CMPS 6100 Lab 10

In this lab, you will learn about network exploration tools on the shell and write your first networked program.

Refer back to the README.md for instruction on git and how to submit properly to get all the points you've earned.

Introduction

In this lab, you will use several shell commands to become familiar computer networking capabilities and with the internet.

We recommend working on our cmps-6100-lab server, but you can also try these commands on your local computer.

Ping

The ping command sends a request packet to a destination using a special ICMP protocol. For example, you could run `ping google.com` or `ping 173.194.33.174` to ping a domain name or IP address.

These packets ask the remote destination to reply. If the remote destination is configured to reply, it will respond with packets of its own. You'll be able to see how long the round-trip time is between your computer and the destination. You'll see a "request timed out" message if packet loss is occurring, and you'll see an error message if your computer can't communicate with the remote host at all. In fact, ping may send several request packets and average the round-trip time for them.

This tool can help you troubleshoot Internet connection problems: if you are having troubles loading a webpage, this may mean several things, e.g.:

- The webserver containing the webpage is down/disconnected from the Internet
- There is a problem with your browser
- Your computer is not connected to the Internet

You can use ping to verify that you're indeed connected to the Internet, although do keep in mind that many devices and servers are configured to not reply to pings.

Execute ping on the following websites (you can quit ping by hitting CTRL-C): - google.com - harvard.edu - bsu.by (Belarusian State University, server presumably located in Belarus) - Foreign website of your choice. A foreign university that you or a friend attended/wish to attend would be a good choice. Unlike businesses which tend to by hosting from American Companies, universities usually host their websites locally.

1. Examine the resulting output of ping command. For each of the websites, create a table and report average roundtrip time and the time of the day when you made the measurements.

   Copy the output from your commands into answers.md

nslookup

The `nslookup` command will look up the IP addresses associated with a domain name. For example, you can run `nslookup google.com` to see the IP address of google.com's server.

Your computer is constantly querying a DNS (domain name service, that is used to find IP addresses corresponding to the human-readable addresses such as google.com or Wikipedia.org) servers to translate domain names to IP addresses. This command just allows you to do it manually.

`nslookup` also allows you to perform a reverse lookup to find the domain name associated with an IP address. For example, `nslookup 192.0.43.7` will show you that this IP address is associated with `icann.org`.

2. Execute nslookup using the following parameters and report the outcome (what is a non-authorative answer?)

- iana.com
- nola.com
- google.com
- 172.253.62.100

  Copy the output from your commands into answers.md

netstat

`netstat` stands for network statistics. This command displays incoming and outgoing network connections as well as other network information.

The `netstat` utility can show you the open connections on your computer, which programs are making which connections, how much data is being transmitted, and other information.

3. Execute `netstat` and wait for it to complete (stop if you wait longer than 5 minutes). Examine the output.

- In the local or foreign address column you might see an IP address and some additional number (like 61262) separated by a colon or period. What is the meaning of that number?

Copy the output from your commands into answers.md (only include the Active Internet connections output. Don't include the UNIX domain sockets)

ip

The `ip` command is an all-purpose command for displaying and manipulating the networking configuration for a device.

For example, `ip address show` displays all network adapters and their IP address, `ip tunnel` allows you to configure network tunnels, and `ip route` displays your device's routing table.

4. Execute `ip address show`.

- What devices are listed? (Hint: the first device is `lo`)
  - What is the IPv4 address of the `lo` device?
  - What does the `/8` at the end of the device indicate?
- What is the IPv4 address of `eth0`?
- What is the IPv6 address of eth0?

  Enter your answers in answers.md

Coding

Warmup

The simple client/server program from the textbook has been provided for you. Study it and answer the following questions.

5. In `server.c` what do each of the following calls do?

- `socket()`
- `bind()`
- `listen()`

- `accept()`

  Enter your answers in answers.md

6. In `client.c` what do each of the following calls do?

- `socket()`
- `connect()`

  Enter your answers in answers.md

Now that you've studied the code, let's run it! First, you will need to edit the code. In both files, the `SERVER_PORT` has been assigned the value of `NULL`, an invalid port number. For any computer, only a single server may be bound to any individual port number. Since we all share the same server, we must each choose our own port number.

To quarantee no conflicts, choose a port number of the format 5#### where the the last four digits are correspond with the last four digits of your student ID.

Update the client and server with your chosen port number. Then compile both the client and the server.

With that done, now start one server and one client, in separate windows.

While the first client is running, start 5 other clients that connect to the same server; these other clients should be started in the background with their input redirected from a file.

6. What happens to these 5 clients? Do their `connect()`'s fail, or time out, or succeed?

   Enter your answer in answers.md

7. Now let the first client exit (`ctrl-c`). What happens?

   Enter your answer in answers.md

Chat Program

Your task is to turn this basic client/server application into a chat program. Your client will ask the user for a username and then connect to the server.

The client will emit a prompt (`"> "`), read in a message from the user, then send it to the server. At this point the client will wait for the server's response.

> **Note:** The client must remove any trailing new line characters from the end of the message before sending it to the server. This will be important uniform implementations so that in the next lab, when you implement a server that can accept multiple clients, you can connect your client to your peers' servers and vice-verse.

The server will print the client's message to the screen in the format:

```
username: user's message
```

The server will emit a prompt (`"> "`), read in a message from the user, and send it to the client.

The client will recieve this message and print it to the screen in the format:

```
server: server's message
```

This will continue until the user kills the programs.

Example Execution

Here is the client's side of the chat:

```
amaus@cmps-6100:~/labs/devel/chat $ gcc client.c -o client && ./client localhost
Welcome to my chat program!
Please enter a username: amaus
Hi amaus!
You may start talking to the server.
> Hi chat program!
server: Hello client user. Shall we chat?
> Ok, although I don't know what to say.
server: I get it, it is odd talking with yourself.
>
```

And here is the server's side:

```
amaus@cmps-6100:~/labs/devel/chat $ gcc server.c -o server && ./server
amaus has connected to chat.
amaus: Hi chat program!
> Hello client user. Shall we chat?
amaus: Ok, although I don't know what to say.
> I get it, it is odd talking with yourself.
```

Resources

Ch. 1.4: Software

Harvard's CS50 Manual Page