CS 485/ECE 440/CS 585 Fall 2015 Lab 2

Due 11:59pm on Wednesday, 21 October 2015

START EARLY, you don't have much time and you'll need to work with a partner so get started on this lab ASAP. We have to move quicker on the labs than we have been if we're going to get to the cool stuff.

Please send your submission for Lab 2 as a gzipped tar ball attachment to an email to "crandall@cs.unm.edu". The subject of the email should contain both the strings "lab2" and "nets". Do not submit your lab writeup to any other address, attach any files that are not the one single tar ball, or put any part of your lab submission in the text body of the email.

You will need exactly one partner for Lab 2. You can choose any partner in the class that you wish, but only one. You will submit your assignments individually and be graded individually, but your pcaps won't be very interesting without you exchanging traffic with your partner (which is required) so choose a good partner and work together to make sure you both get the lab done. Again, START EARLY. The first step you should take is to exchange email addresses and (optionally) other contact information. "I can't get in touch with my partner" won't be an acceptable excuse for not getting Lab 2 done on time.

Before you begin on Lab 2, you must be using the correct ports and IP addresses. So if your get back your graded Lab 1 (hopefully soon) and you didn't get the full points for conforming to the port and IP address usage standard of the class you need to fix that before you start on Lab 2 or it will affect your grade again.

Refer to Lab-2-Setup.pdf for instructions on how to connect your virtual network to your partner's. You'll essentially be dragging a virtual ethernet cable between two new NICs, one in each of your routers, and then setting up dynamic routing so your two networks know how to route packets to each other. The example is Lab-2-Setup.pdf is Raj connecting to Ben. Be sure to break the remaining /25 of IP addresses you have into /30s, because you'll need more subnets to connect to more classmates in future labs.

We'll define a protocol in a mock RFC during class on Wednesday the 14th. You and your partner each need to implement both a client and a server version of the protocol. Feel free to borrow example socket code from the Internet in your chosen language and modify it, but your two programs (the server and the client) should be your own code written by you alone, with only high-level help from your partner or other classmates.

You and your partner must implement your client and server in significantly different languages. For example, you can write your client and server in C and your partner can write their client and server in Python. C and C++ are not significantly different.

The tar ball should contain a single directory, with all files within that single directory. The tar ball

should have your family name, student number, and the string “lab2.” For example, I may or may not post an example tar ball but mine would be named crandall-9-lab2.tar.gz. Your lab 2 submission should contain:

Three text files showing the configuration of each of your VMs:

crandall-9-lab2-showvminfo-moe.txt

crandall-9-lab2-showvminfo-larry.txt

crandall-9-lab2-showvminfo-curly.txt

(produced via “vboxmanage showvminfo moe > crandall-9-showvminfo-moe.txt” and likewise for larry and curly)

One packet capture (how to produce it is described below):

crandall-9-moe-lab2-eth3.pcap

Two text files with the source code for both your server and your client:

crandall-9-lab2-server.[ext]

crandall-9-lab2-client.[ext]

(...where [ext] can be “py”, “c”, “cc”, or whatever is appropriate for the language you chose).

I'll be inferring who your partner is based on IP address and port numbers, so make sure your partner is conforming with the class standards about port number and IP address usage or it may affect YOUR grade.

Lab 2 is worth 100 points, based on the following rubric:

- 10 points for following instructions for how to submit the lab as a tar ball (this will make grading easier, so please pay attention to even the smallest details to make your tar ball look exactly like above).
- 10 points for how well your VM info for your three VMs matches Ben and Raj's. Obviously the names of the VMs, the port numbers, usernames, etc. will be different but modulo that every line in the “vboxmanage showvminfo ...” output files should be identical to the reference output.
- 80 points for having a functioning server and client and a packet capture the demonstrates you and your partner exchanging traffic in a way that conforms with the protocol laid out in the mock RFC.

Your packet capture should be produced on your own network, if I find evidence that the same virtual network and VMs was used to produce packet captures for the submissions of two or more students then all students involved will be considered to have cheated and receive an F in the class (even the one who built the network).

You are expected to do your own work. From modifying your virtual environment to producing the source code for your client and server to producing the outputs, for all phases of this project you should

do your own work. Any instance of not doing your own work will be considered cheating. If you're not sure whether something will be considered cheating or not, ask me before you do it. You are encouraged to discuss the assignment with your classmates at a high level and work closely with your partner on the lab. Exchanging details about specific configuration issues or solutions to specific setup problems is okay. Writing the source code for your partner or allowing them to write yours is NOT okay. As a reminder of the course policy, if you cheat on any assignment in this class including this assignment (cheating includes, but is not limited to, representing somebody else's work as your own or fabricating files to make it look like you completed the assignment) you will receive an F in the class.

To produce the packet capture, you and your partner should be capturing packets at the same time so I can match the packets in your respective captures. If they don't match, that will affect your grade. You should capture packets on your eth3 of your router (moe in my case), but run the server and client on one of your endhosts (larry or curly in my case). You should start by tracerouting to each other from one of your endhosts to one of your partner's endhosts. Then you should have your client connect to your partner's server several times and request different amounts of data.

Your server MUST be able to handle multiple connections at the same time and keep running. The easiest way to implement this is to have a parent process with a listening socket that forks child processes to handle connections.