Emebet Kumsa

Date: May 2025

Lab 2: Password Cracking Report

Task 1: Password Files

1. Permission denied

```
student@pass-crack:~$ more /etc/shadow
more: cannot open /etc/shadow: Permission denied
student@pass-crack:~$
```

2. Root permission is required to view the /etc/shadow file because it contains sensitive information like hashed passwords.

3. The **ID** in the hash is **6**, which is **SHA-512.**

```
student:$6$7voVz3cj$I5AjEXbywvsB.pzjeV4D7m2EdKCeWIefenJ5QDXb0bff0PSHko/6
hKjtat7s5QzC0Zx5hlFQ5OquhfsZHm0C40:17719:0:99999:7:::
student@pass-crack:~$
```

4. The **salt value** used in the generation of the stored digest is **7voVz3cj,** as shown in the shadow file entry.

5. The date when my password was chosen is **July 07, 2018.**

```
student@pass-crack:~$ chage -l student
Last password change                                    : Jul 07, 2018
Password expires                                        : never
Password inactive                                       : never
Account expires                                         : never
Minimum number of days between password change          : 0
Maximum number of days between password change          : 99999
Number of days of warning before password expires       : 7
```

6. The **chage –l** command shows details about my password, like when it was last changed and when it will expire. It helps manage password security by ensuring regular updates and giving warnings before expiration.

Task 2: Dictionary Attacks

7. The users who selected the same password are **alice** and **dave**, since they have the same password hash.

```
student@pass-crack:~$ cat htpasswd-sha1
alice:{SHA}A9Z8JjwnpFPvZbKeMDNHJzM8y80=
bob:{SHA}44rSFJQ9qtHWTBAvrsKd5K/p2j0=
carol:{SHA}drwuv3DCvq6R3jONTBmOYoZDdkU=
dave:{SHA}A9Z8JjwnpFPvZbKeMDNHJzM8y80=
eve:{SHA}zfVH7Uxk5plK81z81pxCBMkiepc=
frank:{SHA}mZw1UrSre5xao86b+zuTBdAlPhA=
george:{SHA}Bjq99Tpg62+4dkPUHGQlFhidFTk=
student@pass-crack:~$
```

Task 2.1: Simple Dictionary Attack

8. The accounts were cracked using tinylist.txt:

- **Username:** alice
  - ★ **Password**: awesome
- **Username:** dave
  - ★ **Password:** awesome

```
student@pass-crack:~$ ./crackSHA.py htpasswd-sha1 tinylist.txt
----------------------------
Pre-processing of input data:
----------------------------
Number of passwords = 7
Number of words     = 109582
Extracting hashes from file
----------------------------
Cracking...
----------------------------
alice password is 'awesome'
dave password is 'awesome'
----------------------------
Found 2 out of 7 passwords.
Processed 109582 words in 0.647 seconds.
student@pass-crack:~$
```

Task 2.2: Common Password Dictionary Attack

9. These are the accounts where the passwords cracked using the **biglist.txt:**

- **Username:** carol
  - ★ **Password**: 2cute4u
- **Username:** alice
  - ★ **Passwod:** awesome
- **Username:** dave
  - ★ **Password:** awesome
- **Username:** bob
  - ★ **Password:** password1
- **Username:** eve
  - ★ **Password:** zaq12wsx

```
student@pass-crack:~$ ./crackSHA.py htpasswd-sha1 biglist.txt
------------------------------
Pre-processing of input data:
------------------------------
Number of passwords = 7
Number of words     = 2198690
Extracting hashes from file
------------------------------
Cracking...
------------------------------
carol password is '2cute4u'
alice password is 'awesome'
dave password is 'awesome'
bob password is 'password1'
eve password is 'zaq12wsx'
------------------------------
Found 5 out of 7 passwords.
Processed 2198690 words in 10.463 seconds.
student@pass-crack:~$
```

10.

- **Number of words attempted:** 2,198,690
- Number of passwords cracked: 5 out of 7
- Time taken: 10.463 seconds

```
Found 5 out of 7 passwords.
Processed 2198690 words in 10.463 seconds.
student@pass-crack:~$
```

11. Using a larger dictionary like biglist.txt improves both the success rate and efficiency of password cracking. The bigger list can crack more passwords, as shown with 5 out of 7

passwords cracked, compared to just 2 out of 7 with the smaller list.

Impact on efficiency:

- **Larger dictionary:** Cracks more passwords but takes a bit longer.
- Smaller dictionary: Faster but less effective.

I recommend using strong, unique passwords and avoiding common passwords like "password1". Also, enable multi-factor authentication (MFA) to add extra protection.

Task 3: Considering Execution Time

12.

- **Number of words attempted: 2,198,690**
- **Number of passwords cracked: 5 out of 7**
- **Time taken: 10.014 seconds**

```
student@pass-crack:~$ ./crackMD5.py htpasswd-md5 biglist.txt
------------------------------
Pre-processing of input data:
------------------------------
Number of passwords = 7
Number of words     = 2198690
Extracting hashes from file
------------------------------
Cracking...
------------------------------
carol password is '2cute4u'
alice password is 'awesome'
dave password is 'awesome'
bob password is 'password1'
eve password is 'zaq12wsx'
------------------------------
Found 5 out of 7 passwords.
Processed 2198690 words in 10.014 seconds.
student@pass-crack:~$
```

13. The speed comparison shows me that MD5 is faster but less secure, while SHA-1 is slower but more secure. For better password security, SHA-256 or SHA-512 are better choices, even though they're slower. Speed shouldn't be more important than protection.

14. Salts are random strings added to passwords before they are hashed. Also, they make sure even the same password has different hashes for each user.

15.

- **Number of words attempted:** 2,198,690
- **Number of passwords cracked:** 5 out of 7
- **Time taken:** 11,523 seconds

```
student@pass-crack:~$ ./crack512.py htpasswd-sha512 biglist.txt
----------------------------
Pre-processing of input data:
----------------------------
Number of passwords = 7
Number of words     = 2198690
Extracting hashes from file
----------------------------
Cracking...
----------------------------
carol password is '2cute4u'
alice password is 'awesome'
dave password is 'awesome'
bob password is 'password1'
eve password is 'zaq12wsx'
----------------------------
Found 5 out of 7 passwords.
Processed 2198690 words in 11.523 seconds.
student@pass-crack:~$
```

16.

- **Number of words attempted:** 3,693
- **Number of passwords cracked:** 5 out of 7
- **Time taken:** 0.045

```
student@pass-crack:~$ ./crackPre.py htpasswd-sha1 calc
------------------------------
Cracking...
------------------------------
alice : password = "awesome"
bob : password = "password1"
carol : password = "2cute4u"
dave : password = "awesome"
eve : password = "zaq12wsx"
------------------------------
Found 5 out of 7 passwords.
Processed 3693 words in 0.045 seconds.
```

17. Using **pre-calculated: -** digests is much faster than dictionary attacks because the hashing is already done, making the process take just milliseconds. And **dictionary attacks: - are** slower since each word needs to be hashed, but they are more flexible and don't depend on pre-computed digests.

18. My password experimentation showed that using commonly used or weak passwords can make my accounts much easier to crack. Since the passwords I chose weren't found in the pre-calculated digest list, they seemed to be strong enough to crack.

19.

- Always create passwords that are long, complex (numbers, letters and characters) and unique for each account. Avoid using simple or common words.
- Adding an extra layer of security through MFA.
- Never use the same password across multiple accounts.
- Regularly update passwords.