CSCI 558L – Laboratory Assignment #8: Custom Router
(Group Assignment)

Instructor: Young H. Cho T.A.: Hao Yu

Due date: October 1, 2016 at 11:55pm

In this assignment you will build a simple IP router in user space (meaning your own code in C, C++ or other programming languages that provide libpcap and/or raw socket interface). In order to build an IP router, you must be able to capture a packet sent to your router, determine the next hop and then send the packet to that next hop. Demo the final router on the due date during the TA's office hours. Please report the daily progress on-line.

To do each step:

- Capture packets: familiarize yourself with libpcap.
  Start here: http://www.tcpdump.org/#documentation. The third tutorial by Tim Carst is a good start.

- Determine the next hop: familiarize yourself with the theory and practice of IP routing. The last assignment should have been a good start.

- Send the packet to the next hop: familiarize yourself with the raw sockets feature provided by Linux (specifically a socket with PF PACKET as the type), or the packet injection feature of libpcap.

Note: For this lab, use the same network configuration you work with in the previous lab with minimum number of nodes.

- Start by instantiating the NS file as an experiment. Your code will eventually go in the node labeled 'rtr3'. However, for now set the default route in each of the nodeX machines to their respective router node (similar to the previous assignment). Then set routes in the three router nodes so that all nodes and routers can ping each other. Note: the routes to reach node3 and node4 should be collapsed into one CIDR route. What is this route in CIDR notation? Which routers did you use it on? When you include this route in your router it must be stored in collapsed form.

- Observe the routing table in the 'rtr3' node. This is the routing table you'll need to duplicate. Start by removing all of the routes that point to non-local IP addresses. Remember to leave the route to 192.168.253.1 or you'll lose connectivity to your node. Note which machines can no longer communicate.

- Use libpcap and raw sockets to build a router. Follow the following steps:

    - Build a routing table. It should contain the destination network, the next-hop IP, and the MAC address of the next-hop. Getting the MAC address can be tricky (google for SIOCGARP or parse /proc/net/arp). Remember you'll need to send at least one packet to the next-hop router to get its MAC address in the arp table.

    - Capture all packets sent to the router.

- If the packet is destined to a locally connected IP or a broadcast or multicast address, ignore it. The Linux kernel will have dealt with it already. { If the packet is destined to an IP address that matches a route in your table, do the following:

    - Set the source MAC address to the MAC address of the ethernet adapter on your router that will be sending the packet.
    - Set the destination MAC address to the MAC address of the next-hop.
    - Decrement the TTL of the packet. If you decrement the TTL to zero, you'll need to send and ICMP message back to the host (and drop the packet).
    - Use a raw socket to write the modified packet out the proper ethernet adapter.

- You will need to figure out how to stop the Linux kernel on the rtr3 node from sending an ICMP "Destination Unreachable" message to the source of a packet on your network. Research ICMP and use TCPDump to observe the 'rtr3' node generating these messages. Why are these generated? Why does this cause a problem? How can you block these messages? Answer these questions in your report. (Hint: you'll need to use a simple firewall rule)

- You must implement enough ICMP support to get traceroute to work properly. At a minimum, you'll need to implement ICMP message "Time Exceeded" (Type 11, Code 0).

- Test your router: once you get it working correctly your network should be fully connected again.

- Now collect the following stats for the competition aspect of this assignment: Packets per Second and Throughput

    - Calculate packets per second by using iperf to send the smallest possible UDP packet. Increase the sending rate until you start to drop packets. Use the output of iperf to calculate the maximum packets per second your router is capable of supporting.

    - Calculate throughput in a similar manner, except use the largest UDP packet as possible and use the output of iperf to calculate the throughput in megabits per second. For this assignment, use mega as 106. Answer the following question in your report: why are file transfer people interested in throughput with mega = 10242 while network people use 106?

- We used one CIDR route in this assignment. Why is CIDR important (i.e. what problem did it solve)?

You must demo a working IP router on September 26th demo date. This router must be able to route an IP packet and that is the minimum requirement for the demo and is meant as a mid-term progress for the lab. On October 3rd, your team must demonstrate your working router with all of the above functions including packet routing, performance tests, and correct ICMP responses with traceroute.

Extra credit: You have an opportunity to earn extra credit (up to 10% of the lab score) by integrating OSPF to your custom IP router with ICMP. You must have all of the above functions working before you can earn extra credit for OSPF.