CS444/544
Programming Project 4 - vos
Due: Friday, May 25 (electronically before class)

This assignment is intended to tie together all of the concepts that we've discussed so far. You're going to again use the vrouter library to build a small routing operating system called "*vos*". Your *vos* operating system can be used for either a router (a machine with multiple interfaces) or a host (with a single interface).
On either type of machine, it must:

1. Respond to ICMPv6 echo requests
2. Respond to Neighbor Discovery requests
3. Send Neighbor Discovery requests to map IPv6 addresses into Ethernet addresses
4. Contain an ARP cache with a configurable timeout (see below)
5. Maintain an IPv6 routing table that it will use when sending IP packets
6. Receive RIPng announcements (see RFC 2080) and use them to populate its routing table
7. Understand and demultiplex the UDP protocol
8. Provide a server on the UDP echo port

On machines with multiple interfaces, it must also:

1. Multicast Router Advertisements so that your non-routers can determine their IPv6 addresses
2. Multicast RIPng routing packets
3. Route packets between its various interfaces using its routing table
4. Send ICMP time exceeded messages in response to hop count expirations (so that traceroute will work)

The intention is that we'll tie your machines into the real network. That means that you'll be able to use the following commands on real computers to test your implementation:

1. ping
2. udpping
3. traceroute

You should use the same library routines that you used for the first project, with the addition of

```
int readipprefixfor(INTERFACE *iface, void *prefix_addr,
                    unsigned int *prefix_length);
```

This routines will obtain the IPv6 prefix and prefix length to use for the machine interface given from "stable storage" for the given machine interface. This routine will only work for computers with multiple interfaces (your "routers"). Your vos machines with a single interfaces (your "hosts") will need to use your RA multicast messages to determine their addresses.

We'll also be modifying the existing `openinterface()` routine so that it takes the interface name **"DSL"**. Packets written to this interface will go to the real Internet. Packets read from this interface will be from the real Internet. Although this interface has a different name, it's a regular ethernet interface like the others and you should treat it the same. You will receive occasional RIPng advertisements from this interface, and to keep it alive, you **must** periodically advertise your local nets out this interface with RIPng as well.

Your vos program must support the following options:

-d
: To print debugging information (more detailed output). In particular, this option must do the following:

    1. When specified at least once, all changes to the routing table must cause the entire table to be printed
    2. When specified at least once, all changes to the ARP table must cause the entire table to be printed

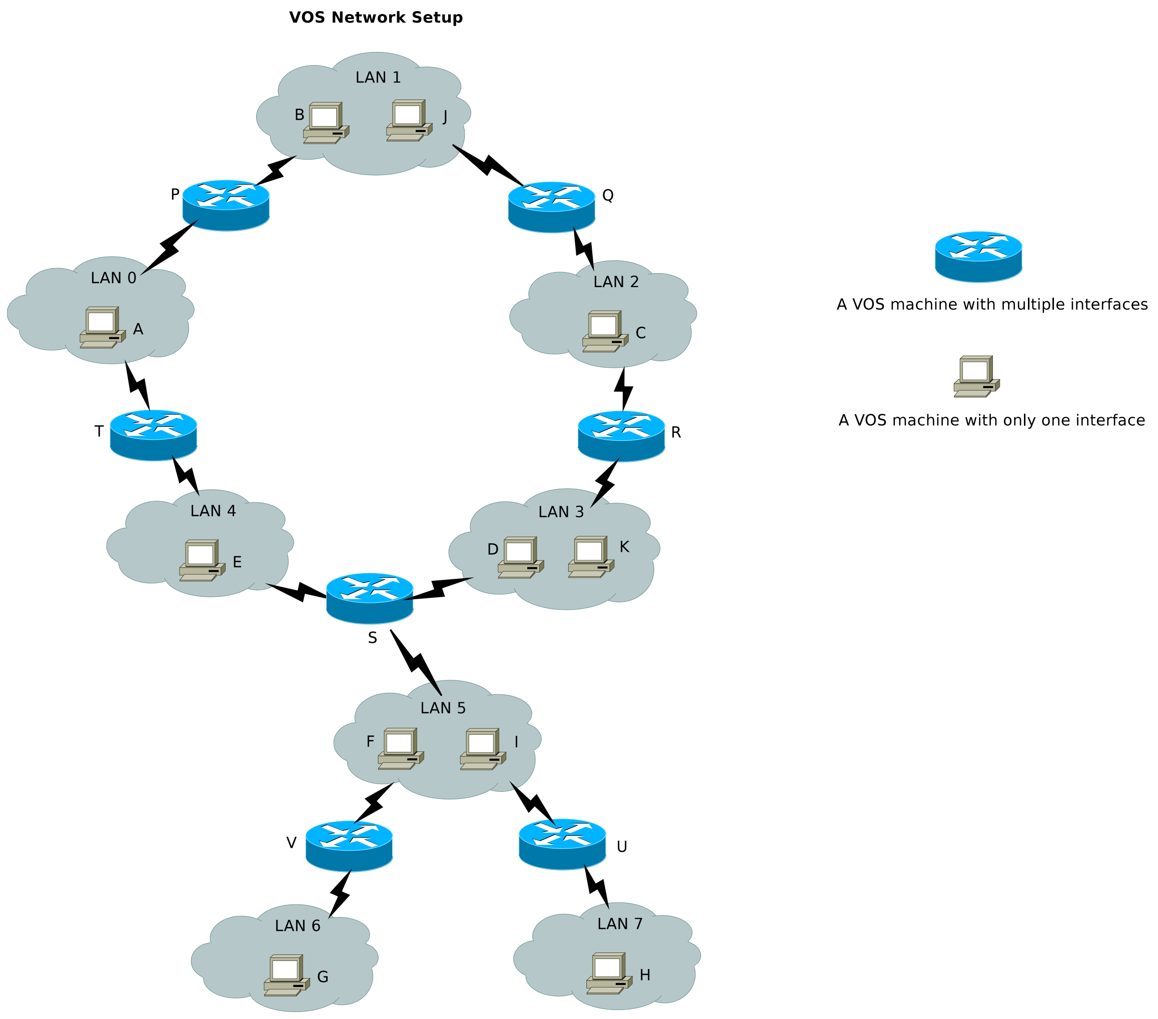
-l
: List all interfaces (names and hardware addresses) and exit

-g NUM
: Use group NUM. By default, you should use the group number that you were assigned.

-D
: Attach this machine to the DSL interface

-a SECONDS
: Specify the number of seconds that an entry in the ARP table is allowed to exist. An entry should be assigned a timestamp (see routine `time()`) when it is ENTERED into the table. After SECONDS (which should default to 15), the entry should not be used anymore (and Neighbor Discovery should be used the next time it is needed).

-h Print a quick summary of the command line arguments and exit. This information should also be printed if you don't understand one of the command line arguments.


VOS Network Setup
LAN 1
B
J
P
Q
LAN 0
A
LAN 2
C
T
R
LAN 4
E
LAN 3
D
K
S
LAN 5
F
I
V
U
LAN 6
G
LAN 7
H
A VOS machine with multiple interfaces
A VOS machine with only one interface

The following table shows the mapping of machines in the "bin lab" in stocker 107.

Diagram Host	Binlab Host
router P	bin01001
router Q	bin01010
router R	bin01011
router S	bin01100
router T	bin01101
router U	bin01110
router V	bin01111
host A	bin00001
host B	bin00010
host C	bin00011
host D	bin00100
host E	bin00101
host F	bin00110
host G	bin00111
host H	bin01000
host I	bin10000
host J	bin10010
host K	bin10011