Secure System Design: Threats and Countermeasures
CS392

Date: 1st Apr 2021 — Assignment 2

Submission Filename: assign2.pdf — Due Date: 9th Apr 2021

Full Marks 40

1 Assignment Overview

The learning objective of this assignment is for students to gain the first-hand experience on developing a more secured authentication system which can check for weak passwords and can also detect password file compromise attack.

1.1 Implement a Honeyword Based System

Implement Erguler's honeyword based scheme in a linux environment.

Following assumptions can be made for this project.

1. A password could be 8 to 12 characters length.
2. In your implementation, you may use $k = 6$ i.e, number of honeywords will be five + the actual password. But you must have an option to change the value of k.
3. For storing the hash value, you may use any standard hash function like `md5`. For this purpose, you may use *md5deep* package. This can be installed by using the following command

   ```
   $sudo apt-get install md5deep
   ```

   To convert into `md5` hash string, you may put the password temporarily in a file say *temp.txt* and then you can use the following command to store the hash value in a file (say, *hashFile.txt*).

   ```
   $md5deep temp.txt >> hashFile.txt
   ```

   This will convert the contents of *temp.txt* file into `md5` hash string and then appends to *hashFile.txt*.

1.2 Use of Password Cracking Tool

A system administrator needs to be careful that users should not use easy to crack passwords. For this purpose, a password cracking tool can be used to check for the weak passwords.

For this experiment, you can use *John The Ripper*[1] tool, also known as `john`. This tool uses a dictionary or a search pattern to check for passwords. To install this tool, you may use the following command

```
$sudo apt-get install john
```

Now, use *su* command and change to root. After then, create a folder named *test*. Change its permission to 777 by using *chmod* command.

Now, goto *test* folder and get a *wordlist* dictionary. You can use the following command to get a dictionary of *wordlist*.

```
#wget http://scrapmaker.com/data/wordlists/dictionaries/rockyou.txt
```

Once the *wordlist* file is downloaded then you can add user using *adduser* command and create an account for a new user. Let's create an account with username `alice` and password `alice`. You can check */etc/shadow* file to check the entry of that new user's account. The password of that new user is now stored using salted hash function. Now, you can use `john` to find whether the password can be cracked or not. If it is available in the *wordlist* file then it should show the corresponding password against the username. To perform this, following command can be used-

```
#john --wordlist=rockyou.txt passwordFile
```

This will try to explore all the hashed entries of passwordFile with specified *rockyou wordlist*. Please note that it is a time taking task. If no *wordlist* is specified then system will use the default *wordlist*. Also, if you want to check the already cracked passwords from a password file then the following command can be used

[1] http://www.openwall.com/john/

```
$sudo john --show passwordFile
```

You have to use this tool to check for weak password entries in your honeyword system.

Now your task is to obtain the passwords and honeywords using `john` tool. Report whether you can obtain all the passwords and also the time needed to crack them. To check time requirement, you can use the *time* command.

2 Submission

You need to submit the followings-

- assign2.pdf: Describe what you have done in short summary form. You need to provide explanation to the observations that are interesting or surprising. Attach supporting snapshots wherever possible. Also, include any assumptions you have considered for the implementation.
- Source codes: C source files for implementing the honeyword based scheme.
- Password File(s): Any password file(s) that need to be provided as input. Otherwise, if the password file is generated automatically then you do not need to submit this file.
- Wordlist File: This file will be used as dictionary for password cracking tool
- Makefile: As you may use more than one source files, so provide a Makefile to build the executable(s) from your source codes

All these files should be archived to assign2.tgz. You can use `tar -cvzf assign2.tgz folderName` to create such *tgz* file.