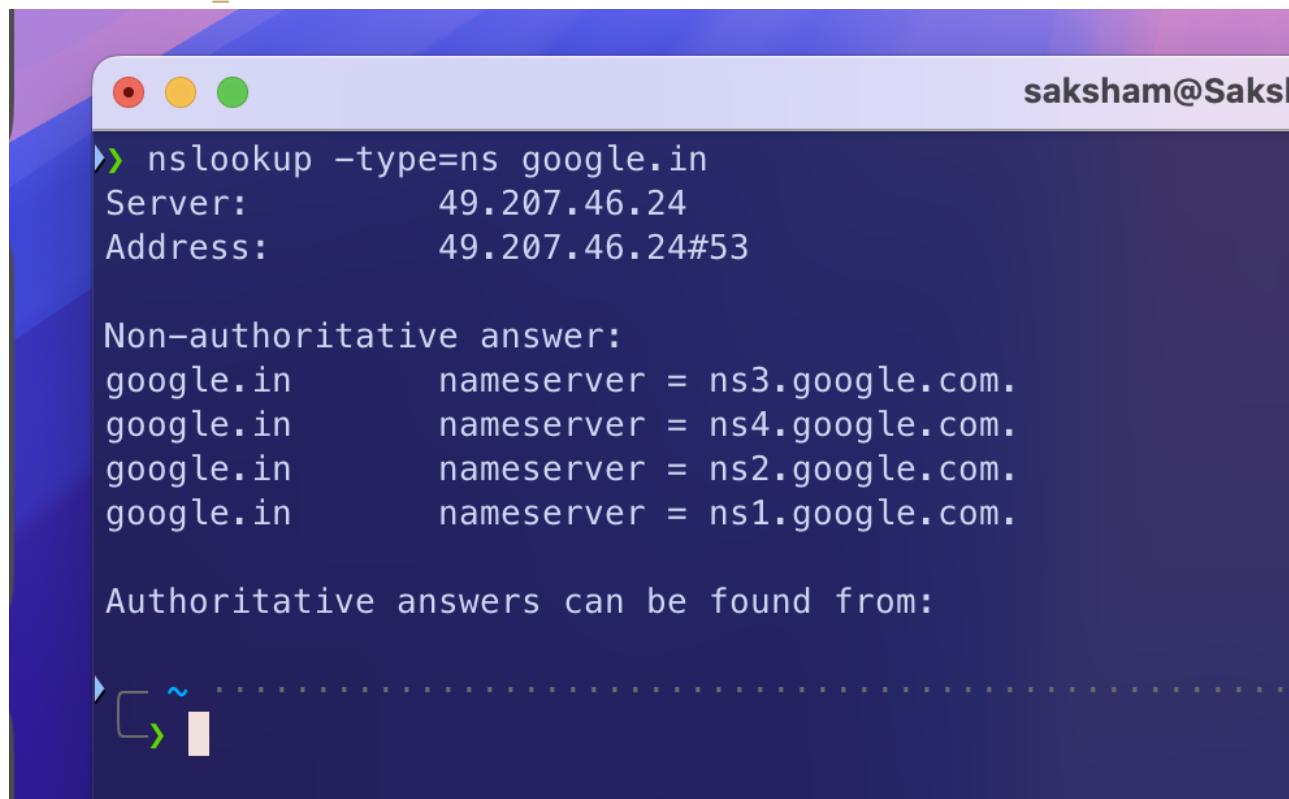
```
nslookup google.in
Server:      49.207.46.24
Address:     49.207.46.24#53

Non-authoritative answer:
Name:  google.in
Address: 142.250.192.228
```

- to get the authoritative result for a domain, we need to first find the DNS server of the domain, and then query that DNS server for the domain name. This can be done by using the option `-type=ns` with the `nslookup` command, which will return the authoritative DNS server for the domain.

```
nslookup -type=ns <domain_name>
```

where `<domain_name>` is the domain name for which the authoritative DNS server is to be found.



```
saksham@Sakshi: ~ % nslookup -type=ns google.in
Server:      49.207.46.24
Address:     49.207.46.24#53

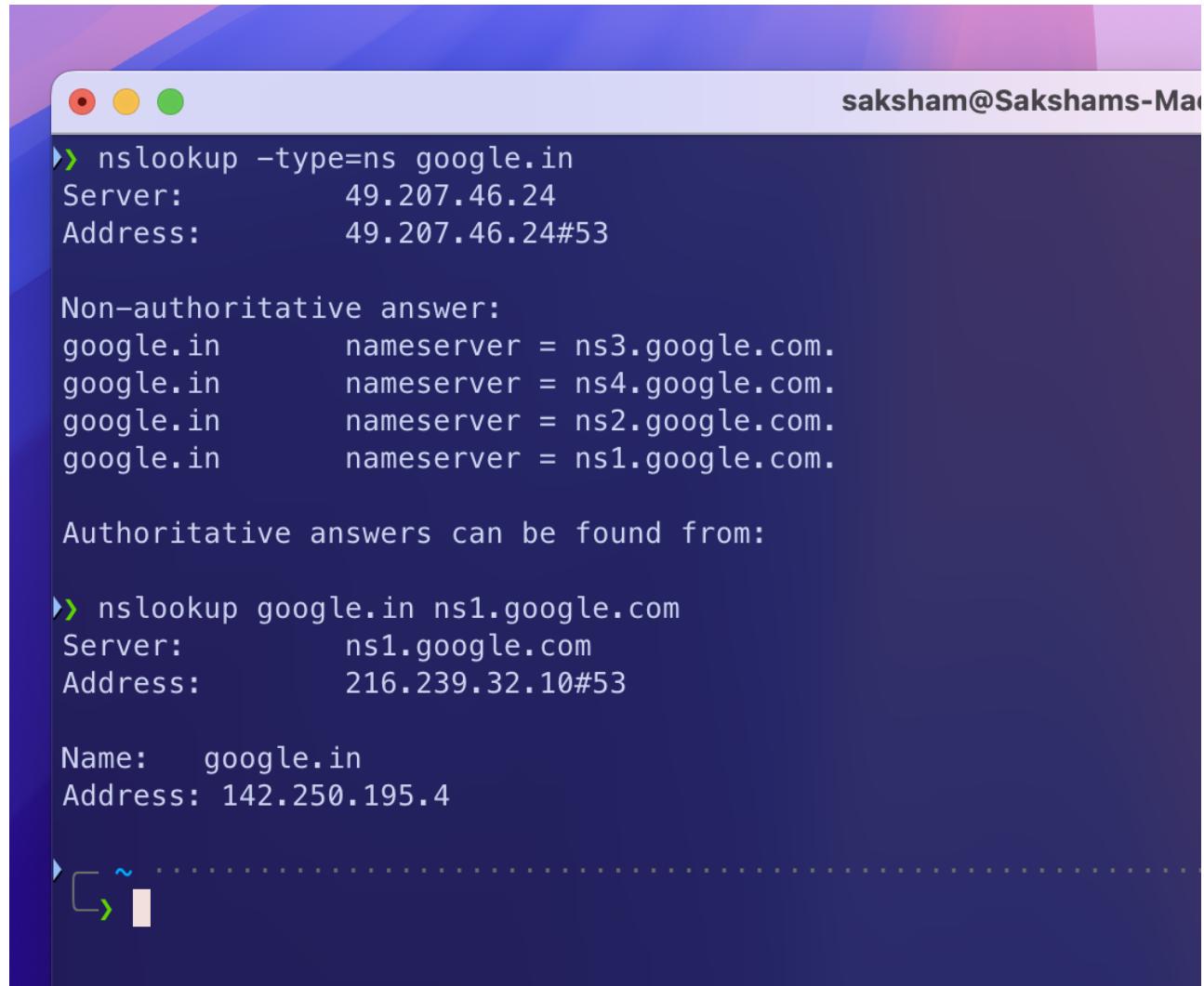
Non-authoritative answer:
google.in      nameserver = ns3.google.com.
google.in      nameserver = ns4.google.com.
google.in      nameserver = ns2.google.com.
google.in      nameserver = ns1.google.com.

Authoritative answers can be found from:
```

Now, we can use one of these DNS servers to query the domain name and get the authoritative result, by following the `<domain_name>` with the DNS server domain.

```
nslookup <domain_name> <dns_server>
```

where `<domain_name>` is the domain name for which the DNS records are to be queried, and `<dns_server>` is the authoritative DNS server for the domain.



saksham@Sakshams-MacBook-Pro:~

```
▶ nslookup -type=ns google.in
Server:          49.207.46.24
Address:         49.207.46.24#53

Non-authoritative answer:
google.in        nameserver = ns3.google.com.
google.in        nameserver = ns4.google.com.
google.in        nameserver = ns2.google.com.
google.in        nameserver = ns1.google.com.

Authoritative answers can be found from:

▶ nslookup google.in ns1.google.com
Server:          ns1.google.com
Address:         216.239.32.10#53

Name:  google.in
Address: 142.250.195.4
```

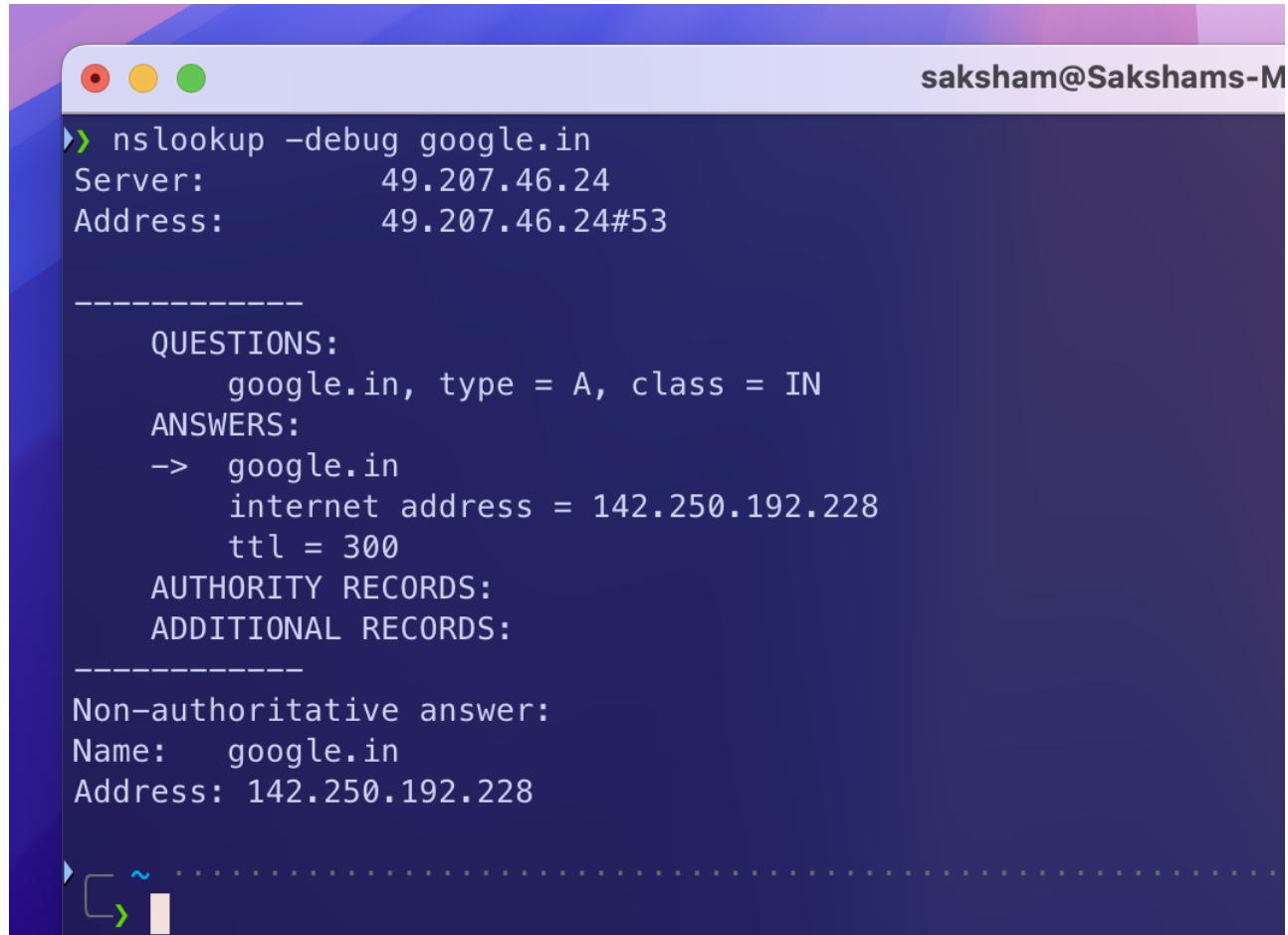
here we can see that the authoritative result for the domain `google.in` is `142.250.195.4`.

#### 4.b Find out the time to live for any website on the local DNS. Put a screenshot. Explain in words (with unit) after how much time this entry would expire from the local DNS server.

- the time to live (TTL) is a value in a DNS record that determines the lifespan of the record in the cache of the DNS server. It is measured in seconds.
- the same can be found using the `debug` option in the `nslookup` command:

```
nslookup -debug <domain_name>
```

where <domain\_name> is the domain name for which the DNS records are to be queried.



```
saksham@Sakshams-M ~ % nslookup -debug google.in
Server:      49.207.46.24
Address:     49.207.46.24#53

-----
QUESTIONS:
    google.in, type = A, class = IN
ANSWERS:
->  google.in
    internet address = 142.250.192.228
    ttl = 300
AUTHORITY RECORDS:
ADDITIONAL RECORDS:
-----
Non-authoritative answer:
Name:  google.in
Address: 142.250.192.228
```

here we can see that the TTL for the domain `google.in` is `300` seconds, which means that the DNS record will expire from the local DNS server after `300` seconds, or `5` minutes. For `iiitd.ac.in`, the TTL is `86400` seconds, which means that the DNS record will expire from the local DNS server after

86400 seconds, or 1 day.

```
▶ nslookup -debug iiitd.ac.in
Server:      49.207.46.24
Address:     49.207.46.24#53

-----
QUESTIONS:
    iiitd.ac.in, type = A, class = IN
ANSWERS:
->  iiitd.ac.in
    internet address = 103.25.231.30
    ttl = 86400
AUTHORITY RECORDS:
ADDITIONAL RECORDS:
-----
Non-authoritative answer:
Name:  iiitd.ac.in
Address: 103.25.231.30

▶ [~] .....
```

## 5. traceroute and ping commands

5.a Run the command, **traceroute google.in**. How many intermediate hosts do you see? What are the IP addresses? Compute the average latency to each intermediate host. Put a screenshot.

- **traceroute** is a command-line tool that tracks the route packets take from one host to another over an IP network. It shows the IP addresses of the intermediate hosts and the latency to each host (the command displays three latency values for each host, for three attempts at making connection, taking average of these values can give the average latency for the jump).

```
traceroute google.in
```

```
saksham@Sakshams-MacBook-Air-5:~
```

```
traceroute google.in
traceroute to google.in (142.250.192.228), 64 hops max, 40 byte packets
 1  192.168.0.1 (192.168.0.1)  3.874 ms  3.426 ms  3.357 ms
 2  * * *
 3  broadband.actcorp.in (49.207.34.226)  28.633 ms  59.628 ms  9.940 ms
 4  broadband.actcorp.in (49.207.47.221)  5.575 ms  8.077 ms  7.272 ms
 5  * * *
 6  172.253.73.194 (172.253.73.194)  28.699 ms
 172.253.67.100 (172.253.67.100)  5.303 ms
 142.251.49.114 (142.251.49.114)  5.789 ms
 7  192.178.83.224 (192.178.83.224)  5.595 ms
 142.251.54.63 (142.251.54.63)  4.882 ms  5.292 ms
 8  del11s13-in-f4.1e100.net (142.250.192.228)  4.692 ms  5.020 ms
 142.251.255.55 (142.251.255.55)  5.104 ms
```

- the ip addresses of the intermediate hosts are as follows:

```
Jump 1: 192.168.0.1 (local router)
    avg latency: 3.552 ms
Jump 2: ----redacted----
Jump 3: 49.207.34.226 (broadband.actcorp.in - ACT Fibernet ISP)
    avg latency: 32.734 ms
Jump 4: 49.207.47.221 (broadband.actcorp.in - ACT Fibernet ISP)
    avg latency: 6.975 ms
Jump 5: ----redacted----
Jump 6: 172.253.73.194 or 172.253.67.100 or 142.251.49.114
    avg latency: 13.264 ms
Jump 7: 192.178.83.224 or 142.251.54.63
    avg latency: 5.256 ms
Jump 8: 142.250.192.228 (google.in) or 142.251.255.55
    avg latency: 4.939 ms
```

---

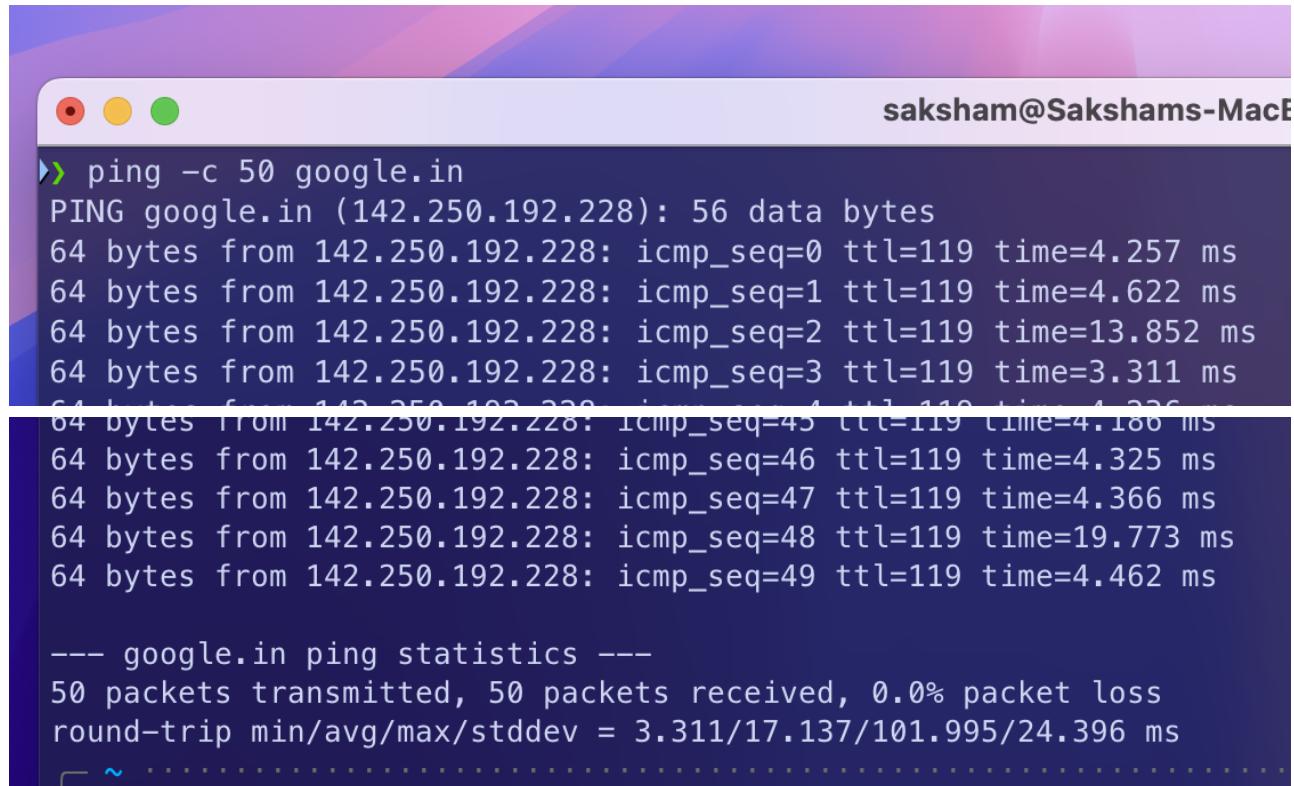
Total hops: 8  
Total latency: 66.720 ms

## 5.b Send 50 ping messages to **google.in**, Determine the average latency. Put a screenshot.

- ping** command is used to check the connectivity between two devices and to measure the round-trip time for the packets to reach the destination and come back.
- any domain or IP address can be pinged by

```
ping <domain_name/ip_addr>          # or
ping -c <count> <domain_name/ip_addr> # for a specific count of
packets
```

where <domain\_name/ip\_addr> is the domain name or IP address of the host to be pinged, and <count> is the number of packets to be sent.



```
saksham@Sakshams-MacBook-Pro: ~
```

```
ping -c 50 google.in
PING google.in (142.250.192.228): 56 data bytes
64 bytes from 142.250.192.228: icmp_seq=0 ttl=119 time=4.257 ms
64 bytes from 142.250.192.228: icmp_seq=1 ttl=119 time=4.622 ms
64 bytes from 142.250.192.228: icmp_seq=2 ttl=119 time=13.852 ms
64 bytes from 142.250.192.228: icmp_seq=3 ttl=119 time=3.311 ms
64 bytes from 142.250.192.228: icmp_seq=4 ttl=119 time=4.186 ms
64 bytes from 142.250.192.228: icmp_seq=5 ttl=119 time=4.325 ms
64 bytes from 142.250.192.228: icmp_seq=6 ttl=119 time=4.366 ms
64 bytes from 142.250.192.228: icmp_seq=7 ttl=119 time=19.773 ms
64 bytes from 142.250.192.228: icmp_seq=8 ttl=119 time=4.462 ms

--- google.in ping statistics ---
50 packets transmitted, 50 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 3.311/17.137/101.995/24.396 ms
```

- the average latency for the 50 packets sent to `google.in` is **17.137 ms**.

### 5.c Add up the ping latency of all the intermediate hosts obtained in (a) and compare with (b). Are they matching, explain?

- the total latency of the intermediate hosts obtained in (a) is **66.720 ms**, and the average latency of the 50 packets sent to `google.in` is **17.137 ms**.
- this change in latencies can be explained by three reasons, which are as follows:
  - roundtrip from each intermediate host**: the `traceroute` command calculates the latency to each intermediate host and back to the source host, while the `ping` command calculates the round-trip time for each packet to the destination host. `traceroute` thus cumulates the latency of each intermediate host, causing the total latency of the intermediate hosts to be higher than the average latency of the `ping` command.
  - smaller amount of packets sent**: the `ping` command sends 50 packets, which is much larger than the number of packets sent by the `traceroute` command, which sends only 3 packets. The average latency of the `ping` command is calculated over a larger number of packets, which gives a more accurate representation of the latency. The `traceroute` command, on the other hand, sends only 3 packets, which may not be representative of the actual latency. This is also seen in the `stddev` in `ping` command, which is **24.396 ms**, which is quite large, indicating that the latency values are spread out over a large range.
  - inconsistencies in the network**: the latency values obtained by the `traceroute` command are the average of the three attempts made to connect to the host. These values may vary due to network congestion, packet loss, or other factors. The latency values obtained by the `ping` command are also affected by these factors, but the average latency smoothens out the effects of these factors, giving a more accurate representation of the latency.

**5.d Take the maximum ping latency amongst the intermediate hosts (in (a)) and compare it with (b). Are they matching, explain?**

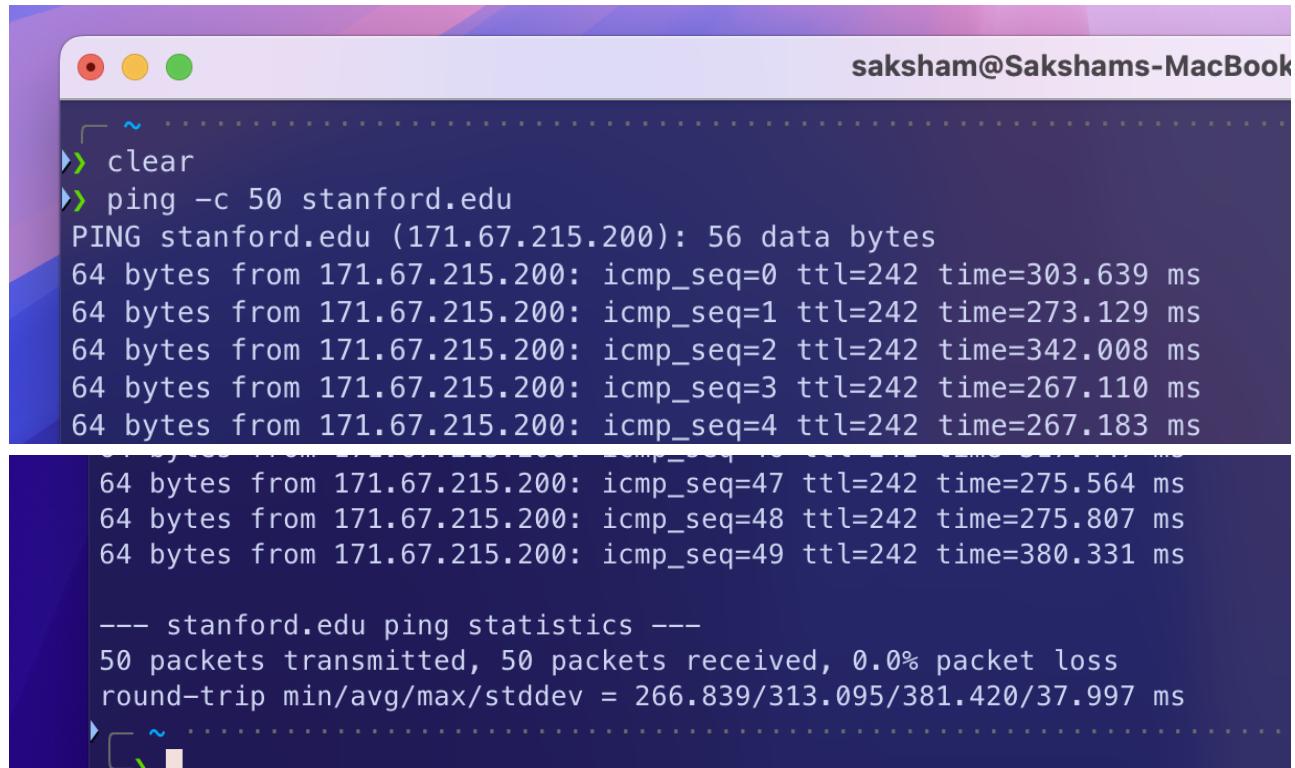
- the maximum latency of the intermediate hosts obtained in (a) is **32.734 ms** in jump 3, and the average latency of the 50 packets sent to **google.in** is **17.137 ms**.
- the maximum latency of the intermediate hosts is higher than the average latency of the **ping** command, which can be explained by:
  - network congestion**: the maximum latency of the intermediate hosts is the maximum latency observed during the three attempts made by the **traceroute** command. This maximum latency may be due to network congestion, packet loss, or other factors that cause delays in the network. The average latency of the **ping** command, on the other hand, is the average round-trip time for each packet, which smoothens out the effects of network congestion and other factors.

**5.e You may see multiple entries for a single hop while using the **traceroute** command. What do these entries mean?**

- the multiple entries for a single hop in the **traceroute** command indicate that there are multiple paths to reach the destination host. At every point in the network, a router may route it to a different path with a different latency, which are all affected by the network conditions like congestion, faults etc. across the network. This path may change for every packet sent, which is why the latency values may vary for each attempt made by the **traceroute** command.

**5.f Send 50 ping messages to **stanford.edu**, Determine the average latency. Put a screenshot.**

- the average latency for the 50 packets sent to **stanford.edu** is **313.095 ms**.



The screenshot shows a terminal window on a Mac OS X desktop. The title bar says "saksham@Sakshams-MacBook". The terminal output is as follows:

```

>> clear
>> ping -c 50 stanford.edu
PING stanford.edu (171.67.215.200): 56 data bytes
64 bytes from 171.67.215.200: icmp_seq=0 ttl=242 time=303.639 ms
64 bytes from 171.67.215.200: icmp_seq=1 ttl=242 time=273.129 ms
64 bytes from 171.67.215.200: icmp_seq=2 ttl=242 time=342.008 ms
64 bytes from 171.67.215.200: icmp_seq=3 ttl=242 time=267.110 ms
64 bytes from 171.67.215.200: icmp_seq=4 ttl=242 time=267.183 ms
64 bytes from 171.67.215.200: icmp_seq=5 ttl=242 time=275.564 ms
64 bytes from 171.67.215.200: icmp_seq=6 ttl=242 time=275.807 ms
64 bytes from 171.67.215.200: icmp_seq=7 ttl=242 time=380.331 ms

--- stanford.edu ping statistics ---
50 packets transmitted, 50 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 266.839/313.095/381.420/37.997 ms
  
```

**5.g Run the command, **traceroute stanford.edu**. Compare the number of hops between **google.in** and **stanford.edu** (between the traceroute result of **google.in** and **stanford.edu**)**

```

❯ traceroute stanford.edu
traceroute to stanford.edu (171.67.215.200), 64 hops max, 40 byte packets
 1  192.168.0.1 (192.168.0.1)  3.755 ms  3.197 ms  3.412 ms
 2  * * *
 3  * broadband.actcorp.in (49.207.47.217)  8.857 ms  6.308 ms
 4  14.143.30.97.static-delhi.vsnl.net.in (14.143.30.97)  4.501 ms  4.964 ms  4.757 ms
 5  * 172.23.183.134 (172.23.183.134)  25.804 ms *
 6  * ix-ae-0-100.tcore1.mlv-mumbai.as6453.net (180.87.38.5)  29.507 ms *
 7  * * if-be-6-2.ecore1.emrs2-marseille.as6453.net (195.219.174.16)  149.236 ms
 8  if-bundle-15-2.qcore1.pye-paris.as6453.net (80.231.154.32)  147.790 ms  149.149 ms  147.651 ms
 9  if-ae-13-2.tcore1.pye-paris.as6453.net (80.231.154.24)  149.026 ms * *
10  if-ae-11-3.tcore1.pvu-paris.as6453.net (80.231.154.149)  157.360 ms *
   if-ae-11-2.tcore1.pvu-paris.as6453.net (80.231.153.49)  146.968 ms
11  * * *
12  stanford-university.e0-62.core2.pao1.he.net (184.105.177.238)  268.399 ms  268.975 ms  277.850 ms
13  campus-east-rtr-vl1118.sunet (171.66.255.228)  269.249 ms
  campus-ial-nets-b-vl1102.sunet (171.66.255.196)  269.467 ms
  campus-east-rtr-vl1118.sunet (171.66.255.228)  271.631 ms
14  * * *
15  web.stanford.edu (171.67.215.200)  296.075 ms  306.206 ms  307.399 ms

```

google.in: 8 hops  
stanford.edu: 15 hops

the number of hops between **stanford.edu** and **google.in** is 7 hops, which is quite a significant difference (nearly as much as the number of hops to **google.in**).

## 5.h Can you explain the reason for the latency difference between google.in and stanford.edu (see (b) & (f))?

- the latency difference between **google.in** and **stanford.edu** can be explained by the following reasons:
  - geographical distance**: the physical distance between the source and destination hosts can affect the latency of the packets. **google.in** is a domain hosted in India, while **stanford.edu** is a domain hosted in the United States. The packets have to travel a longer distance to reach **stanford.edu**, which can increase the latency.
  - number of intermediate hosts**: the number of intermediate hosts between the source and destination hosts can also affect the latency. **stanford.edu** has more intermediate hosts than **google.in**, which can increase the latency as the packets have to pass through more routers and networks.

## 6. Make your ping command fail for 127.0.0.1 (with 100% packet loss). Explain how you do it. Put a screenshot that it failed.

- 127.0.0.1** is the IP address associated with the loopback interface (**lo0** on my device), which is used to test the network stack of the device, hence it is also known as the **localhost**.

```
▶ ifconfig lo0
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
    options=1203<RXCSUM,TXCSUM,TXSTATUS,SW_TIMESTAMP>
    inet6 ::1 prefixlen 128
    inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
    inet 127.0.0.1 netmask 0xff000000
        nd6 options=201<PERFORMNUD,DAD>
```

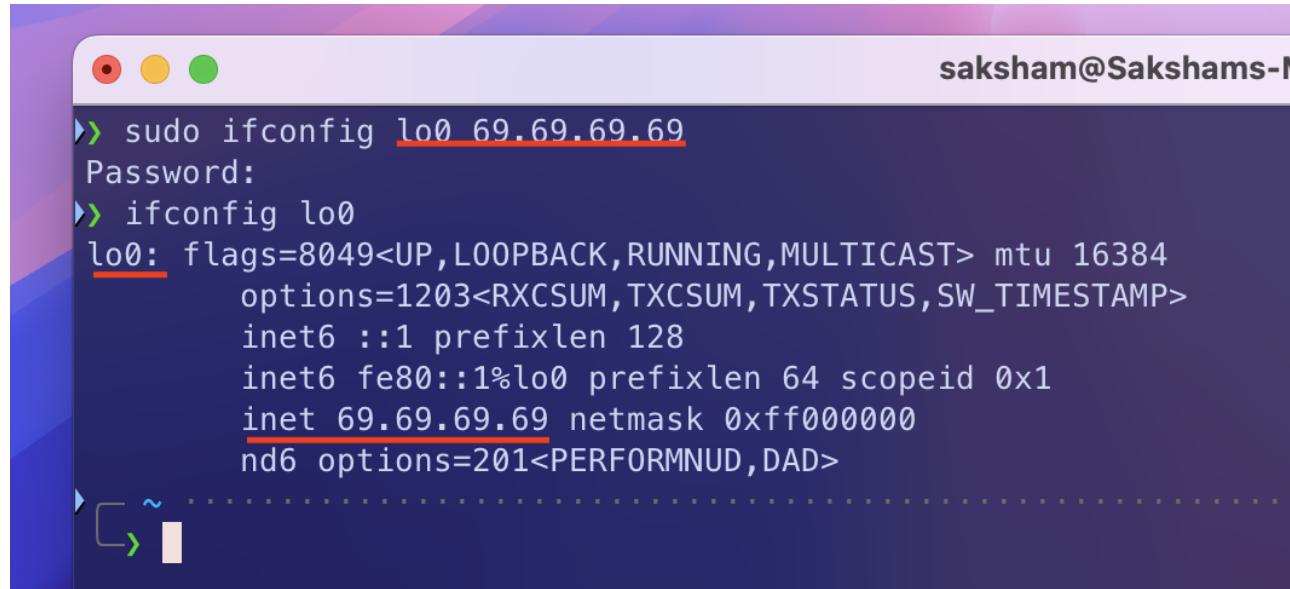
- Normally whenever we ping to **127.0.0.1** it will always connect with very less latency,

```
▶ ping 127.0.0.1
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.104 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.185 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.059 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.129 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.111 ms
64 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=0.125 ms
64 bytes from 127.0.0.1: icmp_seq=6 ttl=64 time=0.082 ms
64 bytes from 127.0.0.1: icmp_seq=7 ttl=64 time=0.156 ms
64 bytes from 127.0.0.1: icmp_seq=8 ttl=64 time=0.044 ms
64 bytes from 127.0.0.1: icmp_seq=9 ttl=64 time=0.077 ms
64 bytes from 127.0.0.1: icmp_seq=10 ttl=64 time=0.131 ms
^C
--- 127.0.0.1 ping statistics ---
11 packets transmitted, 11 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.044/0.109/0.185/0.040 ms
```

- but we can make it fail by **changing the address of lo0 interface**.

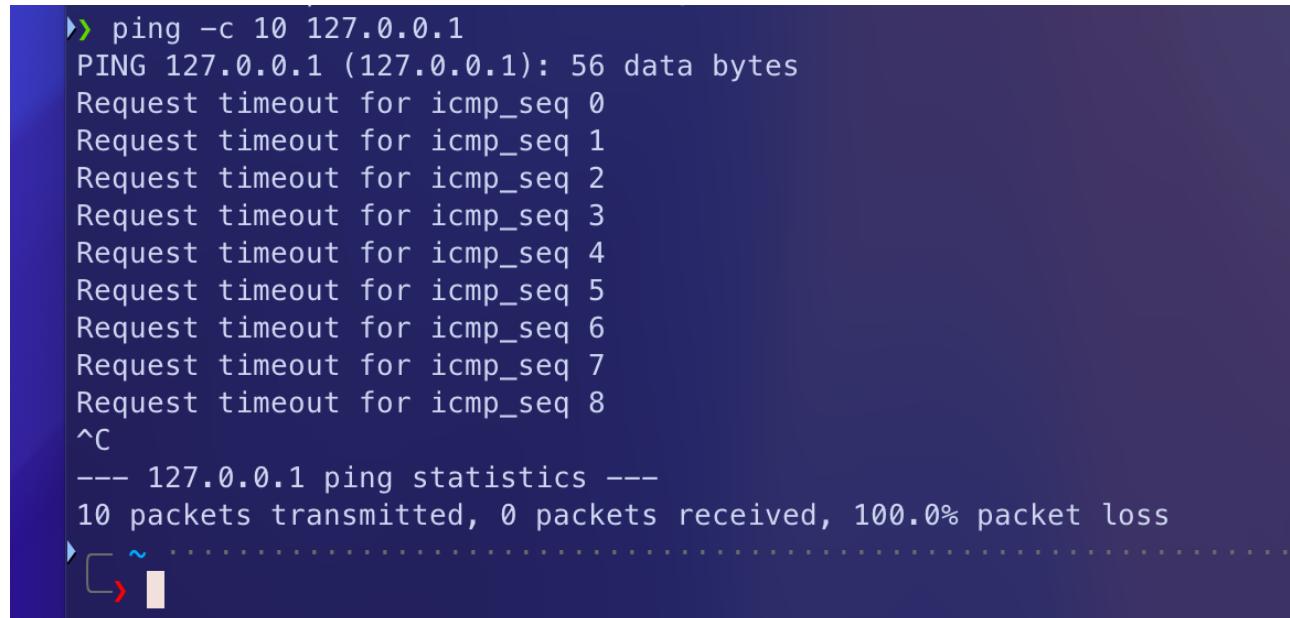
```
sudo ifconfig lo0 <random_ip>
```

this will change the IP address of the loopback interface as follows



saksham@Sakshams-MacBook-Pro ~ % sudo ifconfig lo0 69.69.69.69  
Password:  
saksham@Sakshams-MacBook-Pro ~ % ifconfig lo0  
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384  
 options=1203<RXCSUM,TXCSUM,TXSTATUS,SW\_TIMESTAMP>  
 inet6 ::1 prefixlen 128  
 inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1  
 inet 69.69.69.69 netmask 0xff000000  
 nd6 options=201<PERFORMNUD,DAD>

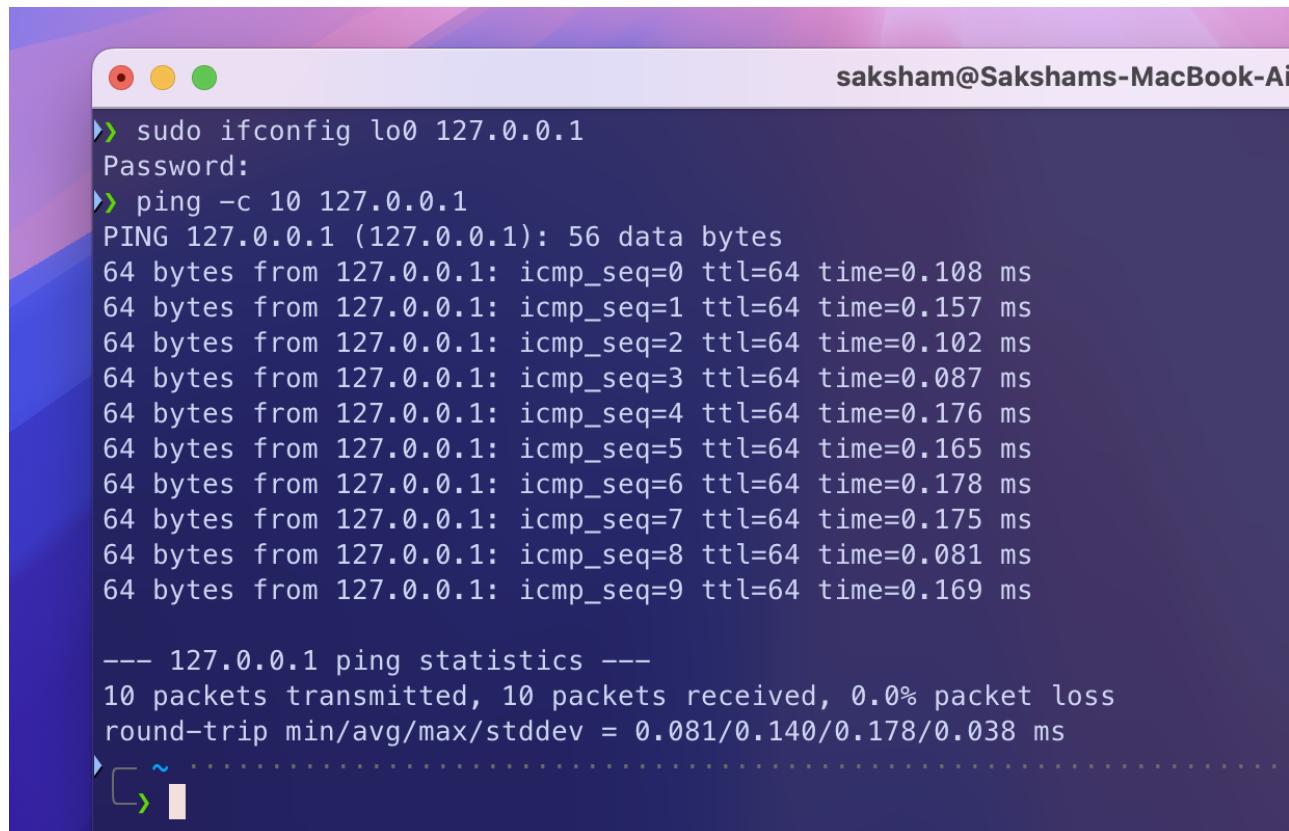
- now when we ping to **127.0.0.1**, it will fail with 100% packet loss



```
saksham@Sakshams-MacBook-Pro ~ % ping -c 10 127.0.0.1  
PING 127.0.0.1 (127.0.0.1): 56 data bytes  
Request timeout for icmp_seq 0  
Request timeout for icmp_seq 1  
Request timeout for icmp_seq 2  
Request timeout for icmp_seq 3  
Request timeout for icmp_seq 4  
Request timeout for icmp_seq 5  
Request timeout for icmp_seq 6  
Request timeout for icmp_seq 7  
Request timeout for icmp_seq 8  
^C  
--- 127.0.0.1 ping statistics ---  
10 packets transmitted, 0 packets received, 100.0% packet loss
```

- to revert back to the original IP address, I can simply do

```
saksham@Sakshams-MacBook-Pro ~ % sudo ifconfig lo0 127.0.0.1  
saksham@Sakshams-MacBook-Pro ~ % ping -c 10 127.0.0.1      # pinging again, it will work as it did before
```



The screenshot shows a terminal window on a Mac OS X desktop. The title bar reads "saksham@Sakshams-MacBook-Ai". The terminal output is as follows:

```
▶ sudo ifconfig lo0 127.0.0.1
Password:
▶ ping -c 10 127.0.0.1
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.108 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.157 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.102 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.087 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.176 ms
64 bytes from 127.0.0.1: icmp_seq=5 ttl=64 time=0.165 ms
64 bytes from 127.0.0.1: icmp_seq=6 ttl=64 time=0.178 ms
64 bytes from 127.0.0.1: icmp_seq=7 ttl=64 time=0.175 ms
64 bytes from 127.0.0.1: icmp_seq=8 ttl=64 time=0.081 ms
64 bytes from 127.0.0.1: icmp_seq=9 ttl=64 time=0.169 ms

--- 127.0.0.1 ping statistics ---
10 packets transmitted, 10 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.081/0.140/0.178/0.038 ms
```

## Thanks

- Saksham Singh
- CSE 2022434
- IIIT-Delhi
- August 2024