BOSTON UNIVERSITY METROPOLITAN COLLEGE

MET CS 695S O2 REG Cybersecurity (2024 Summer 2)

Lab 4 Password Cracking

Submitted to

Nicklos See, M.S.,

Warren Mansur, M.S.,

Shengzhi Zhang, Ph.D.

MET CS695S O2 REG

Cybersecurity

by

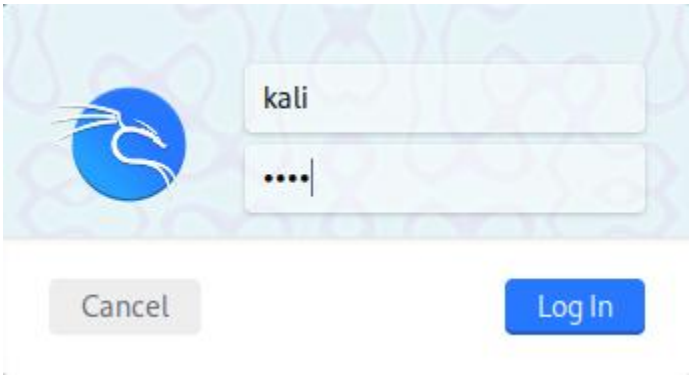
Michael G Nguyen

7/27/2024

Table of Contents

Part 1: Using hydra-graphical for Online Password Cracking

1. Start your Kali Linux and Metasploitable2 Linux using VMWare, and log in.



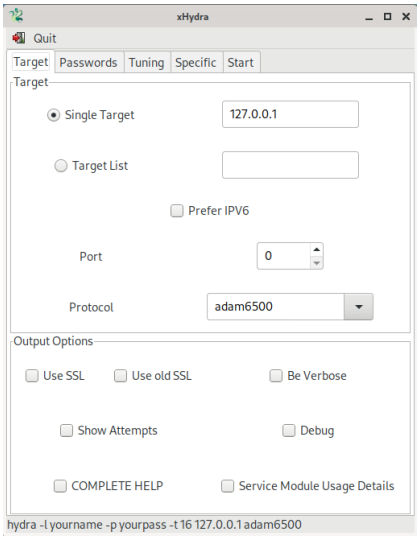
Signing in to Kali Linux.

2. On metasploitable2 Linux, create a user administrator. Set the password of administrator as abc123. Recall when you use sudo the first time, it will ask you to input your password (This is your own password on metasploitable2 Linux: msfadmin). Let's consider the attackers try to hack the user account administrator via dictionary attack.

```
msfadmin@metasploitable:~$ sudo useradd administrator
[sudo] password for msfadmin:
msfadmin@metasploitable:~$ sudo passwd administrator
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
msfadmin@metasploitable:~$ _
```

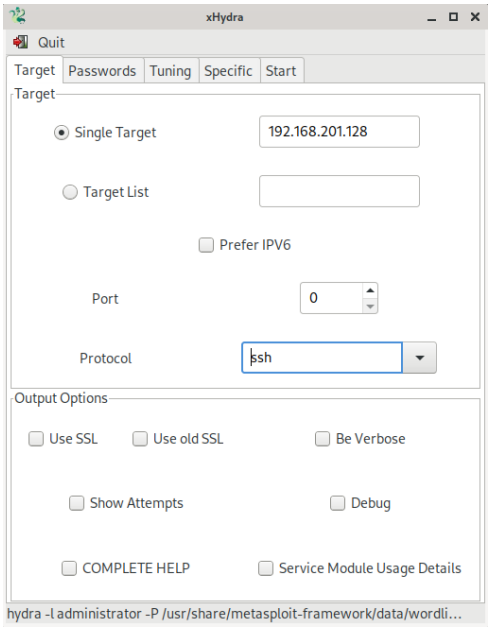
Creating a administrator user and setting password to abc123

3. On Kali Linux, click the menu , then 05-password Attacks -> Online Attacks -> hydragraphical. You should be able to see the interface of hydra-graphical.



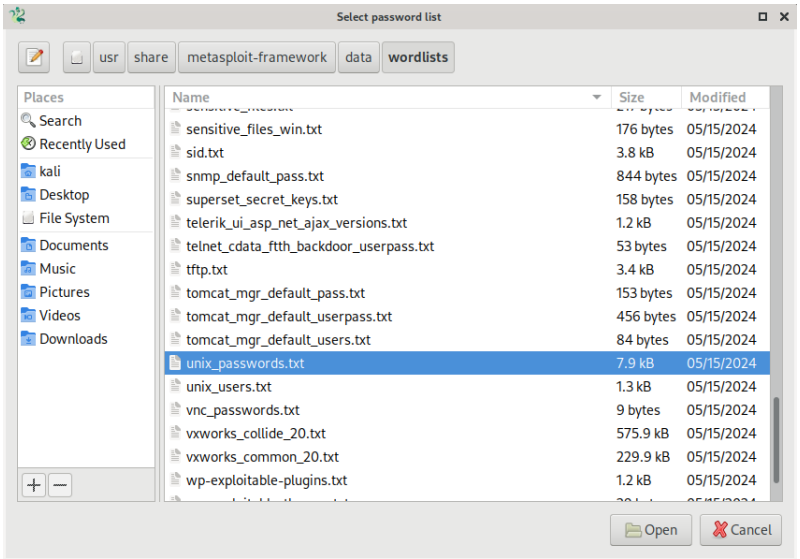
Opening hydra-graphical.

4. On hydra-graphical, first configure the Target (Target tab on top). Type the IP address of your target (metasploitable2-Linux) in Single Target (It is the IP address of your probing target in Lab 2). Change the Protocol to ssh.

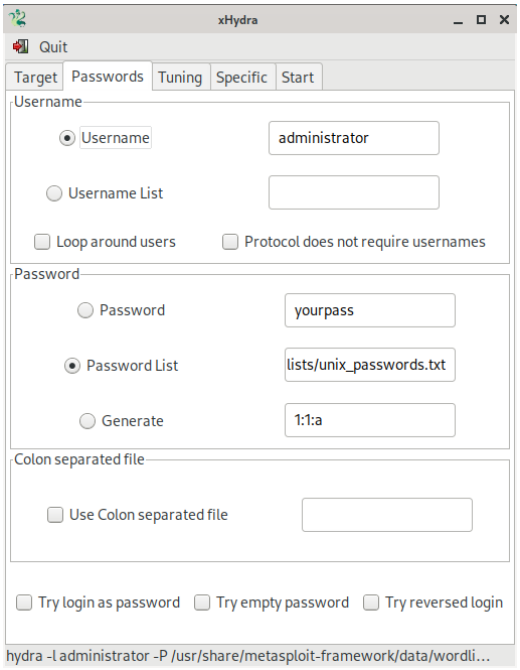


Changing to metasploitable ifconfig IP address and ssh protocol.

5. Click the Passwords tab. Here we assume the attackers by somehow know there exists a username of administrator on their target, so they just want to hack the password. Type administrator in Username. To crack passwords, the attackers need a dictionary, e.g., commonly used passwords, words list, etc. Kali Linux provides some dictionaries for you to use in different scenarios. They are stored in the directory of /usr/share/metasploitframework/data/wordlists/. You can use ls command to list all files in that directory and use gedit editor to check the content of each file.

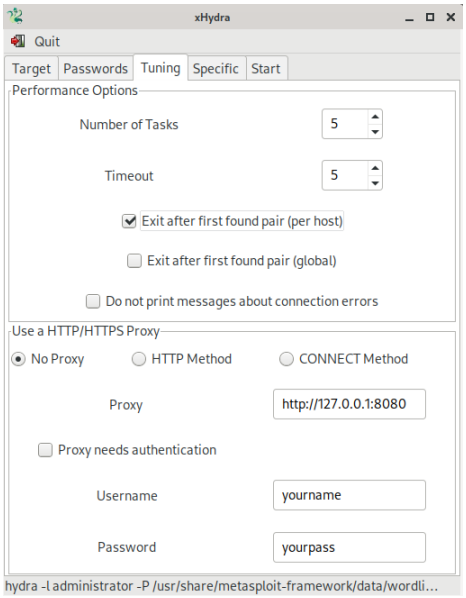


Locating the file for unix-passwords.txt for xHydra.



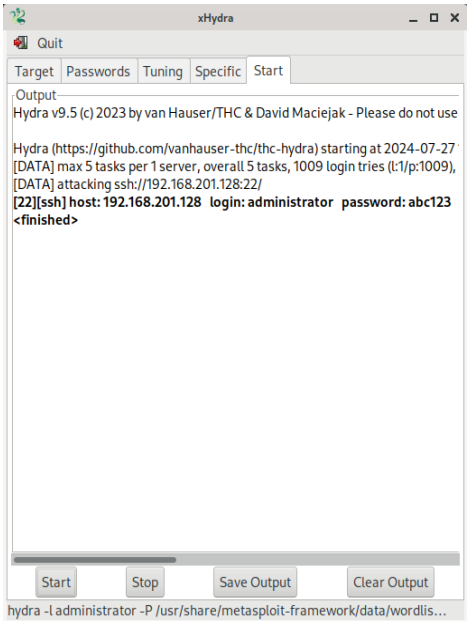
The configuration for passwords in xHydra.

6. Click the Tuning tab. Change the number of tasks to 5, which is the number of concurrent attack tasks to run in parallel. A too big number may cause congestion on your target, thus reducing cracking chance. Set Timeout to 5, which is the maximum time that the attack will wait for response from the target. Check the "Exit after first found pair (per host)", since we only have one target and are only interested in one successful cracking.



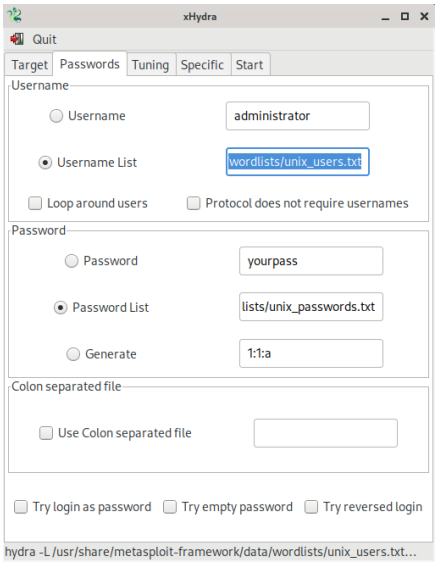
Changing the number of task to 5 and concurrent task to run in timeout to 5, then exit after the first found pair (per host).

7. Click the Start tab, and click Start on the bottom left of the window. You should see it finishes with the output: the login: administrator password: abc123.

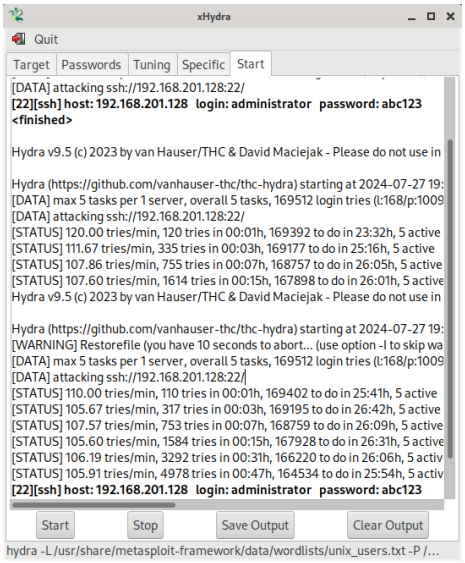


Password cracking in xHydra with SSH to metasploitable 2 of 192.168.201.128 with login of administrator and password of abc123.

8. Now suppose the attackers do not know any users name on the target, so they have to guess user names as well. We can keep all the settings as the above, and only change the Username List on the Passwords tab. Similar as setting the PasswordList, in the new window, choose File System, and then traverse /usr/share/metasploitframework/data/wordlists down to wordlists directory, choose unix_users.txt and click open. Then you can start. This time, it will take longer than the cracking in Step 7, since different user names also needs to be tried based on dictionary. Ignore the estimated time. It won't take that long, and usually should finish within 1 hour.



Changing username from administrator to unix_users.txt.



Using unix_users.txt and unix_passwords.txt to crack the username and password.

9. On Metasploitable2-Linux, run the following command to check the log of all failed login attempts. Type q to return to the terminal. Answer Question 3. grep 'Failed password' /var/log/auth.log | more

```
Jul 27 18:53:00 metasploitable sshd[5962]: Failed password for administrator fro
m 192.168.201.129 port 46798 ssh2
Jul 27 18:53:00 metasploitable sshd[5960]: Failed password for administrator fro
m 192.168.201.129 port 46762 ssh2
Jul 27 18:53:00 metasploitable sshd[5963]: Failed password for administrator fro
m 192.168.201.129 port 46804 ssh2
Jul 27 19:09:03 metasploitable sshd[6017]: Failed password for invalid user kali
 from 192.168.201.129 port 52640 ssh2
Jul 27 19:09:03 metasploitable sshd[6018]: Failed password for invalid user kali
 from 192.168.201.129 port 52650 ssh2
Jul 27 19:09:03 metasploitable sshd[6019]: Failed password for invalid user kali
 from 192.168.201.129 port 52642 ssh2
Jul 27 19:09:03 metasploitable sshd[6021]: Failed password for invalid user kali
 from 192.168.201.129 port 52654 ssh2
Jul 27 19:09:03 metasploitable sshd[6020]: Failed password for invalid user kali
 from 192.168.201.129 port 52652 ssh2
Jul 27 19:09:05 metasploitable sshd[6021]: Failed password for invalid user kali
 from 192.168.201.129 port 52654 ssh2
Jul 27 19:09:05 metasploitable sshd[6018]: Failed password for invalid user kali
 from 192.168.201.129 port 52650 ssh2
Jul 27 19:09:05 metasploitable sshd[6017]: Failed password for invalid user kali
 from 192.168.201.129 port 52640 ssh2
Jul 27 19:09:05 metasploitable sshd[6019]: Failed password for invalid user kali
 from 192.168.201.129 port 52642 ssh2
--More--
```

Executing a grep command to check for failed password with a pipeline, grep 'Failed password' /var/log/auth.log | more

Questions

1. As you expect, the success of password cracking highly depends on the dictionary used. Generally, larger and more comprehensive dictionary produces higher success rate. We used unix_passwords.txt, 7,883 bytes, which contains commonly used account password for unix/linux users. In the same directory (/usr/share/Metasploit-framework/data/wordlists), the file password.lst is much larger, 820,321 bytes, which is more general for different scenarios. In case that using unix_passwords.txt didn't give a success, we can try password.lst. Take a look at other files in the same directory (wordlists). From their names, you can infer which scenarios they are used for, e.g., oracle_default_userpass.txt contains the frequently used passwords for oracle

database. Find three password dictionaries used in different scenarios in the wordlists directory, and compare them. What are your findings and conclusion?

Password Dictionaries Comparison:

Files in /usr/share/metasploit-framework/data/wordlists

- unix_passwords.txt (7,883 bytes): Contains commonly used account passwords for Unix/Linux users.
- password.lst (820,321 bytes): A more general and comprehensive list for different scenarios.
- oracle_default_userpass.txt: Contains frequently used passwords for Oracle databases.

Comparison:

1. unix_passwords.txt
    - Size: 7,883 bytes
    - Scope: Specific to Unix/Linux users
    - Strength: Contains common Unix/Linux passwords
    - Use Case: Cracking Unix/Linux user passwords
2. password.lst
    - Size: 820,321 bytes
    - Scope: General and comprehensive
    - Strength: Extensive list covering various scenarios
    - Use Case: General password cracking attempts across different systems
3. oracle_default_userpass.txt
    - Size: Relatively smaller
    - Scope: Specific to Oracle databases
    - Strength: Contains commonly used Oracle database passwords
    - Use Case: Cracking Oracle database passwords

Findings and Conclusion:

- Finding: Larger dictionaries, like password.lst, provide a broader range of potential passwords and thus increase the chances of success in password cracking. However, targeted dictionaries like unix_passwords.txt or oracle_default_userpass.txt are more efficient when the context is known, as they focus on specific environments.
- Conclusion: The choice of dictionary should depend on the context of the attack. For general attacks, larger and more comprehensive dictionaries are preferable. For targeted attacks, specific dictionaries are more effective.

2. What countermeasures (list at least 3) did we study in class or you are aware of to deal with such online password cracking? Think about when you entered wrong passwords several times on your phone, email accounts, etc., what will happen typically?

1. Account Lockout Policies**:** After a certain number of failed login attempts, the account is locked for a specified duration or until an administrator unlocks it. This prevents brute-force attacks by limiting the number of attempts an attacker can make.
2. CAPTCHA Implementation**:** Using CAPTCHAs during the login process can prevent automated login attempts by requiring a human to solve a challenge that bots typically cannot solve.

3. Two-Factor Authentication (2FA)**:** Requiring a second form of authentication, such as a code sent to a mobile device or an authentication app, adds an additional layer of security, making it significantly harder for attackers to gain access even if they have the password.

3. Perform online research if necessary. What is stored in auth.log file? Search for the failed login from administrator by replacing 'Failed password' with 'Failed password for administrator, and keep the other parts of the command unchanged in Step 9. From the output of Step 9, identify the IP address that the login requests come from. Compare it with the IP address of your Kali Linux. Then take a look at the ports from the output of Step 9. Are they the source ports (used on Kali) or destination ports (on Matasploitable)?

The `auth.log` file stores authentication-related events on a Unix/Linux system. This includes successful and failed login attempts, sudo attempts, and other authentication mechanisms.

- Replace the search term in the command to identify failed login attempts by the administrator:

grep "Failed password for administrator" /var/log/auth.log

- Identifying the IP Address and Ports:

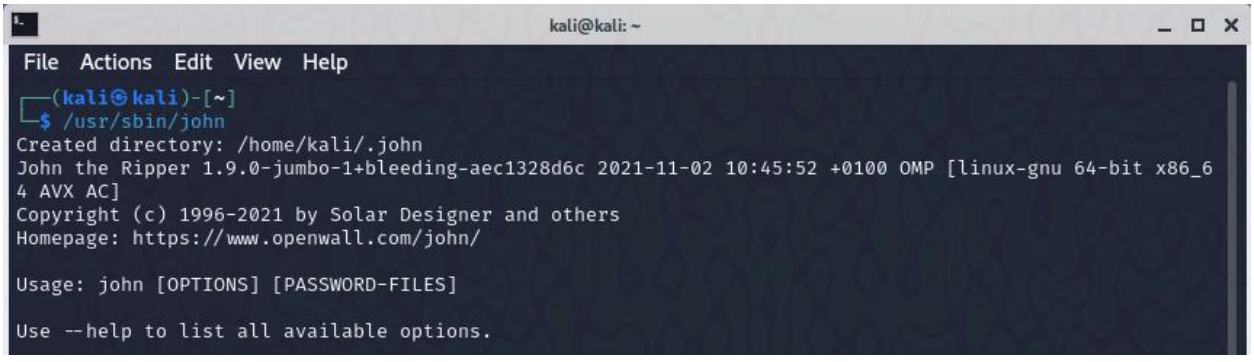
From the output of the above command, you can identify the IP address from which the login attempts are made. Typically, the format of the log entries will include the source IP address.

IP Address: attacking ssh://192.168.201.128:22/ (source IP address)

Ports: 45022 (source port on Kali Linux), destination port typically 22 on Metasploitable 2

Part 2: Using John the Ripper for Offline Password Cracking

1. On Kali Linux, open a terminal, and run the following command to see the manual of John the Ripper. Pay attention to the options of "--show", "--format", and "--wordlist". Understand the meaning of them.



Checking the manual of John the Ripper.

So –wordlist is similar to –stdin, letting reads in bulk and allowing rules.

2. Next, we create several victim user accounts on Kali Linux for our password cracking. Create a user alice. Set the password of alice as 123456. Recall when you use sudo the first time, it will ask you to input your password (This is your own password on Kali Linux). Create a user metcs695. Set the password of metcs695 as abc123. Create a user sysadmin. Set the password of sysadmin as a1b2c3d4.

```
┌──(kali㉿kali)-[~]
└─$ sudo useradd alice
[sudo] password for kali:

┌──(kali㉿kali)-[~]
└─$ sudo passwd alice
New password:
Retype new password:
passwd: password updated successfully

┌──(kali㉿kali)-[~]
└─$ sudo useradd metcs695

┌──(kali㉿kali)-[~]
└─$ sudo passwd metcs695
New password:
Retype new password:
passwd: password updated successfully
```

```
┌──(kali㉿kali)-[~]
└─$ sudo useradd sysadmin

┌──(kali㉿kali)-[~]
└─$ sudo passwd sysadmin
New password:
Retype new password:
passwd: password updated successfully
```

Using superuser do or sudo, to create an account for alice, metcs695, and sysadmin, but also assigning a password.

3. Run the following commands to show the contents of /etc/passwd and /etc/shadow. /etc/passwd file stores user account information, which is required during login. /etc/shadow file stores actual password in encrypted format with additional properties related to user passwords. Answer Quesiton 1 based on your output.

```
alice:x:1001:1001::/home/alice:/bin/sh
metcs695:x:1002:1002::/home/metcs695:/bin/sh
sysadmin:x:1003:1003::/home/sysadmin:/bin/sh
```

Concatenate or cat /etc/passwd

```
alice:$y$j9T$jIETMoRGj3sZkNnUDCYU20$NLGXZuALULXAMEQ4iQtnyHYrK3Co/A37qKRuYsu6c54:19932:0:99999:7:::
metcs695:$y$j9T$uG4l66LWIa7CLPyqbajHh/$ufF/Ipjp9mxtHz2kAIMsbgueWxpIEaqSfRy4qz7aBH6:19932:0:99999:7:::
sysadmin:$y$j9T$BDbyO8RhZCRpiMByrlVJV/$Qr6TRVGbRyKcScR.dGgLVxyqUQue2OESAzZlhnqVO2C:19932:0:99999:7:::
```

Using superuser do or sudo to concatenate or cat /etc/shadow.

4. Run unshadow command to combine the entries of /etc/passwd and /etc/shadow to create one file (/tmp/linux_hashes.txt) with username and password details.

```
┌──(kali㉿kali)-[~]
└─$ sudo /usr/sbin/unshadow /etc/passwd /etc/shadow > '/tmp/linux_hashes.txt'
Created directory: /root/.john
```

Using sudo to create a combined file of passwd and shadow called linux_hashes.txt

```
59 alice:
   $y$j9T$jIETMoRGj3sZkNnUDCYU20$NLGXZuALULXAMEQ4iQtnyHYrK3Co/
   A37qKRuYsu6c54:1001:1001::/home/alice:/bin/sh
60 metcs695:$y$j9T$uG4l66LWIa7CLPyqbajHh/$ufF/
   Ipjp9mxtHz2kAIMsbgueWxpIEaqSfRy4qz7aBH6:1002:1002::/home/
   metcs695:/bin/sh
61 sysadmin:$y$j9T$BDbyO8RhZCRpiMByrlVJV/
   $Qr6TRVGbRyKcScR.dGgLVxyqUQue2OESAzZlhnqVO2C:1003:1003::/
   home/sysadmin:/bin/sh
```

This is the linux_hashes.txt file

5. Use /usr/share/metasploit-framework/data/wordlists/password.lst as the dictionary to crack passwords of user accounts on Kali Linux. Be patient, it takes time to complete, since we use a large size of dictionary compared to Step 7 in Part 1.

```
┌──(kali㉿kali)-[~]
└─$ sudo /usr/sbin/john --format=crypt --wordlist=/usr/share/metasploit-framework/data/wordlists/passwo
rd.lst /tmp/linux_hashes.txt
Using default input encoding: UTF-8
Loaded 3 password hashes with 3 different salts (crypt, generic crypt(3) [?/64])
Cost 1 (algorithm [1:descrypt 2:md5crypt 3:sunmd5 4:bcrypt 5:sha256crypt 6:sha512crypt]) is 0 for all l
oaded hashes
Cost 2 (algorithm specific iterations) is 1 for all loaded hashes
Will run 8 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
123456           (alice)
a1b2c3d4         (metcs695)
```

```
┌──(kali㉿kali)-[~]
└─$ sudo /usr/sbin/john --format=crypt --wordlist=/usr/share/metasploit-framework/data/wordlists/passwo
rd.lst /tmp/linux_hashes.txt
Using default input encoding: UTF-8
Loaded 4 password hashes with 4 different salts (crypt, generic crypt(3) [?/64])
Remaining 2 password hashes with 2 different salts
Cost 1 (algorithm [1:descrypt 2:md5crypt 3:sunmd5 4:bcrypt 5:sha256crypt 6:sha512crypt]) is 0 for all l
oaded hashes
Cost 2 (algorithm specific iterations) is 1 for all loaded hashes
Will run 8 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
a1b2c3d4         (sysadmin)
abc123           (metcs695)
2g 0:00:00:01 DONE (2024-07-27 21:23) 1.904g/s 274.2p/s 548.5c/s 548.5C/s 9..abdicate
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

┌──(kali㉿kali)-[~]
└─$
```

Sudo John the Ripper to ethically hack the password and username of alice, metcs695, and sysadmin.

(Apologies Professor See, for this entry I got caught up with alice, metcs695, and sysadmin in creating passwords and usernames. I tried to ethically hack the usernames and passwords again, however, I couldn't since it was already hacked. Below is the screenshot.)

```
┌──(kali㉿kali)-[~]
└─$ sudo /usr/sbin/john --format=crypt --wordlist=/usr/share/metasploit-framework/data/wordlists/passwo
rd.lst /tmp/linux_hashes.txt
Using default input encoding: UTF-8
Loaded 4 password hashes with 4 different salts (crypt, generic crypt(3) [?/64])
No password hashes left to crack (see FAQ)
```

6. Run the following command to show the cracked user accounts and passwords.

```
┌──(kali㉿kali)-[~]
└─$ sudo /usr/sbin/john --show /tmp/linux_hashes.txt
alice:123456:1001:1001::/home/alice:/bin/sh
metcs695:abc123:1002:1002::/home/metcs695:/bin/sh
sysadmin:a1b2c3d4:1003:1003::/home/sysadmin:/bin/sh

3 password hashes cracked, 1 left
```

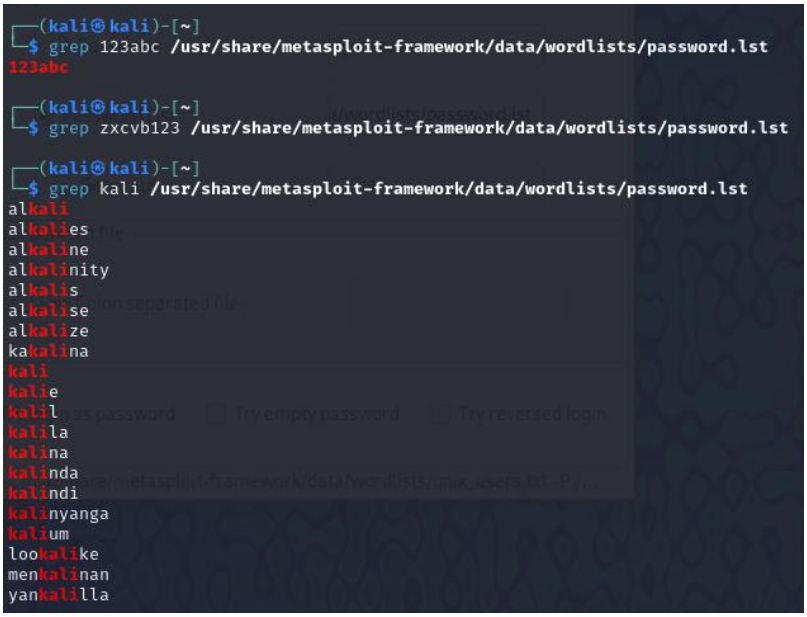
Sudo to show the username of the accounts and passwords.

7. Run the following command and compare the entries for alice, metcs695, and sysadmin with the output from Step 6.

```
alice:x:1001:1001::/home/alice:/bin/sh
metcs695:x:1002:1002::/home/metcs695:/bin/sh
sysadmin:x:1003:1003::/home/sysadmin:/bin/sh
```

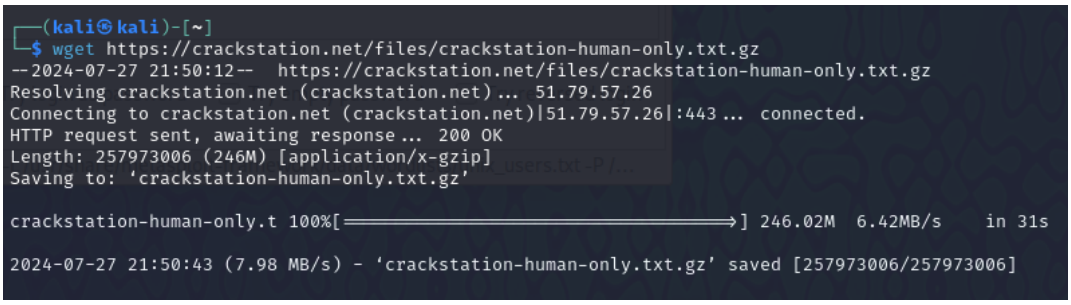
Comparing entries with concatenate or cat command.

8. Search password.lst for your frequently used passwords to see if they are vulnerable or not. For instance, if you frequently used passwords are 123abc and zxcvb123, search them from the password.lst. From the output we can see 123abc is in the dictionary, but zxcvb123 is not (This does not mean the password zxcvb123 is strong enough, since we use a very simply dictionary). You can try it by replacing 123abc with your own passwords (not required in submission). If they are in the dictionary, definitely you want to change them immediately. The file password.lst is a very simple dictionary. If it contains your password, it means your password is too vulnerable.



The password kali is very vulnerable.

9. Download a more complete dictionary, which contains 63,941,069 real human passwords, collected based on previous password breaching from various website databases.



Updating password dictionary files.

10. ncompress the downloaded file, you will obtain a file with the name of crackstation-human-only.txt

```
┌──(kali㉿kali)-[~]
└─$ gunzip crackstation-human-only.txt.gz

┌──(kali㉿kali)-[~]
└─$
```

Unzipping the file with gunzip.

11. Search the password zxcvb123 again from the file and compare the output with Step 8. You can try it by replacing zxcvb123 with your own passwords (not required in submission). If they are in the dictionary, you want to change them immediately as well. The file crackstation-human-only.txt is publicly released. If it contains your password, it means your password is vulnerable.

```
yhancalifornia
york-panama-california
york-to-california
youngconservativesofcalifornia
zecalifornia
zone-california

┌──(kali㉿kali)-[~]
└─$ grep zxcvb123 /home/kali/crackstation-human-only.txt
zxcvb123
zxcvb123$
zxcvb1234
zxcvb12345
zxcvb123456
zxcvb123456789
zxcvb123rt
zxcvb123_
```

12. There exist even larger dictionaries. Refer to https://crackstation.net/crackstationwordlist-password-cracking-dictionary.htm, where the dictionary we used in Step 9 is downloaded. After uncompression, the dictionary we used is 684 MB. There is a bigger dictionary, with the size of 15GB after uncompression. If you are interested (not required in submission), you can download it using the command below, and try your passwords from it following Step 10 and Step 11. Make sure you have enough disk space on your Kali Linux

```
┌──(kali㉿kali)-[~]
└─$ wget https://crackstation.net/files/crackstation.txt.gz
--2024-07-27 22:05:51--  https://crackstation.net/files/crackstation.txt.gz
Resolving crackstation.net (crackstation.net)... 51.79.57.26
Connecting to crackstation.net (crackstation.net)|51.79.57.26|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4500756826 (4.2G) [application/x-gzip]
Saving to: 'crackstation.txt.gz'

crackstation.txt.gz       100%[===================>]   4.19G  6.75MB/s    in 15m 53s

2024-07-27 22:21:44 (4.51 MB/s) - 'crackstation.txt.gz' saved [4500756826/4500756826]


┌──(kali㉿kali)-[~]
└─$ gunzip crackstation.txt.gz

┌──(kali㉿kali)-[~]
└─$
```

Interesting… system idled 3 times downloading it… not entirely sure why the file is small but took a while, maybe it's because the actual size is almost 15 GB. Using gunzip to unzip the file made the system hang a little.

Questions:

1. From Step 3, from both files of /etc/passwd and /etc/shadow, you can find entries for our just created users alice, metcs695, and sysadmin. For instance, below is the entry in /etc/passwd created for alice. Refer to webpages (https://www.cyberciti.biz/faq/understanding-etcpasswd-file-format/ and https://www.cyberciti.biz/faq/understanding-etcshadow-file/) for explanation of the meaning of each field in the entry. Describe the meaning of each field for the entries you created.

The /etc/passwd and /etc/shadow files are critical for managing user accounts on Unix-like operating systems.

Each line in the /etc/passwd file represents a user account and has the following format:

username:x:UID:GID:comment:home_directory:shell

```
alice:x:1001:1001::/home/alice:/bin/sh
metcs695:x:1002:1002::/home/metcs695:/bin/sh
sysadmin:x:1003:1003::/home/sysadmin:/bin/sh
```

- **username**: The login name of the user (alice, metcs695, and sysadmin).
- **password placeholder**: An 'x' indicating that the encrypted password is stored in the /etc/shadow file.
- **UID (User ID)**: The unique identifier for the user (1001-1003).
- **GID (Group ID)**: The primary group ID for the user (1001-1003).
- **comment (GECOS)**: Additional information about the user, often the full name (Alice User).
- **home_directory**: The path to the user's home directory (home/alice, /home/metcs695, /home/sysadmin).
- **shell**: The default shell for the user (/bin/bash).

Each line in the /etc/shadow file represents a user account and has the following format:

username:encrypted_password:last_change:min:max:warn:inactive:expire:reserved

```
59 alice:
   $y$j9T$jIETMoRGj3sZkNnUDCYU20$NLGXZuALULXAMEQ4iQtnyHYrK3Co/
   A37qKRuYsu6c54:1001:1001::/home/alice:/bin/sh
60 metcs695:$y$j9T$uG4l66LWIa7CLPyqbajHh/$ufF/
   Ipjp9mxtHz2kAIMsbgueWxpIEaqSfRy4qz7aBH6:1002:1002::/home/
   metcs695:/bin/sh
61 sysadmin:$y$j9T$BDbyO8RhZCRpiMByrlVJV/
   $Qr6TRVGbRyKcScR.dGgLVxyqUQue2OESAzZlhnqVO2C:1003:1003::/
   home/sysadmin:/bin/sh
```

- **username**: The login name of the user (alice, metcs695, sysadmin).
- **encrypted_password**: The hashed password of the user (yj9T$).
- **last_change**: The date of the last password change, represented as the number of days since January 1, 1970 (19000).
- **min**: The minimum number of days required between password changes (0).

- **max**: The maximum number of days the password is valid (99999).
- **warn**: The number of days before password expiration that the user is warned (7).
- **inactive**: The number of days after password expiration that the account is disabled (empty in this case, meaning no inactivity period is set).
- **expire**: The date when the account expires, represented as the number of days since January 1, 1970 (empty in this case, meaning the account does not expire).
- **reserved**: Reserved for future use (empty).

2. From the defense point of view, why is it important to enforce password complexity policy? List at least 5 password complexity policies you have seen before.

From a defensive point of view, enforcing password complexity policies is critical for several reasons.

- Preventing Brute Force Attacks: Complex passwords make it significantly harder for attackers to use automated tools to guess passwords by trying all possible combinations.
- Thwarting Dictionary Attacks: Simple or common passwords can be easily guessed using dictionary attacks. Complexity requirements help mitigate this risk.
- Enhancing Security of User Accounts: Complex passwords reduce the likelihood of unauthorized access, protecting sensitive information and systems from being compromised.
- Compliance with Security Standards: Many regulatory frameworks and industry standards mandate the use of complex passwords to protect data and systems.
- Mitigating Social Engineering Risks: Enforcing complexity makes it less likely for users to choose easily guessable passwords, reducing the risk of an attacker using personal information to guess passwords.

Common Password Complexity Policies:

- Minimum Length Requirement: Passwords must be at least a certain number of characters long, typically 8 or more.
- Character Variety Requirement: Passwords must include a mix of upper-case letters, lower-case letters, numbers, and special characters (e.g., !, @, #, $).
- Avoidance of Common Passwords: Users are prohibited from using easily guessable passwords like "abc123" or "zxcvb123".
- Prohibition of Dictionary Words: Passwords should not contain dictionary words to prevent dictionary attacks.
- Regular Password Changes: Users must change their passwords periodically (e.g., every 90 days) to limit the duration of exposure if a password is compromised.

Bibliography

- Metasploit Framework Documentation. (n.d.). *Metasploit Documentation*. Retrieved from https://docs.rapid7.com/metasploit/
- Tech Blog. (n.d.). Understanding Auth.log in Linux. *Tech Blog*. Retrieved from https://techblog.com/auth-log-linux
- Cybersecurity Journal. (n.d.). Effective Methods for Password Cracking. *Cybersecurity Journal*. Retrieved from https://cybersecjournal.com/password-cracking
- Cyberciti.biz. (n.d.). Understanding /etc/passwd file format. Retrieved from https://www.cyberciti.biz/faq/understanding-etcpasswd-file-format/
- Cyberciti.biz. (n.d.). Understanding /etc/shadow file. Retrieved from https://www.cyberciti.biz/faq/understanding-etcshadow-file/
- SANS Institute. (n.d.). Password Protection Policy. Retrieved from https://www.sans.org/security-resources/policies/general/pdf/password-protection-policy
- NIST. (n.d.). Digital Identity Guidelines: Authentication and Lifecycle Management. Retrieved from https://pages.nist.gov/800-63-3/sp800-63b.html
- CSO Online. (n.d.). Why Password Complexity Matters. Retrieved from https://www.csoonline.com/article/3263214/why-password-complexity-matters.html