Assignment 3 – Authentication

Assignment reports must be submitted by ***Thursday, August 12, 2021****.*

Assignment Description

In this assignment you will carry out exercises related to the lectures on authentication. You will work on cracking password hashes, as well as investigate two-factor authentication on the Internet.

Assignment Requirements and Setup

Virtual machines. You will need to use the Kali Linux virtual machines. If you have not downloaded and installed it yet, please check the "Assignment 0" document.

John the Ripper. You will need to use a tool called John the Ripper in order to crack password hashes. This tool is pre-installed in the Kali Linux virtual machine. However, John is very computationally intensive, and it might run faster on your main computer rather than inside a virtual machine. If you do decide to install it on your main computer, be sure to use the "jumbo" version, which contains support for many more types of hash algorithms.

- For Windows, you can download a **jumbo** build from the John the Ripper homepage (`http://www.openwall.com/john/`)
- For macOS, you can install John using the command-line package manager "Homebrew". To install Homebrew, visit `https://brew.sh/`. Once you've installed Homebrew, you can install John using the command `brew install john-jumbo`.

Getting text to and from your virtual machine. During this assignment, you may need to copy text to or from your various virtual machines. This can be somewhat annoying. It is possible to set up shared clipboards in VirtualBox (Settings → General → Advanced → Shared Clipboard → Bidirectional), but these are not always reliable. An easy way is to use a web-based clipboard, such as `https://cl1p.net`. Of course, such a web-based clipboard is not secure, so you should not use it for anything sensitive, but it should be fine for these assignments.

To Hand In

Answer the questions labelled "3-1", "3-2", etc.

1 Password hashing and hash cracking

In this section you will use various techniques to try to crack some passwords. Imagine you have successfully hacked into a database (for example, using a SQL injection attack), and you have found that the server is storing password hashes. You have found a few hashes, and your goal is now to find out what the original passwords were.

1.1 Basic password hash cracking

In this task, you must try to crack the following hashed passwords. The first three are very easy to crack – you don't need any specialized tools or software to find the passwords behind these hashes. (In particular, you shouldn't need to use John the Ripper yet.)

a) `e10adc3949ba59abbe56e057f20f883e`
b) `c822c1b63853ed273b89687ac505f9fa`
c) `a57e1b2dc95555d1709d0a324ad145de32150182`

Submit answers to the following questions:

3-1. **[1.5 marks]** What passwords did you find?
3-2. **[2 marks]** What hash algorithm was used for each password? How do you know?

1.2 Password hash cracking using John the Ripper

Now we will use John the Ripper to crack password hashes. As noted in the Assignment Requirements and Setup section above, you can use John inside the Kali Linux virtual machine, or install it on your computer.

To make sure that John is working for you, first try to crack the following basic unsalted hash.

- `5994471abb01112afcc18159f6cc74b4f511b99806da59b3caf5a9c173cacfc5`

Save the above hash to a text file. Here is the command you can use to get going with John: `john --format=Raw-SHA256 myfile.txt`

Sometimes John can detect the hash algorithm used, but for now I've told John that the above hash was computed using SHA-256.

You should see output like the following:

```
Loaded 1 password hash (Raw-SHA256 [SHA256 128/128 SSSE3 4x])
Press 'q' or Ctrl-C to abort, almost any other key for status
12345            (?)
1g 0:00:00:00 DONE 2/3 (2017-09-20 19:57) 100.0g/s 400.0p/s 400.0c/s 400.0C/s 123456..passwor
```

```
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

This tells you that the password was `12345`.

If you run the same command again, the output will be different:

```
Loaded 1 password hash (Raw-SHA256 [SHA256 128/128 SSSE3 4x])
No password hashes left to crack (see FAQ)
```

John maintains a cache and does not try to crack solved hashes again. If you want to see the output even though it's already been cracked, use
`john --format=Raw-SHA256 --show myfile.txt`

Use John the Ripper to crack the following password hash:

d) `$2b$04$XnmzDQC1FAZrKcDI8tdEq.7iZSfu/m1jZuSgZqx6zRC5ALKQAFSgC`

Use John the Ripper to crack the following password hash. (Note that for the purposes of this exercise, please do not run John on both (d) and (e) simultaneously. Do (d) first in one file, then (e) next in a different file.)

e) `$2b$08$QzmOYnRO5fQPXK2oYOFtIuYQVXdXG1MZpuGdQY1Dd8bp5o6cbhyjG`

3-3. **[1 mark]** What passwords did you find for hashes (d) and (e)?

Hashes (d) and (e) are in a format called the Modular Crypt Format, which can represent hashes from many different hash algorithms as described here: https://passlib.readthedocs.io/en/stable/modular_crypt_format.html

3-4. **[1 mark]** Which hashing algorithm was used for hashes (d) and (e)?
3-5. **[2 marks]** Why did (e) take so much longer to crack than (d)?

Here are three more hashes:

f) `$2b$10$zQMG94N8PgMd6KGja9Iw8.zZANtGRc07.2shwzFoTCtCHUtJo3Koa`
g) `$2b$10$zQMG94N8PgMd6KGja9Iw8.rlZFDxNMhLMGrZtRvFYsb2JRQ57LaVy`
h) `$2b$10$zQMG94N8PgMd6KGja9Iw8.aIH4PfRNQmfaCXMA5LpSU2zcEWwdpQS`

3-6. **[2 marks]** If you put hashes (f), (g), and (h) together in a single file and run John on that file, it will run faster than if you put them in separate files and run John on each file.[1] Why?
3-7. **[2 marks]** If you were performing a security assessment of a server and you found hashes (f), (g), and (h) in the password database of a real system, state two weaknesses/misconfigurations you would include in your security assessment report to the system administrator.

[1] To test this, you will need to clear John's cache in between running them in a single file versus running them individually. You can clear John's cache by removing the folder `~/.john` on macOS or Linux.

1.3 Password cracking runtime estimation

Here are two password generation strategies:

a) Pick a random word from the dictionary, and append a random single digit and a random punctuation character.
b) Pick a random ten character password, where each character is chosen uniformly random from `a-zA-Z0-9!@#$%&*()-=,.`

Here are three password hashing scenarios:

i) Hash the password using SHA-1; store the hash.
ii) Pick a random salt and hash the password and the salt using SHA-1; store the hash and the salt. (Assume the attacker knows the salt when trying to crack the password.)
iii) Pick a random salt and hash the password and the salt using PBKDF2 with SHA-1 and iteration counter 10000; store the hash and the salt. (Assume the attacker knows the salt when trying to crack the password.)

Use the command `openssl speed sha1` on your virtual machine to estimate the time it takes to compute SHA-1. You can use the first line of output, which says how many SHA-1 hashes can be computed in 3 seconds.

3-8. **[2 marks]** What is the entropy of each of the two password distributions (a) and (b)? Explain any assumptions you made.
3-9. **[6 marks]** How long would it take on average to find the password for each of the two password generation strategies (a), (b) in each of the three password hashing scenarios (i), (ii), (iii). (6 answers total). Explain the methodology that you used to arrive at your answer and any assumptions you made.

2 Security in your life: 2-factor authentication

Take a look at the following list of online services supporting 2-factor authentication: https://2fa.directory

Pick one service from that list that you use and investigate what it would take to enable 2-factor authentication. (It is up to you whether you actually decide to enable 2-factor authentication.)

3-10. **[1 mark]** What service did you pick? What forms of two-factor authentication does that service support?
3-11. **[1 mark]** What are the risks to the user in enabling that service's two-factor authentication, thinking especially about losing a computer or mobile phone?
3-12. **[1 mark]** Thinking as an attacker, how would you try to circumvent that service's two factor authentication?