Assignment #4

CPEN 442

November 11

Radu Nesiu(55837132)
Department of Electrical and Computer Engineering
University of British Columbia
Vancouver, Canada

I. Problem #1

A. Methods&tools:

I used hashcat, a gpu cracking tool.The command is "**hashcat64.exe hash.txt –m mode –a 3 ?d?d?d?d –o out.txt –potfile-disable**"

Explanation: hash.txt is the file with the hash. The hash needs to match the hashcat format for a specific hash mode.

The mode tells hashcat what algorithm to use and in some cases how is the salt combined with the password.

-a 3 ?d?d?d?d tells hashcat to use the mask mode of attack which tries all combinations of length 4 using the specified charset(e.g. d for all digits)

The –o is for output and –potfile-disable is to append to the output file and not delete it.

B. Password and details

Password: 2583
StudentID: 55837132
It took less than 2 seconds to crack it.
Entropy: log2(10^4) = 13.28

II. Problem #2

A. Methods&tools:

Same tool. The command is "**hashcat64.exe hash.txt –m mode –a 3 ?a?a?a?a?a?a –o out.txt –potfile-disable**"

The only change here is that we use a different charset (a is the one that matches our assignment alphabet, so each letter can be anything from that alphabet).

B. Password and details

Password: KhlaL+
StudentID: 55837132
It took around 20 minutes to crack it.
Entropy: log2(76^6) = 37.48

III. Problem #3

A. Password and details

Password: aWFT!5K5At-rWRD
StudentID: 55837132
I used IDA Demo to look at the binaries and then I noticed that we need to match a specific length. After that we were reading bytes from a string at a hardcoded offset.

B. Patch

Patch is 55837132.program1.dif
Patch devised with IDA Demo using the dif tool and then applied patch by using the ida_patch.exe. I found the source online(Chris Eagle) and compiled the c file.

IV. Problem #4

A. Password and details

Password: sCa_!H
StudentID: 55837132
I used IDA Demo to look at the binaries and then I noticed that we are hashing using SHA1. Looking at the program we are comparing our hash with the hardcoded hash. I stepped through a few comparison by always making the registers match using IDA.
Using hashcat I used a similar command as in 2 to crack it. It took less 15 minutes this time since there was no salt usage.

B. Patch

Patch is 55837132.program2.dif. Using same tools.
It basically just skips the failure jump, instead doing a useless add. Same trick was used in Question 3.

C. Script

The script is password_changer.py and it replaces the hardcoded hash with the new password hash.