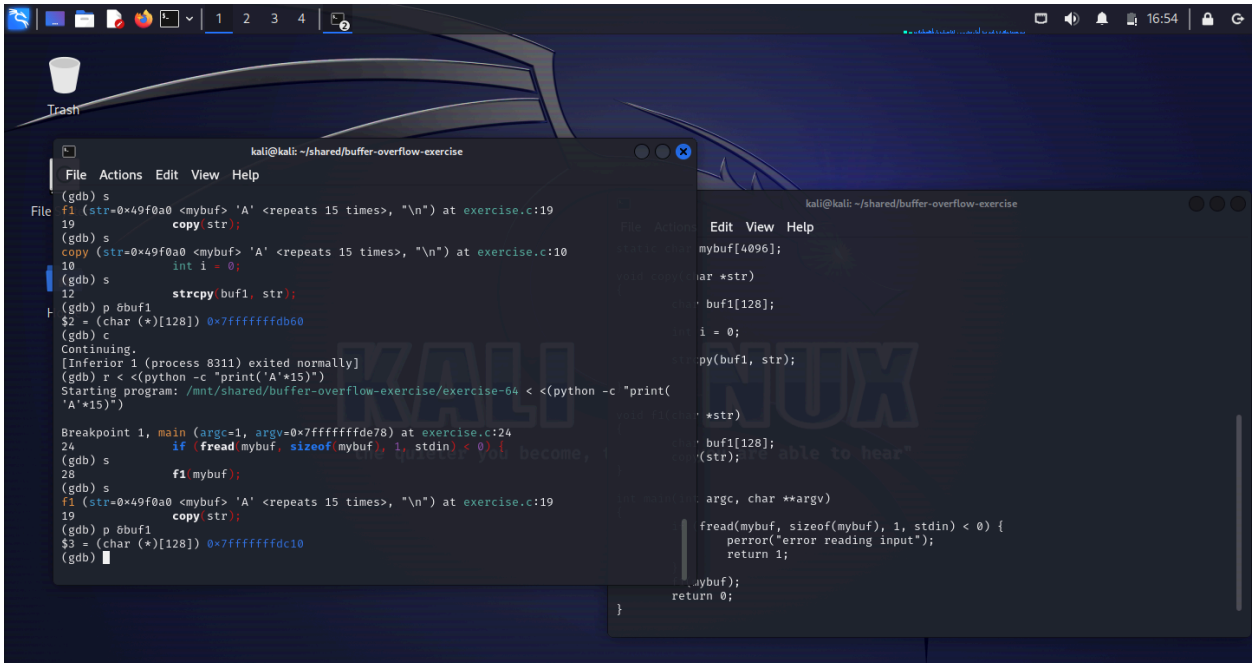
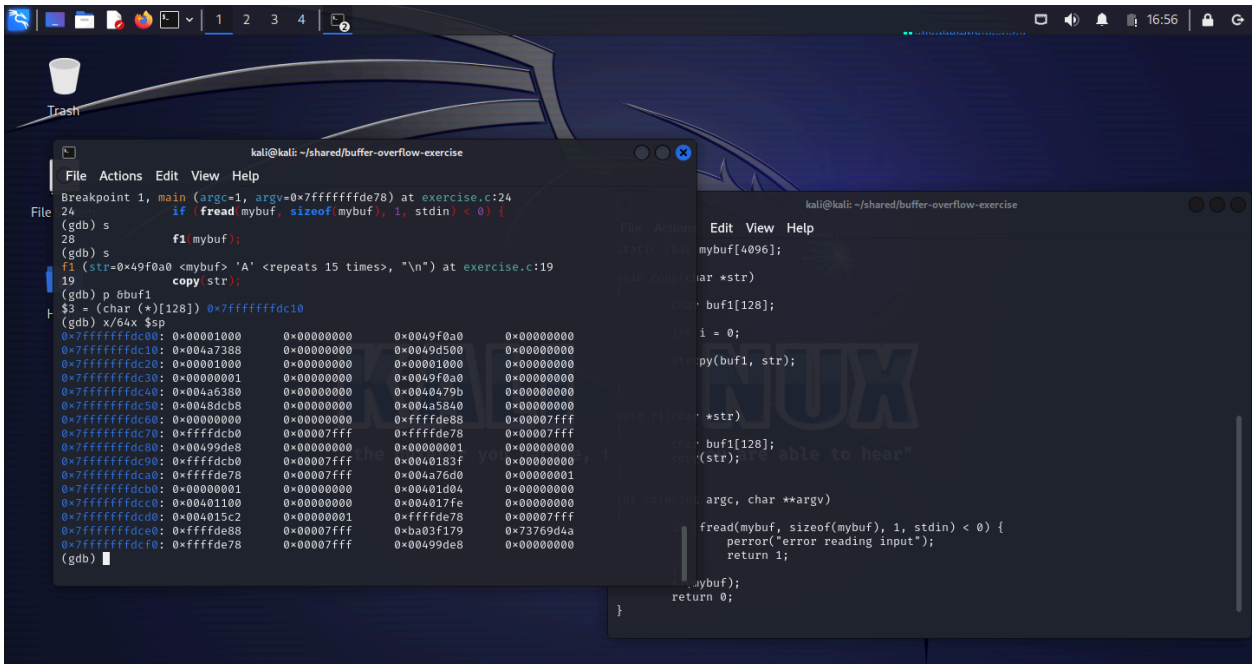
Maximiliano Brizzio
CS-576
Lab 6 - Buffer Overflow

Task 1:

To find the buffer start address, there a few things we need to set up first. I started by running `gdb exercise-64`. Once in gdb, I set a breakpoint at main and run the program using the command `r < <(python -c "print('A'*15)")`. Then I take steps of the program until I enter the f1 function. Once inside of the f1 function, I print out the address of the buf1 variable using the command `p &buf1`. The base address is 0x7fffffffdc10.

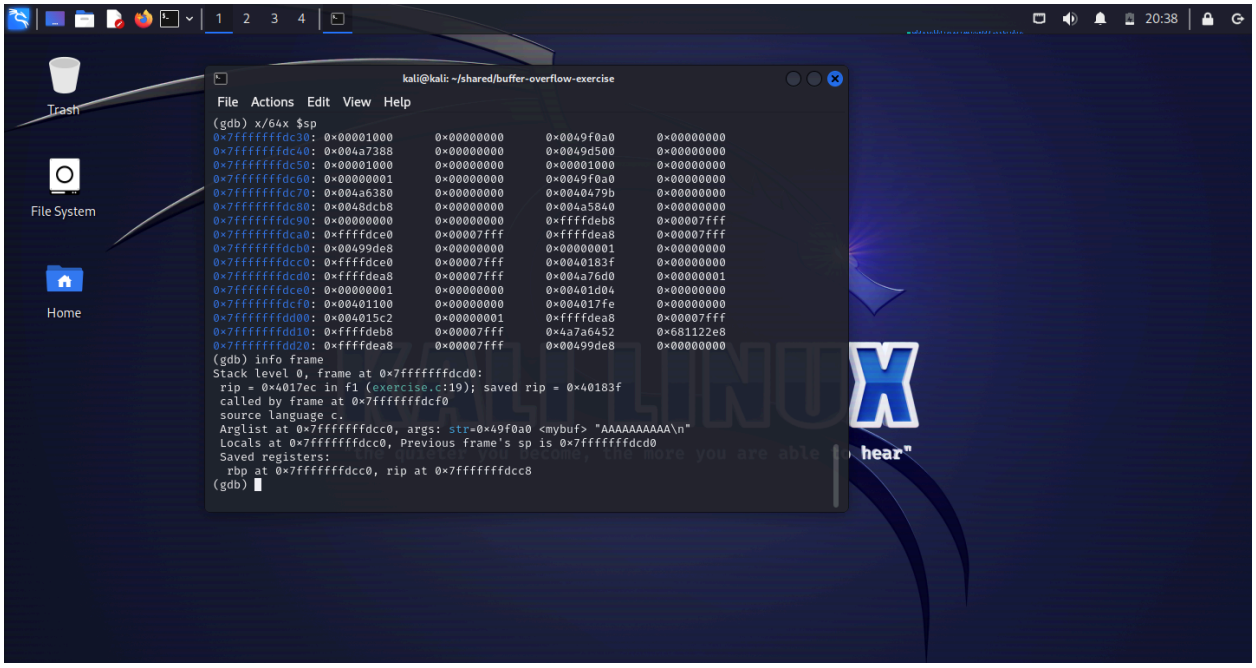


Next, we can print out the stack and check where buf1 is stored:



Task 2:

Because we know the size of buf1 (128 bytes) and it's start address in the stack, we can calculate the bounds of it. But if we want to use buf1 to overwrite the return address of f1, we have to find the return address first. To do this, I run the command `info frame` which under the 'saved rip' value stores the return address of f1. The address is 0x40183f.



Now using the return address, we can cross reference the stack to look for the same hexadecimal string (0x40183f). It appears that from stack address 0x7fffffffdcca-7fffffffdccc (inclusive) is where the return address of f1 is kept.

We know buf1 starts at 0x7fffffffdc10, so to overwrite the return address of f1, we need to make buf1 at least 188 bytes long (0x7fffffffdc10 + BC = 0x7fffffffdccc and BC = 188). Because the size of buf1 is designed to be 128 bytes, we need to overwrite by 188-128 = 60 bytes.