**Buffer Overflow – Team #3**

**Team members:**

* Sharon Greenbaum
* Alon Weinberg
* Ofer Zernichov

**#1 – Clone & Compile**

To clone & compile we’ve gone through the following steps:

1. Forked the Beyond-Cyber-01 repo over to our GitHub account
2. Cloned our repo locally: gh repo clone oferzern/Beyond-Cyber-01
3. Compiled using: gcc challenge2.c -fno-stack-protector

There are built-in security measures within gcc compilation process which protects the stack return address from being overwritten by a longer than expected user input.

Since we do aim to crash the compiled application using buffer overflow exploit we added the “-fno-stack-protector” flag – this disables the built-in security measure that protects the stack.

**#2 – Exploiting the Vulnerability – Crashing challenge2 App**

As a start, we need to understand how the app works – so we’ve looked on its c code:

*#include <stdio.h>*

*#include <string.h>*

*void Success()*

*{*

*printf("\nAccess granted!\n");*

*}*

*void Fail()*

*{*

*printf("\nAccess denied\n");*

*}*

*int main()*

*{*

*char ch[20];*

*printf("What is the password to access this super secret system?\n");*

*scanf("%s", ch);*

*if(strcmp("complex\_password", ch) == 0)*

*{*

*Success();*

*}*

*else*

*{*

*Fail();*

*}*

*}*

Marked with yellow – the array size of the ch variable – any input more than 20 chars have a chance of crashing the app. The scanf function, will assign user’s input to ch, which will hopefully crash the app at the strcmp function.

Started with 30 chars input which was too short to crash the app:

Text

Description automatically generated

We’ve extended the input to 40 chars, which crashed the app:

Text

Description automatically generated

As seen, the fault type is “segmentation fault”:

A segmentation fault (aka segfault) is a common condition that causes programs to crash. Segfaults are caused by a program trying to read or write an illegal memory location.

**#3 – Fixing the Vulnerability**

Since the vulnerability is caused by access illegal memory location, we need to make sure the user’s input which be assigned to the “ch” variable won’t be longer than 20.

Searching online for possible solution to limit variable length assigning we found on StackOverflow site’s how to limit the scanf function to assign up to X chars from the user’s input.

The original scanf:

scanf(“%s”, ch) – this option has no limitation, all of the user’s input will be placed into ch.

The new scanf:

scanf(“%20s”, ch) – this limiting the variable assignment to only the first 20 chars of the user’s input. This, fixing the vulnerability.

We now tested our updated app, with an input of 40 chars – which crashed the app before:

Text

Description automatically generated with low confidence

As can be seen, app kept running