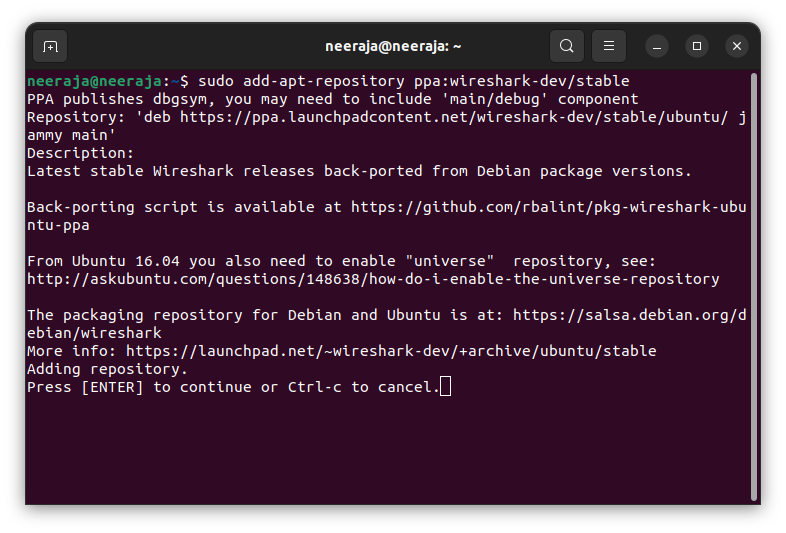
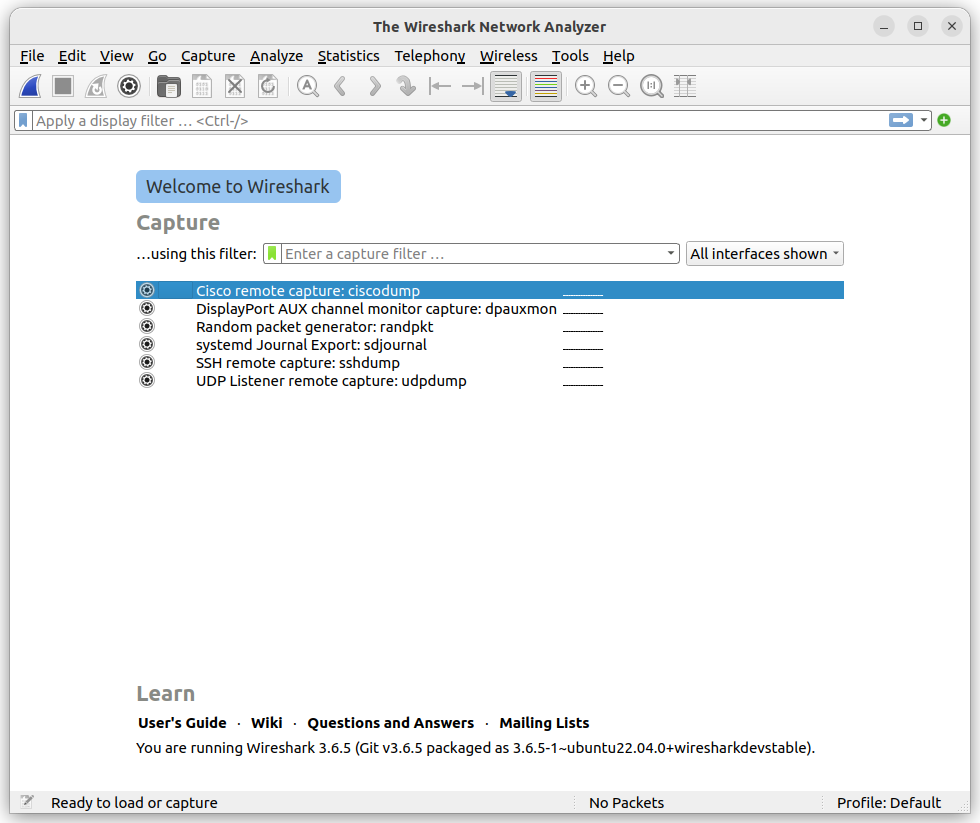
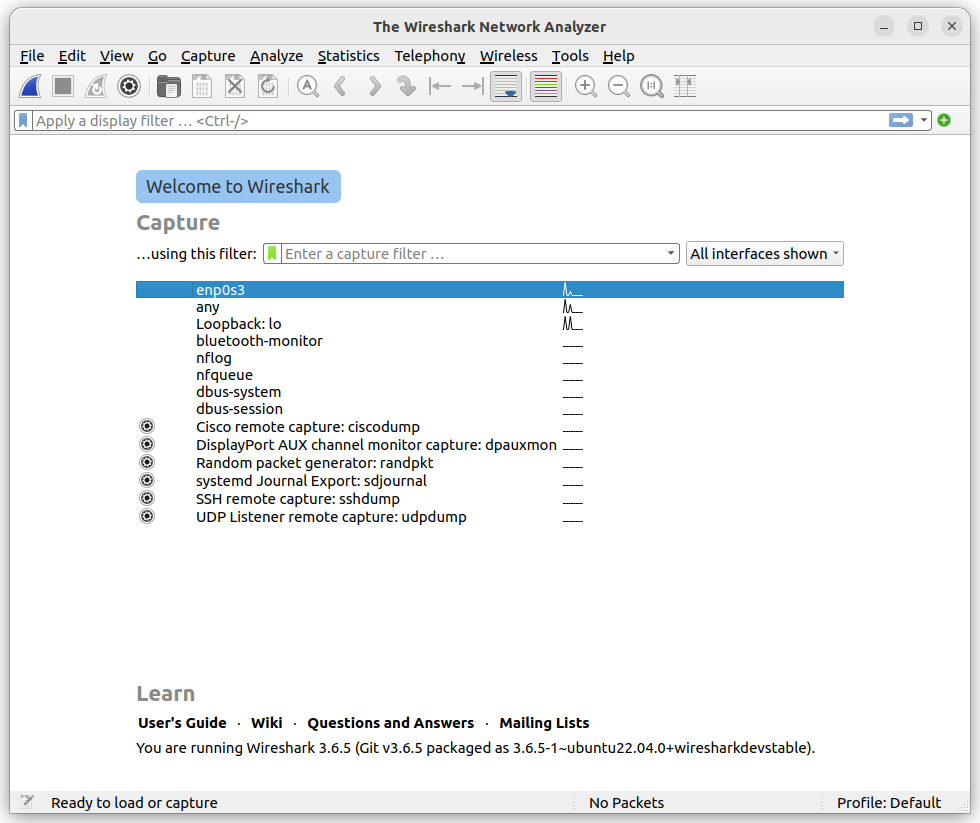
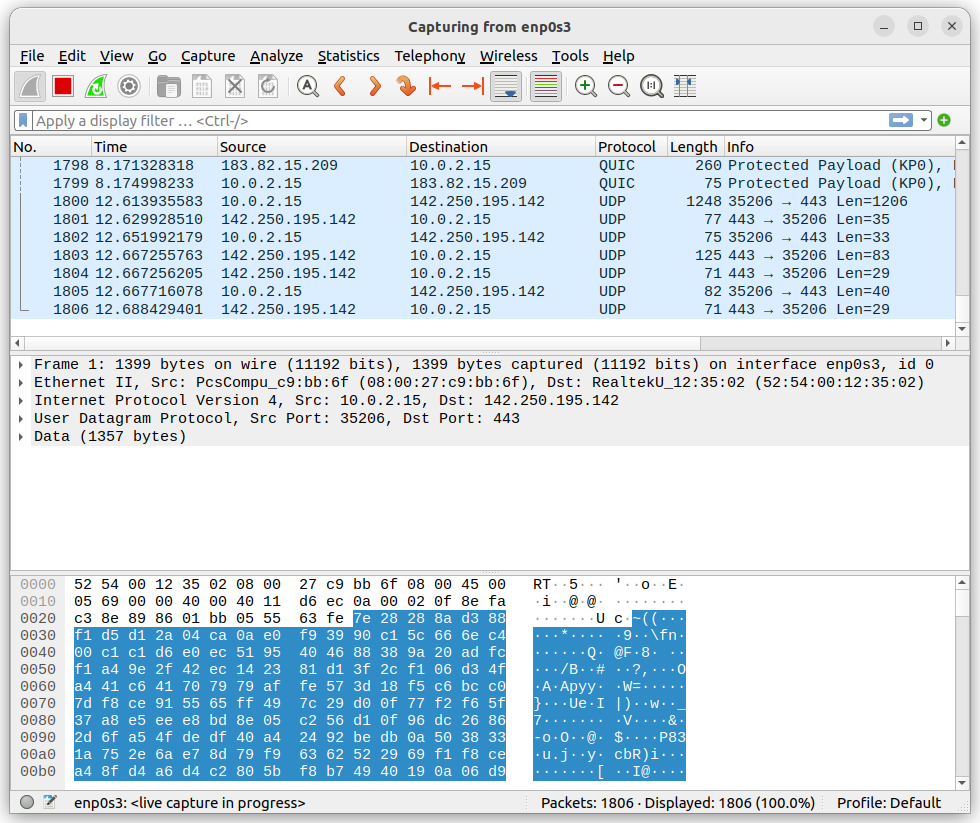
**WIRESHARK:**



DEMONSTRATION:







Demonstrating use of Wireshark:

1. **Packet Capture:** Wireshark listens to a network connection in real time and then grabs entire streams of traffic – quite possibly tens of thousands of packets at a time.
2. **Filtering:** Wireshark is capable of slicing and dicing all of this random live data using filters. By applying a filter, you can obtain just the information you need to see.
3. **Visualization:** Wireshark, like any good packet sniffer, allows you to dive right into the very middle of a network packet. It also allows you to visualize entire conversations and network streams.

The figure below shows an issue on a home network, where the internet connection was very slow.

As the figure shows, the router thought a common destination was unreachable. This was discovered by drilling down into the IPv6 Internet Message Control Protocol (ICMP) traffic, which is marked in black. In Wireshark, any packet marked in black is considered to reflect some sort of issue.

Figure 2: Drilling down into a packet to identify a network problem using Wireshark

In this case, Wireshark helped determine that the router wasn’t working properly and couldn’t find YouTube very easily. The problem was resolved by restarting the cable modem. Of course, while this particular problem didn’t necessitate using Wireshark, it’s kind of cool to authoritatively finalize the issue.

When you take another look at the bottom of Figure 2, you can see that a specific packet is highlighted. This shows the innards of a TCP packet that is part of a transport layer security (TLS) conversation. This is a great example of how you can drill down into the captured packet.

Using Wireshark doesn’t allow you to read the encrypted contents of the packet, but you can identify the version of TLS the browser and YouTube are using to encrypt things. Interestingly enough, the encryption shifted to TLS version 1.2 during the listening.

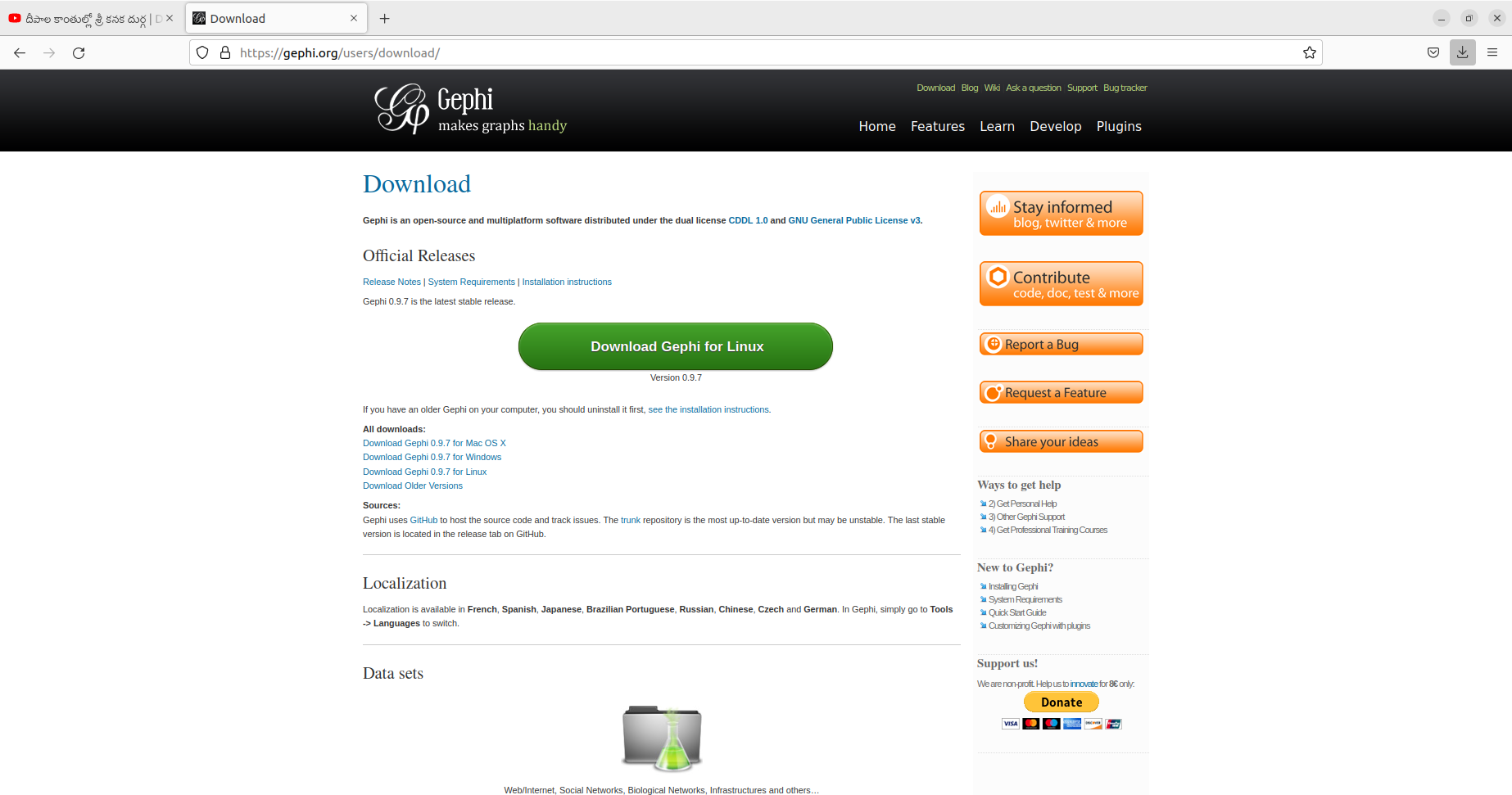
Wireshark is often used to identify more complex network issues. For example, if a network experiences too many retransmissions, congestion can occur. By using Wireshark, you can identify specific retransmission issues, as shown below in Figure 3.

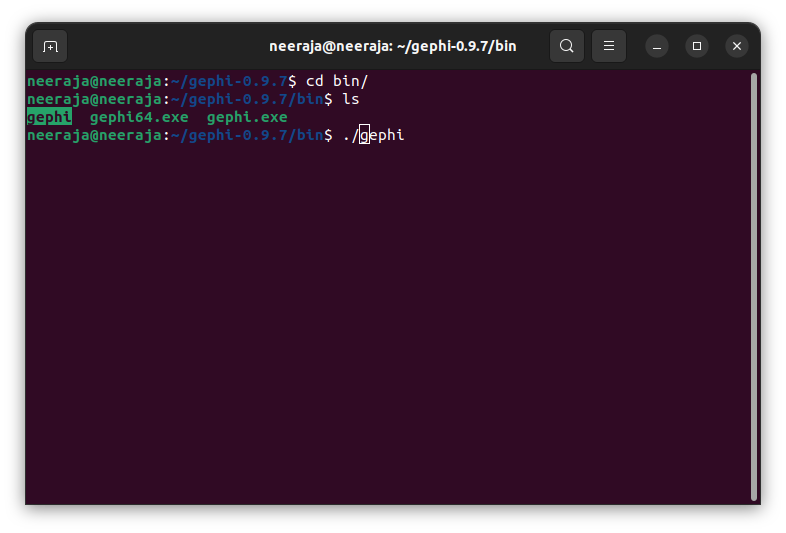
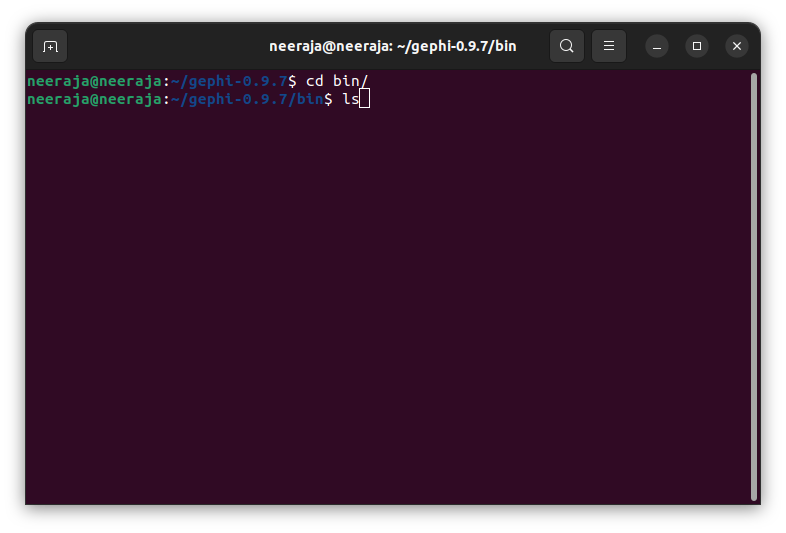
Figure 3: Viewing packet flow statistics using Wireshark to identify retransmissions

By confirming this type of issue, you can then reconfigure the router or switch to speed up traffic.

| **Color in Wireshark** | **Packet Type** |
| --- | --- |
| Light purple | TCP |
| Light blue | UDP |
| Black | Packets with errors |
| Light green | HTTP traffic |
| Light yellow | Windows-specific traffic, including Server Message Blocks (SMB) and NetBIOS |
| Dark yellow | Routing |
| Dark gray | TCP SYN, FIN and ACK traffic |

**Gephi :**



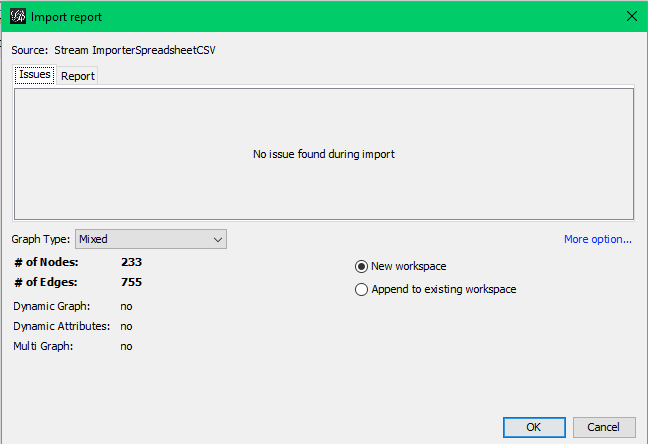


Demonstrating use of Gephi :

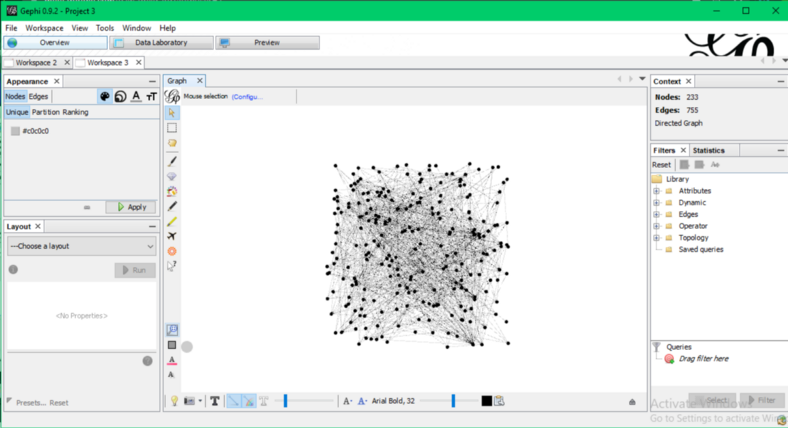
Real-time Visualization.

* Built-in Rendering Engine.
* Native File Formats Support.
* Layout Algorithm.
* Metrics and Statistics.
* Data Laboratory.
* Dynamic Filtering.

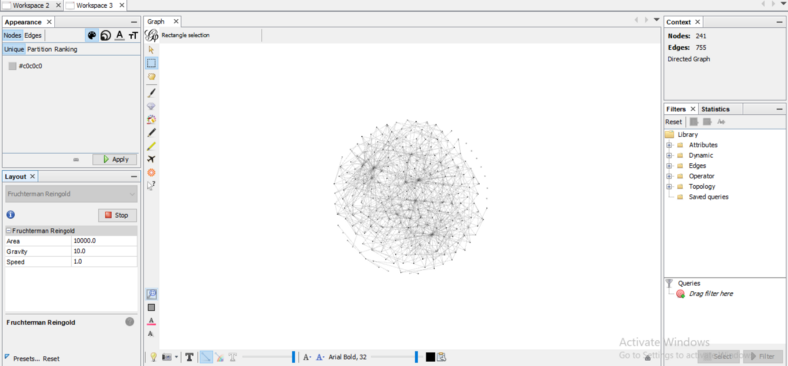
Open Gephi, Create new project and Go to File -> Import Spreadsheet, select the dataset on which you want to visualize. Once the file is imported, report will show metadata of dataset like — no of nodes, edges.  
Here, I have used a Guitar chords fingers dataset.



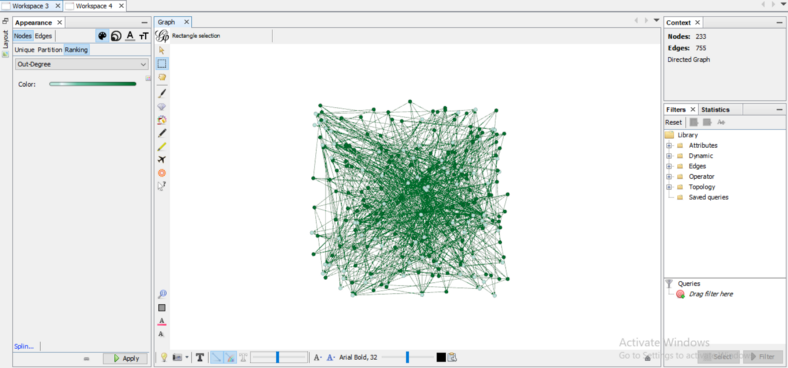
After clicking on OK, it will show the data.



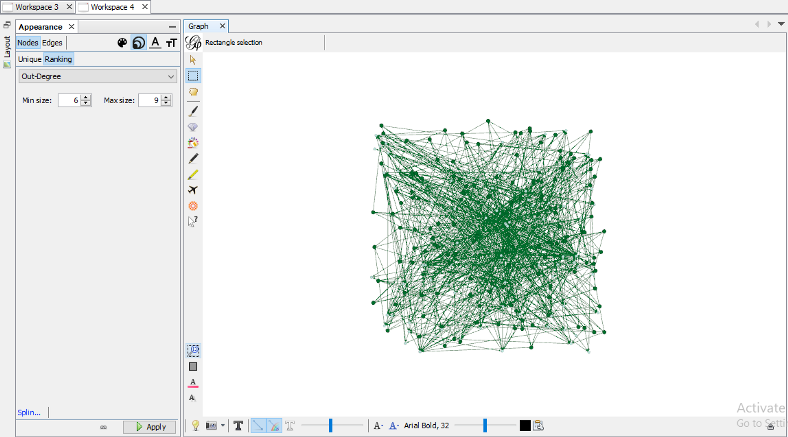
In Gephi, there are various layouts to display data. You can change layout from the left panel. I have selected — **Fruchterman Reingold.**Then run that layout.



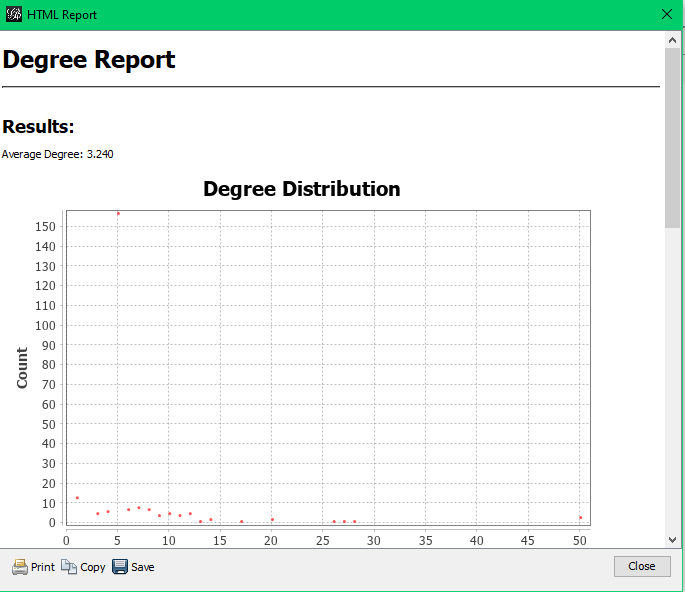
We can show nodes in different color, sizes based on degree, in-degree and out-degree. For that go to left panel, Nodes->Ranking, select color for degree/in-degree/out-degree.



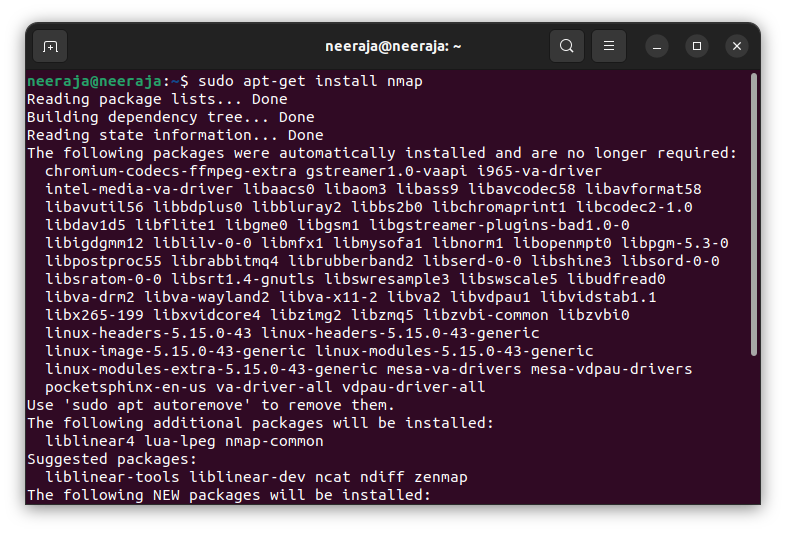
For varying size of nodes, go to Nodes -> Ranking -> Size. Set the Min Size and Max Size.

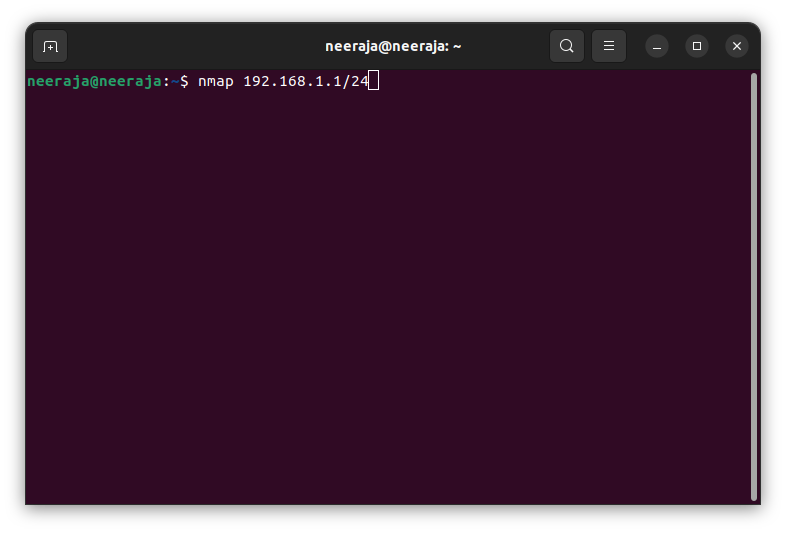


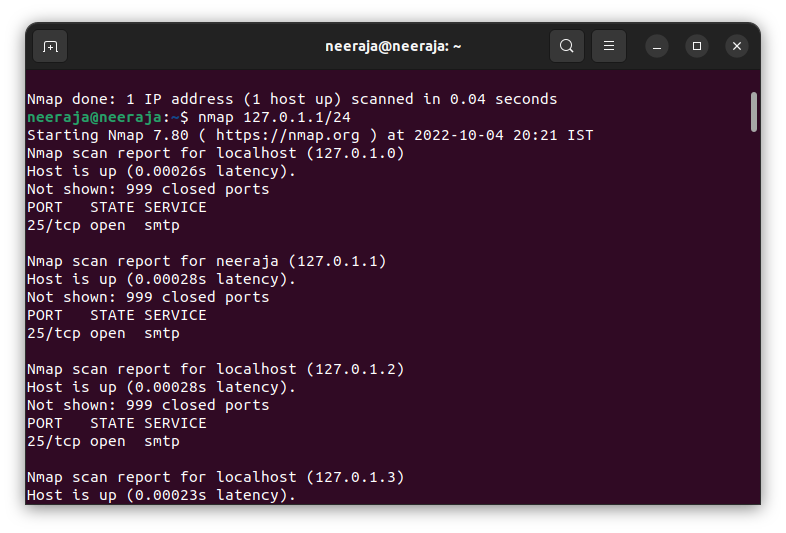
ere, I have displayed **Degree Distribution,**we can view the average degree distribution, in-degree & out-degree distribution as well. For this report, go to right panel, select the statistics tab and run the **Average Degree.**

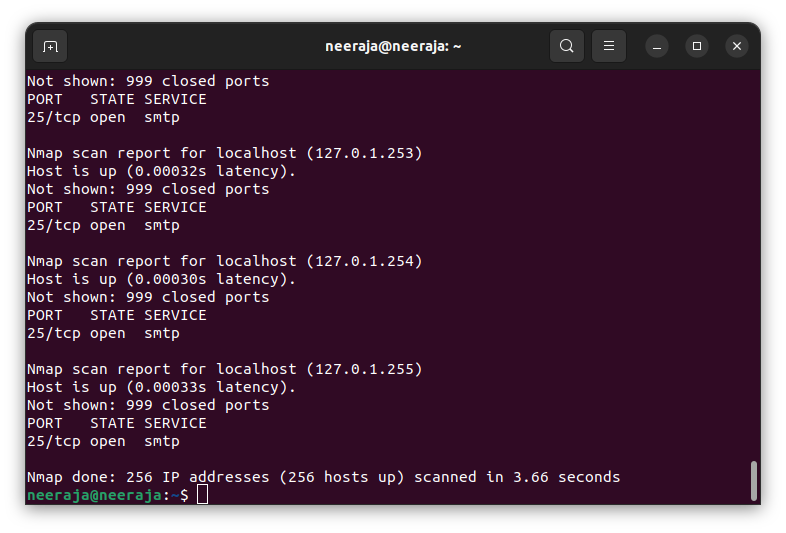


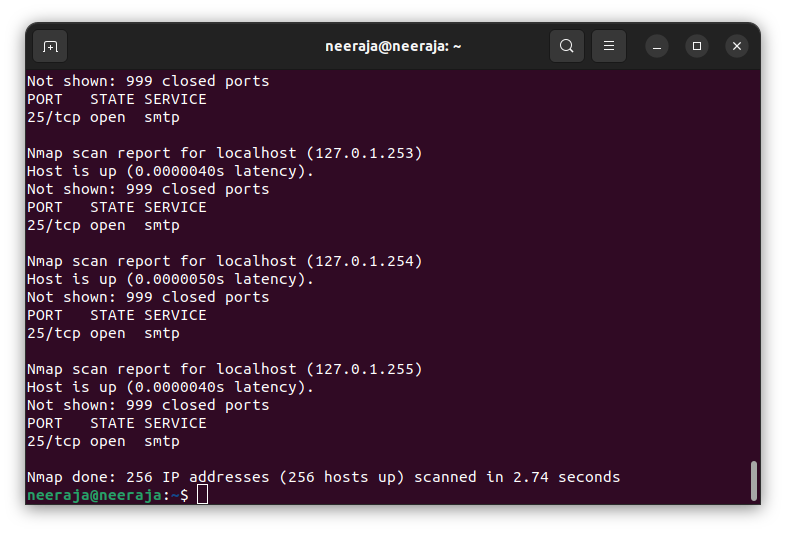
**Nmap :**



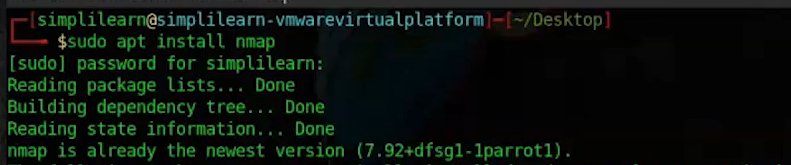




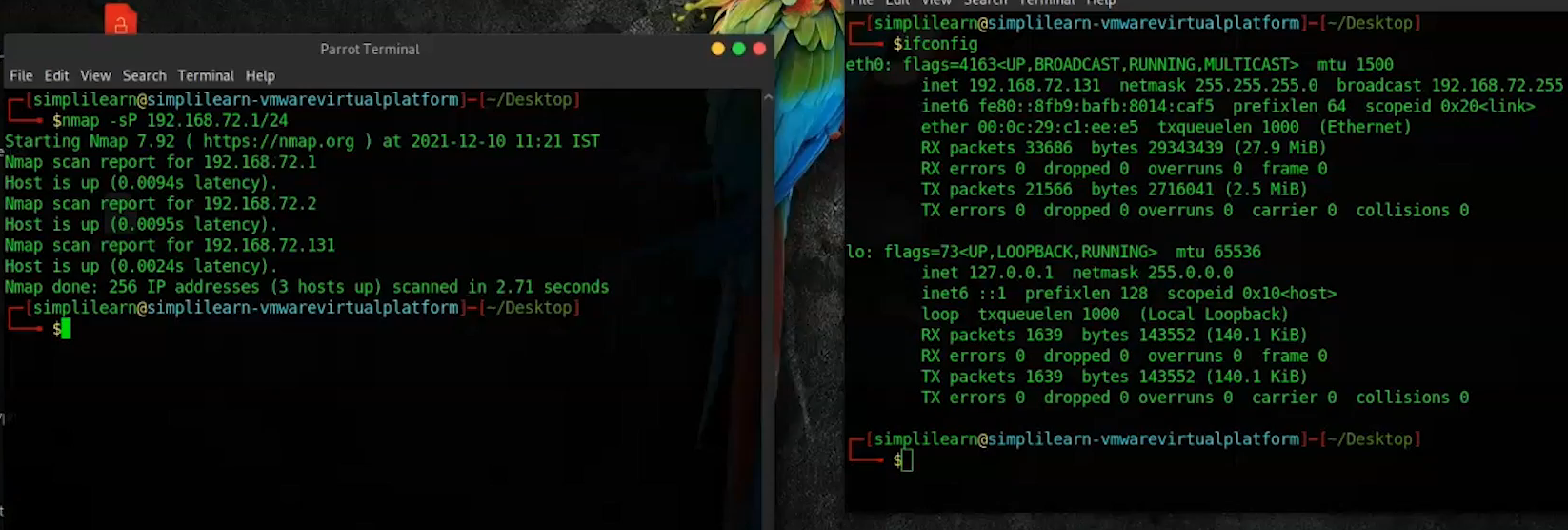


 Demonstration use of Nmap :

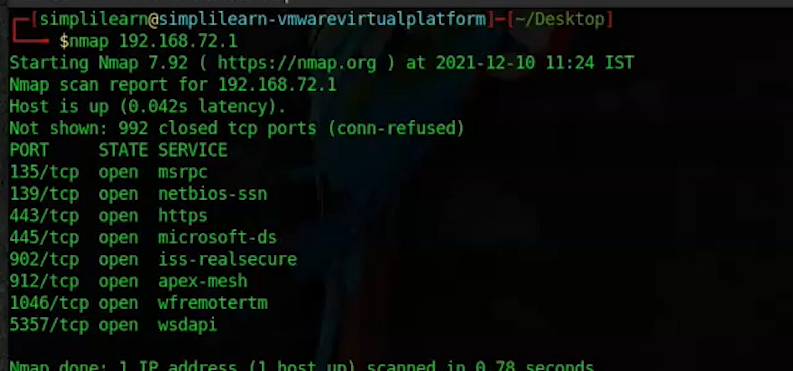
In our demo for what is nmap, you will use [parrot security os](https://www.simplilearn.com/tutorials/cyber-security-tutorial/parrot-security-os), an operating system designed specifically for penetration testers. It comes with all essential [hacking tools](https://www.simplilearn.com/top-5-ethical-hacking-tools-rar313-article) pre-installed. However, should you need to reinstall the software, you can use the command 'sudo apt install nmap' to install Nmap on Debian-based Linux distributions.



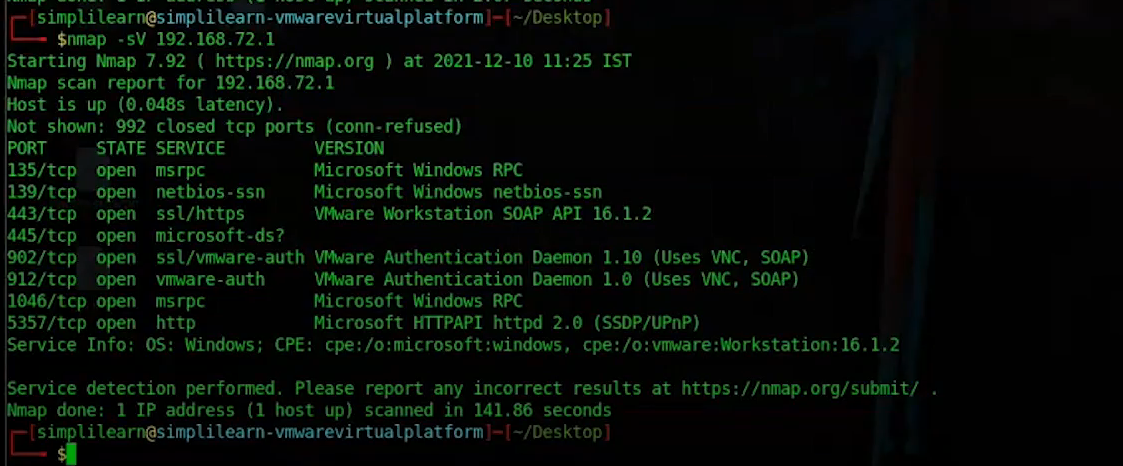
* Once the tool's installation is complete, you can start with a basic host scan. It will check for available IP addresses on a single subnet. As shown below, you find the IP subnet using the 'ifconfig' command and carry out a host scan using the '-sP' flag with Nmap.



* A simple port scan to detect services being run can be carried out using the command 'nmap <target IP address>.

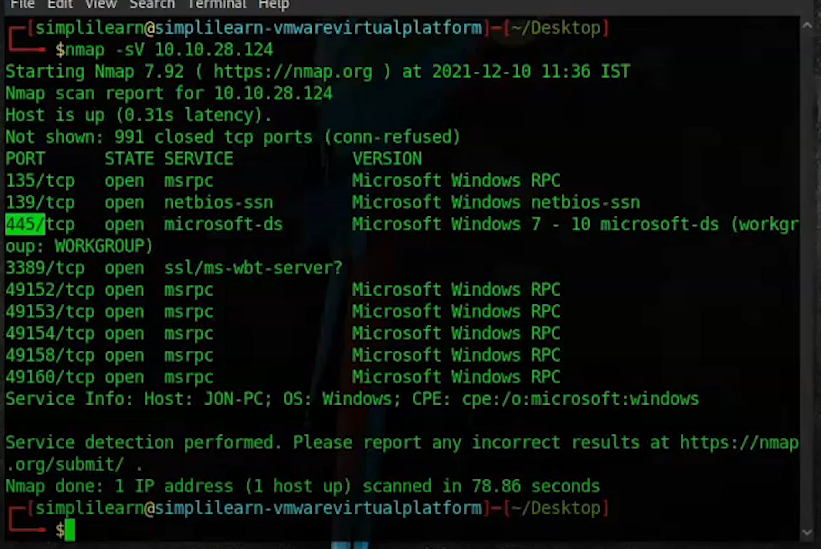
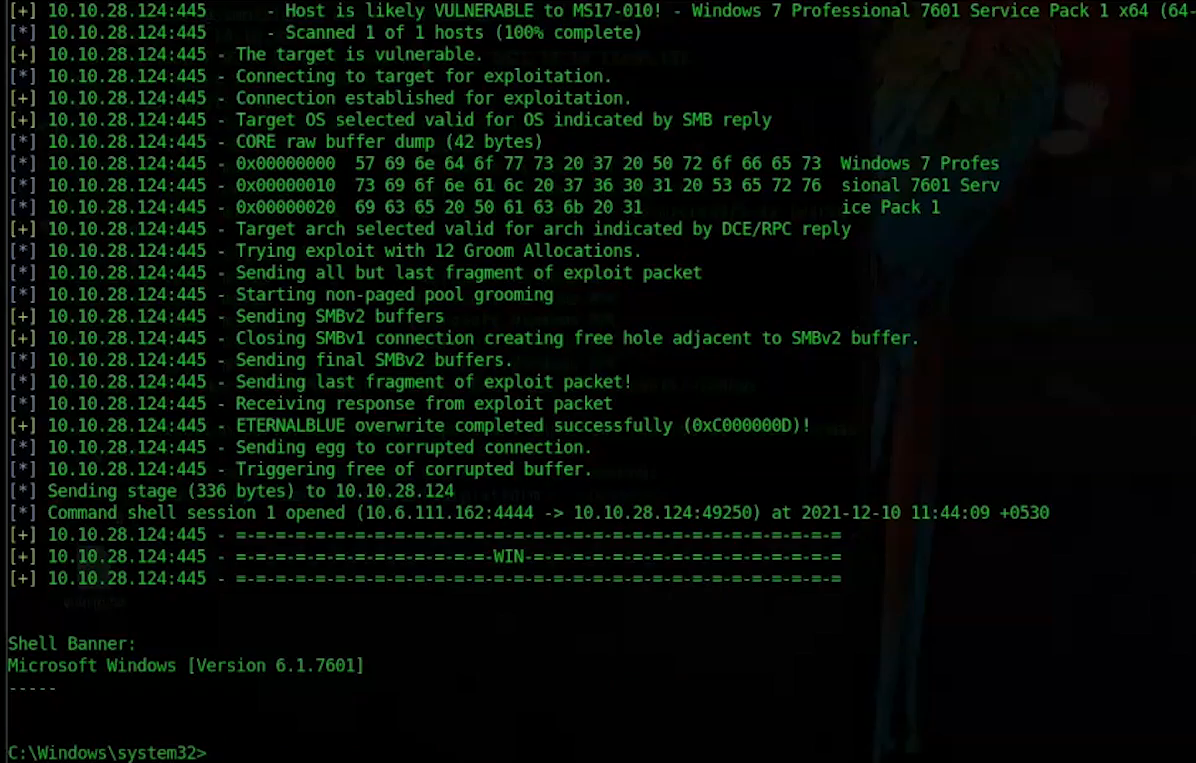


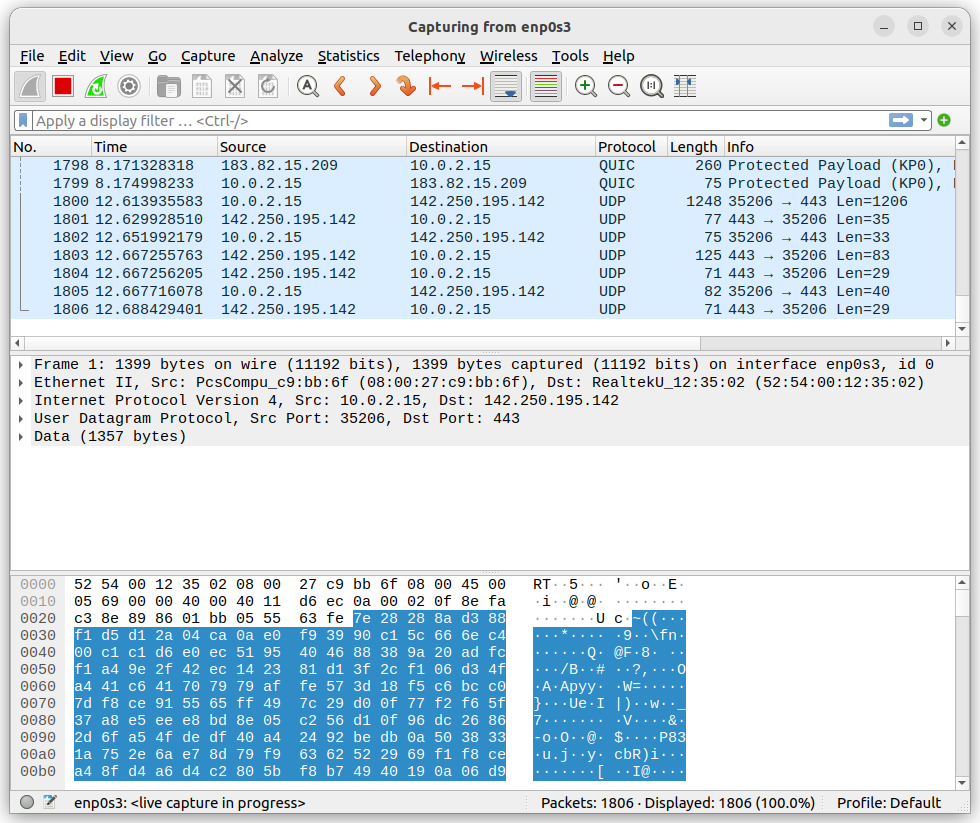
* The operating system of the operating system can also be detected using the '-O' flag. However, this performs TCP/IP fingerprinting, which requires root or sudo privileges when running the command.
* To gather the version number of the services being run on the target, you must use the '-sV' flag. Specific older versions of some software often have vulnerabilities that can be exploited when detected.



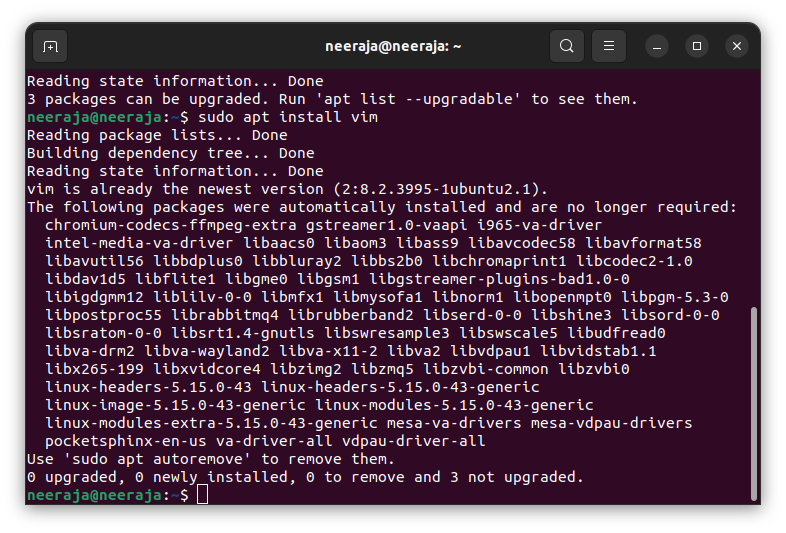
You can scan for particular ports on the target machine using the '-p <port number>' flag with the standard Nmap command. One can also scan multiple ports by separating them using commas like '-p 80,443'.

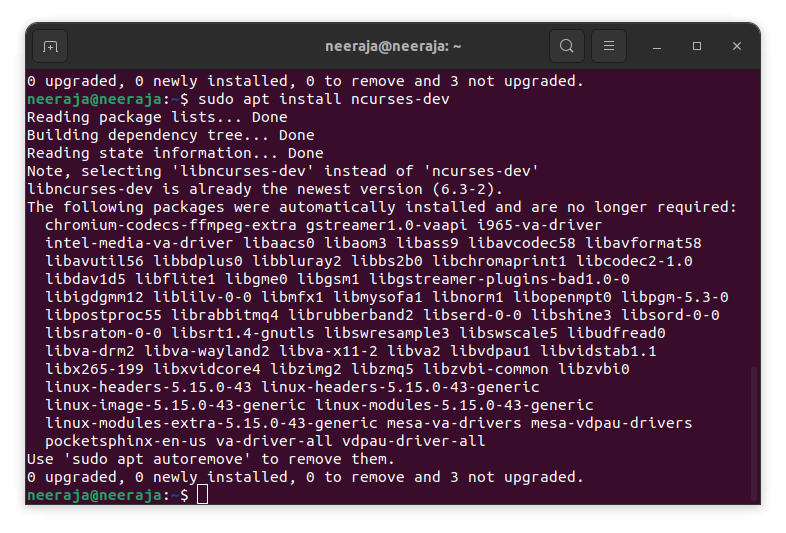
* In the next stage, you must connect to a vulnerable network and scan your target machine using a standard version scan. On seeing ports 139 and 445 open, you can test for a well-known vulnerability on windows machines that run on these ports, known as eternalblue.
* You use Metasploit to run this exploit, and as you can see below, the shell access of your target machine has been achieved. This was only possible because you found the vulnerable ports to be open.

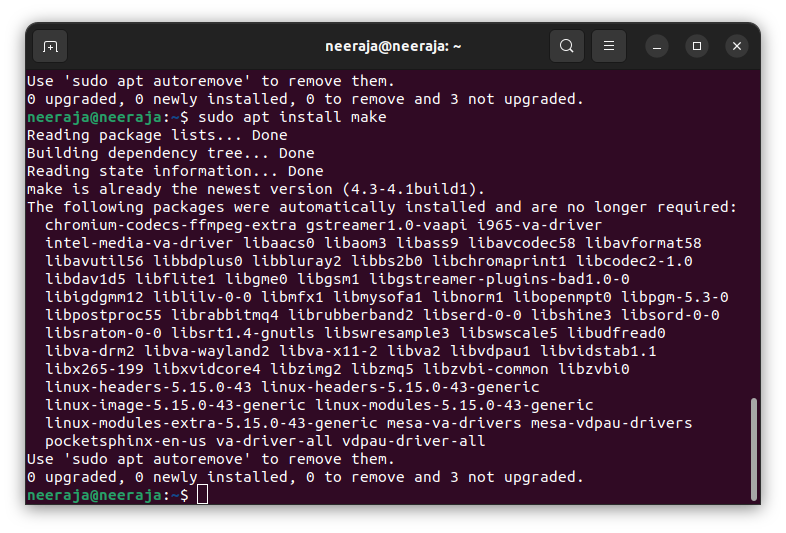
With this, you have reached the end of the tutorial on what is Nmap.

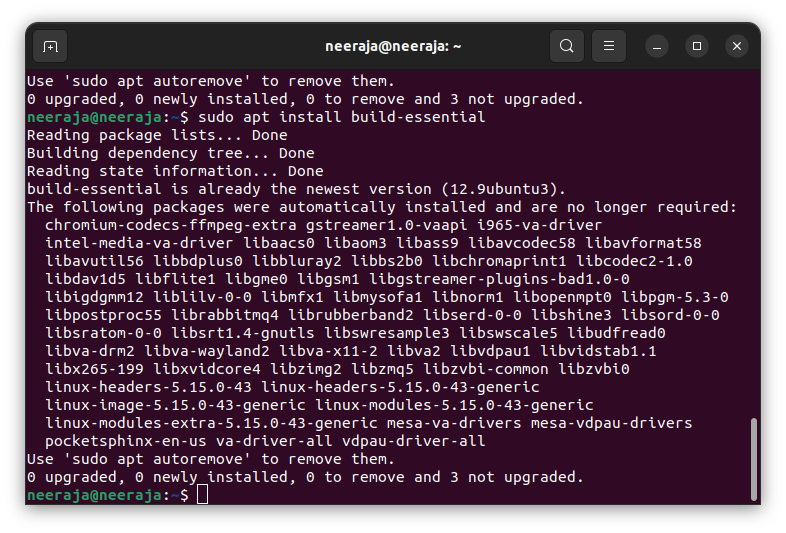


**Vim :**

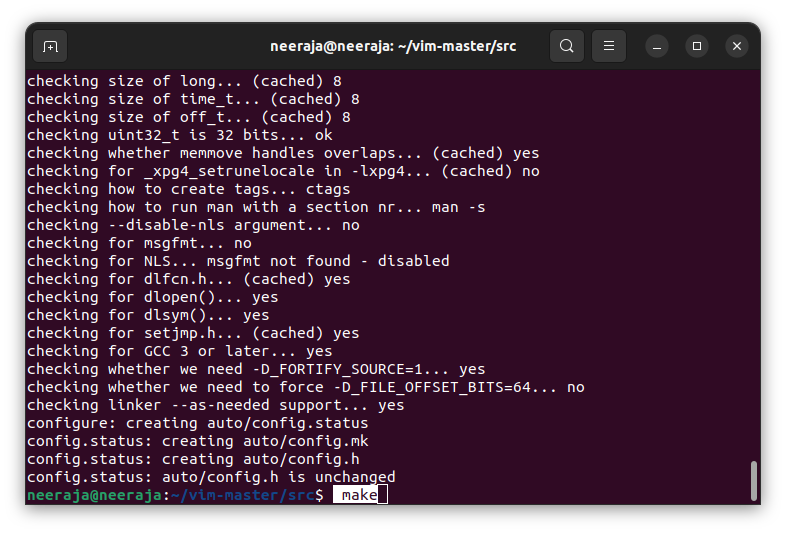


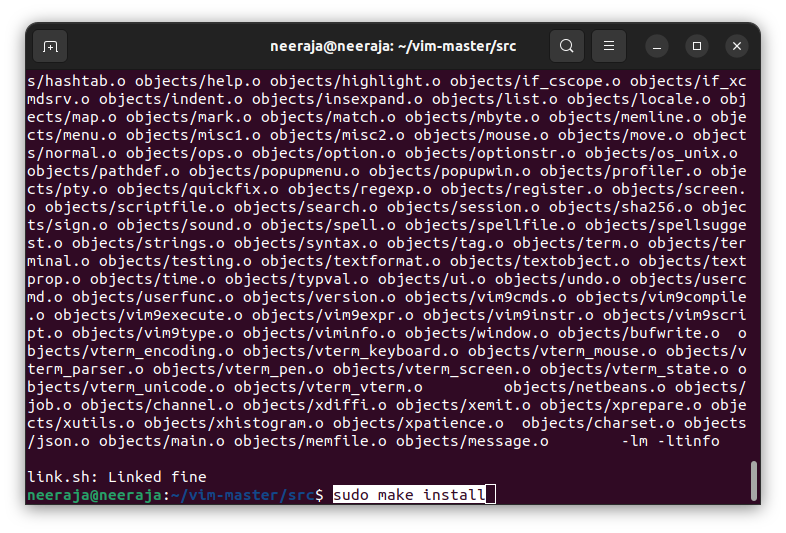


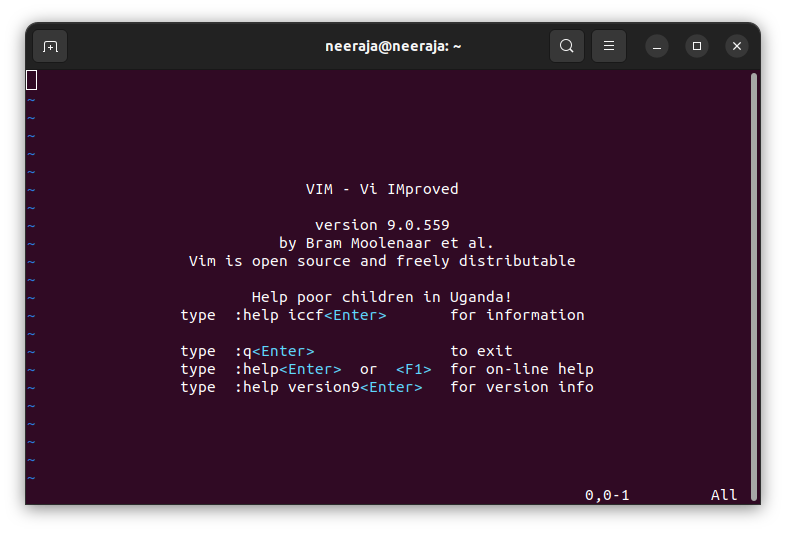


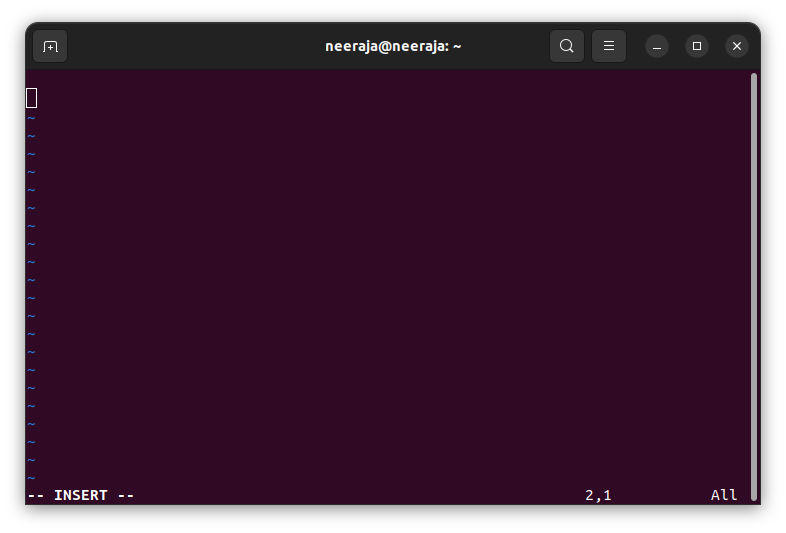














### 1. Basic Vim commands[#](https://www.keycdn.com/blog/vim-commands" \l "1-basic-vim-commands)

The most simple commands allow you to open and close documents as well as saving them. As with most other text editors, there are protections in place to help you avoid exiting the editor without having saved what you're working on.

:help [keyword] - Performs a search of help documentation for whatever keyword you enter

:e [file] - Opens a file, where [file] is the name of the file you want opened

:w - Saves the file you are working on

:w [filename] - Allows you to save your file with the name you've defined

:wq - Save your file and close Vim

:q! - Quit without first saving the file you were working on

### 2. Vim commands for movement[#](https://www.keycdn.com/blog/vim-commands" \l "2-vim-commands-for-movement)

When using movement commands, you can put a number in front of them to make Vim complete a command multiple times. For example, 5h will move your cursor five spaces to the left, and 90j will put your cursor at the beginning of the 90th line down from where your cursor currently is.

h - Moves the cursor to the left

l - Moves the cursor to the right

j - Moves the cursor down one line

k - Moves the cursor up one line

H - Puts the cursor at the top of the screen

M - Puts the cursor in the middle of the screen

L - Puts the cursor at the bottom of the screen

w - Puts the cursor at the start of the next word

b - Puts the cursor at the start of the previous word

e - Puts the cursor at the end of a word

0 - Places the cursor at the beginning of a line

$ - Places the cursor at the end of a line

) - Takes you to the start of the next sentence

( - Takes you to the start of the previous sentence

} - Takes you to the start of the next paragraph or block of text

{ - Takes you to the start of the previous paragraph or block of text

Ctrl + f - Takes you one page forward

Ctrl + b - Takes you one page back

gg - Places the cursor at the start of the file

G - Places the cursor at the end of the file

# - Where # is the number of a line, this command takes you to the line specified

### 3. Vim commands for editing

yy - Copies a line

yw - Copies a word

y$ - Copies from where your cursor is to the end of a line

v - Highlight one character at a time using arrow buttons or the h, k, j, l buttons

V - Highlights one line, and movement keys can allow you to highlight additional lines

p - Paste whatever has been copied to the unnamed register

d - Deletes highlighted text

dd - Deletes a line of text

dw - Deletes a word

D - Deletes everything from where your cursor is to the end of the line

d0 - Deletes everything from where your cursor is to the beginning of the line

dgg - Deletes everything from where your cursor is to the beginning of the file

dG - Deletes everything from where your cursor is to the end of the file

x - Deletes a single character

u - Undo the last operation; u# allows you to undo multiple actions

Ctrl + r - Redo the last undo

. - Repeats the last action

### 4. Vim commands for searching text[#](https://www.keycdn.com/blog/vim-commands" \l "4-vim-commands-for-searching-text)

Like many other text editors, Vim allows you to search your text and find and replace text within your document. If you opt to replace multiple instances of the same keyword or phrase, you can set Vim up to require or not require you to confirm each replacement depending on how you put in the command.

/[keyword] - Searches for text in the document where keyword is whatever keyword, phrase or string of characters you're looking for

?[keyword] - Searches previous text for your keyword, phrase or character string

n - Searches your text again in whatever direction your last search was

N - Searches your text again in the opposite direction

:%s/[pattern]/[replacement]/g - This replaces all occurrences of a pattern without confirming each one

:%s/[pattern]/[replacement]/gc - Replaces all occurrences of a pattern and confirms each one

### 5.Marking text (visual mode)[#](https://www.keycdn.com/blog/vim-commands" \l "6-marking-text-visual-mode)

Visual mode allows you to select a block of text in Vim. Once a block of text is selected you can use visual commands to perform actions on the selected text such as deleting it, copying it, etc.

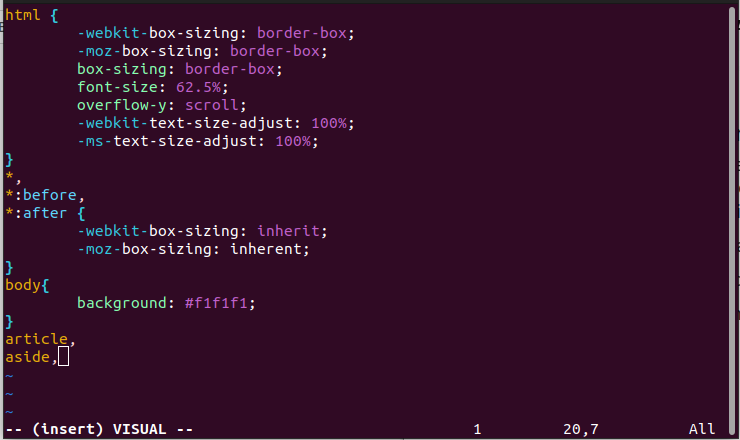
v - starts visual mode, you can then select a range of text, and run a command

V - starts linewise visual mode (selects entire lines)

Ctrl + v - starts visual block mode (selects columns)

Esc - exit visual mode

ENTERING VISUAL MODE:



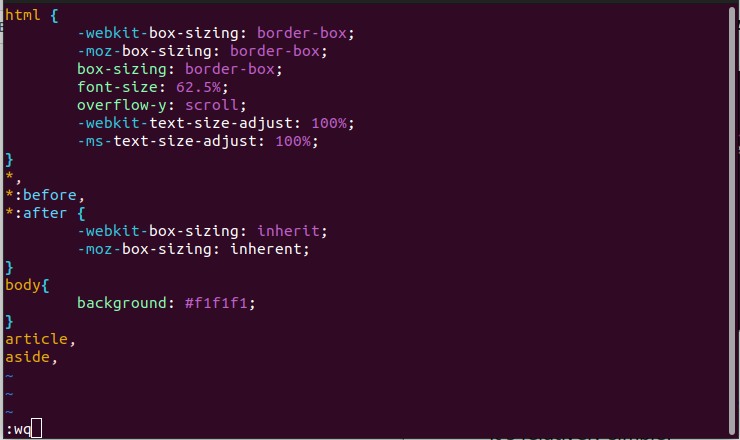
EXITING VISUAL MODE:

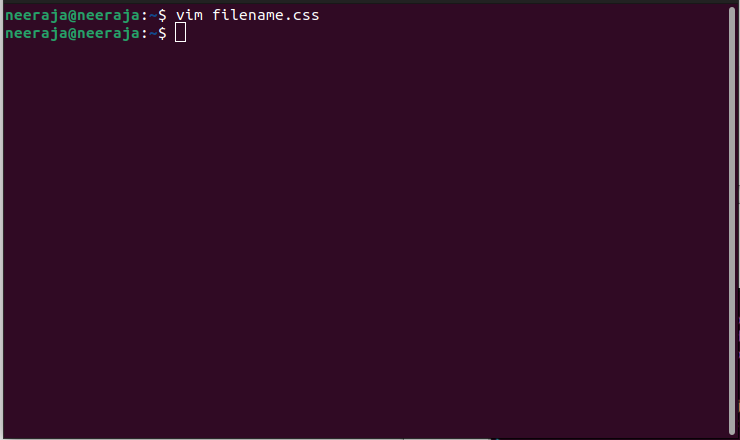


## Simple Vim workflow example[#](https://www.keycdn.com/blog/vim-commands" \l "simple-vim-workflow-example)

It's relatively simple:

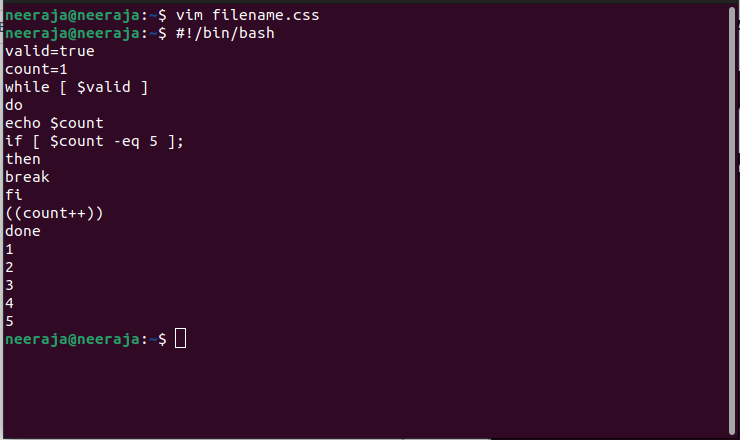
1. Open a new or existing file with vim filename.
2. Type i to switch into insert mode so that you can start editing the file.
3. Enter or modify the text with your file.
4. Once you're done, press the escape key Esc to get out of insert mode and back to command mode.
5. Type :wq to save and exit your file.



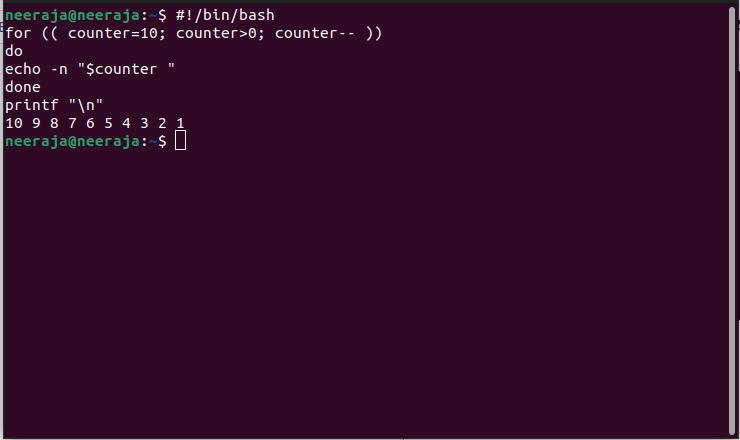


**Bash Scripts :**

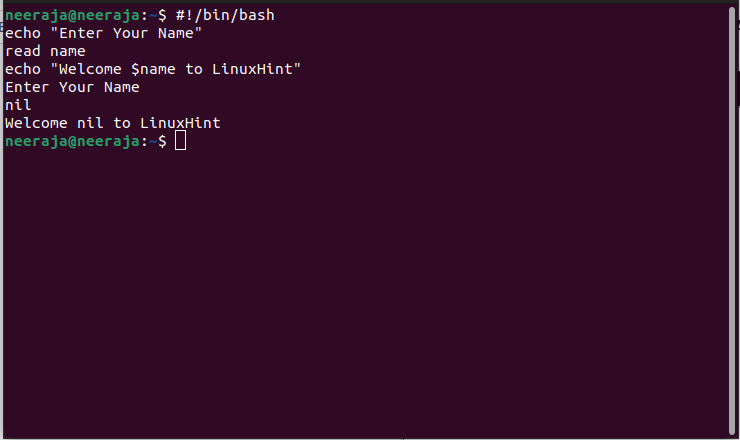
### Using While Loop:



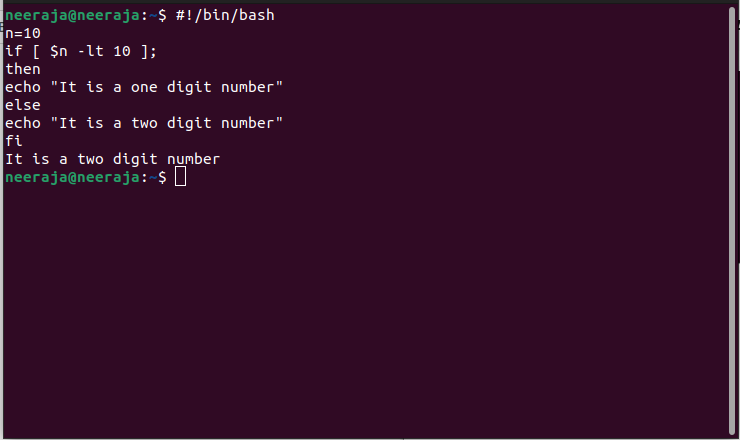
Using For Loop:



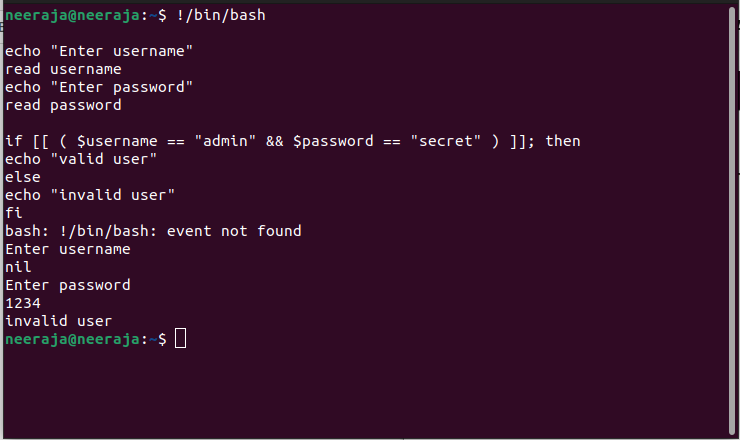
### Get User Input:



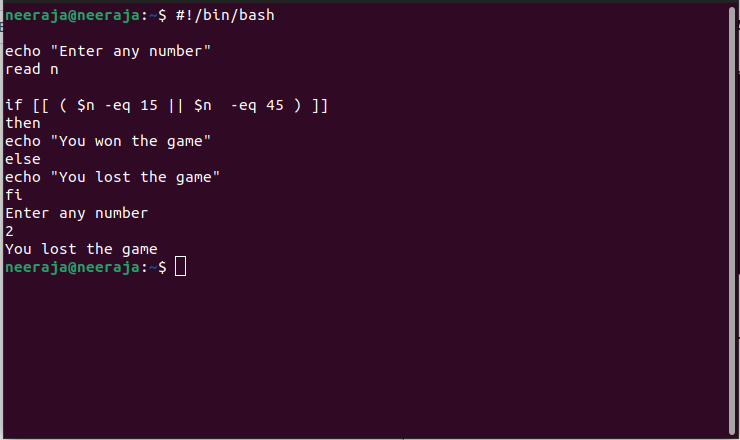
### Using if statement:



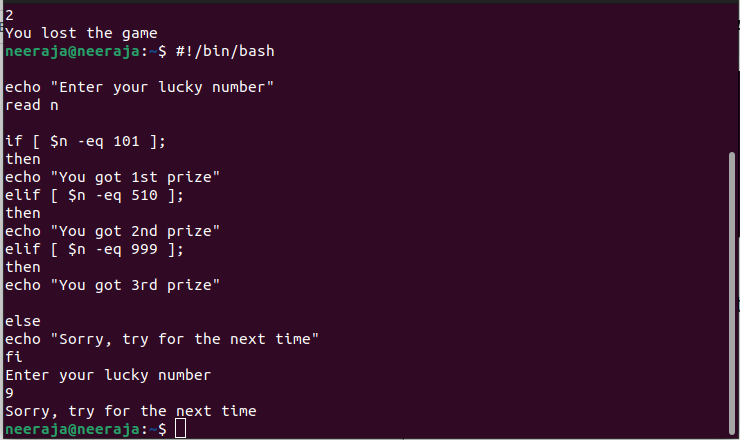
### Using if statement with AND logic:



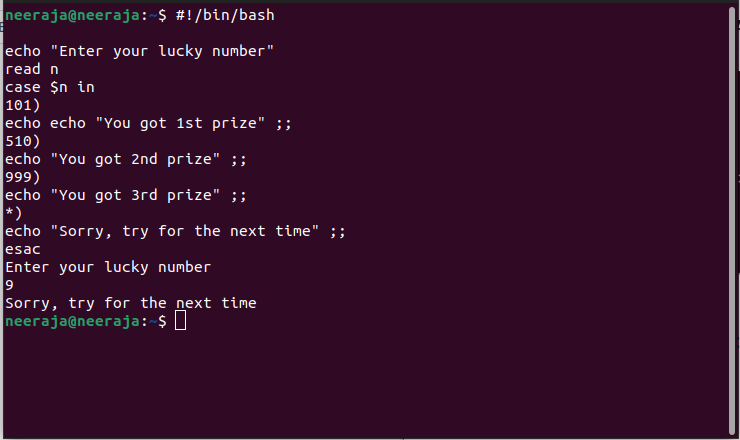
### Using if statement with OR logic:



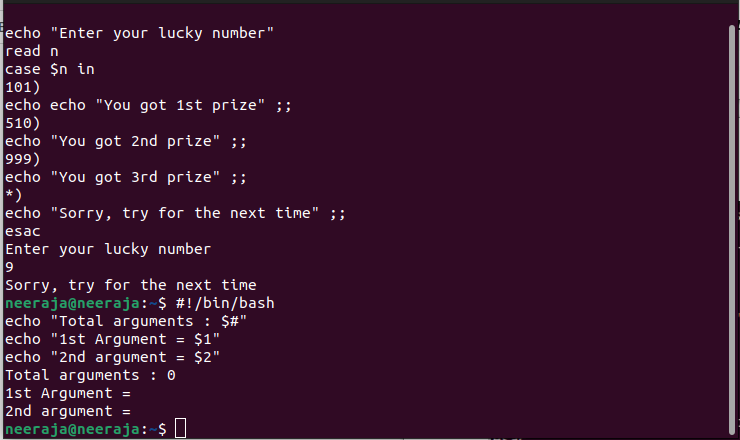
### Using else if statement:



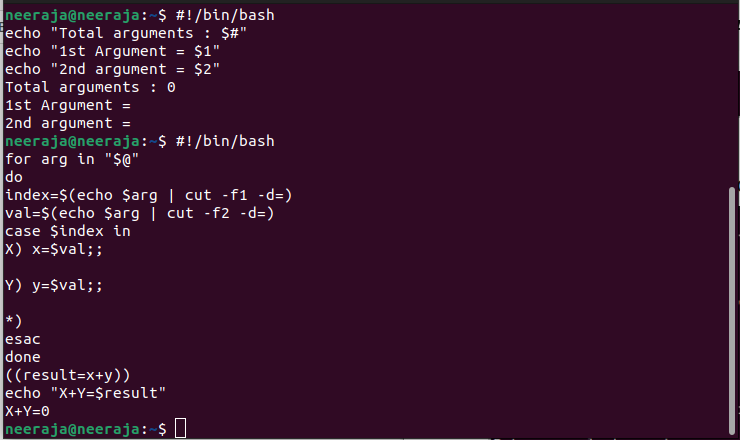
### Using Case Statement:



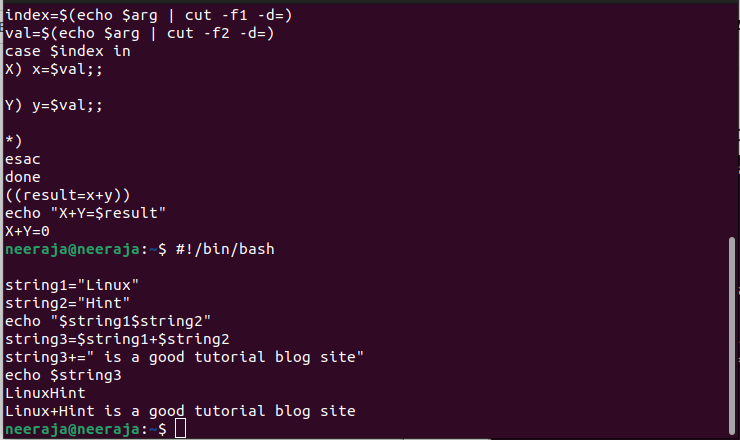
### Get Arguments from Command Line:



### Get arguments from command line with names:



### Combine String variables:



### Get substring of String:

