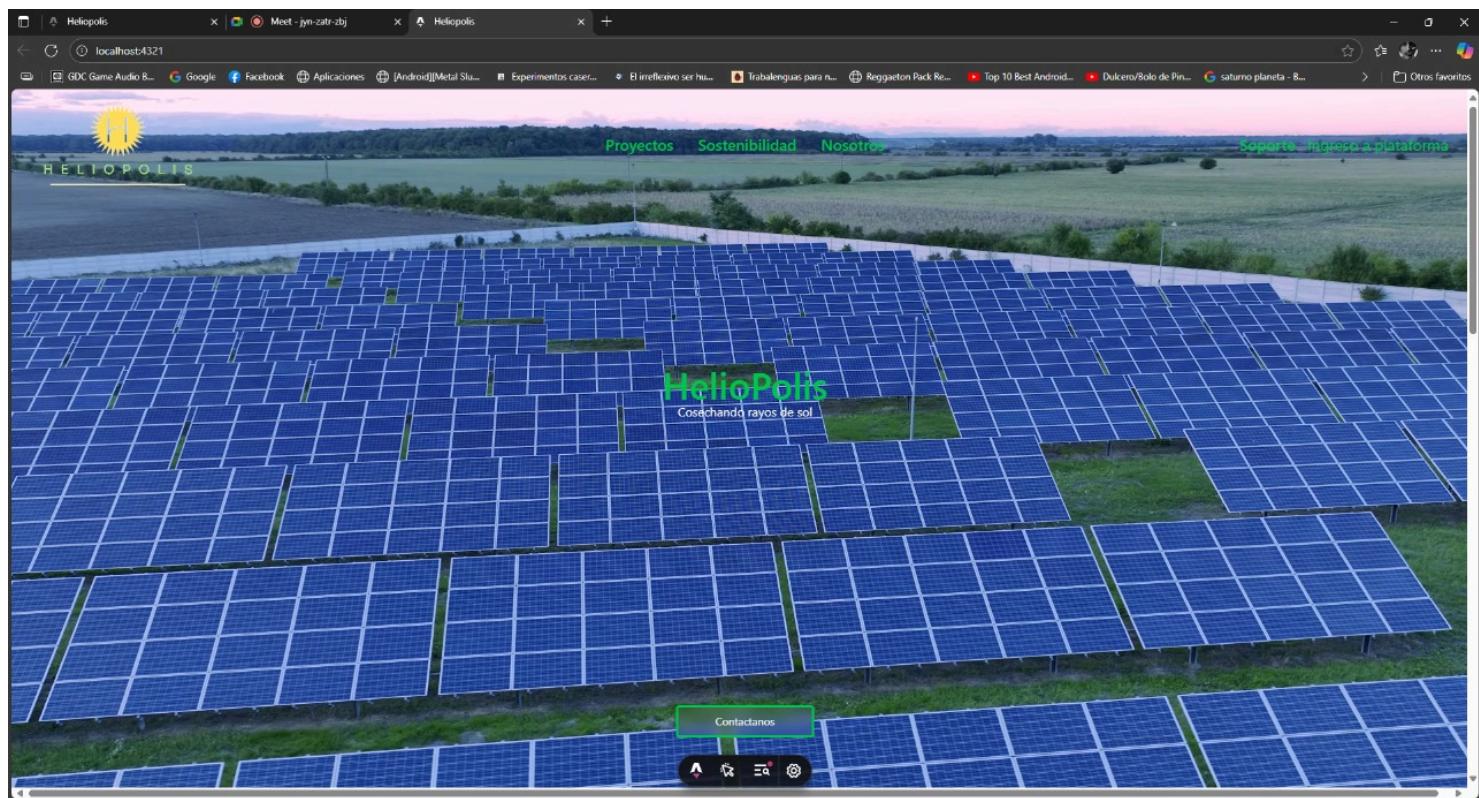


Contexto del proyecto:.....	3
Justificación:.....	3
1. Protección de Infraestructura Crítica:.....	3
2. Confidencialidad y seguridad de los datos:.....	3
3. Prevención de ciberataques avanzados (APT, ransomware, etc.):.....	4
4. Cumplimiento normativo y reputación empresarial:.....	4
Objetivos generales y específicos:.....	4
Objetivo General:.....	4
Objetivos Específicos:.....	4
Vulnerabilidades y controles aplicados:.....	5
❖ Contraseñas en texto plano:.....	5
➤ Control aplicado:.....	5
❖ Sin HTTPS (solo HTTP):.....	5
➤ Control aplicado:.....	5
❖ No hay tokens con firmas o sesiones:.....	5
➤ Control aplicado:.....	5
❖ las rutas sensibles no están protegidas:.....	5
➤ Control aplicado:.....	5
<b>Pruebas de Seguridad y Evaluación Técnica:.....</b>	<b>6</b>
● Herramientas empleadas:.....	6
● Resultados clave:.....	6
1. Contraseñas en texto plano.....	13
2. Sin HTTPS (solo HTTP).....	13
3. No hay tokens o sesiones.....	13
4. las rutas sensibles no están protegidas.....	14
1) Firmar tokens:.....	14
Poblema sin JWT_SECRET (inseguro):.....	14
1. Uso de dotenv (.env) para ocultar secretos: Se movió la clave secreta JWT fuera del código y se almacenó en un archivo .env.	14
<b>Monitoreo y logs:.....</b>	<b>27</b>
<b>Vista de la plataforma:.....</b>	<b>30</b>
Landing-page:.....	30
Login-page:.....	31
Plataforma gestión:.....	32



# HELIOPOLIS

C O S E C H A N D O   R A Y O S   D E   S O L



Heliópolis es una empresa dedicada a la distribución de energía solar, a través de una gestión inteligente de consumo energético, supliendo las variables necesidades de nuestros clientes. Impulsamos la sostenibilidad con tecnología y compromiso ambiental.

## Contexto del proyecto:

HELIÓPOLIS es una empresa innovadora dedicada a la comercialización de energía limpia y renovable, atendiendo principalmente a EPM, CHEC y ELECTRO CARIBE. Este informe desarrolla el proyecto AEGIS TESSERACT, orientado a la planificación e implementación de un SGSI siguiendo ISO/IEC 27001 e incluyendo controles ajustados al sector energético colombiano. El foco principal es asegurar la confidencialidad, integridad y disponibilidad de todos los activos informáticos y estratégicos, mitigando riesgos asociados a la operación crítica y cumplimiento normativo.

## Justificación:

En el contexto actual de transformación digital y transición energética, las empresas que ofrecen distribución de energía solar y servicios de gestión energética enfrentan una creciente exposición a amenazas cibernéticas. Estos sectores dependen cada vez más de infraestructuras digitales, sensores IoT, sistemas SCADA, redes inteligentes (smart grids) y plataformas de monitoreo remoto, lo que amplía notablemente su superficie de ataque.

La implementación de **estrategias de ciberseguridad sólidas** en este tipo de organizaciones tiene un impacto crítico en varios niveles:

### 1. Protección de Infraestructura Crítica:

La generación, almacenamiento y distribución de energía solar se apoya en infraestructuras que, si se ven comprometidas, podrían causar interrupciones en el suministro, afectando a clientes residenciales, comerciales e industriales. Un ciberataque dirigido podría provocar sabotajes operativos, apagones o incluso dañar físicamente equipos mediante sobrecargas o manipulaciones de sistemas automatizados.

### 2. Confidencialidad y seguridad de los datos:

Las empresas de gestión energética recopilan y analizan grandes volúmenes de datos de consumo, eficiencia y comportamiento de los usuarios. La falta de medidas de ciberseguridad adecuadas podría derivar en **violaciones de privacidad**, robo de datos sensibles, o suplantación de identidad digital, afectando la confianza de los clientes y generando consecuencias legales y reputacionales.

### 3. Prevención de ciberataques avanzados (APT, ransomware, etc.):

Dada su importancia estratégica, estas organizaciones son objetivos potenciales de ataques persistentes avanzados (APT), ataques con motivación financiera (ransomware), o incluso de ciberterrorismo. Asegurar sus activos digitales es crucial para mantener la integridad, disponibilidad y continuidad del servicio.

### 4. Cumplimiento normativo y reputación empresarial:

La implementación de políticas de ciberseguridad demuestra el cumplimiento de estándares internacionales como ISO 27001, NERC CIP **North American Electric Reliability Corporation** (Corporación de Confiabilidad Eléctrica de Norteamérica) o la IEC 62443 **International Electrotechnical Commission** (Comisión Electrotécnica Internacional) y fortalecer la imagen de la empresa como un proveedor confiable. Esto no solo reduce el riesgo de sanciones legales, sino que también se traduce en una **ventaja competitiva** en un mercado cada vez más consciente de los riesgos tecnológicos.

#### Objetivos generales y específicos:

##### Objetivo General:

Implementar un plan integral de ciberseguridad que permita proteger los activos digitales, operativos y de información de una empresa dedicada a la distribución de energía solar y servicios de gestión energética, garantizando la continuidad del servicio, la protección de los datos y la resiliencia ante amenazas ciberneticas.

##### Objetivos Específicos:

1. **Analizar la infraestructura tecnológica actual** de la empresa para identificar vulnerabilidades en sus sistemas de información, redes, plataformas de monitoreo energético.
2. **Evaluar los riesgos ciberneticos** asociados al sector energético, considerando amenazas específicas como ataques a sistemas SCADA, interrupciones de servicio, ransomware y robo de datos.
3. **Diseñar e implementar políticas de seguridad informática**, incluyendo control de accesos, cifrado de datos, segmentación de red, y mecanismos de autenticación y monitoreo.

4. **Establecer un protocolo de respuesta ante incidentes ciberneticos**, que incluya detección, contención, recuperación y notificación ante posibles ataques.
5. **Fomentar una cultura de ciberseguridad** dentro de la organización, mediante programas de capacitación y concientización para empleados sobre buenas prácticas digitales.
6. **Garantizar el cumplimiento normativo** con estándares y marcos de referencia aplicables (como ISO 27001, NIST o IEC 62443), asegurando la conformidad con los requisitos legales y sectoriales.
7. **Evaluar y probar periódicamente** la efectividad del plan de ciberseguridad a través de auditorías, simulacros de ataque (pentesting) y análisis forense en caso de incidentes.

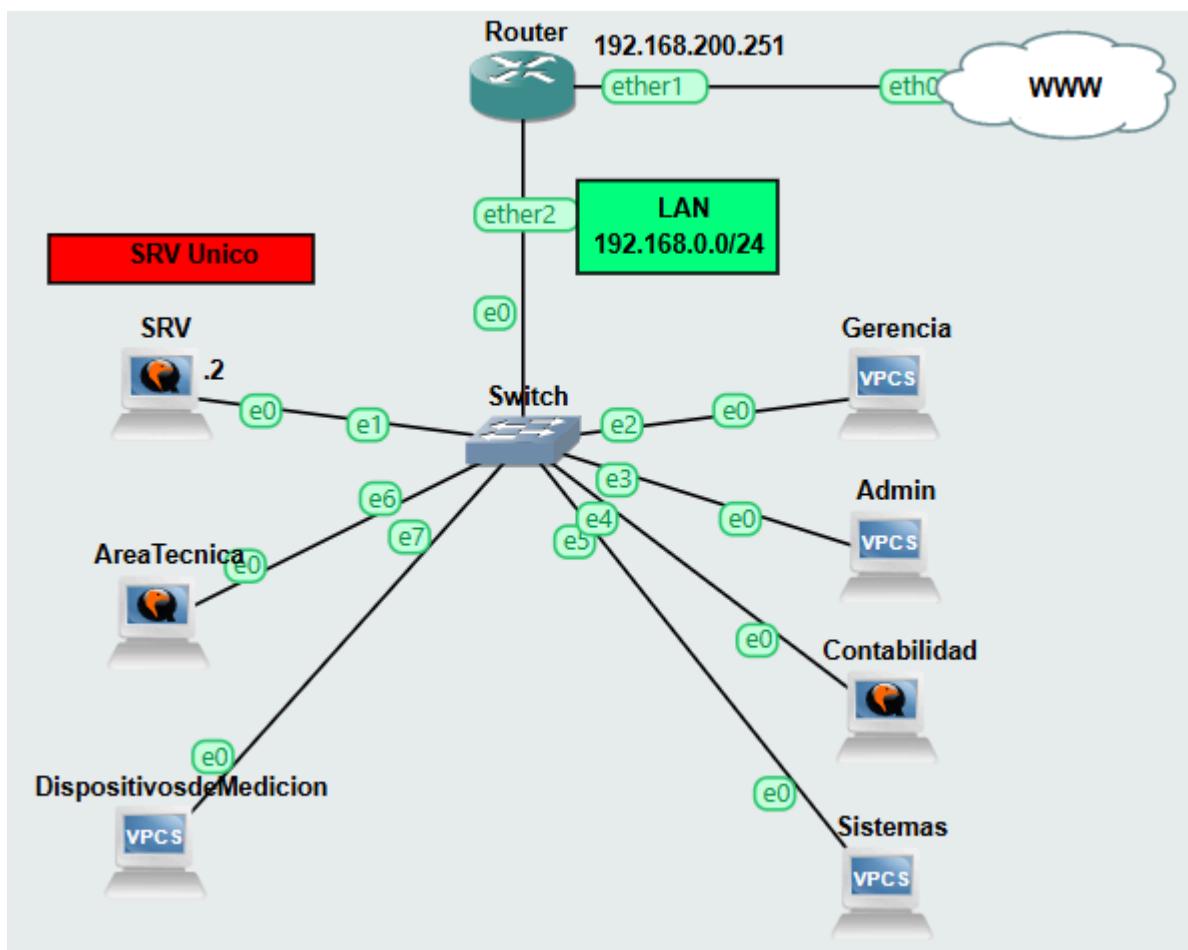
### Vulnerabilidades y controles aplicados:

- ❖ Contraseñas en texto plano:
  - *Control aplicado:*  
Firmar tokens.
- ❖ Sin HTTPS (solo HTTP):
  - *Control aplicado:*  
Instalación de certificados SSL/TLS (certbot).
- ❖ No hay tokens con firmas o sesiones:
  - *Control aplicado:*  
Uso de dotenv (.env) para ocultar secretos.
- ❖ las rutas sensibles no están protegidas:
  - *Control aplicado:*  
middleware para rutas protegidas.

# Pruebas de Seguridad y Evaluación Técnica:

- Herramientas empleadas:  
Nmap, Nessus, Zenmap, Wireshark.
- Resultados clave:
  - Servicios expuestos (HTTP, Telnet, FTP inseguros).
  - Vulnerabilidades clasificadas por criticidad (ver gráficos de Nessus/escaneos detallados).
  - Tráfico inseguro captado y documentado por Wireshark.

Para iniciar con el análisis de la empresa y revisando su topología actual nos damos cuenta que tiene muchos factores a mejorar ya que manejan una sola red y tienen todo en un solo servidor, se realizan las pruebas iniciales dentro de la LAN y observamos los siguientes resultados.



## Escaneo con NMAP a Red Vulnerable:

Comando nmap 192.168.0.0/24 para escanear toda la red.

```
(root㉿kali)-[~/home/kali]
# nmap 192.168.0.0/24
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-30 13:44 -05
Nmap scan report for 192.168.0.1
Host is up (0.0064s latency).
Not shown: 994 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
80/tcp    open  http
2000/tcp  open  cisco-sccp
8291/tcp  open  unknown

Nmap scan report for 192.168.0.2
Host is up (0.010s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
```

comando nmap 192.168.0.2 srv

```
2-virtualbox-amd64 [Corriendo] - Oracle VirtualBox
File  Actions  Edit  View  Help
root@kali: ~/home/kali

(root㉿kali)-[~/home/kali]
# nmap 192.168.0.2
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-30 13:37 -05
Nmap scan report for 192.168.0.2
Host is up (0.12s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 11.22 seconds

(root㉿kali)-[~/home/kali]
#
```

Sondeo de ping

nmap -sP 192.168.0.0/24

```
[root@kali]# nmap -sP 192.168.0.0/24
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-30 13:49 -05
Nmap scan report for 192.168.0.1
Host is up (0.0033s latency).
Nmap scan report for 192.168.0.2
Host is up (0.0065s latency).
Nmap scan report for 192.168.0.251
Host is up (0.0045s latency).
Nmap scan report for 192.168.0.252
Host is up (0.0045s latency).
Nmap scan report for 192.168.0.253
Host is up (0.0045s latency).
Nmap scan report for 192.168.0.254
Host is up (0.0043s latency).
Nmap done: 256 IP addresses (6 hosts up) scanned in 4.00 seconds
```

Se le realiza al servidor la verificación de servicio por puerto.

nmap -sV -sC 192.168.0.2

```
[root@kali]# nmap -sV -sC 192.168.0.2
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-30 13:59 -05
Nmap scan report for 192.168.0.2
Host is up (0.0068s latency).
Not shown: 998 closed tcp ports (reset) at 2025-07-30 13:49 -05
PORT      STATE SERVICE VERSION
22/tcp    open  ssh   OpenSSH 8.2p1 Ubuntu 4ubuntu0.13 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey: 1024 0b:da:64:05:bf:48:4b:10:28:7d:5c:98:6e:f1:1a:02 (RSA)
|_ 256 66:94:0a:9c:72:8f:ef:76:a4:01:74:2c:e5:4e:17:4b (ECDSA)
|_ 256 bf:d5:a7:02:d0:83:0b:40:e0:f0:ea:67:9d:aa:ba:64 (ED25519)
80/tcp    open  http  Apache httpd 2.4.41 ((Ubuntu))
|_http-title: Heliopolis
|_http-generator: Astro v5.11.1
|_http-server-header: Apache/2.4.41 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
Host script results:
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.49 seconds
```

Para realizar el escaneo de manera Sigilosa usamos el mismo comando y agregamos -sS

nmap -sS -sV -sC 192.168.0.0/24

```
root@kali: /home/kali
Screenshot taken
File Actions Edit View Help
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.49 seconds
└─(root㉿kali)-[~/home/kali]
└─# nmap -sS -sV -sC 192.168.0.2
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-30 14:03 -05
Nmap scan report for 192.168.0.2
Host is up (0.039s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.13 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 0b:da:64:05:bf:48:4b:10:28:7d:5c:98:6e:f1:1a:02 (RSA)
|   256 66:94:0a:9c:72:8f:ef:76:a4:01:74:2c:e5:4e:17:4b (ECDSA)
|_  256 bf:d5:a7:02:d0:83:0b:40:e0:f0:ea:67:9d:aa:ba:64 (ED25519)
80/tcp    open  http     Apache httpd 2.4.41 ((Ubuntu))
|_http-generator: Astro v5.11.1
|_http-title: Heliopolis
|_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 14.82 seconds
└─(root㉿kali)-[~/home/kali]
└─#
```

Para poder cambiar la mac virtualmente ingresamos spoof

```
nmap --spoof-mac 00:11:22:33:44 -sS -sV -sC 192.168.0.2
```

```
root@kali: /home/kali
File Actions Edit View Help
2 packets transmitted, 2 received, 0% packet loss, time 1019ms
rtt min/avg/max/mdev = 4.559/5.896/7.234/1.337 ms
└─(root㉿kali)-[~/home/kali]
└─# nmap --spoof-mac 00:11:22:33:44 -sS -sV -sC 192.168.0.2
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-31 09:29 -05
Spoofing MAC address 00:11:22:33:44:DB (Cimsys)
Nmap scan report for 192.168.0.2
Host is up (0.0061s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.13 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 0b:da:64:05:bf:48:4b:10:28:7d:5c:98:6e:f1:1a:02 (RSA)
|   256 66:94:0a:9c:72:8f:ef:76:a4:01:74:2c:e5:4e:17:4b (ECDSA)
|_  256 bf:d5:a7:02:d0:83:0b:40:e0:f0:ea:67:9d:aa:ba:64 (ED25519)
80/tcp    open  http     Apache httpd 2.4.41 ((Ubuntu))
|_http-title: Heliopolis
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-generator: Astro v5.11.1
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 7.47 seconds
└─(root㉿kali)-[~/home/kali]
└─#
```

## Conclusiones del escaneo nmap:

Vemos que todos los servicios están arriba en una sola máquina, esto es muy peligroso ya que un hacker con tan solo apoderarse de la máquina puede acceder a toda la información de los demás servicios.

PORT	STATE	SERVICE
21/tcp	open	ftp
22/tcp	open	ssh
23/tcp	open	telnet
80/tcp	open	http
2000/tcp	open	cisco-sccp
8291/tcp	open	unknown

## Procedemos a hacer un análisis de Nessus a la red vulnerable:

Sev	CVSS	VPR	EPS	Name	Family	Count	
LOW	2.1	2.2	0.0037	ICMP Timestamp Request Remote Date Disclosure	General	1	🔗
INFO	...	...	...	HTTP (Multiple Issues)	Web Servers	3	🔗
INFO	...	...	...	SSH (Multiple Issues)	General	2	🔗
INFO	...	...	...	SSH (Multiple Issues)	Misc.	2	🔗
INFO	...	...	...	SSH (Multiple Issues)	Service detection	2	🔗
INFO				Nessus SYN scanner	Port scanners	2	🔗
INFO				Service Detection	Service detection	2	🔗
INFO				Apache HTTP Server Version	Web Servers	1	🔗
INFO				Backported Security Patch Detection (WWW)	General	1	🔗
INFO				Common Platform Enumeration (CPE)	General	1	🔗
INFO				Device Type	General	1	🔗
INFO				Nessus Scan Information	Settings	1	🔗
INFO				OpenSSH Detection	Misc.	1	🔗
INFO				OS Fingerprints Detected	General	1	🔗

SRV Heliopolis Vulnerable / Plugin #10114

[← Back to Vulnerabilities](#)

Hosts 1 | Vulnerabilities 19 | History 1

LOW ICMP Timestamp Request Remote Date Disclosure

**Description**

The remote host answers to an ICMP timestamp request. This allows an attacker to know the date that is set on the targeted machine, which may assist an unauthenticated, remote attacker in defeating time-based authentication protocols.

Timestamps returned from machines running Windows Vista / 7 / 2008 / 2008 R2 are deliberately incorrect, but usually within 1000 seconds of the actual system time.

**Solution**

Filter out the ICMP timestamp requests (13), and the outgoing ICMP timestamp replies (14).

**Output**

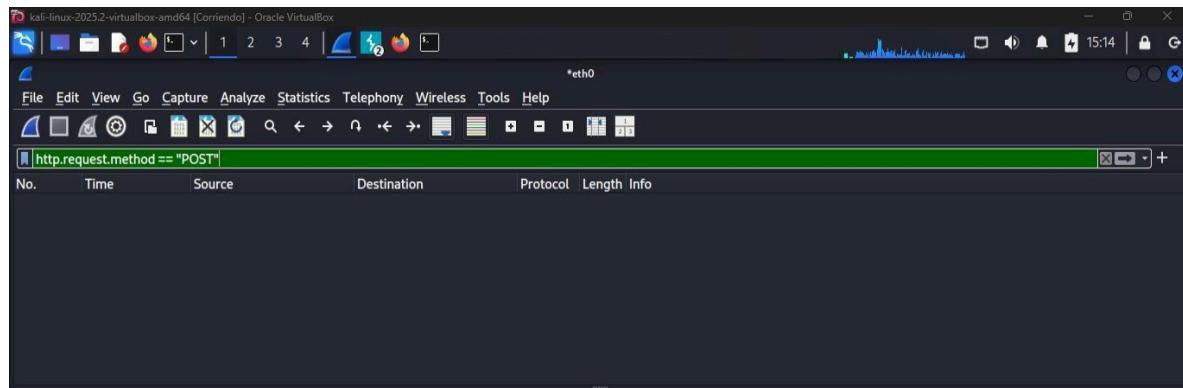
The difference between the local and remote clocks is -1 seconds.

To see debug logs, please visit individual host

Port ▲ Hosts

En cuanto a la red se encontraron vulnerabilidades pequeñas pero que dan mucha información a un posible atacante, ya que entrega muchos datos de valor, como versiones, servicios que hay activos, también que los puertos en los que están los servicios son los comunes. Estos errores a pesar de ser pequeños lo recomendable es corregirlos.

## Wireshark:



En cuanto al Wireshark se analizó que había una vulnerabilidad crítica en el aplicativo, cuando el servidor hacia una petición post, concretamente en la parte de conexión del login con el backend, lo que pasaba era que con el wireshark al servidor no estar protegido con certificaciones HTTP de cifrado (**HTTPS**) y tampoco tener implementaciones de hash en las credenciales, el atacante podría obtener los datos de ingreso de la plataforma ya que los paquetes que viajan por la red no están cifrados. Este tipo de vulnerabilidad permite ataques del tipo **Man-in-the-Middle (MITM)**.

Ya verificando la red y yéndonos un poco más a fondo de los procedimientos que tienen dentro de la empresa podemos observar que no tienen un servidor de archivos unificado y que todos los equipos tienen sus archivos de forma local sin realizar backup de ellos.

También podemos observar que no tienen un servidor de dominio para darles permisos a los equipos y manejar políticas de dominio tanto a usuarios como a equipos.

## **Recomendaciones realizadas y procedimientos de mejora:**

Como recomendación principal iniciamos con la segmentación de la red para que los diferentes procesos no tengan comunicación entre sí y así blindamos un poco a nivel de red. En cuanto al desarrollo de la página web, encontramos muchos huecos de seguridad los cuales se explicaran a continuación:

### **¿Por qué la implementación actual es insegura?**

#### **1. Contraseñas en texto plano**

- Si un atacante accede a la base de datos (por ejemplo con **SQL injection** o credenciales robadas), obtendrá las contraseñas tal como el usuario las escribió.
- Además, muchas personas usan las mismas contraseñas en múltiples sitios.

**Solución:** Hash con **bcrypt + salting automático**. Esto evita ataques como: Esta solución será aplicada concretamente en el backend, en el archivo server.js.

- Ataques de diccionario
- Fuerza bruta
- Rainbow tables (precomputados)

#### **2. Sin HTTPS (solo HTTP)**

- Toda la información viaja en **texto plano** (correo, contraseña, token).
- Un atacante en la misma red (Wi-Fi pública, LAN interna) podría hacer un ataque de **MITM (Man In The Middle)** y leer los datos.

**Solución:** Certificado SSL (Let's Encrypt o Nginx Reverse Proxy con HTTPS).

#### **3. No hay tokens o sesiones**

- No puedes controlar qué usuarios están logueados.
- No puedes invalidar un login anterior.
- Un atacante puede hacer **replays** o **secuestro de sesión** fácilmente.

**Solución:** JWT (con expiración + verificación + firma secreta).

#### 4. las rutas sensibles no están protegidas

- Cualquiera puede acceder a /Plataforma\_gestion o cualquier otra ruta sin estar logueado.

**Solución:** Middleware que valide el token antes de acceder.

Después de este listado, procederemos a corregir aspectos de la página web para volverla segura:

### CORRECCIONES:

#### 1) Firmar tokens:

##### Poblemas sin JWT\_SECRET (inseguro):

- El backend antes generaba tokens JWT sin firmarlos (o con una clave visible en el código), cualquier atacante podría:
  - Analizar un token válido desde el navegador (porque los JWT son fácilmente legibles, aunque estén codificados).
  - Modificar el contenido del token (por ejemplo, cambiar "rol": "usuario" a "rol": "admin").
  - Volver a enviarlo al servidor, que aceptaría el token sin saber que fue alterado.

**Resultado:** El atacante podría acceder a datos privados, robar cuentas, incluso escalar privilegios.

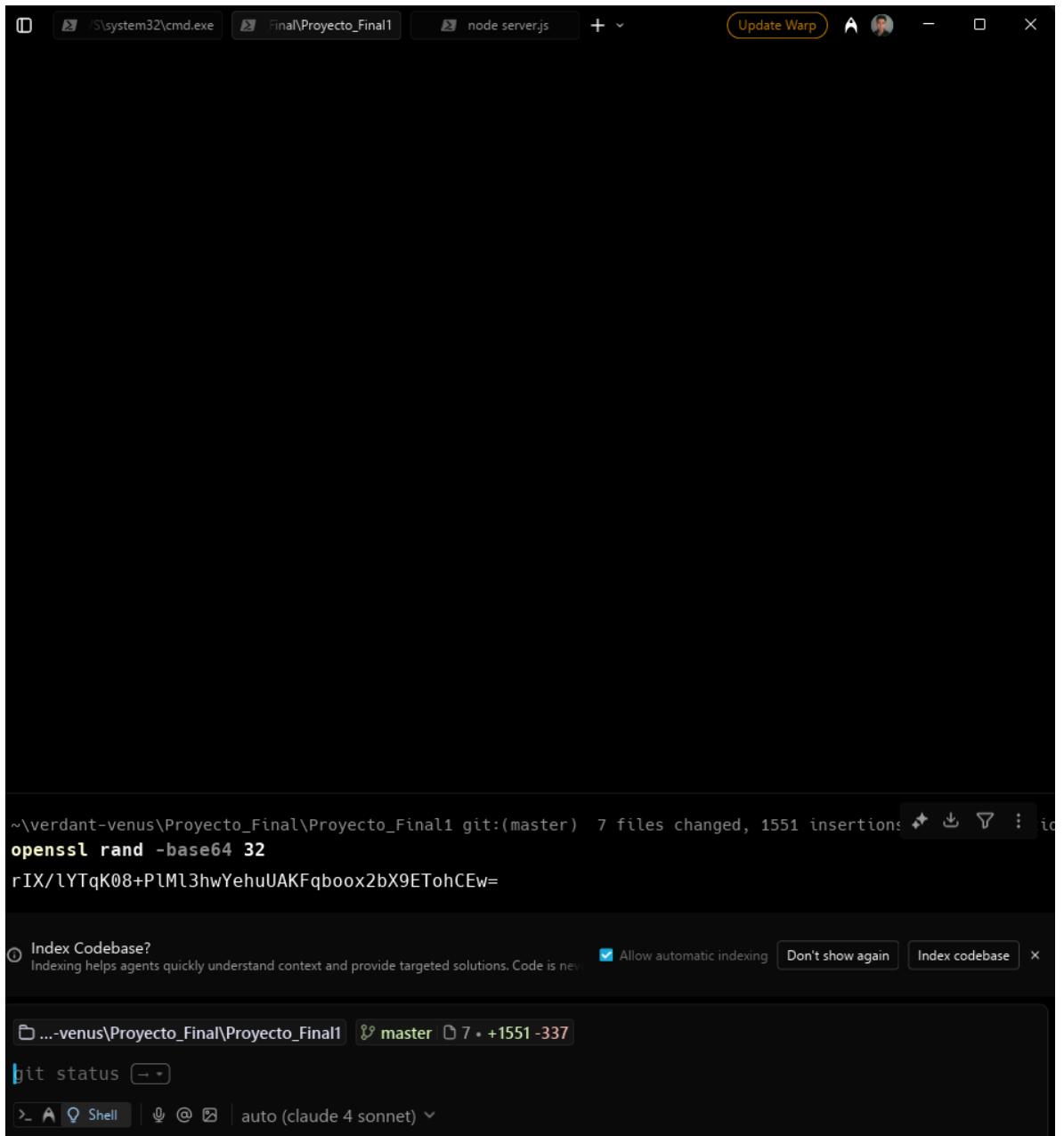
Para corregir este problema se hizo lo siguiente:

#### 1. Uso de dotenv (.env) para ocultar secretos:

Se movió la clave secreta JWT fuera del código y se almacenó en un archivo .env.

En ese archivo .env el cual está en la raíz, se creó una variable llamada **JWT\_SECRET** y se genera una clave utilizando el siguiente comando en bash: **openssl rand -base64 32**. De esta

forma se asegura de generar una firma segura para los tokens.



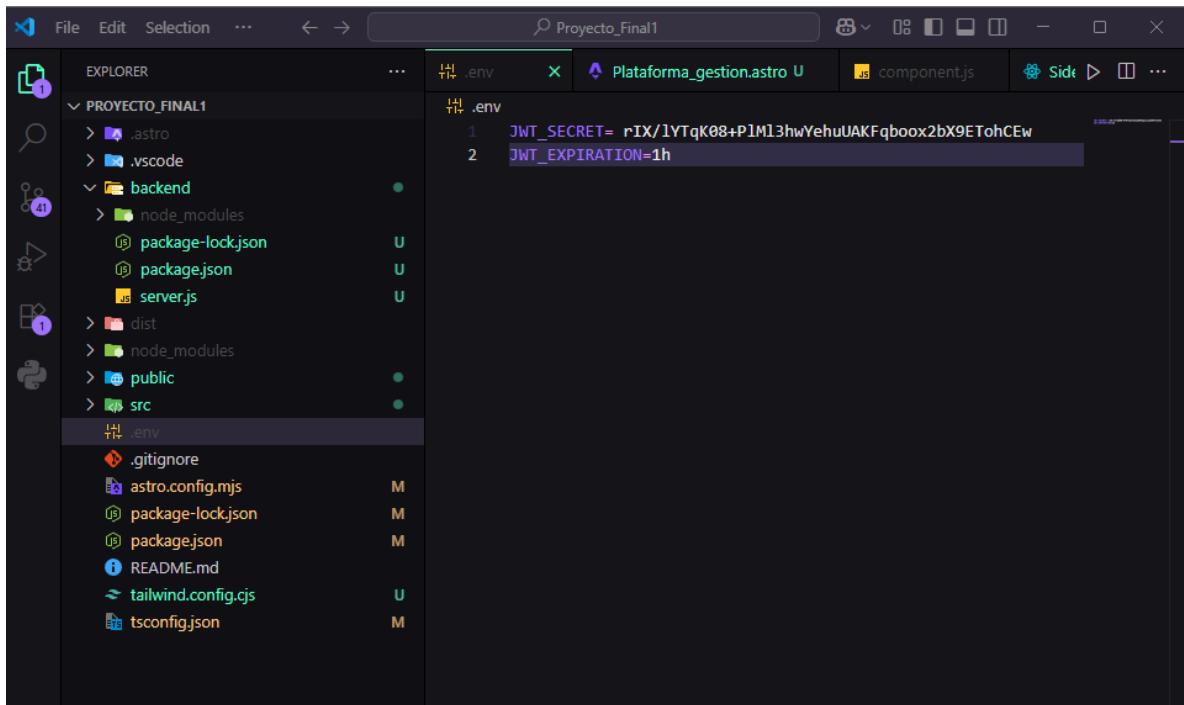
The screenshot shows a terminal window with several tabs at the top: 'S\system32\cmd.exe', 'Final\Proyecto\_Final1', and 'node server.js'. The main pane displays a command-line interface. The user has run the command 'openssl rand -base64 32' and is awaiting the output. The output is a long string of base64 encoded random bytes: 'rIX/lYTqK08+PlMl3hwYehuUAKFqboox2bX9ETohCEw='.

```
~\verdant-venus\Proyecto_Final\Proyecto_Final1 git:(master) 7 files changed, 1551 insertions: openssl rand -base64 32 rIX/lYTqK08+PlMl3hwYehuUAKFqboox2bX9ETohCEw=
```

Below the terminal, there is a UI element for indexing the codebase. It includes a tooltip 'Index Codebase?', a message 'Indexing helps agents quickly understand context and provide targeted solutions. Code is never shared.', an 'Allow automatic indexing' checkbox (which is checked), a 'Don't show again' button, and an 'Index codebase' button.

At the bottom of the terminal window, there is a status bar showing the path '...-venus\Proyecto\_Final\Proyecto\_Final1', the branch 'master', and statistics '7 +1551 -337'. Below the status bar are navigation icons for back, forward, search, and other shell functions.

También crearemos una variable que será el tiempo en que se expira:

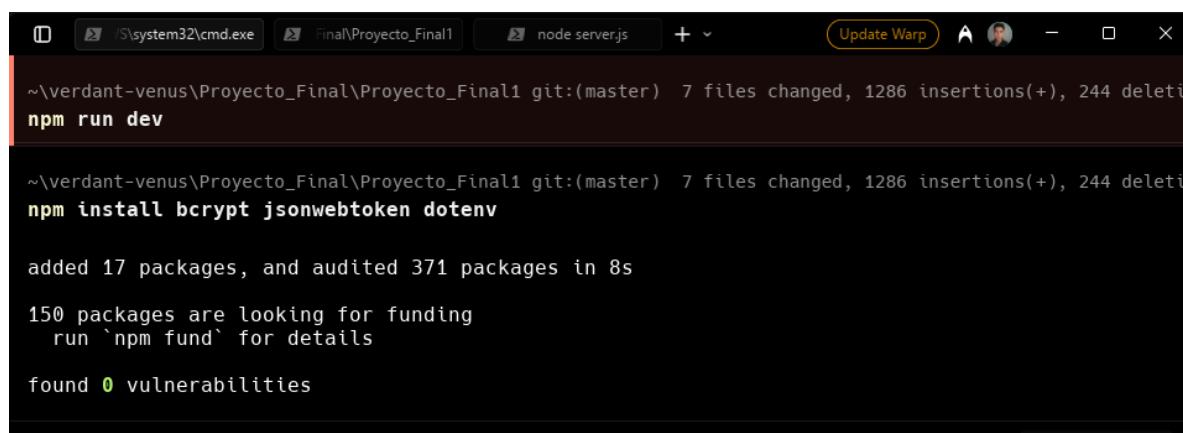


The screenshot shows the VS Code interface with the project structure on the left and the .env file content on the right. The .env file contains the following code:

```
JWT_SECRET= rIX/1YTqK08+P1Ml3hwYehuUAKFqboox2bX9ETohCEw
JWT_EXPIRATION=1h
```

## 2. Hash + salt + instalación de certificados SSL/TLS:

Para esta parte se harán varios pasos, es importante ya que una de las vulnerabilidades de la implementación actual es la forma en que se envían los datos para la verificación. Estos en el servidor vulnerable no tienen aplicados ningún tipo de hash y por lo tanto un atacante que esté en la red puede leer de manera fácil la información (texto plano) del paquete post que se envía a la hora del logueo. La primera medida que se hará es la implementación de la librería bcrypt en nuestro backend. De esta manera se puede encriptar de una manera segura y eficiente los datos de acceso. El primer paso será instalar el bcrypt en nuestro proyecto, nos vamos a una terminal bash y colocamos el siguiente comando: **npm install bcrypt jsonwebtoken dotenv**



```
~\verdant-venus\Proyecto_Final\Proyecto_Final1 git:(master) 7 files changed, 1286 insertions(+), 244 deletions(-)
npm run dev

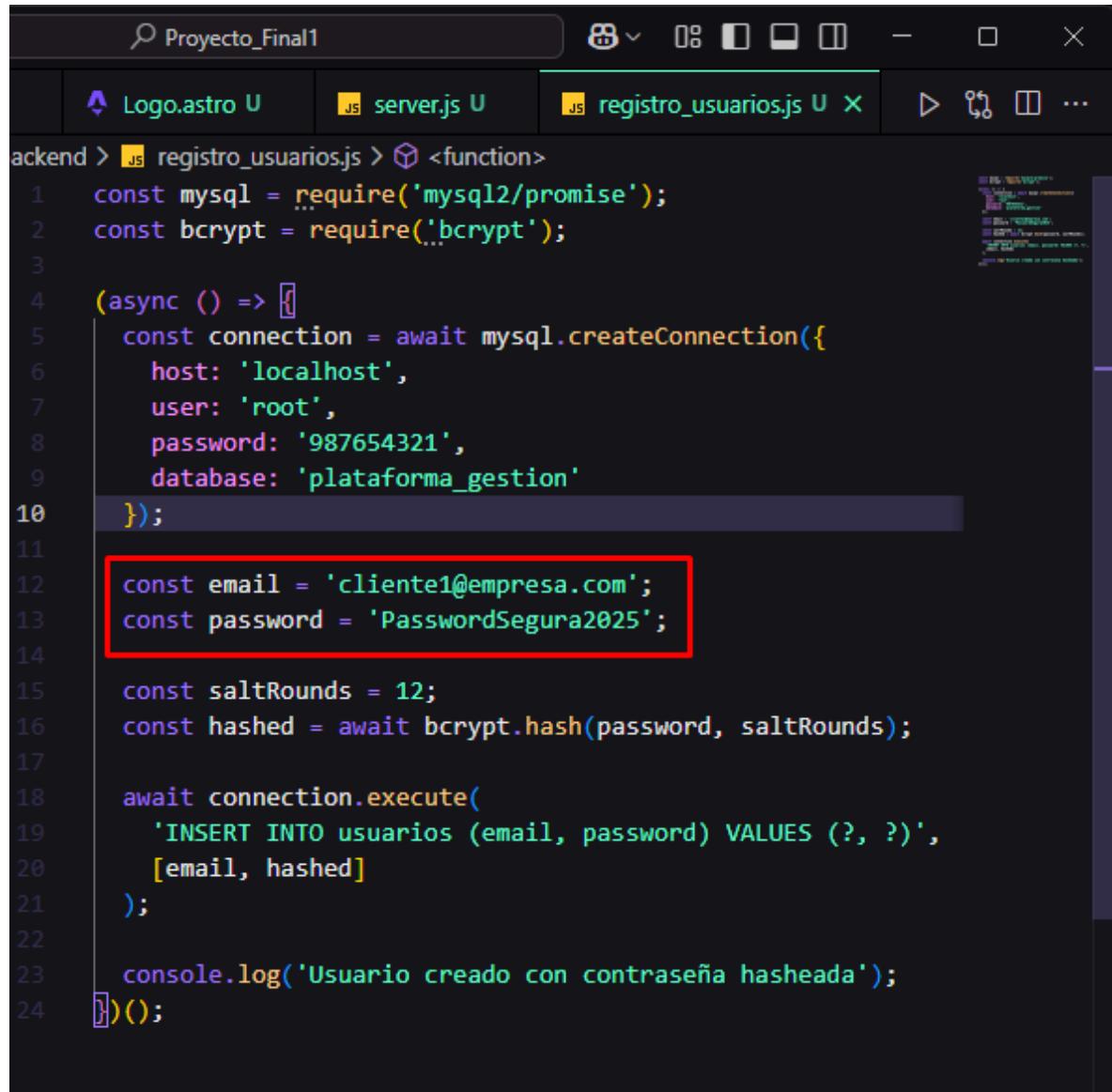
~\verdant-venus\Proyecto_Final\Proyecto_Final1 git:(master) 7 files changed, 1286 insertions(+), 244 deletions(-)
npm install bcrypt jsonwebtoken dotenv

added 17 packages, and audited 371 packages in 8s
150 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities
```

Con esta instalación lo que se procede a hacer será modificar la estructura del manejo de registro de la empresa. Anteriormente la empresa al rentar la plataforma de gestión energética lo que hacía era

crear la base de datos directamente en mysql, en dicha base de datos creaban los usuarios en texto plano, sin ninguna medida de protección a los datos.

El cambio que se hizo fue desarrollar una nueva forma de registro que implementa directamente una forma de encriptación a las contraseñas de los usuarios. Por lo tanto cuando una empresa contrate los servicios, el encargado de registrar a los usuarios lo hará de manera interna a través de un script:



```
Proyecto_Final1
Logo.astro U server.js U registro_usuarios.js U X ...
backend > registro_usuarios.js > <function>
1 const mysql = require('mysql2/promise');
2 const bcrypt = require('bcrypt');

3 (async () => {
4   const connection = await mysql.createConnection({
5     host: 'localhost',
6     user: 'root',
7     password: '987654321',
8     database: 'plataforma_gestion'
9   );
10
11   const email = 'cliente1@empresa.com';
12   const password = 'PasswordSegura2025';
13
14   const saltRounds = 12;
15   const hashed = await bcrypt.hash(password, saltRounds);
16
17   await connection.execute(
18     'INSERT INTO usuarios (email, password) VALUES (?, ?)',
19     [email, hashed]
20   );
21
22
23   console.log('Usuario creado con contraseña hasheada');
24 })();
```

En la parte señalada en rojo el encargado del registro colocará las credenciales de los nuevos usuarios, esto lo que hará será agregarlos directamente a la base de datos, pero con una diferencia, con este nuevo método las contraseñas quedarán con un hash aplicado en la base de datos de MySQL.

```
mysql> SELECT * FROM usuarios;
+----+-----+-----+
| id | email           | password          |
+----+-----+-----+
| 1  | prueba@correo.com | 123456            |
| 2  | cliente1@empresa.com | $2b$12$d7Hc/6nSbn/0ncLFemFCzuVrtL1e0JmNGc7AXiQffpuDhYjiU2Xl2 |
+----+-----+-----+
2 rows in set (0.009 sec)
```

En la anterior imagen se puede observar un usuario registrado de manera directa a través de mysql y el otro registrado a través del script para registrar usuarios.

```
app.post('/login', (req, res) => {
  const { email, password } = req.body;
  db.query('SELECT * FROM usuarios WHERE email = ?', [email], async (err, results) => {
    if (err || results.length === 0) return res.status(401).json({ message: 'Credenciales inválidas' });
    const user = results[0];
    const match = await bcrypt.compare(password, user.password);
    if (!match) return res.status(401).json({ message: 'Credenciales inválidas' });
  });
});
```

Cuando el usuario intenta loguear lo que hará el proceso interno en el backend será hacer una comparación del email, luego de esto procederá a comparar las contraseñas. El usuario introduce una contraseña y bcrypt le hará el mismo procedimiento de hashing y comprueba si al convertirla da el mismo resultado que la contraseña hasheada que se guardada en la base de datos.

### 3. middleware para rutas protegidas:

Se protegió el acceso a la ruta /Plataforma\_gestion del lado del cliente mediante una validación estricta del JWT. Esto se hizo debido a que se podía acceder a la plataforma de gestión sin antes loguearse.

Se añadió un script que:

- Oculta el contenido de la página hasta verificar el token con el backend.
- Redirige automáticamente al login (/) si el token no es válido o ha expirado.
- Muestra el contenido únicamente si la validación es exitosa.

Esto lo hace mediante la verificación del token como vimos anteriormente creamos un .env que contiene la “firma digital del token”, complementario a esto en el backend (Express), se creó una **ruta protegida** /api/protected que solo responde si el token es válido. Asegura que las peticiones sensibles sólo puedan realizarse si el usuario tiene un token legítimo. Por lo tanto si alguien intenta saltarse el logueo colocando la ruta exacta de la página, no podrá acceder ya que no habrá un token existente ni firmado, este se crea exclusivamente cuando el logueo se concreta.

**Código modificado:**

```
app.post('/login', (req, res) => {

  const { email, password } = req.body;

  db.query('SELECT * FROM usuarios WHERE email = ?', [email], async (err, results) => {

    if (err || results.length === 0) return res.status(401).json({ message: 'Credenciales inválidas' });

    const user = results[0];

    const match = await bcrypt.compare(password, user.password);

    if (!match) return res.status(401).json({ message: 'Credenciales inválidas' });

    const token = jwt.sign({ userId: user.id, email: user.email }, process.env.JWT_SECRET, {

      expiresIn: process.env.JWT_EXPIRATION

    });

    res.json({ success: true, token });

  });

});
```

Como podemos ver hace primero todo el proceso de verificación y por último le asigna un token al usuario, para después proceder a firmarlo con el contenido de nuestro archivo `.env`.

### Código del Middleware:

```
const authMiddleware = (req, res, next) => {

  const auth = req.headers.authorization;

  if (!auth) return res.status(401).send({ message: 'No autorizado' });

  const token = auth.split(' ')[1];

  jwt.verify(token, process.env.JWT_SECRET, (err, decoded) => {

    if (err) return res.status(403).send({ message: 'Token inválido o expirado' });

    req.user = decoded;

    next();
  });
};
```

#### 4. Limitación de Intentos de Inicio de Sesión con **express-rate-limit**:

Antes un usuario podía intentar loguearse infinitas veces, esto es muy peligroso ya que da entrada a que los atacantes logren vulnerar el sistema mediante fuerza bruta, es por esto que se colocó un límite de intentos.

### Código agregado:

```
const limiter = rateLimit({

  windowMs: 15*60*1000,

  max: 10,

  message: 'Demasiados intentos, prueba más tarde.'

});

app.use('/login', limiter);
```

El rate limit es una función proporcionada por el paquete **express-rate-limit** que permite establecer límites en la cantidad de solicitudes que un cliente puede hacer a un endpoint específico dentro de un intervalo de tiempo determinado.

#### Parámetros utilizados:

- **windowMs: 15 \* 60 \* 1000**  
Define una ventana de tiempo de 15 minutos (en milisegundos).  
Durante ese intervalo, se contará cuántas veces un cliente intenta acceder a **/login**.
- **max: 10**  
Permite un máximo de **10 solicitudes por IP** dentro de esa ventana de 15 minutos.  
Si se excede este límite, el servidor **bloqueará temporalmente** los intentos posteriores.
- **message: 'Demasiados intentos, prueba más tarde.'**  
Es el mensaje que se enviará al cliente cuando supere el límite de intentos.

#### La línea:

```
app.use('/login', limiter);
```

indica que el middleware se aplicará únicamente a las solicitudes dirigidas a **/login**, lo que protege específicamente el endpoint de inicio de sesión sin afectar otras rutas de la aplicación.

Estas fueron todas las medidas de seguridad aplicadas al código de la página web.

También se realiza la división de servicios en diferentes servidores para no tener un único punto de falla y en caso de ser vulnerada nuestra red no accedan a todos los servicios en un solo servidor.

Aclaración por el cual no se realiza uso de Router, no se usa el router ya que el Firewall usado en este caso permite configuración de enrutamientos capa 3 por ende nos evitamos un punto de falla adicional y lo dejamos solo para la administración general de la red.

La configuración de las VLAN en los equipos de core quedaron asignados así:

## En el Firewall:

Interfaces			
WAN	10Gbase-T <full-duplex>	192.168.200.228	
LAN	10Gbase-T <full-duplex>	192.168.50.1	
VLAN100DMZ	10Gbase-T <full-duplex>	172.10.16.1	
VLAN101GERENCIA	10Gbase-T <full-duplex>	192.168.15.1	
VLAN102ADMIN	10Gbase-T <full-duplex>	192.168.16.1	
VLAN103CONTABILIDAD	10Gbase-T <full-duplex>	192.168.17.1	
VLAN104SISTEMAS	10Gbase-T <full-duplex>	192.168.18.1	
VLAN105OPERACIONES	10Gbase-T <full-duplex>	192.168.19.1	
VLAN106MONITOREO	10Gbase-T <full-duplex>	192.168.20.1	
VLAN107INVITADOS	10Gbase-T <full-duplex>	192.168.21.1	

## En nuestro SW

SwL3#sh vlan			
VLAN	Name	Status	Ports
1	default	active	Gi3/1, Gi3/2
99	DMZ	active	Gi0/1, Gi0/2, Gi0/3, Gi1/0 Gi1/1
101	Gerencia	active	Gi1/2
102	Admin	active	Gi1/3
103	Contabilidad	active	Gi2/0
104	Sistemas	active	Gi2/1
105	Operaciones	active	Gi2/2
106	Monitoreo	active	Gi2/3
107	Invitados	active	Gi3/0
200	VLAN0200	active	
300	VLAN0300	active	
1002	fddi-default	act/unsup	
1003	trcrf-default	act/unsup	
1004	fddinet-default	act/unsup	
1005	trbrf-default	act/unsup	

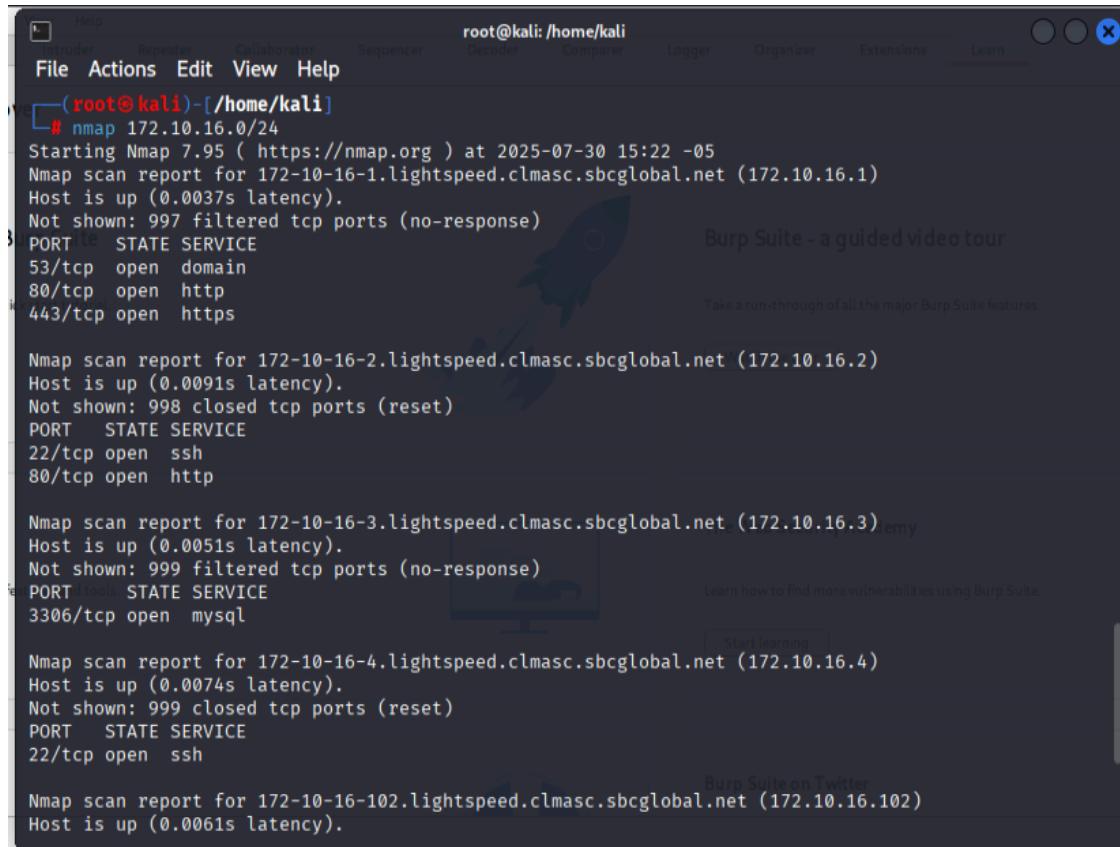
Dentro de los servicios que tenía adicionalmente se implementa un servidor de dominio para poder crear políticas de dominio y tener centralizado los usuarios de la empresa, allí mismo configuraremos un file server para que toda la data de los usuarios esté centralizada y poder realizar un backup óptimo.

Configuramos un equipo que se encargará de centralizar los backups realizados al file server y así administrar de manera fácil nuestros backups y realizar revisiones periódicas.

Se instala un Firewall para poder generar ACL y tener un mayor control de permisos a nivel general de nuestra red.

## NMAP RED SEGURA:

nmap 172.10.16.0/24



```
(root@kali)-[~/home/kali]
# nmap 172.10.16.0/24
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-30 15:22 -05
Nmap scan report for 172-10-16-1.lightspeed.clmasc.sbcglobal.net (172.10.16.1)
Host is up (0.0037s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE
53/tcp    open  domain
80/tcp    open  http
443/tcp   open  https

Nmap scan report for 172-10-16-2.lightspeed.clmasc.sbcglobal.net (172.10.16.2)
Host is up (0.0091s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Nmap scan report for 172-10-16-3.lightspeed.clmasc.sbcglobal.net (172.10.16.3)
Host is up (0.0051s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE
3306/tcp  open  mysql

Nmap scan report for 172-10-16-4.lightspeed.clmasc.sbcglobal.net (172.10.16.4)
Host is up (0.0074s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh

Nmap scan report for 172-10-16-102.lightspeed.clmasc.sbcglobal.net (172.10.16.102)
Host is up (0.0061s latency).
```

Escaneo Sigiloso nmap -sS -sV -sC 172.10.16.0/24

```
File Actions Edit View Help
Nmap scan report for 172.10.16.2.lightspeed.clmasc.sbcglobal.net (172.10.16.2)
Host is up (0.050s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 9.6p1 Ubuntu 3ubuntu13.13 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|_ 256 90:58:3c:2c:fe:ee:81:1b:75:df:e4:3d:28:73:1d:d9 (ECDSA)
|_ 256 26:43:a8:b5:4b:ca:94:e2:a8:21:89:1f:6c:94:42:06 (ED25519)
80/tcp    open  http     Apache httpd 2.4.58 ((Ubuntu))
|_http-server-header: Apache/2.4.58 (Ubuntu)
|_http-generator: Astro v5.11.1
|_http-title: Heliopolis
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Nmap scan report for 172.10.16.3.lightspeed.clmasc.sbcglobal.net (172.10.16.3)
Host is up (0.011s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
3306/tcp  open  mysql   MariaDB 10.3.23 or earlier (unauthorized)

Nmap scan report for 172.10.16.4.lightspeed.clmasc.sbcglobal.net (172.10.16.4)
Host is up (0.054s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 9.6p1 Ubuntu 3ubuntu13.13 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|_ 256 7f:43:52:09:32:ae:0f:23:3e:3c:9c:a2:5c:eb:b8:ba (ECDSA)
|_ 256 3f:27:64:c6:57:8a:16:7a:2b:22:f0:a7:9c:78:49:0a (ED25519)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

```

Nmap scan report for 172-10-16-102.lightspeed.clmasc.sbcglobal.net (172.10.16.102)
Host is up (0.0058s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE      VERSION
135/tcp    open  msrpc        Microsoft Windows RPC
139/tcp    open  netbios-ssn  Microsoft Windows netbios-ssn
445/tcp    open  microsoft-ds?
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
| smb2-time:
|_  date: 2025-07-30T20:28:21
|_  start_date: N/A
| smb2-security-mode:
|_  3:1:1:
|_  Message signing enabled but not required

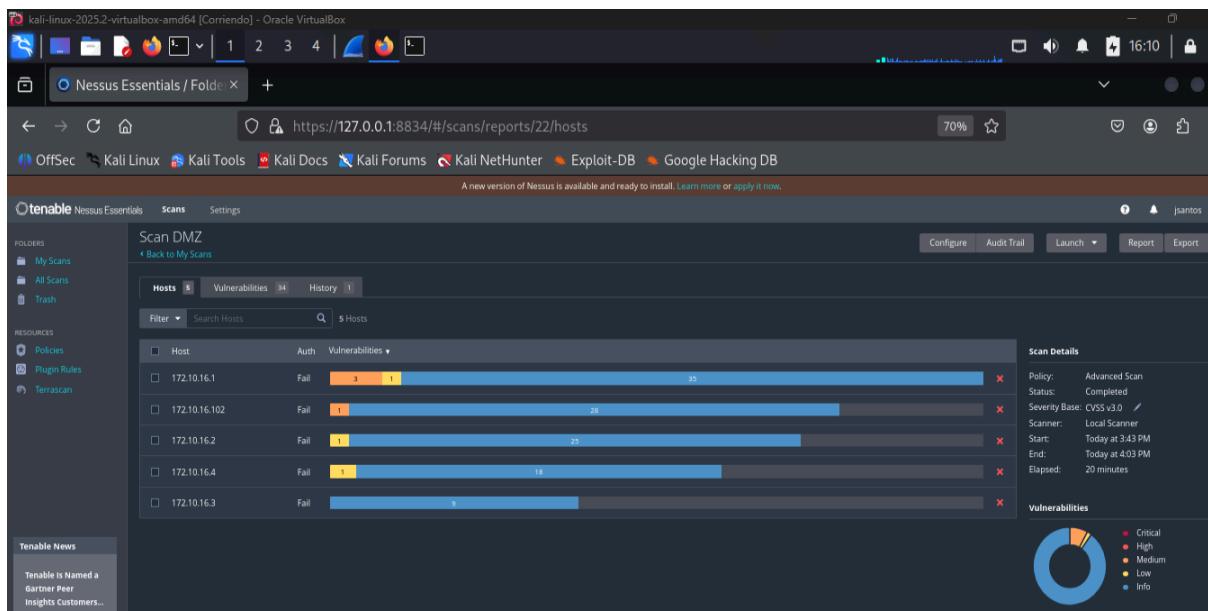
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 256 IP addresses (5 hosts up) scanned in 71.61 seconds

```

(root@kali)-[/home/kali]

Se puede analizar que ya todo está descentralizado, cada servidor se encarga de un servicio en específico, por este motivo se mitigan riesgos.

## Análisis de Nessus a red DMZ:



172.10.16.2 SRV WEB

**Vulnerabilities** 19

Sev	CVSS	VPR	EPSS	Name	Family	Count
Low	2.1	2.2	0.037	KICP Timestamp Request Remote Date Disclosure	General	1
Info	...	...	...	HTTP (Multiple Issues)	Web Servers	3
Info	...	...	...	SSH (Multiple Issues)	General	2
Info	...	...	...	SSH (Multiple Issues)	Misc.	2
Info	...	...	...	SSH (Multiple Issues)	Service detection	2
Info				Nessus SYN scanner	Port scanners	2
Info				Service Detection	Service detection	2
Info				Apache HTTP Server Version	Web Servers	1
Info				Backported Security Patch Detection (WWW)	General	1

**Host Details**

- IP: 172.10.16.2
- DNS: 172.10.16.2.lightspeed.dmasc.sbcg1.obal.net
- OS: Cisco Catalyst 9200 Series Switches, Cisco Catalyst 9300 Series Switches, Cisco Catalyst E9300 Rugged Series, Nutanix
- Start: Today at 3:43 PM
- End: Today at 3:50 PM
- Elapsed: 7 minutes
- KB: Download
- Auth: Fail

**Vulnerabilities**

## 172.10.16.3 DB

**Vulnerabilities** 9

Sev	CVSS	VPR	EPSS	Name	Family	Count
Info				Common Platform Enumeration (CPE)	General	1
Info				Host Fully Qualified Domain Name (FQDN) Resolution	General	1
Info				Nessus Scan Information	Settings	1
Info				Nessus SYN scanner	Port scanners	1
Info				OS Fingerprints Detected	General	1
Info				OS Identification	General	1
Info				Service Detection	Service detection	1
Info				TCP/IP Timestamps Supported	General	1
Info				Traceroute Information	General	1

**Host Details**

- IP: 172.10.16.3
- DNS: 172.10.16.3.lightspeed.dmasc.sbcg1.obal.net
- OS: Cisco Catalyst 9200 Series Switches, Cisco Catalyst 9300 Series Switches, Cisco Catalyst E9300 Rugged Series, Nutanix
- Start: Today at 3:43 PM
- End: Today at 3:50 PM
- Elapsed: 7 minutes
- KB: Download
- Auth: Fail

**Vulnerabilities**

## 172.10.16.4 APLICACIONES

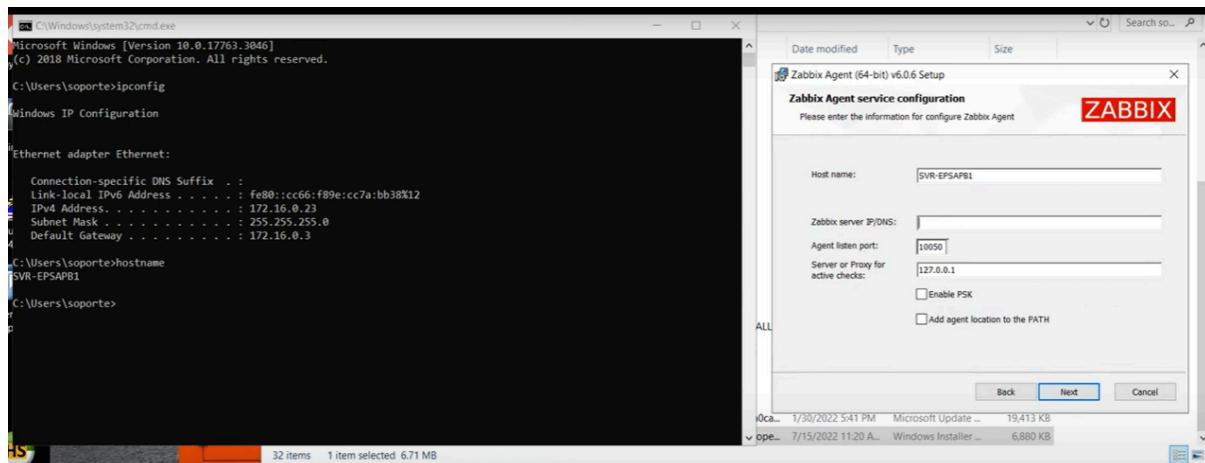
Podemos ver que en los servidores, la mayoría de vulnerabilidades son únicamente informativas, ocurre lo mismo que en la insegura, lo mejor sería proceder a hacer algunos cambios para que nuestros servidores no entreguen tanta información a los posibles atacantes y así prevenir.

## Monitoreo y logs:

Para el sistema de control y monitoreo con logs se utilizó Zabbix, nos permite monitorear en todo momento distintas métricas, muy útil para entornos e implementaciones empresariales como es el caso de la empresa energética Heliópolis.

### Instalación:

Ejecutamos el instalador y debemos saber la ip del equipo en el cual lo vamos a ejecutar, ya seguido es darle en siguiente y aceptar.



Una vez instalado y que se haya iniciado procedemos con agregar los host o equipos los cuales vamos a monitorear.

The screenshot shows the 'New host' dialog box with the 'Host' tab selected. The 'Host' tab contains the following fields:

- \* Host name: SVR
- Visible name: SVR-EP
- Templates: (Search field and Select button)
- \* Groups: (Search field and Select button)
- Interfaces: No interfaces are defined. (Add button)
- Description: (Large text area)
- Monitored by proxy: (no proxy) (Dropdown menu)
- Enabled: (Checkmark)

At the bottom right of the dialog box are two buttons: 'Add' (blue) and 'Cancel' (white).

Una vez agregados los equipos aparecen listados en la parte inferior del panel y ya se puede proceder a configurar todos los parámetros que se quieren monitorear, este paso ya es dependiendo las necesidades de cada empresa.

Screenshot of the Zabbix web interface showing the host configuration page. The left sidebar includes links for Monitoring, Inventory, Reports, Configuration, Administration, Support, Integrations, and Help.

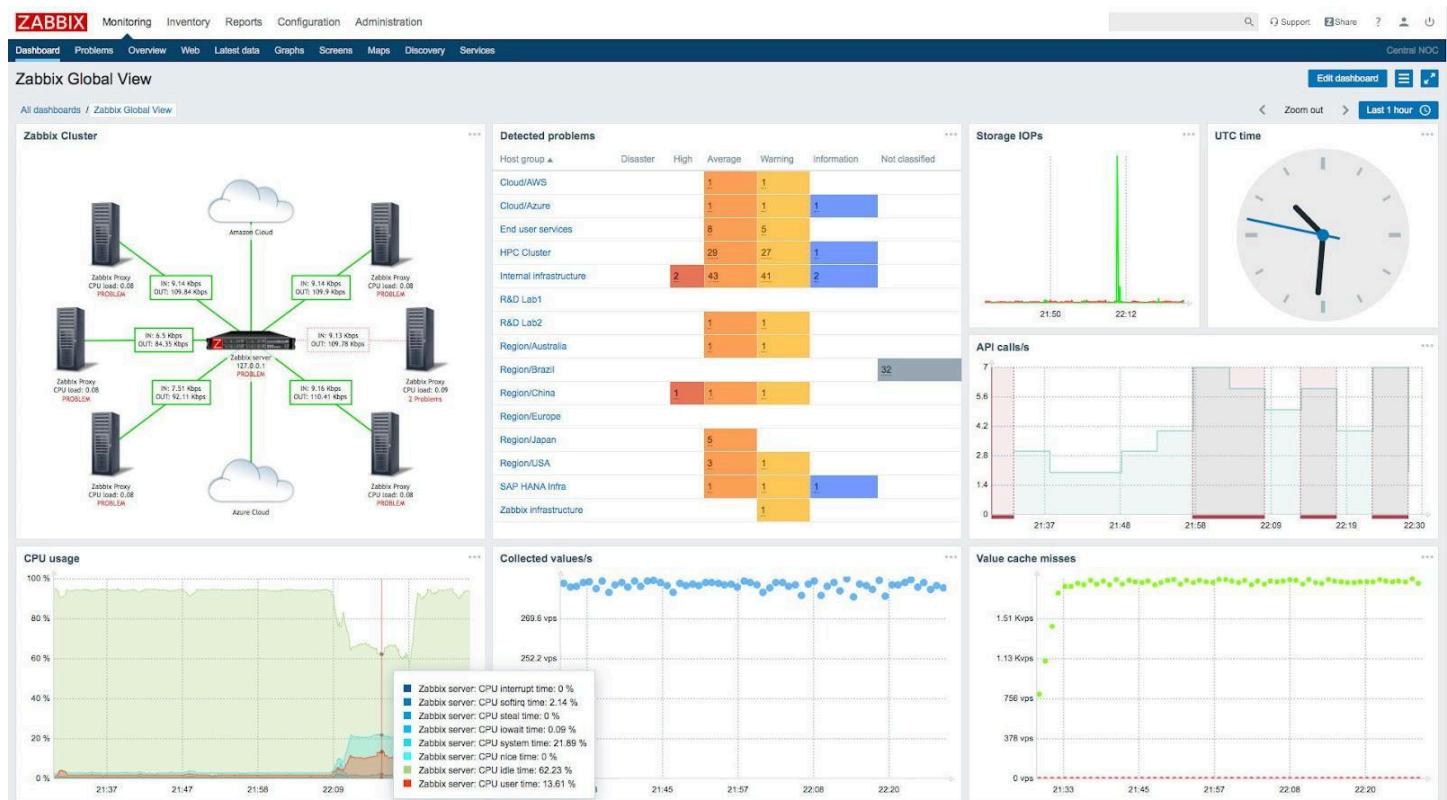
The main search bar at the top has fields for Name, Host groups, IP, DNS, Port, Status (Any, Enabled, Disabled), Tags (And/Or, Or), Severity (Not classified, Information, Warning, Average, High, Disaster), and checkboxes for Show hosts in maintenance and Show suppressed problems.

The host list table displays the following data:

Interface	Availability	Tags	Status	Latest data	Problems	Graphs	Dashboards	Web
172.16.0.5:10050	ZBX	class:os   target: windows	Enabled	Latest data 32	1	Graphs 5	Dashboards 2	Web
172.16.0.6:10050	ZBX	class:os   target: windows	Enabled	Latest data 32	1	Graphs 5	Dashboards 2	Web
172.16.0.1:10050	ZBX	class:os   target: windows	Enabled	Latest data 134	1	Graphs 16	Dashboards 2	Web
172.16.0.14:10050	ZBX	class:os   target: windows	Enabled	Latest data 32	1	Graphs 5	Dashboards 2	Web
127.0.0.1:10050	ZBX	class:os   class:software   target: linux ***	Enabled	Latest data 129	1	Graphs 25	Dashboards 4	Web

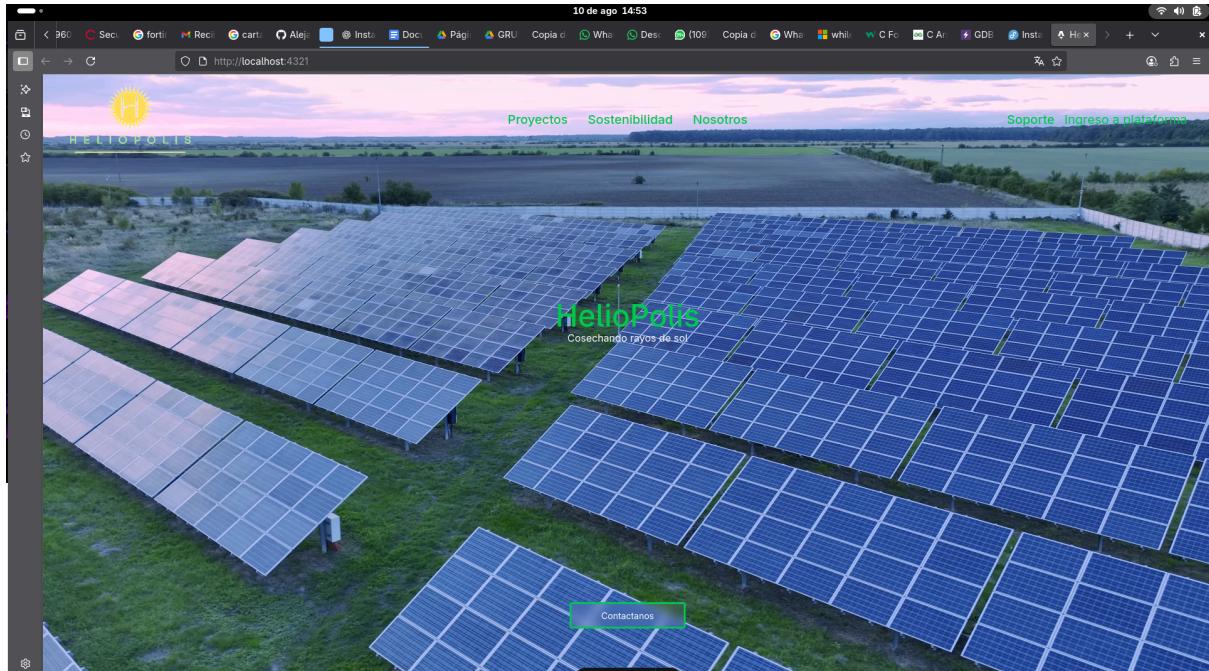
Displaying 5 of 5 found.

Ya configurados algunos parámetros en el panel se puede observar unas gráficas de mediciones las cuales se pueden ajustar a medida.



# Vista de la plataforma:

## Landing-page:



10 de ago 14:53

http://localhost:4321

HELIOPOLIS

Oficinas

\*\*Pereira\*\*  
Edif. Torre Central  
Cra 10 No. 17-35 Piso2  
contactenos@ehp.com.co

\*\*Cartago\*\*  
Carrera 4 # 16 – 104  
CC Nuevo Cartago  
contactenoscartago@ehp.com.co

\*\*Otros correos\*\*  
notificacionesjudiciales@ehp.com.co  
lneasetica@ehp.com.co

Línea de atención

606 3151515  
Marcando desde un fijo: 115  
Opción 1 Pereira  
Opción 2 Cartago

Enlaces de interés

Meeo - movilidad eléctrica  
Sistemas solares fotovoltaicos  
Realiza una PQRS  
Mapa del sitio  
Políticas institucionales  
Sistema Único de Información de Trámites

Pacto Global Red Colombia

ISO 14064-1:2018  
BUREAU VERITAS Certification

## Login-page:

10 de ago 14:54

http://localhost:4321/Login

Login

Aprovecha de nuestros servicios

cliente1@empresa.com

\*\*\*\*\*

Sign In

¿Problemas con tus datos de ingreso?  
Contáctanos

## Plataforma gestión:

