#define \_GNU\_SOURCE /\* See feature\_test\_macros(7) \*/

#include <unistd.h>

#include <sys/syscall.h> /\* For Sys\_xxx definitions \*/

/\* headers came from: https://manpages.debian.org/stretch/manpages-dev/syscall.2.en.html \*/

#include <stdio.h> // adding this due to printf error

//using code model from recitation

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

long callHello;

callHello = syscall (333);

printf("System call returned # %ld. \n", callHello);

return callHello;

}

* arch/x86/entry/syscalls/ syscall\_64.tbl
  + 334 64 cs3753\_add sys\_cs3753\_add

#include <linux/kernel.h>

#include <linux/linkage.h>

#include <asm/uaccess.h> //(copy\_to\_user header)

* Arch/x86/kernel cs3753\_add.c

asmlinkage long sys\_cs3753\_add (int a, int b, int\* place)

{

int answer = a + b;

printk(KERN\_ALERT "test...enterest asmlinkage long sys\_cs3753\_add");

if(answer < 32768 && answer > -32768){

copy\_to\_user(place, &answer, sizeof(int));

printk(KERN\_ALERT "Number 1 (a) + Number 2 (b) = %d.\n", answer);

}

else

printk(KERN\_ALERT "Function call executed, but numbers were outside max-range.");

return 0;

}

* Include/linux
  + Asmlinkage long sys\_cs3753\_add(int a, int b, int\*place);

#include <stdio.h>

#include <unistd.h>

#include <sys/syscall.h>

#include <stdlib.h> // need this for malloc

int main()

{

int a, b;

long callAdd;

printf("Enter the first number value:");

scanf("%d", &a);

printf("Enter the second value:");

scanf("%d", &b);

int \*location = malloc(sizeof(int));

callAdd = syscall(334, a, b, location);

printf("System call returned # %ld \n", callAdd);

return callAdd;

}