Wargame 2 | COMP6447 20T2

The following wargames will provide you with exercises where you will be required to:

1. Solve buffer overflow challenges

2. Solve buffer overflow challenges which contain stack canaries

3. Reverse engineer an binaryninja flow graph into C code

Before you start, we recommend attempting the Lab 2 challenges as we can only provide assistance with challenges from the labs.

You can download the challenges here:

https://cloudstor.aarnet.edu.au/plus/s/40zm2zPKoXkR33r

These challenges are a zip file with the password: **IHatePasswords1!**

There are **4 exploitation challenges** and **1 reverse engineering challenge** this week!

Try to solve the **exploitation** challenges locally first, then connect to our servers to obtain the flags. To get full marks you must get the flag from our servers.

Challenge	IP:PORT
jump (source provided)	plsdonthaq.me:2001
blind (source provided)	plsdonthaq.me:2002
bestsecurity	plsdonthaq.me:2003
stack-dump	plsdonthaq.me:2004

Each **exploitation** challenge has a flag to submit. The flag is in the format FLAG{XXX}. To get full marks in this wargame, you need to submit all flags.

Reverse engineering challenge

This week we also have a reverse engineering challenge. The challenge is in the form of a screenshot of IDA graph view output.

The screenshot is of a **single function** . You are required to fully reverse engineer this function into C code. You're submitted C code should contain a **completed** function, it shouldn't contain trivial errors such as typos, undefined variables. To get full marks for the reverse engineering challenge, your C code must follow the

same logic as the C code we used to generate the image. It does not have to be exactly the same! **This RE challenge is similar to the RE challenge that will be in the final exam, so try to get it as close to the real C as possible.**

Variable and function names are up to your discretion. However if a function or variable name is obvious (can be found in the image) you should use these names. Any arguments should be shown for function calls

ie: the following two code segments are the same in our eyes.

```
int main() {
    int a = 10;
    printf("%d\n", a);
}
```

and

```
int main() {  printf("%d\n", 10); }
```

Submission Instructions

A markdown document (.md) containing the following for each challenge:

We are interested in proof that you understood the challenge, the vulnerabilities and how to exploit them. This is not intended as a formal bug report.

````
chal1
============================
Flag: FLAG{hi}
General overview of problems faced
-----------------------------------
Had to hack the program
Script/Command used
------------------
```
print "hello_world"
```
re challenge
=============
General overview of problems faced
-----------------------------------
needed to use man page to find arguments for atoi
```C
int main(int argc, char** argv) {
    int a = atoi(argv[1][0]);
    if (a > 20) {
        printf("%d\n", a + b);
    }
}
```
````

Please submit the document as a markdown file on give. You may submit as many times as you like. Only your most recent submission will be marked.

Submission

```
give cs6447 war2 war2.md
```

Marking scheme

This week's wargames are worth 3 marks in total.

Due date

The wargames are due **17:59 Tuesday 16th June (Sydney time).** This is in Week 3.

Late Penalty

Late submissions will have marks deducted from the maximum achievable mark at the rate of 1 mark *per day* that they are late.

Resource created 25 days ago, last modified a day ago.