ECS 189M
SQ 2017
Programming Assignment #1
Due: Tuesday, May 9, 2017, 10 AM, Via SmartSite

My plan is for you to demonstrate your solutions to me and the TA during a 15 minute slot we will assign to you, most likely on May 9.

You may work alone or with a partner.

There are some additional details on the problems I willl provide, likely in a day or so.

Problem 1:

Refer to the *first* security notes, Pg. 717, problem 46. I assume you are familiar with shell scripts and/or C programs that call *Unix System calls;* if not, let me try to devote a discussion section (not this week) to this topic.

The idea of this problem is to have a program (or a process) send a message of bits to another process via a covert channel, where the channel involves the permission bits of a file. Even though the two processes do not share the file, one process can change the permission bits of the file and the other can observe the permission bits. You will want to interleave the actions of the two processes so the first (sender) process goes first; it sends a bit by changing (or not) the permission bits of a file. Then it goes to sleep. The receiving process goes second and checks the permission bits of the same file and prints the bit it believes it has been sent. Then the first process goes again and so on. Assume the first process wants to send a bit string of length 5 it reads in from the terminal.

Play with the "sleep" times to see if you can make the channel unreliable or totally reliable.

Problem 2: Experiments with computer viruses in Python;

Please refer for the following web pages:

https://cranklin.wordpress.com/2012/05/10/how-to-make-a-simple-computer-virus-with-python/

https://codingsec.net/2016/11/make-simple-computer-virus-python/

First of all, be careful when you run these viruses as you could easily damage Python source files.

Part 1: Create several (say 3) Python source files that contain mostly simple Python programs and place these files in your home directory.   Run the Python virus program to infect the several dummy Python source files you have created. Compile and run these dummy files to exhibit the infection and to cause other dummy files to become infected.

Part 2: Write a simple *(anti-virus AV)* program that will check a Python source program for the presence of this virus.

Part 3: Make your virus polymorphic and, consequently, more difficult for your AV program to detect. One approach is to have the *marker* change each time the virus infects another file, where the change is effected by an "encryption key" hidden in the virus.

Part 4: Stay tuned.