EN3240: Embedded Systems Engineering
Assignment 6 — Validation/Verification & Security

Name: Thalagala B. P.
Index No: 180631J

October 7, 2022

This is an individual assignment!
Due Date: 7 October 2022 by 11.59 PM

Instructions

Please read the instructions and questions carefully. Write your answers directly in the space provided. Compile the tex document and submit the resulting PDF. This is an individual assignment. You are not allowed to take or give any help in completing this assignment.

Problem 1 (1 Point)

How many input patterns (tests) are required (minimum) to verify a 3-input OR gate completely when only binary inputs are allowed (each input can be 0 or 1)? List the inputs and expected outputs.

Problem 2 (2 Points)

How many input patterns (tests) are required (minimum) to verify a 3-input OR gate completely when only ternary inputs are allowed (each input can be 0 or 1 or x; x implies unknown which can be 0 or 1)? List the inputs.

Problem 3 (2 Points)

There are two types of processor simulation techniques - functional and cycle-accurate. Given an input assembly program, the functional simulation produces the correct output but does not provide cycle-by-cycle details. On the other hand, the pipelined simulation provides a cycle-by-cycle simulation of the pipeline to eventually produce the final result. Which one is faster in terms of performance (simulation time) and why? Why do people use the slower one, then?

Answer:

1. In terms of simulation time, Functional Simulator is faster than the Cycle-Accurate Simulator. Because functional simulators are used only to validate the correctness of the design rather than performance characteristics of the design. Their simulation results may not provide accurate timing details such as the delays and the latencies. This is because functional simulators either have no or only a limited model of the microarchitecture. As an example a simple `ADD` instruction of the target processor, may be translated to an `ADD` instruction on the host, and be simulated that way.

2. In contrast to the functional simulators, cycle-accurate simulators include a model of the target microarchitecture and simulation happens on a cycle-by-cycle basis. Theses simulations are chosen over the aforementioned functional simulators when the performance characteristics such as

precision in timing are of the interest. Moreover they provide more complex simulation metrics such as Cycles Per Instructions (CPI), Cache hit/miss rates and are capable of simulating Branch Prediction and Out-of-order Execution of the target microarchitecture.

Problem 4 (2 Points)

There are only two ways of combining compression and encryption:

- CASE I: compression followed by encryption.
- CASE II: encryption followed by compression.

Please indicate which is beneficial for both code size reduction and security improvement. Please explain why the other one is not suitable.

Answer: *compression followed by encryption* (compression first, then encryption), is beneficial.

Encryption followed by compression is not beneficial in terms of code size reduction and security improvement because of the following two simple facts.

- **Compression relies on patterns** found in the original data. Therefore, for optimal compression original data must contain more patterns than random data. Higher the randomness of data, lower the value of compression.
- **Purpose of encryption is to destroy any pattern** in the data and make it non-vulnerable to various attacks. Therefore, encryption essentially increases the randomness of data.

Therefore, encrypting the data at first place destroys the patterns that can be used to effectively compress the data. This eventually make the compression not valuable as it does not yield a significant reduction in code size. Therefore, it can be concluded that compression should be done prior to encryption to get more benefits in terms of code size reduction and security improvement at the same time.

Problem 5 (8 Points)

1. Download the shadow.hex file from the "assignment6-resources" folder. This file has been encrypted using RC4 encryption. 40 bits hexadecimal key: 6D 69 74 72 65. Decrypt the file using any available decryption tool (e.g., cryptool). Add a screenshot of the decrypted shadow.hex file content.

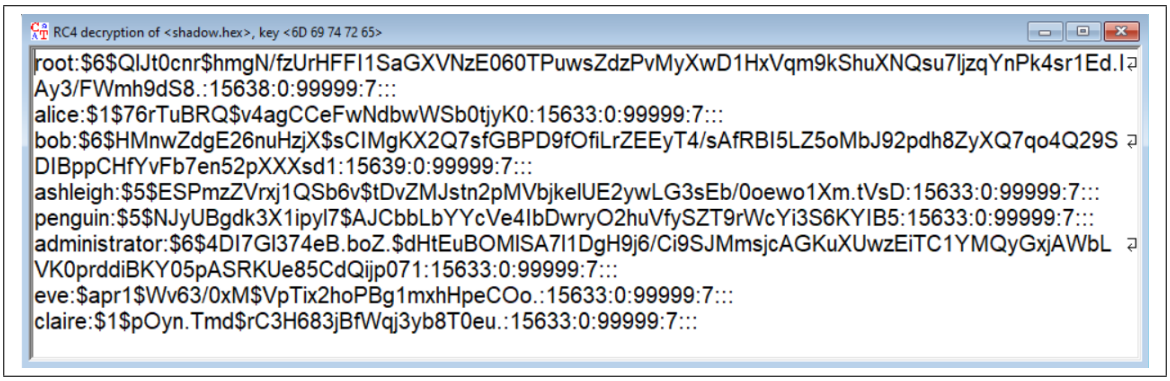

Figure 1: Content of the decrypted "`shadow.hex`" file

2. Run "John the Ripper" password cracking utility to crack the passwords in the decrypted shadow file with the help of the dictionary "rockyou.txt"(Link). What is the command you used to crack the passwords in the shadow file?

"*Note that John can't crack hashes of different types at the same time. If you happen to get a password file that has more than one hash type, you have to invoke John once for each hash type and you need to use this option to make John crack hashes of types other than the one it would auto-detect by default.*" - abstracted from here.

Answer:

Hash ID	Hash format in John	Users
`$6$`	`sha512crypt`	root, bob, administrator
`$5$`	`sha256crypt`	ashleigh, penguin
`$1$, $apr1$`	`md5crypt`	alice, eve, claire

Table 1: Additional `--format` argument that should be passed to crack password

Note: "`resources`" in the following command is a user created directory to keep the password list and the decrypted shadow file.

Following commands were invoked one after the other to crack all the possible passwords in the given shadow file.

```
$ ./john --wordlist=resources/rockyou.txt --rules resources/Cry-RC4-shadow.txt --format=md5crypt
$ ./john --wordlist=resources/rockyou.txt --rules resources/Cry-RC4-shadow.txt --format=sha256crypt
$ ./john --wordlist=resources/rockyou.txt --rules resources/Cry-RC4-shadow.txt --format=sha512crypt
```

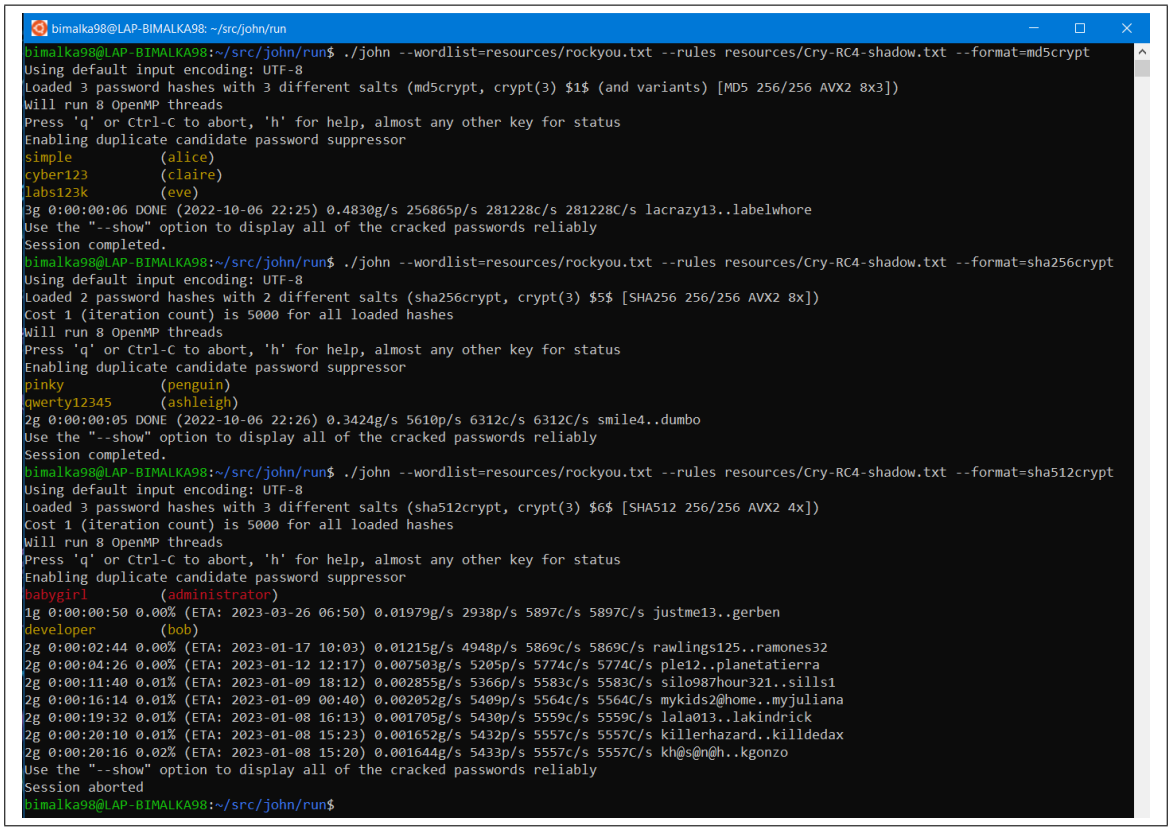

Figure 2: Status of *John the Ripper* while cracking the passwords

It was observed that, the Estimated Time of Arrival(ETA) to crack the password of the *root* user is 2023-01-XX and its progress increments really slowly. Therefore the program was aborted without trying to crack it.

3. Add a screenshot of all the cracked passwords.

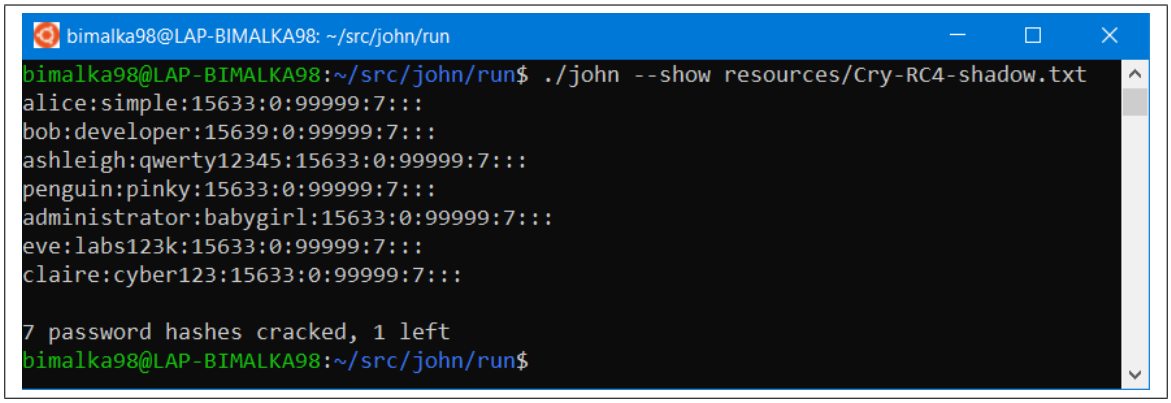

Figure 3: All the cracked passwords

4. Provide recommendations to enhance the strength of the passwords.

   Answer: By observing time duration the program took to crack each password, following recommendations can be provided.

   - When creating a password a combination of numbers, letters (upper case and/or lower case) and special characters must be used and the password must be sufficiently long to make it stronger.
   - When selecting an encryption method, SHA-512 is preferred over SHA-256 and the MD5. Because the latter two methods were comparably weak and the passwords of the *ashleigh, penguin, alice, eve* and *claire* were cracked in less than 10 seconds. Whereas it took 50 seconds to guess the password of *administrator* and nearly two minutes to crack the password of the *bob*.
   - When creating a password common words (such as *simple, cyber,labs (found above) and etc*) and patterns (such as *123, 12345 (found above) and etc*) must be strictly avoided. As these words can be easily guessed and cracked in no time as seen in the above scenario.
   - A password must not contain personal information such as name, birth year and etc. as those can easily be combined to form a candidate set of passwords easily.