

Source: http://www.wikiwand.com/en/Replay\_attack, only for picture A and E

Flawfinder Tool with insecure C function’s database C/C++ Source Code

Flawfinder Tool

/\* C library functions considered Harmful \*/

strcpy() // Buffer overflow risks

strcat() // Buffer overflow risks

gets() // Buffer overflow risks

sprintf() // Buffer overflow risks

scanf() // Buffer overflow risks

printf() // Format string problems

snprintf() // Format string problems

system() // Potential shell metacharacter dangers

popen()//Potential shell metacharacter dangers

/\* Vulnerable Program \*/

#include <stdio.h>

//statements

void getName()

{ char str[5];

printf( "Enter your name :");

gets( str );

printf( "\nYou entered: ");

puts( str );

}

int main( )

{

//Statements

getName() /\* Function Call \*/

//Statements

return 0;

}

Match found

by Lexical Analysis

Correct output:

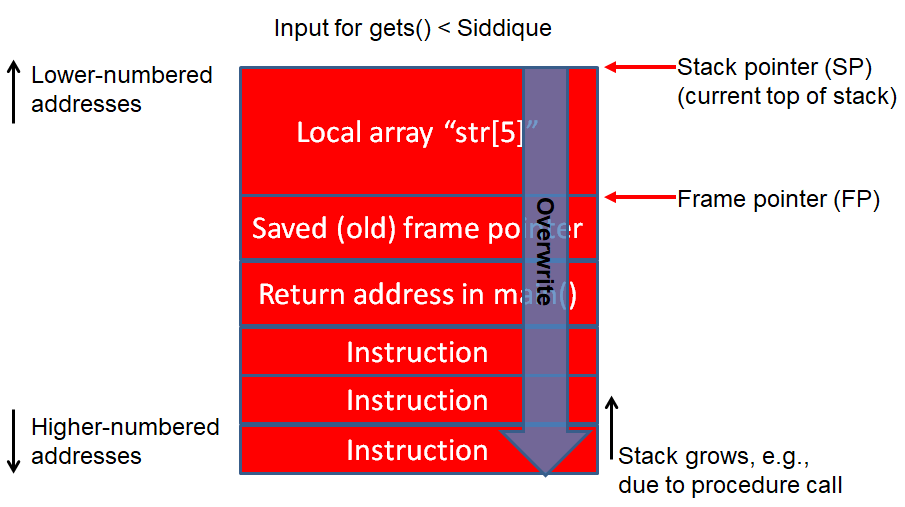
Enter a value : Khan

You entered: Khan

Program crash:

Enter a value: Siddique

**Segmentation Fault**



**Example : For Tool Demonstration**

1. Examine ﬁle only report vulnerabilities level 4 and up

$ flawfinder -m 4 vuln.c | less

2. Examine ﬁle to report the functions that take inputs

$ flawfinder -I vuln.c| less

3. Examine ﬁle to report including even the hits marked for ignoring in the code comments

$ flawfinder -n vuln.c | less

4. Examine file to report all hits in CSV format

$ flawfinder --csv vuln.c >csvData.csv

5. Examine file to report only the actual results (removing the header and footer of the output).

$ flawfinder -QD vuln.c | less

6. Examine file to report only the actual results

$ flawfinder -QDSC vuln.c|less

7. Examine the ﬁle and produce HTML formatted results

$ flawfinder -QHc vuln.c>result.html

8. Examine ﬁle save the resulting hitlist in savehit.txt file

$ flawfinder -Q --savehitlist savehit.txt vuln.c|less

9. Examine ﬁle and show hits that were already in the ﬁle savehit.txt

$ flawfinder --loadhitlist savehit.txt vuln.c|less

10. Examine file, but only report hits where CWE-119 or CWE-120 apply.

$ flawfinder --regex "CWE-119|CWE-120" vuln.c | less