Buffer Overflow assignment

# 1.Program Exploit.c

#include "stdio.h"

#include "time.h"

#include <unistd.h>

// Prints the date and time

void shashadhar() {

time\_t t;

time(&t);

printf("Shashadhar Das\n");

printf("M Tech CSE - First year\n");

printf("Date and Time: %s", ctime(&t));

}

//Main function

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

{

// calling the testme program

execlp("./testme", "testme",argv[1], (char \*)NULL);

return( 0 );

}

# 2. Program Testme.c

#include <stdio.h>

#include <string.h>

#include<time.h>

// Explotable function

int exploitable( char \*arg ) {

// Make some stack space

char buffer[10];

// Now copy the buffer

strcpy( buffer, arg );

printf( "The buffer says .. [%s/%p].\n", buffer, &buffer );

// Return everything fun

return( 0 );

}

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

{

// Make some stack information

char a[100], b[100], c[100], d[100];

// Call the exploitable function

exploitable( argv[1] );

// Return everything is OK

return( 0 );

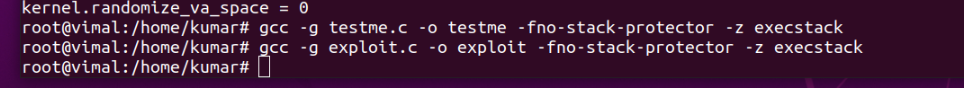
}

# Steps to attack buffer overflow

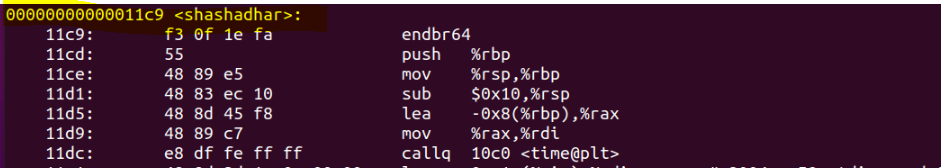
1. Disable address space randomization



1. Run the exploit.c program and get the address of the function Shashadhar() by giving command “objdump -d exploit”

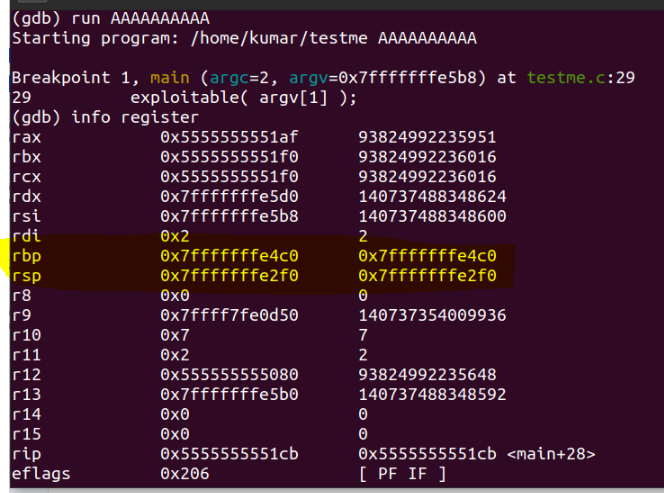


Address of the function is = 00000000000011c9 , we have to override this address to the return address



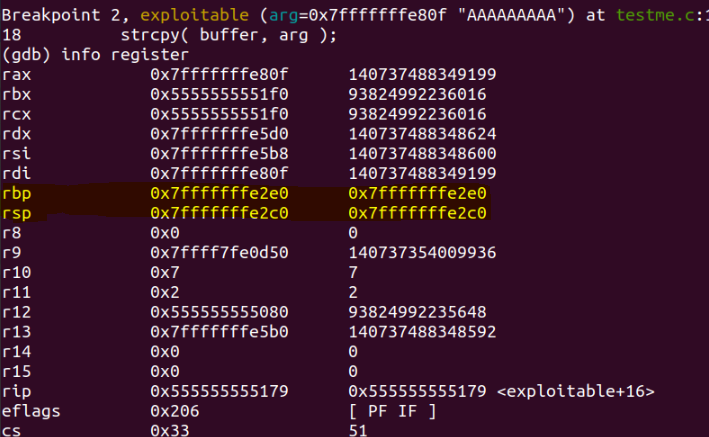
1. Get the address of ebp and esp for the exploitable function

To get the address we run the program using gdb and added breakpoints in main and exploitable function

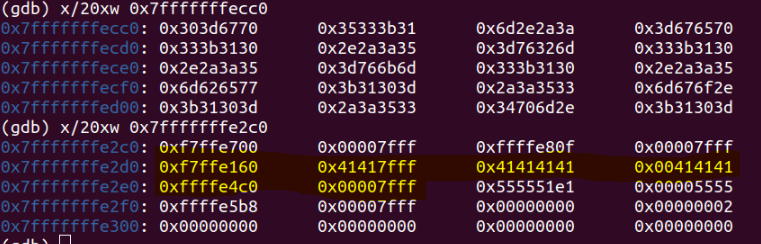


I can see the ebp of main is 7fffffffe4c0, that is the address which should be store in the stack of expoitable function.

1. Proceed to expoitable function and get the ebp and stack esp address



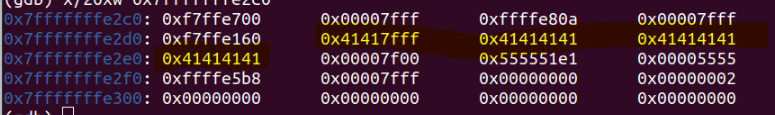
We can see that the esp and ebp. Lets see the contents of rsp .



I can clearly see the char “A” – hex value 41 is stored and ebp is storing the ebp of main function(7fffffffe4c0)

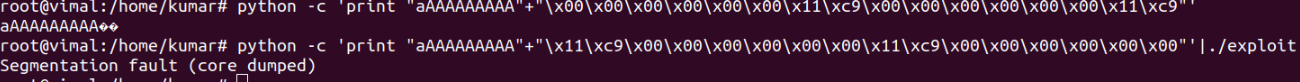
My task is to overwrite that ebp and and next 64 bit with the return address of “Shashadhar”

1. I run again with the more no of “AAAAAAAAAAAAAA”s to check if it is overwriting or not



It is clearly visible that ebp address is getting overwritten after 10s .

1. Now we need to provide the input with the address of Shashadhar function



And I can see the overflow happened

