Unix for Telecommunications

Portfolio Task – P-Lab-06-PCAP-Programming
Distinction Level Task

I. Introduction

In this lab you will learn how to use the **libpcap** (or similar) library with either the C or Python programming languages to parse packets. You will develop a tool to parse pre-captured PCAP files.

II. Purpose

To gain and/or enhance the following practical skills:

- Build a customised packet parsing tool/program instead of relying on existing tools (e.g tcpdump, tshark)
- Develop a better understanding of packet capture/sniffing tools
- Able to understand software documentation/examples and apply them
- Develop a deeper understanding of network traffic behaviour

III. Preparation

You can prepare for this lab by reading some of the documentation and examples available at:

- http://www.tcpdump.org/manpages/pcap.3pcap.html
- http://www.tcpdump.org/pcap.html
- https://eecs.wsu.edu/~sshaikot/docs/lbpcap/libpcap-tutorial.pdf
- http://recursos.aldabaknocking.com/libpcapHakin9LuisMartinGarcia.pdf

And for Python

- https://dpkt.readthedocs.io/en/latest/
- https://dpkt.readthedocs.io/en/latest/api/index.html

You should also review the basic concepts of:

- TCP/IP packets
- Packet sizes / lengths
- Existing packet analysers, such as tcpdump and tshark

IV. Requirements

You are to develop a second software tool to parse a PCAP file and display a summary of flow information within that file. The output format is standardised to allow for automated assessment of your work.

A. *Program Requirements*

Your program must be placed in the following directory:

```
/home/student/pcap_lab
```

Your program must be able to be executed directly from the command line and take one command line parameter. The command line parameter is the name of the PCAP file to parse. Execution should be as per:

```
./tcpsummary filename.pcap
```

Your program should open the PCAP file specified, process the contents of the file, and then print the required information.

B. Extracting and Displaying Flow Information

For each unique flow in the PCAP file, you are required to extract and print the following information:

- Flow tuple (source IP/Port, destination IP/Port, protocol)
- Timestamp of the first packet in the flow
- Total packets within the flow
- Total bytes within the flow

All packet information is to be displayed to the screen (**stdout**) in the following format:

```
--------------- PER-FLOW INFO -----------------

Flow: 172.16.11.72:35192-172.16.10.62:5000
Start time: 2018-07-25 01:25:32.370918
Total packets: 50078
Total bytes: 75110020

Flow: 172.16.10.62:5000-172.16.11.72:35192
Start time: 2018-07-25 01:25:32.392489
Total packets: 28154
Total bytes: 1504588

Flow: 172.16.11.72:32903-172.16.10.62:5001
Start time: 2018-07-25 01:25:32.483847
Total packets: 51247
Total bytes: 76864608

Flow: 172.16.10.62:5001-172.16.11.72:32903
Start time: 2018-07-25 01:25:32.520527
Total packets: 28662
Total bytes: 1529660

-----------------------------------------------
```

To enable automated assessment of your work, it is imperative that your output matches the format exactly. It is also important the order that flow information is printed is consistent, you are required to sort your flows in chronological order based on the timestamp of the first packet in the flow.

To simplify your code, you should consider packets flowing in the reverse direction of the flow being part of a different flow and have their own summary (eg. in a http flow between `192.168.0.1:45678` and `192.168.0.2:80`, packets from `192.168.0.1:45678` to `192.168.0.2:80` form a different flow to those from `192.168.0.2:80` to `192.168.0.1:45678`)

In order to complete this task you will need to consider:

- How to store information for all flows for later output?
- How to correctly order the output of flow information?
- How to update flow information as each packet is processed?
- How much extra consideration to put into the design of your solution to complete this task?
- If you are preparing a lab report, also answer how you might modify your program such that bi-directional packets of the same flow are combined into a single flow

C. PCAP files

Please develop and test your program using the same two PCAP files provided with the **P-Lab-06-PCAP-Programming-P** task resources.

When you execute the marking script to verify your program, it will execute your program against against the same three PCAP files as for the **P-Lab-06-PCAP-Programming-P** Pass Level task, keeping the contents and output of the third PCAP file hidden.

V. Choosing your Programming Language

You are free to choose whichever programming language you like to develop your software. Before making your selection, it is strongly recommended you consider the task and read the associated links for information. Unlike the **Pass Level** task, completing this **Distinction Level** task will be easier using a higher level programming language like Python. Otherwise the same issues apply as per the Pass task with your choice of programming language.

A. Sample Programs

As with the **Pass Level** task, we have provided two sample programs (in both C and Python) within the task resources available on Doubtfire. Both these sample programs contain three functions (**`flow_summary_header()`**, **`flow_summary_header()`** and **`flow_summary_single()`**). These functions will print parameterised variables to exactly match the format required to pass the lab assessment. It is strongly recommended you use these functions to help ensure your output meets the requirements for assessment.

VI. Assessment

The due date for completion of practical work is **11:00pm**, exactly **six** days after your scheduled class.

Note: *The nominated submission day/time holds regardless of whether that day is a non-teaching day or public holiday*

A. Self Assessment

You can self-assess your progress at any time via the marking script available at
http://ruleprimary1.caia.swin.edu.au

B. Completion of task in Doubtfire

Download the PDF output of the marking script from http://ruleprimary1.caia.swin.edu.au and submit it to Doubtfire. Your tutor will confirm completion of the lab by examing the rule marking log files on the rule server.

If you complete the task during class beforehand, you may demonstrate completion in class to your tutor.

Note: *The downloaded PDF is not evidence of successful completion of the lab, it is a document to demonstrate completion within your portfolio. Your tutor will assess the evidence via either direct confirmation via the marking script or via the log files generated when you run the marking script*

C. Tutor Discussion

In order for the submission to be marked as complete, you must discuss your work with the tutor