CS 432 “Smashing the Stack”

Setting up the VM

1) To work on your laptop (Windows, Linux, Intel Mac), download and install VirtualBox (free from Oracle). On an M1/M2 Mac, you should download and install UTM. On the lab machines, you should use VirtualBox. Type “virtualbox” in the terminal to start the player. It is already installed.

2) Download the VM image from the course website. Note you must be on campus (or using the proxy) to access it. **PLEASE STORE THE VM IMAGE IN /home/scratch/{your unix ID here}/ ON THE LAB MACHINES RATHER THAN IN YOUR HOME DIRECTORY!** The easiest way to do this is to:

```
$ cd /home/scratch/jeannie (insert your unix id instead of jeannie)
$ wget http://dept.cs.williams.edu/~jeannie/cs432/assignments/downloads/assignment3.ova
```

3) Import the VM archive into VirtualBox/UTM and run the VM. You should basically just pick the default option for all windows and options that pop up during import.

4) Log in to the virtual machine. There are two accounts, root with the password root, and user with the password user. Log in as “user” for now.

5) I already copied the project 3 tarball from the course website (proj3.tar.gz) onto the virtual machine. Verify that it is there and has been installed in the user’s home directory.

6) Use the Makefile to build the targets in ~/proj3/targets/ and install them in /tmp. (cd ~/proj3/targets; make; make install). Note: You’ll want to run make install as root (type su to become root).

```
$ cd ~/proj3/targets
$ make
$ su (password is root)
# make install
# exit
```

7) Warning: Every time you reboot your VM, you'll have to set up the targets in the VM's /tmp directory again because it'll have been wiped clean. Just rerun the make install command (as root).

Smashing the Stack Starter Guide

We are going to go over sploit1 and target1 in class. That should provide enough background for you to get started on the project. However, if you'd like more background info, here are some suggestions.

1) Start with the Aleph One text (http://insecure.org/stf/smashstack.html). If you haven't already done so, read it! Make sure you understand the general framework.

2) I recommend trying to replicate example1. Generate assembly code. It probably looks different than Aleph's. Why do you think this is? Note that the "important" parts haven't changed, however.

3) Continue reading Aleph's text. Read example2, but you don't have to actually do that one (unless you want to). Now look at example3. If you're feeling ambitious, see if you can use gdb to replicate the behavior that Aleph describes.

4) Read through the discussion about generating shellcode. Play with gdb to see how the x/bx command can be used to generate your own shellcode. You might want to experiment with shellcodeasm.c. You will have to change the syntax to make it compile. Try getting testsc.c to work on your system. Ultimately make sure you have a file containing shellcode without null characters that works on your VM. Why do you need null-free shellcode?

5) Carefully read through the discussion relating to buffer overflow exploits. Try exploit4 on target1. Does it work? If so, try to understand exactly what's going on. Why does it work using environment variables?

6) Can you rewrite exploit4 without using environment variables and get_esp()? Hint: You can instrument the target to help you find your address. Don't blindly copy code. Trial and error brute force programming is tedious. Understand what is happening in exploit4 and try to reproduce it. You should not use get_esp() or environment variables.