

CSIS0234B Computer and Data Communication (Class B)

Tutorial 8

Setting up a NAT router

This week we focus on something that you can do at home if you have an Internet connection and what to share it among computers. While there are commercial products that can do the same task, it is usually more flexible to have a computer act as a NAT router, as you can configure it as a firewall which allows many variations to suit your network requirements.

Your tasks

Form groups among yourselves, each group with three or four students, and operate on four computers. Three of them (A, B and C) forms a private network, and the fourth (D) is outside the private network preparing to try tapping into that private network. Among the 3 computers of the private network, one of them (C) acts as a NAT router, and one of them (B) provide a service that is accessible through C.

Your task is to **write startup scripts** on C to set up NAT and protect the private network against D. Note that you should be writing startup scripts, i.e., a shell script that will *reset everything and configure the network packet filter from the beginning*. This is much less time consuming than having to find and modify the chains of the tables, because you can reset it any time you want by just running the script. Remember that you can write rules with LOG targets, and thus gain some idea about what is happening to packets that passes through the NAT router.

Part 1: Basic NAT setup

1. Setup the network parameters for A, B and C: configure them to use the IP addresses in 192.168.50.x (24 bit for network part) to communicate among themselves. For C, setup the extra interface to use 172.16.9.y to connect to the rest of the department, where y is the computer number of the computer. Test connectivity using ping.
2. Setup NAT at C. To do this, stop `ipchains` and start `iptables`, enable forwarding, and create a shell script that setup a MASQUERADE action in the `nat` table. The firewall will be suspended in the process—we will soon see how to enable the firewall again. Make sure the other computer can surf the web and can use ssh.
3. Use `ethereal` on both computers to determine whether a connection to outside have the source IP address, destination IP address, source port number and destination port number modified. (Hint: when capturing packets for the external interface of C, you can prevent most irrelevant packets by specifying `not multicast` as capture filter.)

Part 2: Basic firewall setup

1. The original Redhat default “medium-level” firewall is configured to drop any **incoming** packet to the UDP and TCP port ranges 0–1023 (privileged ports, most services run there), TCP ranges 6000-6100 (X11 ports) and 7100 (X11 font server). Add rules to your script at C to provide the same functionality, by modifying the `INPUT` chain of `filter`. Every dropped packet should be logged.
2. Now stop the `ipchains` services of A and B (which stops their firewalls). Try to use D to connect to the ssh port of B. This is supposed not to work, since those computers are in “private network”. Explain what stops it from connecting to B.

3. Without a proper fire wall, the above safety is not real. Try to modify the routing table of D to gain access to be able to connect to B via ssh.
4. Add more rules in C to prevent the above from happening. To do this, try to add rules in the `forward` chain of the `nat` table in the NAT router, similar to those you add to the `INPUT` chain. Remember that A and B still need to access the network outside, so be selective when dropping packets.

Part 3: Dynamic NAT setup

1. Suppose you want a service, say ssh, to be provided via the NAT router, but implemented by B. Setup dynamic NAT in C, so that all traffic to TCP port 22 (ssh) of the public interface will be forwarded to one of the computers in the internal computer. You will also need to relax the filter a little bit. Use the picture in the reading material to determine whether you should relax the `INPUT` or `FORWARD` chain of the filter table.
2. Try to use that ssh service from D. Make sure that it gets a shell of B, not of C.
3. Try to use that ssh service from A. Explain why it doesn't work by looking at the packets captured by `ethtereal`, and try to correct the situation by making some further modification to C.