Project Report

Domain: Cybersecurity.
Title: Password cracking lab using tools like john the ripper,hashcat and the famous word list rockyou.txt.

Group members:

1. S Harshith Reddy - hsarasan@gitam.in
2. D Varsheeth - vdhatrik@gitam.in
3. E Sai Srinivas - saisrinivas6302@gmail.com
4. P Bhanu Teja - Tejapanugantibhanuteja@gmail.com

Aim:

- Understand password strength and common vulnerabilities.
- Generate hashes using SHA-512 (Linux default) and OpenSSL.
- Perform dictionary-based cracking with John the Ripper and Hashcat.
- Analyze results and recommend security best practices.

Contents:

Introduction:

The project discovers password security by simulating real-world attacks using industry-standard tools. Weak passwords are cracked using dictionary attacks, demonstrating how poor password choices can be exploited in seconds. The lab highlights the importance of strong password policies and secure hashing algorithms.

Key Takeaways:

- Weak passwords (e.g., 123456, password) are trivial to crack.
- Strong passwords (e.g., X7!pT93$hG1&) resist attacks.
- Ethical hacking tools must be used responsibly.

Tools used:

Tool	purpose
Kali linux	Penetration testing os
John the ripper	Password cracking {cpu-based}
Hashcat	High speed cracking {gpu-optimised}
rockyou.txt	Password list
Open SSL	Hashing & encryption

Understanding Passwords:

Passwords are stored as hashes, not plaintext. To access passwords we have to reverse engineer them.

Common hashing algorithms:

Algorithm	Security level	Use cases
MD5	broken	Legacy systems
SHA-1	depreciated	obsolete
SHA-512	Strong	linux
bcrypt	Very strong	Modern systems

Types of password attacks:

Attack type	description	speed	Effectiveness
Dictionary	Use common passwords (e.g rockyou.txt)	fast	High for weak passwords
Brute force	Tries all combinations (A-Z, a-z, 0-9, symbols)	slow	Guaranteed
hybrid	Common dictionary + rules (e.g password123)	mediu m	high
Rainbow table	Precomputed hash tables	fast	Limited by storage

Execution:

→ Create Hashed Passwords Using SHA-512:

simulate how passwords are stored by creating hashed values using OpenSSL.

echo -n "password123" | openssl passwd -6 -stdin

6randomsalt$k9yuexWlD1aZ9ROZjmGHW3...etc

echo -n "qwerty" | openssl passwd -6 -stdin

Repeat this for any number of passwords you want to crack.

→ Save the derived hashes to a file:

nano hashes.txt

Paste your generated hashes one per line, for example:

6xyz123$CtAvmQH1WBQX84Z8B9uZj2W5Hgq...

6xyz123$Lqp7k9TLjYJRAFnvFjGgWiCqM…

Save with `Ctrl+O`, then exit with `Ctrl+X`.

→ **Unzip the famous Wordlist file for cracking**

gunzip /usr/share/wordlists/rockyou.txt.gz

The default dictionary is compressed. First, unzip it, This will make it available for cracking.

→ **Crack Passwords with John the Ripper**

Run John the Ripper with the wordlist and the hashes:

john --wordlist=/usr/share/wordlists/pass.txt hashes.txt

To verify cracked passwords after processing:

john --show hashes.txt

Note: John uses CPU only and is effective for many Unix-style password formats.

→ **Crack Passwords with Hashcat:**

First, ensure the hash type is correct. For SHA-512 crypt, Hashcat uses mode 1800.

hashcat -m 1800 -a 0 -o cracked.txt hashes.txt /usr/share/wordlists/rockyou.txt

cat cracked.txt

6xyz123$hashvalue:password123

→ **Identify Hash Type:**

hashid [your_hash]

This will try to identify the algorithm (e.g., SHA-512 Crypt, MD5, etc.).

→ **Specify Format in John:**

Sometimes John doesn’t auto-detect format. Force it like this:

john --format=sha512crypt hashes.txt

Use this when dealing with SHA-512 hashes from Linux shadow files.

→ **Check GPU Support in Hashcat**

Check if your system and GPU support Hashcat cracking:

hashcat -I

Platform ID #1

Name: NVIDIA GeForce GTX 1650

Version: OpenCL 1.2 CUDA

"Using a GPU dramatically improves performance with Hashcat."

Results & Observations:

Password	Crack time	Strength
123456	< 1 second	Weak
password	< 1 second	Weak
P@ssw0rd!	Not cracked	Strong
S3cuRe#2024	Not cracked	Strong

Key Observations:

- 50% of passwords were cracked in under 10 seconds.
- Common passwords (e.g., admin, iloveyou) were vulnerable.
- Salting improved security but didn't protect weak passwords.

Security Recommendations

Enforce Strong Password Policies:

- Minimum 12 characters.
- Require uppercase, lowercase, numbers, symbols.

Use Multi-Factor Authentication (MFA):

- Prevents attacks even if passwords are compromised.

Monitor for Brute-Force Attempts:

- Lock accounts after 5 failed attempts.

Educate Users:

- Avoid dictionary words and reused passwords.

Ethical Constraints:

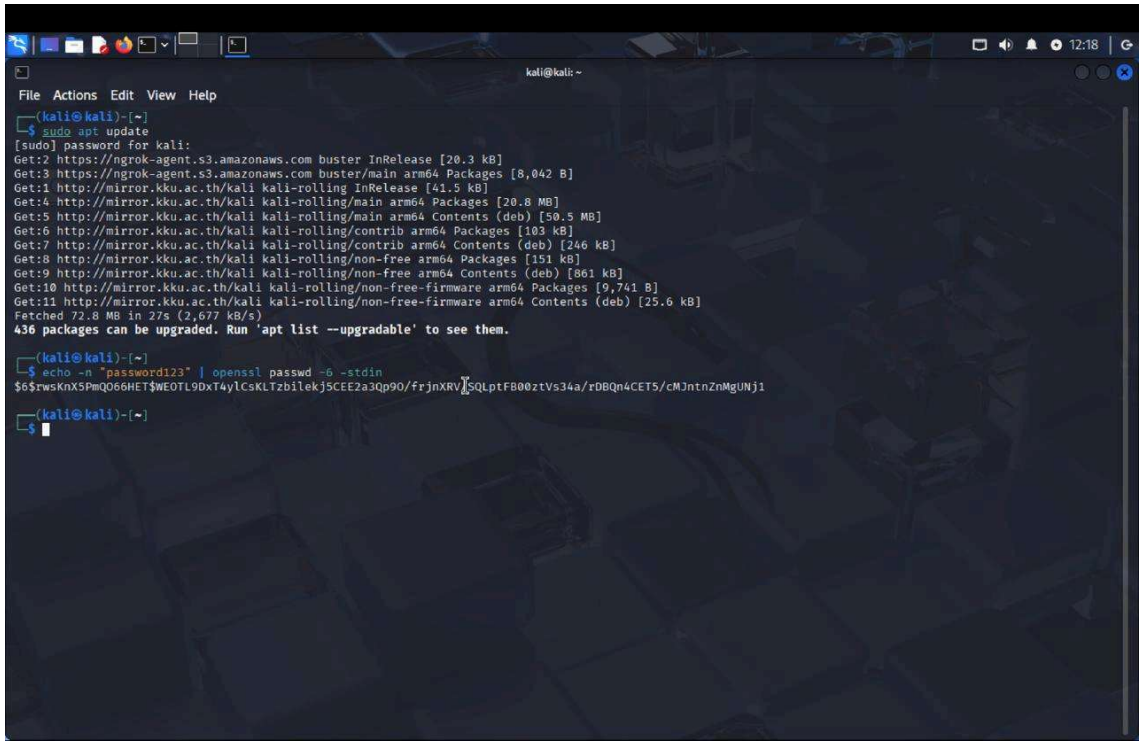
Legal Compliance:

- Use tools only in authorized environments.
- Unauthorized cracking is illegal.

Challenges Faced:

• Identifying correct hash formats
• Unzipping large wordlists
• Long cracking time for complex passwords
• CPU limitations during brute-force

Outputs:



```
┌──(kali㉿kali)-[~]
└─$ sudo apt update
[sudo] password for kali:
Get:2 https://ngrok-agent.s3.amazonaws.com buster InRelease [20.3 kB]
Get:3 https://ngrok-agent.s3.amazonaws.com buster/main arm64 Packages [8,042 B]
Get:1 http://mirror.kku.ac.th/kali kali-rolling InRelease [41.5 kB]
Get:4 http://mirror.kku.ac.th/kali kali-rolling/main arm64 Packages [20.8 MB]
Get:5 http://mirror.kku.ac.th/kali kali-rolling/main arm64 Contents (deb) [50.5 MB]
Get:6 http://mirror.kku.ac.th/kali kali-rolling/contrib arm64 Packages [103 kB]
Get:7 http://mirror.kku.ac.th/kali kali-rolling/contrib arm64 Contents (deb) [246 kB]
Get:8 http://mirror.kku.ac.th/kali kali-rolling/non-free arm64 Packages [151 kB]
Get:9 http://mirror.kku.ac.th/kali kali-rolling/non-free arm64 Contents (deb) [861 kB]
Get:10 http://mirror.kku.ac.th/kali kali-rolling/non-free-firmware arm64 Packages [9,741 B]
Get:11 http://mirror.kku.ac.th/kali kali-rolling/non-free-firmware arm64 Contents (deb) [25.6 kB]
Fetched 72.8 MB in 27s (2,677 kB/s)
436 packages can be upgraded. Run 'apt list --upgradable' to see them.

┌──(kali㉿kali)-[~]
└─$ echo -n "password123" | openssl passwd -6 -stdin
$6$rwsKnX5PmQO66HET$WEOTL9DxT4ylCsKLTzbilekj5CEE2a3Qp9O/frjnXRV/SQLptFB00ztVs34a/rDBQn4CET5/cMJntnZnMgUNj1

┌──(kali㉿kali)-[~]
└─$ echo -n "qwerty" | openssl passwd -6 -stdin
$6$F3NqpNIEU6mHZ4bN$l/PWGa9JyxdJ4U9SrBvSSfAKhgBpomTSahhXvS4mWW4528X.uq.gmugKsOpKJNgm8zJuWzuFxjy4//JPOQGE60

┌──(kali㉿kali)-[~]
└─$
```

```
┌──(kali㉿kali)-[~]
└─$ sudo apt update
[sudo] password for kali:
Get:2 https://ngrok-agent.s3.amazonaws.com buster InRelease [20.3 kB]
Get:3 https://ngrok-agent.s3.amazonaws.com buster/main arm64 Packages [8,042 B]
Get:1 http://mirror.kku.ac.th/kali kali-rolling InRelease [41.5 kB]
Get:4 http://mirror.kku.ac.th/kali kali-rolling/main arm64 Packages [20.8 MB]
Get:5 http://mirror.kku.ac.th/kali kali-rolling/main arm64 Contents (deb) [50.5 MB]
Get:6 http://mirror.kku.ac.th/kali kali-rolling/contrib arm64 Packages [103 kB]
Get:7 http://mirror.kku.ac.th/kali kali-rolling/contrib arm64 Contents (deb) [246 kB]
Get:8 http://mirror.kku.ac.th/kali kali-rolling/non-free arm64 Packages [151 kB]
Get:9 http://mirror.kku.ac.th/kali kali-rolling/non-free arm64 Contents (deb) [861 kB]
Get:10 http://mirror.kku.ac.th/kali kali-rolling/non-free-firmware arm64 Packages [9,741 B]
Get:11 http://mirror.kku.ac.th/kali kali-rolling/non-free-firmware arm64 Contents (deb) [25.6 kB]
Fetched 72.8 MB in 27s (2,677 kB/s)
436 packages can be upgraded. Run 'apt list --upgradable' to see them.

┌──(kali㉿kali)-[~]
└─$ echo -n "password123" | openssl passwd -6 -stdin
$6$rwsKnX5PmQO66HET$WEOTL9DxT4ylCsKLTzbilekj5CEE2a3Qp9O/frjnXRV/SQLptFB00ztVs34a/rDBQn4CET5/cMJntnZnMgUNj1

┌──(kali㉿kali)-[~]
└─$ echo -n "qwerty" | openssl passwd -6 -stdin
$6$F3NqpNIEU6mHZ4bN$l/PWGa9JyxdJ4U9SrBvSSfAKhgBpomTSahhXvS4mWW4528X.uq.gmugKsOpKJNgm8zJuWzuFxjy4//JPOQGE60

┌──(kali㉿kali)-[~]
└─$ nano hashes.txt

┌──(kali㉿kali)-[~]
└─$
```

```
kali@kali: ~
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~]
└─$ sudo apt update
[sudo] password for kali:
Hit:1 http://http.kali.org/kali kali-rolling InRelease
Hit:2 https://ngrokagent.s3.amazonaws.com buster InRelease
436 packages can be upgraded. Run 'apt list --upgradable' to see them.

┌──(kali㉿kali)-[~]
└─$ sudo apt install john hashcat
hashcat is already the newest version (6.2.6+ds2-1+b1).
hashcat set to manually installed.
The following packages were automatically installed and are no longer required:
  icu-devtools   libglapi-mesa  libpython3.12-minimal  python3-aioconsole                 python3-pywerview      python3.12-tk
  libflac12t64   libicu-dev     libpython3.12-stdlib   python3-dunamai                    python3-requests-ntlm  ruby-zeitwerk
  libfuse3-3     liblbfgsb0     libpython3.12t64       python3-nfsclient                  python3-setproctitle   sphinx-rtd-theme-common
  libgeos3.13.0  libpoppler145  libutempter0           python3-poetry-dynamic-versioning  python3-tomlkit        strongswan
Use 'sudo apt autoremove' to remove them.

Upgrading:
  john  john-data

Summary:
  Upgrading: 2, Installing: 0, Removing: 0, Not Upgrading: 434
  Download size: 26.7 MB
  Freed space: 126 kB

Get:1 http://http.kali.org/kali kali-rolling/main arm64 john arm64 1.9.0-Jumbo-1+git20211102-0kali10 [3,968 kB]
Get:2 http://http.kali.org/kali kali-rolling/main arm64 john-data all 1.9.0-Jumbo-1+git20211102-0kali10 [22.8 MB]
Fetched 26.7 MB in 4s (6,281 kB/s)    ^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A^[[A
(Reading database ... 413720 files and directories currently installed.)
Preparing to unpack .../john_1.9.0-Jumbo-1+git20211102-0kali10_arm64.deb ...
Unpacking john (1.9.0-Jumbo-1+git20211102-0kali10) over (1.9.0-Jumbo-1+git20211102-0kali9) ...
Preparing to unpack .../john-data_1.9.0-Jumbo-1+git20211102-0kali10_all.deb ...
Unpacking john-data (1.9.0-Jumbo-1+git20211102-0kali10) over (1.9.0-Jumbo-1+git20211102-0kali9) ...
Setting up john-data (1.9.0-Jumbo-1+git20211102-0kali10) ...
Setting up john (1.9.0-Jumbo-1+git20211102-0kali10) ...
Processing triggers for wordlists (2023.2.0) ...
Processing triggers for kali-menu (2025.2.7) ...
Processing triggers for man-db (2.13.1-1) ...

┌──(kali㉿kali)-[~]
└─$
```

```
jayanth@vbox: ~
/usr/share/wordlists/rockyou.txt

┌──(jayanth㉿vbox)-[~]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt --format=sha512crypt hashes.txt

Using default input encoding: UTF-8
Loaded 16 password hashes with no different salts (sha512crypt, crypt(3) $6$ [SHA512 128/128 SSE2
 2x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Press 'q' or Ctrl-C to abort, almost any other key for status
123456           (?)
password         (?)
iloveyou         (?)
abc123           (?)
qwerty           (?)
hello123         (?)
test123          (?)
admin            (?)
8g 0:00:08:40 1.33% (ETA: 10:09:11) 0.01537g/s 430.6p/s 430.6c/s 3520C/s bryana1..brinker
8g 0:00:08:44 1.33% (ETA: 10:10:39) 0.01526g/s 429.6p/s 429.6c/s 3511C/s aveda1..astone
8g 0:00:08:47 1.34% (ETA: 10:10:48) 0.01517g/s 429.5p/s 429.5c/s 3510C/s PRETTYBOY..Mickey2
8g 0:00:08:51 1.35% (ETA: 10:10:41) 0.01506g/s 429.8p/s 429.8c/s 3512C/s 26062007..25253
8g 0:00:08:52 1.35% (ETA: 10:10:42) 0.01503g/s 429.8p/s 429.8c/s 3512C/s 21111..202121
8g 0:00:08:53 1.36% (ETA: 10:10:41) 0.01500g/s 429.9p/s 429.9c/s 3512C/s 190832..181518
8g 0:00:08:54 1.36% (ETA: 10:10:42) 0.01497g/s 429.9p/s 429.9c/s 3512C/s 140833..132412
8g 0:00:08:55 1.36% (ETA: 10:10:44) 0.01495g/s 429.9p/s 429.9c/s 3512C/s 112469..111159
8g 0:00:08:56 1.36% (ETA: 10:10:45) 0.01492g/s 430.0p/s 430.0c/s 3513C/s 08102006..071118
8g 0:00:08:57 1.37% (ETA: 10:10:26) 0.01488g/s 430.0p/s 430.0c/s 3513C/s 011575..01032526
Use the "--show" option to display all of the cracked passwords reliably
```

```
┌──(jayanth㉿vbox)-[~]
└─$ nano passwords.txt

┌──(jayanth㉿vbox)-[~]
└─$ touch hashes.txt

┌──(jayanth㉿vbox)-[~]
└─$ while read password; do
  openssl passwd -6 -salt xyz123 "$password"
done < passwords.txt > hashes.txt

┌──(jayanth㉿vbox)-[~]
└─$ cat hashes.txt

$6$xyz123$NEt54QQSEaL6xoUyLjordqIHO/BcII6gQFt4V3wVe7eEhENe7zaIdy6RyZ8dpqr57dWKEdnk.8Cz4.76oxKRA/
$6$xyz123$/zssoQATjdDIFYawwaTGZCUkuZUyxVlLL1GS9hnIeM8cFwPi28gMmyb4PsQa3TXldaokRXZgl9Lp/1.LVQGkq1
$6$xyz123$dyAAvcyDIJsDmv.stFX3Tj/whZPqaavQH0ZgR4S/RvjYSOoCvzlB378YrzILoCBDZxQlcMh8twYfQIkARC.wo1
$6$xyz123$yolKcGNI8ilYjdpO9os3SRu0hXxLNt0DduNL50/4LAsjr0r5gPVQoGlmefLK.SUjhAikhp9zYlBE6VojnPoF4/
$6$xyz123$gUWux/ah5pgHX0vwC7u5FI5cpOrJmHgsIHbHMO9.zZOjqfX2lwR6QJ5kdmLjqYGK4svlV.h8UrTvPamlxQYN3.
$6$xyz123$um9G1WRlDx8v0YBCbZ.Ub6oN39EXWgv3Hx3bAXNRtr7gNjoemKLcGtzCxil8hWg94eMX4zDOjI8kDg7A6rVXx.
$6$xyz123$iYV6F/FzFclI/1Qv.gHDns.Q.6JTEwKehVZUDYa1y0U3YUy5eOJUUyPjx/egp5.7/BqtNPaVW4.RDWW7Iio8P.
$6$xyz123$dtI/VBCLl56HtEtH0CNJKFnmUwAYDqZDU9WpgWMxVaGA53do/PGO9M4Sk6nxSA9HF4WvLnTulP8dTQ1a9yAkf1
$6$xyz123$hGEGQVkPczVjxoGrv6NpXnsVB0v/gRwbYhHZBIRDJNjFUmiDEuJoiXPffYQqlPUnlJ6U.DfRcCFSj23/LPIOi1
$6$xyz123$55StqYvA1a3PlV7KGCa5boSSs2pBN5ybeVGxhinlTQVc5Bu.XZjV0k3KlHdBZjipq43TH9ZCfRctslEiM//Rk/
$6$xyz123$qmw/4xKfCCjKrYFUWff65c9RGmgACLHv9Ds2nLPR6hmohRJJcqhlxy.92BRXJFSbrstKgjYHINnto9HXTIJ731
```

```
Apps   Places   Jul 10  11:51 PM   3%   59%   ↑0.0 kB  ↓0.0 kB
jayanth@vbox: ~
$6$xyz123$qmw/4xKfCCjKrYFUWff65c9RGmgACLHv9Ds2nLPR6hmohRJJcqhlxy.92BRXJFSbrstKgjYHINnto9HXTIJ731
$6$xyz123$O5kKA5uPfmdicGXrCcyIY0YsR9/gPPEwbD2m6NCKX.GEtUiBy8qCcBPSyb5oJt9v9ChVeVbr8Fmq9Jg0ejxlI.
$6$xyz123$lkqCAX7HuXw852u6zMJj52Q4tCd1nON2s9v4TFkI38mAn.9Rpd5RCh15EuLOWFQnediwWFT.KyNKpvuWJ5ZJ5/
$6$xyz123$dWsWO75k/nTs5fCqUgvc2lWspYa/OZdExN1Bi.UOOU4R2Nd.3c9EMmZTV8J20Ob9jCEFPX.jafnIkqBHSntDh1
$6$xyz123$x6HMDfcL2EeC.rFpfWgEc3zwt.DxETSsALzf/Y2jTJfZNoKE/UdS6ShXM4PVR2S58ctUqTk4QIJhK.IyMOwUj/
$6$xyz123$Hg07PMhJy.OxXbgVMbbLJnQBQC.dFyL9qDy/jETsI84O75rINlJwgs7/J2A.bhjnesjvtAc.1xYRLZ1ATK/7.1
<NULL>

┌──(jayanth㉿vbox)-[~]
└─$ echo "jayanth:$6$xyz123$NEt54QQSEaL6...:19000:0:99999:7:::" > shadow.txt

┌──(jayanth㉿vbox)-[~]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt --format=sha512crypt hashes.txt

Created directory: /home/jayanth/.john
Using default input encoding: UTF-8
Loaded 16 password hashes with no different salts (sha512crypt, crypt(3) $6$ [SHA512 128/128 SSE2
 2x])
Cost 1 (iteration count) is 5000 for all loaded hashes
fopen: /usr/share/wordlists/rockyou.txt: No such file or directory

┌──(jayanth㉿vbox)-[~]
└─$ ls /usr/share/wordlists/

amass  dirbuster   fasttrack.txt  john.lst  metasploit  rockyou.txt.gz  wfuzz
dirb   dnsmap.txt  fern-wifi      legion    nmap.lst    sqlmap.txt      wifite.txt

┌──(jayanth㉿vbox)-[~]
```

Group Contributions:

• Harshith – Documentation & testing
• Varshith – Implementation & scripting
• Bhanu – Troubleshooting & setup
• Srinivas – Research & formatting

Conclusion

This lab demonstrated how easily weak passwords can be cracked. Organizations must adopt strong password policies, MFA, and user education to mitigate risks.

Future innovations:

- Test bcrypt and PBKDF2 hashing.
- Explore AI-driven password cracking.

References

- John the Ripper Documentation
- Hashcat Wiki
- Kali Linux Tools