Lab 1: IP Lookup Algorithms

I. Abstract

In Lab 1, students will study several well-known IP lookup algorithms discussed in course lecture, and implement them using C programming language. To test code functionality, the algorithm modules will be integrated with packet capturer APIs provided by PCAP library, to allow reading IP packets from the laptop's Network Interface Card (NIC).

II. Lab Description

(1) Grouping and Assignment

In Lab 1, each group is required to implement and demonstrate two IP lookup algorithms. The algorithms to each group is based on the sum of the last digit of your and your partner's NYU student ID number. Your question number is the sum of the two digits modulus 7 and that value plus one.

For example Group A has Nxxxxxxx5 ; N xxxxxxxx3. The sum of their last two digits is 8.

8 mod 7 = 1 So group A should do question 1 and 2 which are

1. Disjoint Prefix Binary Trie
2. Path Compressed Trie

Group B has Nxxxxxxx7 ; N xxxxxxxx6 : (7+6) mod 7 = 6 So group B should do question 6 and 0.

The question set is shown as follows:

- 1. Disjoint Prefix Binary Trie
- 2. Path Compressed Trie
- 3.Multi Bit Trie
- 4.Multi Bit Trie with Leaf Pushing
- 5.Level Compression Trie
- 6.Binary Search on Prefix Lengths
- 0. Binary Search on Prefix Range

An implementation of basic **Binary Trie** lookup algorithm is provided in the reference code package.

(2) Lab Description

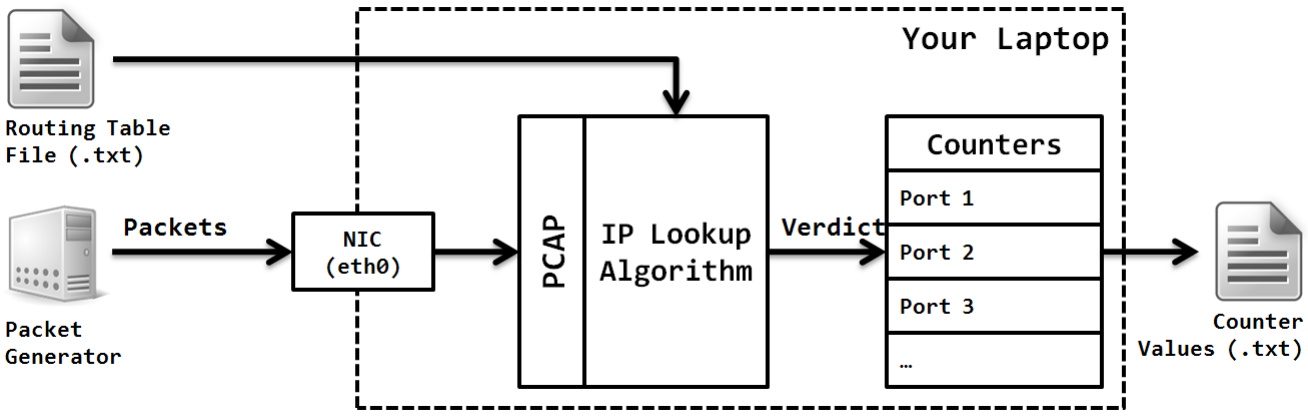
Before starting the lab, you are suggested to install a Linux-based system on your laptop. You can either have a native installation or a virtual machine, with the virtual network interface bridged to your laptop's physical interface (which is the default setup for most virtual machines).

You will download a reference code package. In the reference code package, some **routing table files** will be provided. It will be in the format of plain text. Each line contains two columns: an **IP prefix** expression and its associated **output port number**. Your program is required to read from the file, parse the file, and construct IP your lookup data structures accordingly.

After that, your program shall receive a stream of packets from your laptop's **Network Interface Card (NIC)**. In most of the Linux-based systems, the default interface is named `eth0`. We recommend using **PCAP library**, which is available by default in most Linux-based systems, to implement packet I/O functionalities.

Upon receiving an IP packet, your code shall extract its destination IP, and perform IP lookup. Please note that your lookup algorithms should follow the rule of **longest prefix matching**. Your program shall keep a counter for each possible output port number. For example, whenever there is a packet matched to Port 1, the port's counter will increment by one.

The **demo environment** is shown in the following figure:



Please note that the "ports" mentioned above merely refer to counters in your program. You don't need to forward the packets to another physical NIC. When you demo your codes, you are required to print the contents of the counters to a text file. If your code is working properly, the counter values should be correct.

(3) Testing Your Codes

For your convenience, you will be provided a simple routing table file and a packet trace, such that you can test your codes during development. You can use the offline mode of PCAP (i.e. read packets from a trace, not an NIC) to test your codes. Please note that we will use random packet generator and a different routing table during demo, so please do not try to hard code the rules in your program.

III. Deliverables

You are required to submit your source codes and a lab report after demo. Both your codes and reports should be submitted via **NYU Classes** system. No e-mail submission will be accepted. The formats of codes and reports will be detailed during lab hours this semester.
Please not that each group should bring at least one laptop to lab sessions.

IV. Notes

- Students caught with code plagiarism will face serious penalty.
- Only codes implemented with C, C++ or Python will be graded.
- We strongly recommend that you work on Linux-based systems, preferably Ubuntu distributions. Mac OS will do, but Windows-based systems are strongly discouraged.