Assignment 9, Fall 2018
CS4630, Defense Against the Dark Arts
Return-oriented Programming Attack:
RIP Ropa Curse

Purpose

In this assignment, you will learn the power of return-oriented programming attacks.

Due

This assignment is due on **Thursday, 29-NOV-2018 at 11:59 pm**

Assignment Prerequisites

1. Review your solution to Assignment 08 (buffer overflows).
2. Download the code: `dumbledore_rop.exe`, `hacked.sh`, and `ROPGadget.tgz` from the class Collab site.

Assignment Details

1. This assignment must be completed using the **64-bit Ubuntu 18.04.1 LTS OS** you installed on your VM for Assignment #1. This environment is where we will test your submitted code. It is possible, due to the sensitivity of vulnerabilities to the operating environment, that exploit code developed for one environment will work not correctly in a slightly different environment.
2. The class Collab site has the file that you should download. You must attack the supplied version of `dumbledore_rop.exe`.
3. Ubuntu has ASLR (Address Space Layout Randomization) turned on as a defense. To simplify your task, we will turn it off. The command
   `setarch 'uname -m' -RL bash`
   will disable ASLR for that shell. It does not affect any other shells.
4. Examine `dumbledore_rop.exe`. It contains an obvious buffer overrun vulnerability in the `readString` function. NOTE: This code is slightly different from `dumbledore.exe`, so please be sure to examine it carefully.
5. A ROP attack, or return-oriented programming attack, is a type of arc-injection attack where the attacker causes "gadgets" already existing in the code to be executed. If sufficient gadgets exist in the executable being attacked, the attacker can cause arbitrary programs to be executed. The goal of our attack is to exec a shell and execute a shell command (`hacked.sh`).

6. Extract the files from `ROPgadget.tgz` in the directory where you will be doing your work. Extract and build ROPgadget using the following commands.

```
$ tar xvzf ROPgadget.tgz
... <list of files extracted> ...
$ cd ROPgadget
$ make clean
$ make
```

7. Make sure the program is built by issuing the following command:

```
./ROPgadget --help
```

8. You will need to read the document to understand how to use `ROPgadget`. There are some YouTube videos demonstrating its use.

9. As mentioned, you will notice that `dumbledore_rop.exe` is very similar to the program you attacked in Assignment 8. So you should know how to craft an attack string that will kick off the execution of a sequence of gadgets. The file `hacked.sh` needs to be in the directory where you carry out the attack (i.e., same directory as the binary you are attacking).

10. The `ROPgadget` tool will generate a program that will create most of the attack string. Here is the command:

```
./ROPgadget/ROPgadget -csyn dumbledore_rop.exe /bin/sh hacked.sh
>rop_attack.c
```

    You will need to add some code to the program that `ROPgadget` generates (in this case `rop_attack.c`) where it says Padding goes here. Basically, this area is the beginning of the attack string – your name and some padding to fill up the buffer so that the first address that `ROPgadget` generated overwrites the return address in function `readStrings` activation. We are essentially doing an arc-injection attack to the first gadget and then `ROPgadget` has set up successive addresses to invoke the appropriate gadgets. Call this modified program (where you have inserted the necessary padding), `rop_attack.c`. The following illustrates an attack run assuming the attack string generated by `ROPgadget` was written to a file called `rop_attack.c` and it was compiled to produce `rop_attack.exe`.

```
$ ./ROPgadget/ROPgadget -csyn dumbledore_rop.exe /bin/sh hacked.sh
>rop_attack.c
...  <edit rop_attack.c to put in padding> ...
$ gcc -m32 -o rop_attack.exe rop_attack.c
$ ./rop_attack.exe >rop_attack.txt
$ ./dumbledore_rop.exe < rop_attack.txt
You succeeded in this challenge. Hearty congratulations!
$
```

    Notice that because the attack code causes a shell to be execed (causing the shell script contained in `hacked.sh` to be executed), the normal output of the program does not appear.

Requirements

1. Submit your attack generation code via the Collab. It must be named `rop_attack.c`.
2. This assignment is pledged.

Items to Submit

This assignment is due on **Thursday, 29-NOV-2018 at 11:59 pm** .

1. Upload your `rop_attack.c` file.

It is mandatory that you use the file names given and adhere to the given API to ease the task of grading multiple different student submissions of this assignment. Throughout the semester, you will be given file names and sample execution output. All assignments will be submitted using the class Collab Website.