Project Specification for the George Varghese Espresso Prize

To be eligible for the George Varghese Espresso Prize, your project should first satisfy the original *Simple Router* project specifications mentioned in the course site. While it is not necessary to receive a perfect score on the standard assignment, the basic functionality must be generally correct.

If you are interested in competing for the prize, you will need to implement one or both of the following two additional pieces of functionality:

1. Firewall
2. Network Address Translation

The prize will be awarded to the submission that, in the judgement of the instructor and the TAs, best implements the specifications below. In the case that more than one submission faithfully implements the initial aspects of both components, preference will be given to the project that earned the highest score on the base project. Should multiple projects receive perfect scores, the prize will go to the project that has implemented the most additional functionality in an elegant fashion. Multiple awards may be made at the sole discretion of the Instructor. The winner(s) will be announced at the final exam.

1. Firewall :

A firewall is a hardware or software security system that inspects packets flowing to and from the network device, checks compliance of each packet against a set of rules, and decides if the packet can be sent through to the destination. There are three levels in which a firewall can operate :

1. Packet Filter - Based on the contents of the packet headers.
2. Stateful inspection - Based on the state of a TCP connection.
3. Application Layer Filters - Based on the application layer information carried by each packet.

Out of the three mentioned above, you should start with the first type - Packet Filter.

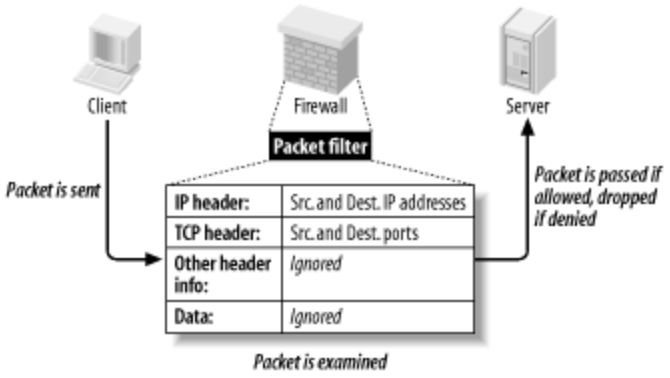

Image from http://eduunix.ccut.edu.cn

Assume that all the clients in your network topology belong to a student organization named *Triton*. Triton wants to safeguard its network from malicious requests from the internet. You, the owner of Triton's edge router, have negotiated with Triton and have come to an understanding that Triton's board will give you a set of filtering rules in the following format :

```
<action> <protocol> from <incoming IP block> [<optional incoming
port>] to <outgoing IP block> [<optional outgoing port>] [<optional
direction>]
```

Direction (can be "in" or "out") is specified with respect to the way the connection is first established. For example, an "inbound" flow is one whose first packet comes from <incoming IP block>.

Eg: `deny ip from 127.0.0.0/8 to any` - This means that all the IP packets from 127.0.0.0/8 directed to any IP address should be rejected by the router.

Triton's board has kindly accepted to give the firewall rules in the form of a file which your router should read to create its firewall rules. For example, the firewall spec file can look like the following (assuming Triton's network has a network address of 20.0.0.0/8).

```
deny tcp from 20.0.0.0/8 to any 22 out
```
// DROP outgoing SSH connections from Triton network to the public network.

```
allow tcp from any to 20.0.0.0/8 80 in
```
// ALLOW HTTP access from public network to Triton network.

```
deny tcp from any to 20.0.0.0/8 in
```
// DROP all other inbound TCP connections from public network to Triton network.

```
allow tcp from any to any
```
// ALLOW all other TCP connections.

You should apply these rules in order. In other words, if a packet matches more than one rule, you should apply the action specified by the first rule matched.

Your router should be able to change its firewall policies based on Triton board's specification no later than 1 second after the change was made to the firewall spec file.

Further information regarding firewall operation can be found in the following links :

- http://en.wikipedia.org/wiki/Firewall_(computing)
- http://www.freebsd.org/cgi/man.cgi?query=ipfw&sektion=8
- Chapter 8.5 in P&D.

You are welcome to implement additional firewall functionality; such as stateful transport or application-layer packet inspection. Please document any extensions required to the configuration language to support your features in a README.

2. Network Address Translation (NAT):

NAT is a way to spoof all traffic going in and out of a network as though they are originating from and destined to a single IP address. For example, most home networks are set up such that the cable/DSL modem has one IP address to the external world and connections from all devices on the home network appear to the external world as though they originate from a single IP address (that of the modem).

Task:

Some switches in your topology will be designated to be NAT switches. In each such switch, all but one of the ports will be connected directly to clients on the network. The last port will be dedicated to connect outside the network (uplink port). You are required to perform NAT operations when machines within the network communicate with machines outside the network.

For example, lets take the following list of clients on a network:
A - 172.168.12.15
B - 172.168.12.25
C - 172.168.12.32

Hosts A, B and C are all connected to switch S which in turn is connected to a router R via the interface which has a network address of 172.168.12.1 (called "private interface/network") and is the default gateway for A, B and C. R also has an uplink port (or multiple uplink ports) which is connected to other networks (called "public network" from here on) via an interface with an IP address 10.0.0.4.

When A makes a TCP request from port 10000 to an external address, say port 80 on 1.2.3.4:

- The packet is sent to R's interface with address 172.168.12.1.
- R determines that the packet is destined to the public network and the private IP address, port needs to be translated.
- R should create or reuse a translation entry in its Port Allocation Table (this is a 1 to 1 mapping between internal network IP, port number to public IP, port number). In this case, the translation entry can look like (172.168.12.15, 10000, 10.0.0.4, <external port>). External port number is randomly generated between number 1024 and 65535.
- So, a new packet is crafted with source address as the router's uplink IP address and source port as the translated port number.

When R receives a packet destined to its uplink IP address and port, p:

- It checks the Port Allocation Table to see if an translation entry exists for the port, p.
- If yes, update the destination IP address and port based on the matched translation entry and send it inside the network.
- If not, drop the packet.

Basic NAT functionality works only for flows that are originated from inside the NAT. Some protocols (such as FTP) subsequently attempt to establish flows from the remote server back to the initiating host. Unless the NAT has explicit support for these protocols, they will not work. You are welcome to extend your NAT to support FTP as well as any other protocols with similar behavior if you desire.

Further information regarding NAT can be found in the following links :

- http://en.wikipedia.org/wiki/Network_address_translation
- http://www.ipprimer.com/nat.cfm
- Network Address Translation on P&D p-335.