

TCP/IP Practicals - End Term

Name : Aditya Shivram Mahajan
Branch : Computer Science
Reg No. : 2017BCS001
Roll No. : A10
Year : Final Year



Index

Practical 1 : Write a program for Subnet design
(Classless addressing). - [link](#)

Practical 2 : Write a program for IP Header Checksum
Calculation. - [link](#)

Practical 3 : Write a program for Socket programming. - [link](#)

Practical 4 : Write a script to create a static Web Document/Web page
(For example: Your Resume). - [link](#)

Practical 5 : Study the following commands and utilities - Ping,
Traceroute/Tracert, Ipconfig/Ifconfig. - [link](#)



Practical 1 : Write a program for Subnet design (Classless addressing).

Theory:

The concept of subnet was introduced to address the following requirement. Consider an internet that includes one or more WANs and a number of sites, each of which has a number of LANs. We would like to allow arbitrary complexity of interconnected LAN structures within an organization while insulating the overall internet against explosive growth in network numbers and routing complexity.

One approach to this problem is to assign a single network number to all of the LANs at a site. From the point of view of the rest of the internet, there is a single network at that site, which simplifies addressing and routing. To allow the routers within the site to function properly, each LAN is assigned a subnet number. The host portion of the internet address is partitioned into a subnet number and a host number to accommodate this new level of addressing.

To reduce the wastage of IP addresses in a block, we use subnetting. What we do is that we use host id bits as net id bits of a classful IP address. We give the IP address and define the number of bits for the mask along with it (usually followed by a '/' symbol), like, 192.168.1.1/28. Here, the subnet mask is found by putting the given number of bits out of 32 as 1, like, in the given address, we need to put 28 out of 32 bits as 1 and the rest as 0, and so, the subnet mask would be 255.255.255.240.

Some values calculated in subnetting :

1. Number of subnets : Given bits for mask – No. of bits in default mask
2. Subnet address : AND result of subnet mask and the given IP address
3. Broadcast address : By putting the host bits as 1 and retaining the network bits as in the IP address
4. Number of hosts per subnet : $2(32 - \text{Given bits for mask}) - 2$
5. First Host ID : Subnet address + 1 (adding one to the binary representation of the subnet address)

Program :

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

void extractIpAddress(unsigned char * sourceString, short *
ipAddress) {
    unsigned short len = 0;
    unsigned char oct[5] = {0}, cnt = 0, cnt1 = 0, i, buf[8];
    len = strlen(sourceString);
    for (i = 0; i < len; i++) {
        if (sourceString[i] != '.') {
            buf[cnt++] = sourceString[i];
        }
        if (sourceString[i] == '.' || i == len - 1 || sourceString[i]
== '/') {
            buf[cnt] = '\0';
            cnt = 0;
            oct[cnt1++] = atoi(buf);
        }
    }
    ipAddress[0] = oct[0];
    ipAddress[1] = oct[1];
    ipAddress[2] = oct[2];
    ipAddress[3] = oct[3];
    ipAddress[4] = oct[4];
}

int main() {
    unsigned char ip[20] = {0};
    short ipAddress[5];
    short ipAddress1[4];
    short ipAddress2[4];
    short ipAddress3[4];
    short nwprefix, suffix, newnwprefix;
    short initaddr[4];
    int mask;
    int ntk[32];
    int subs, sizeofblock, remaining, addr;
```

```

printf("Enter a Classless IP Block: ");
scanf("%s", ip);
printf("Enter number of subnets: ");
scanf("%d", & subs);
extractIpAddress(ip, & ipAddress[0]);
printf("Ip Address: %
d. % d. % d. % d / % d\n
",ipAddress[0],ipAddress[1],ipAddress[2],ipAddress[3],ipAddress[4]);
sizeofblock = (int) pow(2, 32 - ipAddress[4]);
if (ipAddress[3] % sizeofblock == 0) {
printf("It is a valid CIDR block\n");
} else {
printf("It is not a valid CIDR block\n");
exit(0);
}
newnwprefix = ipAddress[4] + (int)(floor(log(subs) / log(2)));
printf("Length of new subnet mask=%d\n", newnwprefix); remaining = 32
- newnwprefix; addr = (int) pow(2, remaining); printf("Number of
addresses per subnet: %d\n", addr); ipAddress1[0] = (addr - 1 >> 24)
& 0xFF; ipAddress1[1] = (addr - 1 >> 16) & 0xFF; ipAddress1[2] =
(addr - 1 >> 8) & 0xFF; ipAddress1[3] = addr - 1 & 0xFF;
printf("Range of first subnet: %d.%d.%d.%d/%d - %
d. % d. % d. % d / % d\n
",ipAddress[0],ipAddress[1],ipAddress[2],ipAddress[3],newnwprefix,ipA
ddress[
0] + ipAddress1[0], ipAddress[1] + ipAddress1[1], ipAddress[2]
+ ipAddress1[2], ipAddress[3] + ipAddress 1[3], newnwprefix);

ipAddress1[0] = (sizeofblock - 1 >> 24) & 0xFF;
ipAddress1[1] = (sizeofblock - 1 >> 16) & 0xFF;
ipAddress1[2] = (sizeofblock - 1 >> 8) & 0xFF;
ipAddress1[3] = sizeofblock - 1 & 0xFF;
ipAddress2[0] = ipAddress1[0] + ipAddress[0];
ipAddress2[1] = ipAddress1[1] + ipAddress[1];
ipAddress2[2] = ipAddress1[2] + ipAddress[2];
ipAddress2[3] = ipAddress1[3] + ipAddress[3];
ipAddress3[0] = (addr - 1 >> 24) & 0xFF;
ipAddress3[1] = (addr - 1 >> 16) & 0xFF;
ipAddress3[2] = (addr - 1 >> 8) & 0xFF;

```

```

        ipAddress3[3] = addr - 1 & 0xFF;

        printf("Range of last subnet: %d.%d.%d.%d/%d - %d.%d.%d.%d/%d",
ipAddress2[0] -
        ipAddress3[0], ipAddress2[1] - ipAddress3[1], ipAddress2[2] -
ipAddress3[2], ipAddress2[3] -
        ipAddress3[3], newnwprefix, ipAddress2[0], ipAddress2[1],
ipAddress2[2], ipAddress2[3], newnwprefix);
    }

```

Output :

```

skystone@skystone-HP-Notebook: ~/Desktop/TCP IP practicals
skystone@skystone-HP-Notebook:~/Desktop/TCP IP practicals$ gcc -o 1 1.c -lm
1.c: In function 'extractIpAddress':
1.c:16:27: warning: implicit declaration of function 'atoi' [-Wimplicit-function-declaration]
   16 |         oct[cnt1++] = atoi(buf);
      |                             ^~~~~~
1.c: In function 'main':
1.c:47:9: warning: implicit declaration of function 'exit' [-Wimplicit-function-declaration]
   47 |         exit(0);
      |         ^~~~~
1.c:47:9: warning: incompatible implicit declaration of built-in function 'exit'
1.c:4:1: note: include '<stdlib.h>' or provide a declaration of 'exit'
    3 | #include <math.h>
    +++ |+#include <stdlib.h>
    4 |
skystone@skystone-HP-Notebook:~/Desktop/TCP IP practicals$ ./1
Enter a Classless IP Block: 160.85.16.0/20
Enter number of subnets: 16
Ip Address: 160. 85. 16. 0 / 20
It is a valid CIDR block
Length of new subnet mask=24
Number of addresses per subnet: 256
Range of first subnet: 160.85.16.0/24 - 160. 85. 16. 255 / 24
skystone@skystone-HP-Notebook:~/Desktop/TCP IP practicals$ 

```

Practical 2 : Write a program for IP Header Checksum Calculation.

Theory :

The IPv4 header checksum is a checksum used in version 4 of the Internet Protocol (IPv4) to detect corruption in the header of IPv4 packets. It is carried in the IP packet header, and represents the 16-bit result of summation of the header words.

What is Checksum :

A check sum is basically a value that is computed from a data packet to check its integrity. Through integrity, we mean a check on whether the data received is error free or not. This is because while traveling on the network a data packet can become corrupt and there has to be a way at the receiving end to know that data is corrupted or not. This is the reason the checksum field is added to the header. At the source side, the checksum is calculated and set in the header as a field. At the destination side, the checksum is again calculated and cross checked with the existing checksum value in the header to see if the data packet is OK or not.

IP Header Checksum :

The IPv6 protocol does not use header checksums. Its designers considered that the whole-packet link layer checksumming provided in protocols, such as PPP and Ethernet, combined with the use of checksums in upper layer protocols such as TCP and UDP, are sufficient.[1] Thus, IPv6 routers are relieved of the task of recomputing the checksum whenever the packet changes, for instance by the lowering of the Hop limit counter on every hop.

0	4	8	16	19	31
Version	Header Length	Service Type	Total Length		
Identification			Flags	Fragment Offset	
TTL	Protocol		Header Checksum		
Source IP Addr					
Destination IP Addr					
Options				Padding	

Program :

```

#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <sys/socket.h>
#include <netinet/ip.h>
#include <netinet/udp.h>
#include <arpa/inet.h>
#define PKT_LEN 8192

struct ipheader {
    unsigned char iph_ihl: 5, iph_ver: 4;
    unsigned char iph_tos;
    unsigned short int iph_len;
    unsigned short int iph_ident;
    unsigned char
    iph_flag;
    unsigned short int iph_offset;
    unsigned char iph_ttl;
    unsigned char iph_protocol;
    unsigned short int iph_checksum;
    unsigned int iph_sourceip;
    unsigned int iph_destip;
    unsigned short ip_offset;
    unsigned short ip_checksum;
};

unsigned short csum(unsigned short * buf, int nwords) {
    unsigned long sum;
    for (sum = 0; nwords > 0; nwords--)
        sum += * buf++;
    sum = (sum >> 16) + (sum & 0xffff);
    sum += (sum >> 16);
    return (unsigned short)(~sum);
}

int main(int argc, char * argv[]) {
    int sd;
    char buffer[PKT_LEN];

```

```

    struct ipheader * ip = (struct ipheader * ) buffer;
    ip -> iph_ihl = 5;
    ip -> iph_ver = 4;
    ip -> iph_tos = 0;
    ip -> ip_offset = 0;
    ip -> ip_checksum = 0;
    ip -> iph_flag = 0;
    ip -> iph_len = sizeof(struct ipheader) + atoi(argv[3]);
    ip -> iph_ident = htons(atoi(argv[4]));
    ip -> iph_ttl = atoi(argv[5]);
    ip -> iph_protocol = atoi(argv[6]);
    ip -> iph_sourceip = inet_addr(argv[1]);
    ip -> iph_destip = inet_addr(argv[2]);
    ip -> iph_chksum = csum((unsigned short * ) buffer, ip ->
iph_len);
    printf("%x\n", ip -> iph_chksum);
}

```

Output :

```

skystone@skystone-HP-Notebook: ~/Desktop/TCP IP practic...
skystone@skystone-HP-Notebook:~/Desktop/TCP IP practicals/Practical 2$ gcc 2.c
2.c: In function 'main':
2.c:47:19: warning: assignment to 'unsigned char' from 'char *' makes integer fr
om pointer without a cast [-Wint-conversion]
   47 |     ip -> iph_ttl = argv[5];
      |                   ^
2.c:48:24: warning: assignment to 'unsigned char' from 'char *' makes integer fr
om pointer without a cast [-Wint-conversion]
   48 |     ip -> iph_protocol = argv[6];
      |                       ^
skystone@skystone-HP-Notebook:~/Desktop/TCP IP practicals/Practical 2$ ./a.out 1
95.168.1.1 195.168.1.3 40 100 2 17
3d2a
skystone@skystone-HP-Notebook:~/Desktop/TCP IP practicals/Practical 2$

```


Practical 3 : Write a program for Socket programming.

Theory :

What is socket programming?

Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket(node) listens on a particular port at an IP, while other socket reaches out to the other to form a connection. Server forms the listener socket while client reaches out to the server.

Program : Server.py

```
import socket
import threading

HEADER = 64
PORT = 5050
SERVER = "192.168.1.105"
#SERVER = socket.gethostname(socket.gethostname())
ADDR = (SERVER, PORT)
FORMAT = 'utf-8'
DISCONNECT_MESSAGE = "!DISCONNECT"

server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind(ADDR)

def handle_client(conn, addr):
    print(f"[NEW CONNECTION] {addr} connected.")

    connected = True
    while connected:
        msg_length = conn.recv(HEADER).decode(FORMAT)
        if msg_length:
            msg_length = int(msg_length)
            msg = conn.recv(msg_length).decode(FORMAT)
            if msg == DISCONNECT_MESSAGE:
                connected = False

            print(f"[{addr}] {msg}")
            conn.send("Msg received".encode(FORMAT))

    conn.close()
```

```

def start():
    server.listen()
    print(f"[LISTENING] Server is listening on {SERVER}")
    while True:
        conn, addr = server.accept()
        thread = threading.Thread(target=handle_client, args=(conn, addr))
        thread.start()
        print(f"[ACTIVE CONNECTIONS] {threading.activeCount() - 1}")

print("[STARTING] server is starting...")
start()

```

Program : Client.py

```

import socket

HEADER = 64
PORT = 5050
FORMAT = 'utf-8'
DISCONNECT_MESSAGE = "!DISCONNECT"
SERVER = "192.168.1.105"
ADDR = (SERVER, PORT)

client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(ADDR)

def send(msg):
    message = msg.encode(FORMAT)
    msg_length = len(message)
    send_length = str(msg_length).encode(FORMAT)
    send_length += b' ' * (HEADER - len(send_length))
    client.send(send_length)
    client.send(message)
    print(client.recv(2048).decode(FORMAT))

send("Hello World!")
input()
send("Hello Everyone!")
input()

```

```
send(DISCONNECT_MESSAGE)
```

The image shows a Visual Studio Code editor window with the following components:

- Explorer Panel:** Displays the file structure with folders for `server.py`, `client.py`, and `socket-programming`. The `client.py` file is selected.
- Editor Panel:** Shows the code for `client.py`. The code is as follows:

```
1 import socket
2
3 HEADER = 64
4 PORT = 5050
5 FORMAT = 'utf-8'
6 DISCONNECT_MESSAGE = "!DISCONNECT"
7 SERVER = "192.168.1.105"
8 ADDR = (SERVER, PORT)
9
10 client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
11 client.connect(ADDR)
12
13 def send(msg):
14     message = msg.encode(FORMAT)
15     msg_length = len(message)
16     send_length = str(msg_length).encode(FORMAT)
17     send_length += b' ' * (HEADER - len(send_length))
```
- Terminal Panel:** Shows the output of the command `python3 client.py`. The output is:

```
(base) skystone@skystone-HP-Notebook: /opt/lampp/htdocs/github/s
ocket-Programming$ source /home/skystone/anaconda3/bin/activate
(base) skystone@skystone-HP-Notebook: /opt/lampp/htdocs/github/s
ocket-Programming$ python3 client.py
Msg received
Msg received
Msg received
(base) skystone@skystone-HP-Notebook: /opt/lampp/htdocs/github/s
ocket-Programming$
```

Practical 4 : Write a script to create a static Web Document/Web page (For example: Your Resume).

HTML Code :

```
<?php include './includes/header.php'; ?>

<style type="text/css">
    .trans-card{
        background-color: transparent;
    }

    html,
    body,
    header,
    .view {
        height: 100%;
    }

    .intrests:hover {
        box-shadow: 0px 0px 40px 16px rgba(18,18,18,1.00);
    }

    @media (max-width: 740px) {
        html,
        body,
        header,
        .view {
            height: 1000px;
        }
    }

    @media (min-width: 800px) and (max-width: 850px) {
        html,
        body,
        header,
        .view {
            height: 650px;
        }
    }

    @media (min-width: 800px) and (max-width: 850px) {
        .navbar:not(.top-nav-collapse) {
            background: #1C2331!important;
        }
    }
</style>
```

```

<!-- Full Page Intro -->
<div class="view full-page-intro" style="background-image:
url('https://mdbootstrap.com/img/Photos/Others/images/78.jpg'); background-repeat:
no-repeat; background-size: cover;">

    <!-- Mask & flexbox options-->
    <div class="mask rgba-black-light d-flex justify-content-center
align-items-center">

        <!-- Content -->
        <div class="container">
            <br><br><br>
            <!--Grid row-->
            <div class="row wow fadeIn">
                <!--Grid column-->
                <div class="col-md-6 mb-4 white-text text-center text-md-left">
                    <h1 class="display-4 font-weight-bold">Hello There <br>Aditya
Mahajan Here... !!!</h1>
                    <hr class="hr-light">
                    <p>
                        <strong>Well inshort about me?? &#128521; </strong>
                    </p>
                    <p class="mb-4 d-none d-md-block">
                        <strong>Currently pursing B.Tech Computer Science from
Shri Guru Gobind Singhji Institute of engineering and technology,Vishnupuri,
Nanded, Maharashtra. </strong>
                    </p>

                    <a href="myProjects.php" class="btn btn-indigo btn-lg">Projects
                        <i class="fas fa-graduation-cap ml-2"></i>
                    </a>
                </div>
                <!--Grid column-->
                <!--Grid column-->
                <div class="col-md-6 col-xl-5 mb-4">
                    <!--Card-->
                    <div class="card trans-card z-depth-5">
                        <!--Card content-->
                        <div class="intrests card-body">
                            <!-- Form -->
                            <form name="" class="white-text opacity-50 ">
                                <!-- Heading -->
                                <h3 class="text-center">
                                    <strong>Things I like to Do...</strong>
                                </h3>
                                <hr>

```

```

fa-code"></i></li>

<ul>
<li>Coding (Mostly in C++ and Python) <i class="fa

</ul>
<ul>
<li>Problem Solving (Algorithms)</li>
</ul>
<ul>
<li>Photography <i class="fa fa-camera"></i></li>
</ul>
<ul>
<li>Cooking <i class="fa fa-cutlery"

aria-hidden="true"></i></li>

</ul>
<ul>
<li>Travelling</li>
</ul>
<ul>
<li>Lawn Tennis, Gymnastics, Skating </i></li>
</ul>

<!-- <div class="md-form">
<i class="fas fa-user prefix grey-text"></i>
<input type="text" id="form3"

class="form-control">

<label for="form3">Your name</label>
</div>
<div class="md-form">
<i class="fas fa-envelope prefix grey-text"></i>
<input type="text" id="form2"

class="form-control">

<label for="form2">Your email</label>
</div>
<div class="md-form">
<i class="fas fa-pencil-alt prefix grey-text"></i>
<textarea type="text" id="form8"

class="md-textarea"></textarea>

<label for="form8">Your message</label>
</div>
<div class="text-center">
<button class="btn btn-indigo">Send</button>
<hr>
<fieldset class="form-check">
<input type="checkbox"

class="form-check-input" id="checkbox1">

```

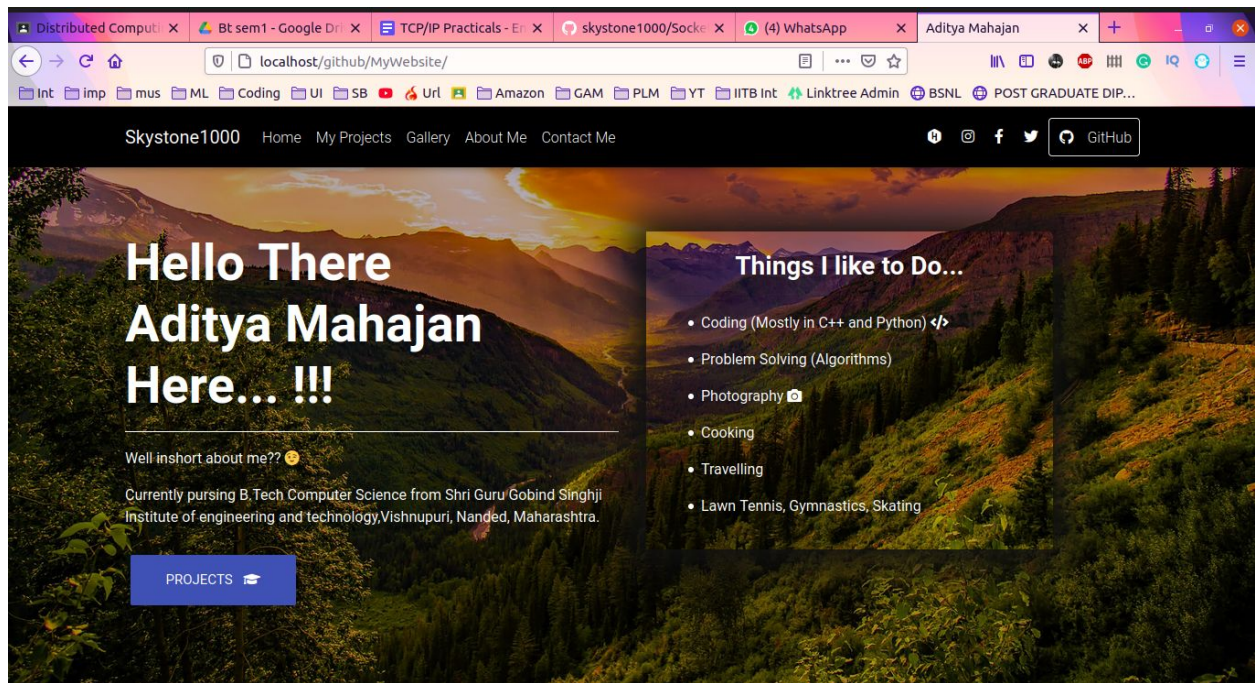
```

                <label for="checkbox1"
class="form-check-label dark-grey-text">Subscribe me to the newsletter</label>
            </fieldset>
        </div> -->
    </form>
    <!-- Form -->
</div>
<!--/.Card-->
</div>
<!--Grid column-->
</div>
<!--Grid row-->
</div>
<!-- Content -->
</div>
<!-- Mask & flexbox options-->
</div>
<!-- Full Page Intro -->

<?php include './includes/footer.php'; ?>

```

Output :



Practical 5 : Study the following commands and utilities

Ping, Traceroute/Tracert, Ipconfig/Iconfig.

Theory :

Ping

Ping is a computer network administration software utility used to test the reachability of a host on an Internet Protocol (IP) network. It is available for virtually all operating systems that have networking capability, including most embedded network administration software.

Ping measures the round-trip time for messages sent from the originating host to a destination computer that are echoed back to the source. The name comes from active sonar terminology that sends a pulse of sound and listens for the echo to detect objects under water.[1]

Ping operates by sending Internet Control Message Protocol (ICMP) echo request packets to the target host and waiting for an ICMP echo reply. The program reports errors, packet loss, and a statistical summary of the results, typically including the minimum, maximum, the mean round-trip times, and standard deviation of the mean.

The command-line options of the ping utility and its output vary between the numerous implementations. Options may include the size of the payload, count of tests, limits for the number of network hops (TTL) that probes traverse, interval between the requests and time to wait for a response. Many systems provide a companion utility ping6, for testing on Internet Protocol version 6 (IPv6) networks, which implement ICMPv6.

```
Microsoft(R) Windows DOS
(C)Copyright Microsoft Corp 1990-2001.

C:\DOCUME~1\CHAD\DESKTOP>ping www.youtube.com

Pinging youtube-ui.l.google.com [74.125.127.113] with 32 bytes of data:

Reply from 74.125.127.113: bytes=32 time=53ms TTL=247
Reply from 74.125.127.113: bytes=32 time=55ms TTL=247
Reply from 74.125.127.113: bytes=32 time=54ms TTL=247
Reply from 74.125.127.113: bytes=32 time=53ms TTL=247

Ping statistics for 74.125.127.113:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 53ms, Maximum = 55ms, Average = 53ms

C:\DOCUME~1\CHAD\DESKTOP>
```


Traceroute/Tracert

In computing, traceroute and tracert are computer network diagnostic commands for displaying possible routes (paths) and measuring transit delays of packets across an Internet Protocol (IP) network. The history of the route is recorded as the round-trip times of the packets received from each successive host (remote node) in the route (path); the sum of the mean times in each hop is a measure of the total time spent to establish the connection. Traceroute proceeds unless all (usually three) sent packets are lost more than twice; then the connection is lost and the route cannot be evaluated. Ping, on the other hand, only computes the final round-trip times from the destination point.

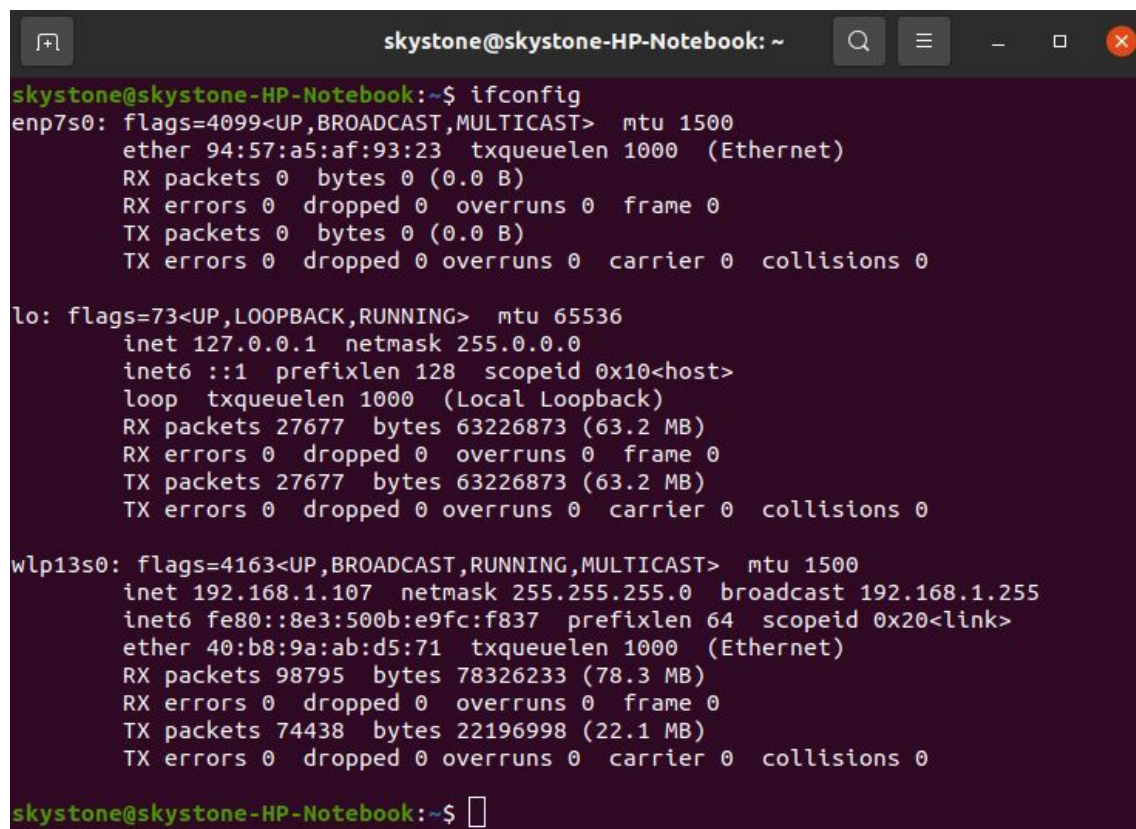
For Internet Protocol Version 6 (IPv6) the tool sometimes has the name traceroute6 or tracert6.

```
$ traceroute -w 3 -q 1 -m 16 example.com
```

Ipconfig/Ifconfig

ifconfig is a system administration utility in Unix-like operating systems for network interface configuration.

The utility is a command-line interface tool and is also used in the system startup scripts of many operating systems. It has features for configuring, controlling, and querying TCP/IP network interface parameters. Ifconfig originally appeared in 4.2BSD as part of the BSD TCP/IP suite.

A terminal window titled 'skystone@skystone-HP-Notebook: ~' with search, menu, and window control icons. The terminal shows the command 'ifconfig' being executed. The output displays details for three network interfaces: 'enp7s0' (Ethernet), 'lo' (Local Loopback), and 'wlp13s0' (Wireless LAN). Each interface listing includes flags, MTU, IP address, netmask, broadcast address, MAC address, and statistics for RX and TX packets, bytes, errors, dropped, overruns, carrier, and collisions.

```
skystone@skystone-HP-Notebook:~$ ifconfig
enp7s0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether 94:57:a5:af:93:23 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    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 27677 bytes 63226873 (63.2 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 27677 bytes 63226873 (63.2 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlp13s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.107 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::8e3:500b:e9fc:f837 prefixlen 64 scopeid 0x20<link>
    ether 40:b8:9a:ab:d5:71 txqueuelen 1000 (Ethernet)
    RX packets 98795 bytes 78326233 (78.3 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 74438 bytes 22196998 (22.1 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

skystone@skystone-HP-Notebook:~$
```