**CSE 484 In-section Worksheet #2**

Q1. Draw the stack at the beginning of function foo.

*int foo(char \*argv[]){*

*char buf[320];*

*strcpy(buf, argv[1]);*

*}*

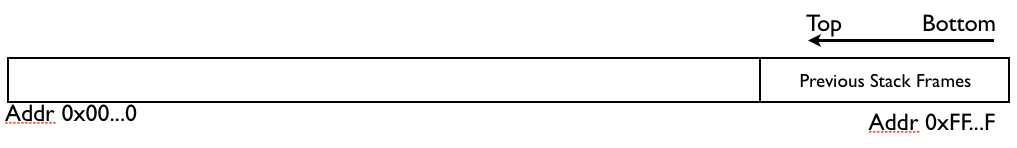
*int main(int argc, char \*argv[]){*

*if (argc != 2) {*

*fprintf(stderr, "target1: argc != 2\n");*

*exit(EXIT\_FAILURE);*

*}*

 *foo(argv);*

*return 0;*

*}*

Q2. How is this program vulnerable? How can we structure buf to exploit this vulnerability? Fill in the program below.

*#define TARGET "/bin/target0"*

*#define EIP \_\_\_\_\_ // How do we find this out? What does it represent?*

*#define BUFLEN \_\_\_\_\_*

*int main(void){*

*char \*args[3];*

*char \*env[1];*

*char buf[\_\_\_\_\_]; // What size should this be?*

*memset(\_\_\_\_\_\_\_\_,\_\_\_\_\_\_\_\_,\_\_\_\_\_\_\_\_);*

*// How does strcpy know when to stop?*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*memcpy(\_\_\_\_\_\_\_\_,\_\_\_\_\_\_\_\_,\_\_\_\_\_\_\_\_);*

*// Where do we put EIP?*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*args[0] = TARGET;*

*args[1] = buf;*

*args[2] = NULL;*

*env[0] = NULL;*

*if (0 > execve(TARGET, args, env))*

*perror("execve failed");*

*return 0;*

*}*

Q3. What gdb (or cgdb) statement and sequence of commands will let us step through these programs if the files are ~/sources/target0.c and ~/sploits/sploit0.c while the executables are /bin/target0 and ~/sources/sploit0?