CS416 Project 2: Implementation of IP Forwarding

Learning objectives:

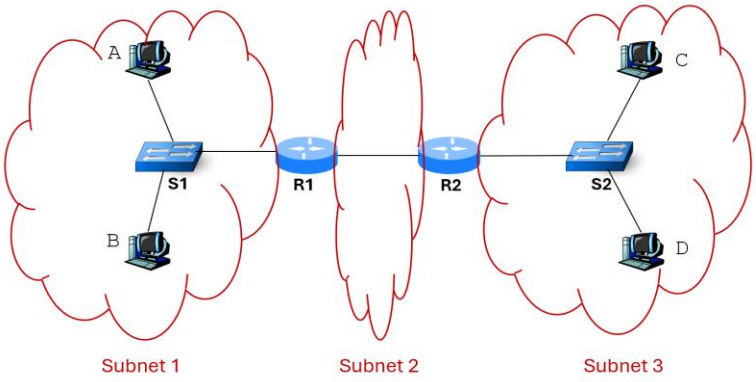
1. Implement the end-to-end IP forwarding process.
2. Construct an internetwork using routers to interconnect subnets.

What to submit:

1. The java source files.
2. The config file.
3. A note summarizing:
    a. Each member's attendance at group meetings, including in-class collaboration sessions and any additional meetings scheduled by the group.
    b. Each member's contribution to the project as agreed upon by all members.

(Each group only needs to submit one copy on Canvas. Please include a submission comment listing the names of all group members.)

Instructions:



1. Provide the config file that encodes the above internetwork topology. More specifically, the config file must specify the following information:
    a. For each device, specify its:
        - Real IP address (e.g., 10.222.56.42)
        - Real UDP port number (e.g., 3000)
        - ID (e.g., A, R1), which doubles as the virtual MAC address
    b. For routers and hosts only, specify their virtual IP addresses:
        - A host has a single virtual IP in the form of ***subnet.host***. For example, *net1.A*, *net3.D*
        - For a router, each virtual port has a separate virtual IP in the form of ***subnet.router***. For example, router R1 has two virtual IPs: *net1.R1* for its left port, and *net2.R1* for its right port.

- (A switch does not need any virtual IPs, as it operates purely at layer 2.)

c. For each host, specify the virtual IP of its gateway router.
   - The gateway router for A & B is *net1.R1*; for C & D, it is *net3.R2*

d. Specify how devices are connected, i.e., all the links.

2. When a device starts, it gets its ID (e.g., S1, A, R1) as a command-line argument; for example, this can be specified through the "run configuration" in IntelliJ.

3. Upon starting, a device parses the config file and finds out its neighbors. For example, router R1 has two neighbors: S1 and R2. It reads the real IP address and real port number of each neighbor from the config file.

4. A switch or router may have multiple virtual ports, and each virtual port can be "named" using the real IP address *and* the real port number of the *neighboring* device.
   a. As an example, R1's left virtual port may be named as: "*S1RealIP*:*S1RealPort*".

5. Modify the virtual host from Project 1.

   a. A host is always waiting for the user to specify a destination virtual IP and a short message. The host then generates a virtual frame that has five elements:

      i. Virtual source MAC address
      ii. Virtual destination MAC address
      iii. Virtual source IP address
      iv. Virtual destination IP address
      v. Short message

      As an example, if the user on host A specifies that the destination is "net3.D" and the short message is "hello!", host A should generate the following virtual frame:

      i. Virtual source MAC address: **A**
      ii. Virtual destination MAC address: **R1** (i.e., the virtual MAC address of the default gateway router)
      iii. Virtual source IP address: **net1.A**
      iv. Virtual destination IP address: **net3.D**
      v. Short message: **hello!**

   b. The host then creates a UDP packet to encapsulate the virtual frame and sends the UDP packet to the neighboring switch.

   c. In a separate thread, a host is always waiting to receive packets from the neighboring switch. When a packet is received, the host first checks to see if the frame's virtual destination MAC address matches its own virtual MAC address:
      i. If so, the host prints out the short message and the source host.
      ii. If not, the host prints out a debug message (it must be a flooded frame).

6. Provide the Java implementation of a virtual router.

   a. The router's primary task is to perform a lookup in its forwarding table and determine the outgoing port for each packet it receives.

b. The router table may be hard coded or read from the config file.

    i. In the next project, your router will run a routing protocol to automatically generate the table. Therefore, ensure your code is extensible, i.e., having a clear separation between obtaining this table and the rest of the router code.

c. The router table contains one entry for each subnet. For a directly connected subnet, the *virtual exit port* is specified. For a non-directly connected subnet, the *next-hop router's virtual IP* is specified.
For example, *R1*'s table should look like the following:

Subnet prefix	*Next-hop or Exit port*
net1	left port
net2	right port
net3	net2.R2

d. The router needs to re-write the source and destination MAC addresses of the virtual frames. To determine the new destination MAC address, the router should extract the ID from the next-hop virtual IP listed in its forwarding table. For example, extract 'R2' from 'net2.R2' and use R2 as the new destination MAC address before sending the virtual frame to R2.

e. Every time a router receives a virtual frame, it should print out all five elements of the frame; every time a router forwards a virtual frame out, it should print out all five elements of the frame as well.

Grading:

This project is worth a total of 100 points, distributed as follows:

- 75 points for implementation (detailed rubric will be included in the demo score sheet)
- 25 points for participation & contribution:
    - Attendance at all collaboration sessions, both in class and as scheduled by your group (10 points)
    - Attendance at the demo session. (5 points)
    - Actively contribute to the project in a meaningful way. (10 points)

Demo session: you will be required to give a live demonstration of your work, which will be the primary basis for grading the implementation. A demo score sheet will be provided separately.

Partial credit: If your implementation is only partially correct, you will receive partial credit proportional to the level of functionality achieved.