CS 353: Programming Assignment 2

Revision History:
2/10: Release
2/11: Added clarification about timestamp formatting in webserver files

The main goal of this assignment is to give you hands-on experience in developing tools for network traffic analysis and management. In this assignment you will develop a libpcap-based tool to analyze and balance network traffic at a webserver farm.

Develop a tool, called `balancer`, that runs in two modes; the analysis mode and the balancing mode. In the analysis mode, it observes the traffic and outputs a summary report at the end. In the balancing mode, it enforces the directive of the configuration and manages the traffic seen on the network by balancing packets across the servers in the webserver farm.

The Analysis Mode

Develop a tool, called `balancer`, to display packet and bytes counts for IPv4 packets. Based on the command line options, the tool analyzes the packets in a file or on the network interface and generates a summary report before quitting. The summary report has two, three or four columns of data based on the command line options as explained below.

Command Line Options:

> `balancer [-r filename] [-i interface] [ -l filename ] [-p] [-b] [-s] [-d]`

where

```
-r, --read          Read the specified pcap file
-i, --interface     Listen on the specified interface
-l, --logfile       Logfile with the summary report
-p, --packet        Output packet counts
-b, --byte          Output byte counts
-s, --src           Analyze based on source IP
-d, --dst           Analyze based on destination IP
```

When run without any options, it should print the usage instructions and exit.

Input:

The tools accepts two forms of network traffic for analysis. It accepts a pcap file specified with the –r filename. It also analyzes live network traffic specified with –i interface as input for analysis.

Output:

The tool generates an output report upon the completion of the analysis. The analysis is completed either when the pcap file is completely read or when the user presses Ctrl+C in the terminal window when capturing live network traffic.

The summary report is written in the logfile specified with –l filename on the command line. The output format of the summary report is specified below.

Operation:
The number of columns of analysis in the summary report is based on the command line options.

Two Columns are generated when the tool is run with either **one** of the –s (source IP) or the –d (destination IP) option along with either **one** of the –p (packets) or the –b (bytes) options along. There are a total of four variations, namely "–s –p" or "-s –b", or "-d –p" or –d –b". For example, assume the tool is run with the –s –p option. Then the first column is the observed source IP addresses and the second column is the number of packets that originated from the IP address. When the tool is run with the –d –p option, the first column is the observed destination IP addresses and the second column is the number of packets that were sent to the IP address. Similarly for byte counts.

Three Columns are generated when three command line options are used. When the tool is run with **one** of the –s (source IP) or the –d (destination IP) option along with **both** the –p (packets) and the –b (bytes) options. Then the first column should be the source/destination IP addresses, followed by a packet count column and a byte count column respectively. There are a total of two variations, namely, "-s –p –b" or "–d –p –b" .
When the tool is run with **both** –s (source IP) and the –d (destination IP) option along with **one** of the –p (packets) and the –b (bytes) options. The first column should be the source IP address column, followed by the destination IP address column, followed by the packet/byte count columns. This will yield two variations, namely, "-s –d –p" and "-s –d –b"

Four Columns: There is only one variation in this case. When the tools is run with all four command line options, namely, "-s –d –p –b". The first column should be the source IP address, the second column should be the destination IP address, followed by a packet count column and a byte count columns respectively.

The Balancing Mode

Extend the `balancer` tool to support balancing network traffic flows. Each flow is defined as a five-tuple, consisting of source IP, destination IP, source port, destination port, and protocol type. The configuration percentage on the command line indicates how the tool balances the flows in a file or on the network interface. It creates a log to record the balancing decisions and generates two or more files to balance the traffic across the servers in the web farm.

Command Line Options:
`> balancer [-r filename] [-i interface] [-w num] [ -l filename ] [-c configpercent]`
where

```
 -r, --read          Read the specified pcap file
 -i, --interface     Listen on the specified interface
 -l, --logfile       Logfile with the summary report
 -c, --config        Percentage of flows to balance across each server
 -w, --webserver     The number of webservers to balance across
```

When run without any options, it should print the usage instructions and exit.

Input:
The tools accepts two forms of network traffic. It accepts a pcap file specified with the –r filename. It also balances live network traffic specified with –i interface.

The –c specifies the balancing percentage for the tool. The string will consist of ":" separated traffic percentage values. For example, the options –w 3 –c 33:33:34 would imply, there are three servers in the webserver farm, the first server receives 33% of the traffic flows, the second receives 33% of the flows and the third receives 34% of the traffic flows.

Output:
The tool generates a log to record the balancing decision for each packet. The log is written in a logfile specified with –l filename on the command line.

The tool also generates two or more webserver files. The number of webserver files is based on the -w option. Each webserver file should be named webserver.X, where X is the index of the server. For example, -w 3 would create three webserver files, namely, webserver.1, webserver.2 and webserver.3.

Operation:
For each packet received by the tool, the logfile creates a 3-tuple entry on a new line. The first value in the tuple is the packet index. The packet index starts from 1 and is used to indicate the total number of packet received by the balancer tool at any point in the operation. The packet index is incremented by one for each packet received by the tool. The second value in the tuple indicates the flow index. The flow index also starts from 1 and is incremented every time a new flow is observed. As mentioned earlier, a flow is based on a unique combination of source IP, destination IP, source port, destination port and protocol type. Hence there will be several packets belonging to the same flow and should be assigned to the same server. The last value in the tuple indicates the index of the server the flow is assigned to. For example, the first packet received by the balancer tool will have the log entry as "1 1 1" to indicate first packet and first flow assigned to server 1.

Once the packet is assigned to the correct server, the tool should also create an entry in the webserver file. Each entry should be on a new line. The entry is an 8-tuple consisting of the packet index, time, source IP, destination IP, source port, destination port, protocol type, and packet length in bytes. The time value indicates the packet capture time as recorded by the kernel in the pcap header. Print the time in UNIX time format as "seconds.microseconds"

Existing Tools and Libraries:
The libpcap library is widely used for network traffic analysis. "Tcpdump" , http://www.tcpdump.org is a popular network traffic analysis tool based on libpcap. Wireshark, http://www.wireshark.org also provides graphical user interface.

The wireshark wiki http://wiki.wireshark.org/Tools has an extensive list of tools based on the libpcap library. There is excellent primer on programming with pcap at http://www.tcpdump.org/pcap.html

Code and Collaboration:

You can also discuss the assignment and coding strategies with your classmates. However, your solution **must** be coded and written by yourself. Please refer to the plagiarism policy in the course syllabus.

The submissions will be run through code similarity tests. Any flagged submissions will result in a **failing score**. Keeping your code private is your responsibility.

Submission Instructions

Create a compressed tar file which includes a README, Makefile, and the source code. To submit, create a folder called assign2 with the code, Makefile, and README. Tar/zip the folder assign2. Submit the tar/zip file on blackboard.

The README must contain: USCID, compiling instructions, additional notes on usage if needed.

To evaluate your project we will untar and type
% make

It is a project requirement that your code build without any warnings (when com- piled with -Wall flag). Structure the Makefile such that it creates the balancer executable (not a.out). For more information please read the make(1) man page.

We will then run your program using a suite of test inputs. After running the program, we will grade your work based on the output. It is recommended that your implementation be somewhat modular. This means that you should follow good programming practices—keep functions relatively short, use descriptive variable names.

Deadlines

Due on March 4th by 6pm