CS 458 A1 Milestone
Patrick Wong
20317267

sploit1

sploit1 exploits a character buffer in submit's copy_file command. copy_file creates a character buffer of size 1024 and copies the contents of the submitted file into the buffer. From the memory architecture of a C program in the ugster05 environment, the memory address higher than that of the buffer is the stack frame pointer and return address, contained in four bytes of memory respectively. sploit1 creates a file containing a large string of length 1032 to exploit the character buffer, by overwriting the return address with the first address that contains the shellcode.

The first character in the long string is a null character; this is to bypass submit's virus checker, check_for_viruses. The initial condition in check_for_viruses's while loop checks if it is reading an EOF character. The long string is prefixed with a null character and check_for_viruses exits without printing a warning message. The second part of the long string contains only "A"'s, of length 978 (1032 - 45 (length of shellcode) - 1 (null character) - 8 (sfp, ret)). The third part of the string is the shellcode. The second last part contains four "A"'s, used only for debugging when inspecting the program's memory in gdb. "A" is represented as "41" in hexadecimal, making it easy to notice when debugging. The last part of the string is the contains the first address where the shellcode is located.

sploit2

sploit2 exploits a character buffer in the print_usage command. print_usage creates a character array of size 172. To overflow the buffer, a string of length 173 is needed. The submit command accepts three arguments. The first argument is sent to print_usage. The process of exploiting this is the same as sploit1. A long string containing "A" is created, concatenated with the shellcode and the address where the shellcode starts.