**Ashesh Kumar**

**501254**

**I.T. – 6**

**Emulate Buffer Overflow Attack**

**BufferOverflow.c**

#include <stdio.h>

#include <string.h>

#define MAX 9

int main()

{

//good for storing upto eight characters excluding '\0', NULL character

char buffer[MAX];

int i;

//prints the allocated memory location for each array element

for(i=0;i<MAX;i++) {

printf("Memory location of buffer[%d] = %p; \n",i,&buffer[i]);

}

//assigns memory location for flag just next to last array element, buffer[MAX-1]

char\* flag=(char\*)&buffer[i];

printf("\n Memory location of flag = %p;",flag);

//set the flag

\*flag='0';

printf("\n flag = '%c'",\*flag);

printf("\n\n");

//store the input string into the array

printf("Enter the password: ");

gets(buffer);

//compare input to password

if(strcmp(buffer, "password")) {

//this block exectes if the input is inccorect

printf ("\nWrong password!\n");

} else {

//this block exectes only if the input is exactly the same as password

printf ("\nCorrect password!\n");

\*flag = '1';

}

//print the contents of the array

printf("\n");

printf("buffer = { ");

for(i=0;i<MAX;i++){

if(i!=0)

printf(", ");

printf ("'%c'", buffer[i]);

}

printf(" }");

printf("\n");

//check for overflow by examining the flag

//fails if flag is over written with 1, eg. passwordx1

if(\*flag!='0' && \*flag!='1') {

printf ("\nFlag was overwritten. flag = '%c'",\*flag);

printf("\nNon fatal Buffer Overflow has occured.\n\n");

printf ("Login Successful.");

}

if(\*flag=='1') {

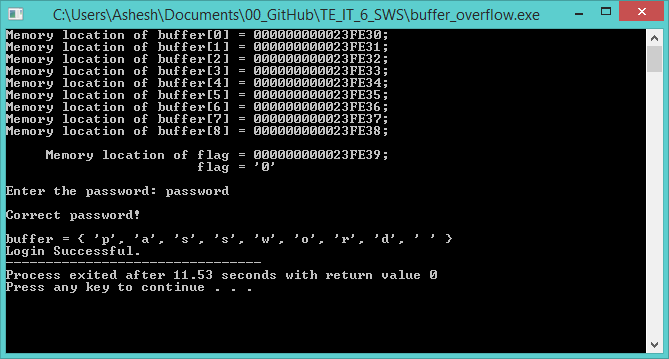
printf ("Login Successful.");

}

return (0);

}

**OUTPUT:**

****

