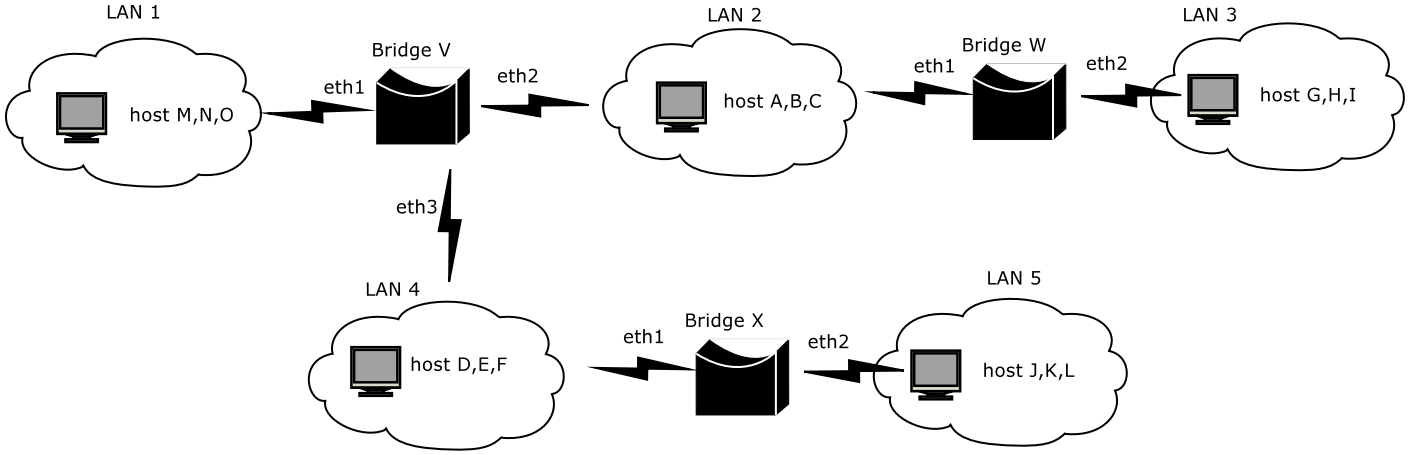
CS444/544
Programming Project 2 - vbridge
Due: Tuesday, Feb 9

For your next network programming project, you're going to modify your vdump programs to perform smart bridging on a simple network that looks like this:



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

```
int writepacket(INTERFACE *iface, char *buf, int len);
```

This routine writes one packet to the interface specified from the buffer `buf` (of size `len`). It returns the number of bytes written or -1 if it fails.

The following table shows the mapping of machines in the "bin lab" in stocker 307.

Diagram Host	Binlab Host
host A	odd07
host B	odd09
host C	odd11
host D	odd13
host E	odd15
host F	odd17
host G	odd19
host H	odd21
host I	odd23
host J	odd25
host K	odd27
host L	odd29
host M	odd31
host N	odd33
host O	odd35
Bridge V	odd01
Bridge W	odd03
Bridge X	odd05

All of the machines that are NOT bridges will have a single interface defined (eth1) on the lan shown in the figure. The 3 bridges will use 2 or 3 interfaces, as shown. Your bridge should act as a "learning bridge", as we discussed in class, and should use the source address of Ethernet frames to create a cache of the locations of the machines on the network. Those cache entries should time out (some number of seconds AFTER they were first entered) as discussed below. When asked to forward a frame for which no location is cached, your bridge should send the frame out over every interface other than the one on which it arrived. You should also support broadcast and multicast forwarding.

Your program should only be run on the 3 bridge machines (V,W,X), and it should bridge ALL of the interfaces that it finds on the machine. Your vbridge program must support the following options:

-d
To print debugging information (more detailed output)

-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.

-t SECONDS
Specify the number of seconds that an ETHERNET address is allowed to be used in the address-to-interface binding table. 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.

-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.

You will test your program, vbridge, by using your existing vdump program to print packets and a second testing program that you need to write called vsend that is called as follows:

```
./vsend [−d] [−g NUM] [−i INTERFACE] [−t MSECS] [−c COUNT] aa:bb:cc:dd:00:11
```

Vsend will send frames to the ethernet address given. By default, it should send a packet every 1000 milliseconds (see the usleep() routine), or with the optional interval given. If should send packets infinitely (until killed), or COUNT packets if specified.
To test your network, you will run your `vbridge` program on the 3 bridges, your `vsend` program on the machines on the LANs to send packets and your `vdump` program on the machines on the LANs to watch and see where they go.