UNIVERSITY OF CALIFORNIA, DAVIS

BERKELEY • DAVIS • IRVINE • LOS ANGELES • RIVERSIDE • SAN DIEGO • SAN FRANCISCO SANTA BARBARA • SANTA CRUZ

EEC173A/ECS152A Computer Networks **Winter 2020**

Project 1 (Due: Feb 2, 2020)

You should work in teams of *two* for this project. Each team only needs to submit one joint report via Canvas. Please remember to list the names and student numbers of both team members on your submission.

Part A (50 points). Wireshark Labs

Please refer to separate attachment:

a) (20 points) PI_Wireshark_IP.pdf
b) (30 points) P1_Wireshark_ICMP.pdf

Part B (50 points). BGP routing prefixes and routing table lookup

In Homework #1, you used *whois* command to find out which organization is assigned a particular AS number, and which IP address block is allocated to that organization (or AS). This lab shows an alternate method for mapping a particular IP address to an AS number that owns that address space by leveraging BGP routing tables.

In Chapter 4, you learned about Internet routing hierarchy, and how routers in the wide-area Internet figure out the forwarding path to a destination IP address by using the BGP protocol to exchange routing information. A typical BGP routing table at a router X may look like this:

```
Network          Next Hop        Metric  LocPrf  Weight  Path
…
*4.21.254.0/23   208.30.223.5    49      110     0       1239 1299 10355
*4.23.84.0/22    208.30.223.5    112     110     0       1239 6461 20171
…
```

Here is how you would interpret the first entry:
From router X's perspective, the shortest way to reach destination prefix 4.21.254.0/23 (i.e. all IP addresses that belong to this contiguous block) is to forward it along a path that will cross three networks (at the high level), identified by their AS numbers: 1239, 1299, and then 10355. Within each of these networks, there can be multiple routers that are involved in routing the packets, but that level of details (intra-domain routing) is masked and hidden from router X. The last network that you need to reach (in this case 10355) is typically known as the *home AS* that owns the destination prefix (in this case, 4.21.254.0/23).

If you run a traceroute experiment from router X to 4.21.254.0, you will get a router-level path between router X to 4.21.254.0, showing every single router that the ICMP packet traverses. If you run *whois* with the IP addresses of these routers as input, you will find that a group of routers belong to the first AS (1239), another group belongs to 1299, and the 3rd group belongs to 10355.

Given the rich connectivity of the Internet backbone, there can be multiple paths to reach the same destinations. Since Classless Inter Domain Routing (CIDR) allows prefixes of arbitrary length, overlapping prefixes can exist in the same routing table. Routers forward packets to the most specific forwarding information, called *longest prefix match*. In this part of the lab, you will write a program that (i) takes an IP address as an input, (ii) performs the longest prefix match on a compressed version of the routing table, and (iii) look up & output the *home AS* number for that IP address.

- First, download the compressed table:
  Canvas / File / Programming #1/ DB_091803.txt

  It contains three columns: IP prefix, prefix length, and AS number. For example, the third line of the file shows:
  ```
      3.218.160.0    20        13953
  ```
  which means prefix 3.218.160.0/20 belongs to AS number 13953.

- Download the **README** file
  Canvas ➔ File / Programming #1 / readme.txt

  The README file explains the requirements of a software tool, IP2AS, which would map IP addresses (given as input in a text file) to the home AS Number by performing longest prefix match using the compressed table DB_091803.txt.

- Your task is to write your own IP2AS software using your favorite programming/scripting language (C, C++, Java, Pearl, Python, etc). It should read in a list of IP addresses from a file, look up the database (DB_091803.txt) to map each IP address to an AS number, and print out the results to a file (or standard output).

  You can find an example input file at:
  Canvas ➔ File / Programming #1 / IPlist.txt

  If your tool works correctly, when you run
  ```
  ➢ ip2as DB_091803.txt IPlist.txt
  ```

  Your output should looks like:
  Canvas ➔ File / Programming #1 / output.txt