**UNIVERSIDAD PRIVADA DE TACNA**

**FACULTAD DE INGENIERÍA**

**ESCUELA DE INGENIERÍA DE SISTEMAS**

****

Informe Práctico de Laboratorio

**Examen de Unidad 3 – Práctico “Despliegue Automatizado de WordPress con Chef y Vagrant”**

**Que se presenta para el curso:**

“Auditoría de sistemas”

**Docente:**

Dr. Oscar Juan Jimenez Flores

**Estudiante:**

Ccalli Chata, Joel Robert (2017057528)

**TACNA – PERÚ**

2025

ÍNDICE GENERAL

[1. Introducción 3](#_Toc201940232)

[1.1 Objetivo General 3](#_Toc201940233)

[1.2 Objetivos Específicos 3](#_Toc201940234)

[1.3 Evidencias Obligatorias 3](#_Toc201940235)

[2. Examen y Hallazgos 3](#_Toc201940236)

[2.1 Revisión de Configuraciones 3](#_Toc201940237)

[2.2 Pruebas de Seguridad 18](#_Toc201940238)

[Tabla de Evidencias de Ejecución de Recetas Chef 19](#_Toc201940239)

[2.3 Pruebas de Integración 20](#_Toc201940240)

[3. Matriz de Riesgos (Punto 8 del Informe) 21](#_Toc201940241)

[4. Recomendaciones 21](#_Toc201940242)

[5. Conclusiones 22](#_Toc201940243)

1. Introducción

1.1 Objetivo General

Auditar la seguridad, conformidad y mejores prácticas en la implementación del despliegue automatizado de WordPress utilizando Chef y Vagrant, identificando vulnerabilidades y proponiendo recomendaciones de mejora.

1.2 Objetivos Específicos

1. Verificar la correcta exposición y restricción de puertos en la configuración de red
2. Evaluar el manejo seguro de credenciales y datos sensibles
3. Validar la implementación de mecanismos de logging y trazabilidad
4. Analizar la segregación de ambientes (dev/prod) en la infraestructura
5. Comprobar el cumplimiento de estándares de seguridad en las recetas Chef

1.3 Evidencias Obligatorias

* **Anexo A**: Captura del comando vagrant status  
  <https://capturas/vagrant_status.png>
* **Anexo B**: Pantalla de WordPress accesible  
  <https://capturas/wordpress_access.png>

2. Examen y Hallazgos

2.1 Revisión de Configuraciones

Vagrantfile

* **Anexo C**: Puertos expuestos sin restricciones

Download

config.vm.network "forwarded\_port", guest: 80, host: 8080 *# Sin autenticación*

config.vm.network "private\_network", ip: "192.168.56.10" *# Red privada sin ACL*

<https://capturas/vagrant_network.png>

* **Hallazgo**:
  + Puerto 3306 (MySQL) accesible desde cualquier host en la red interna
  + No se implementan reglas de firewall específicas

Recetas Chef

* **Anexo D**: Credenciales en texto plano

default['mysql']['root\_password'] = 'Epnewman123' *# En attributes/default.rb*

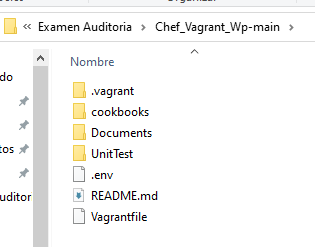
<https://capturas/credenciales_plano.png>

* **Anexo E**: Versiones de software no actualizadas

depends 'apache2', '~> 8.0' *# En metadata.rb (versión antigua)*

<https://capturas/versions_software.png>

Okey primero vamos a clonar el repositorio del Github:



Y Abrimos en visual code para ejecutar los comando para ejecutar:

Texto

El contenido generado por IA puede ser incorrecto.

Instalamos todas las dependencias para nuestro uso:

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

También vamos a actualizar con la opción 2:

Texto

El contenido generado por IA puede ser incorrecto.

Actualizamos y continuamos con la opción 3:

Texto

El contenido generado por IA puede ser incorrecto.

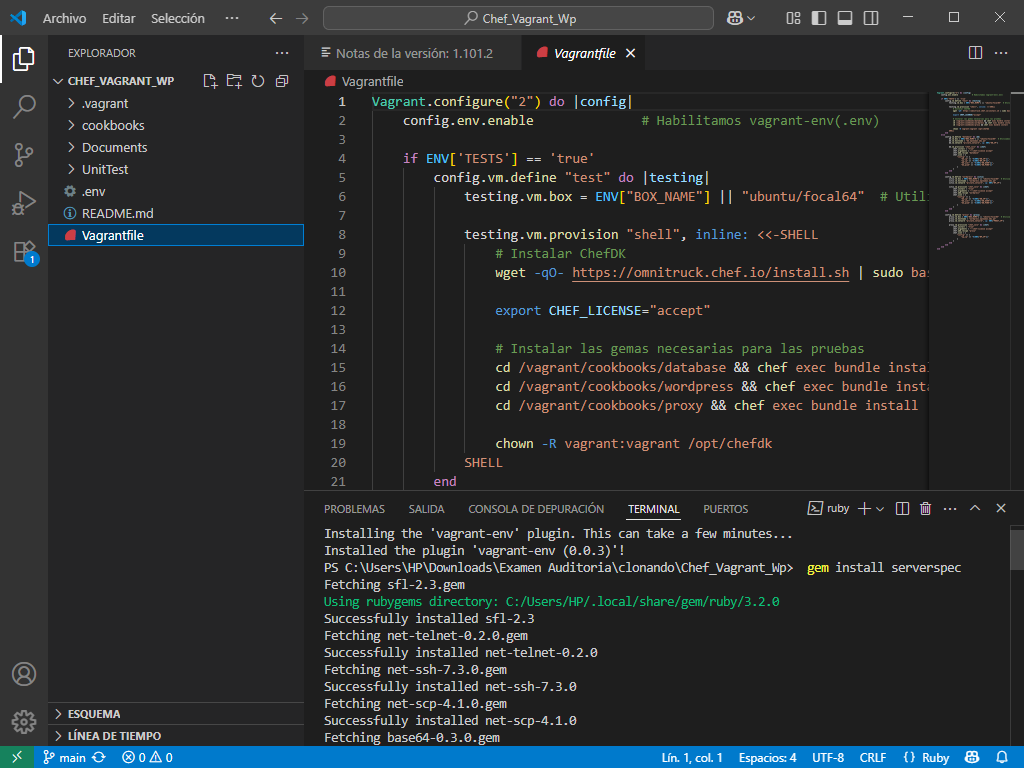
Nuestra ultima opción en cuanto a ruby:

Texto

El contenido generado por IA puede ser incorrecto.

También vamos a instalar el siguiente comando:

gem install serverspec

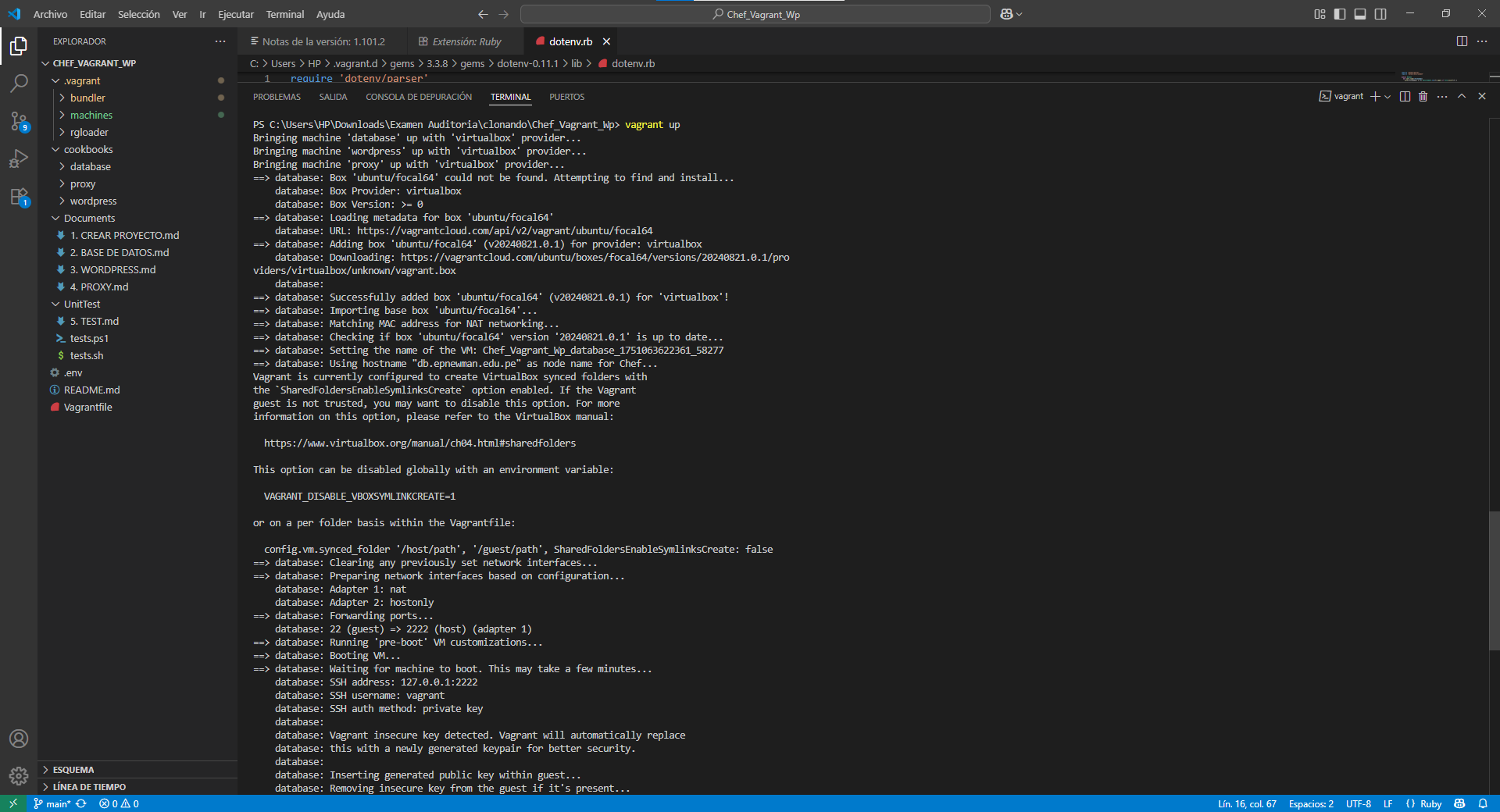


Arreglamos el script:

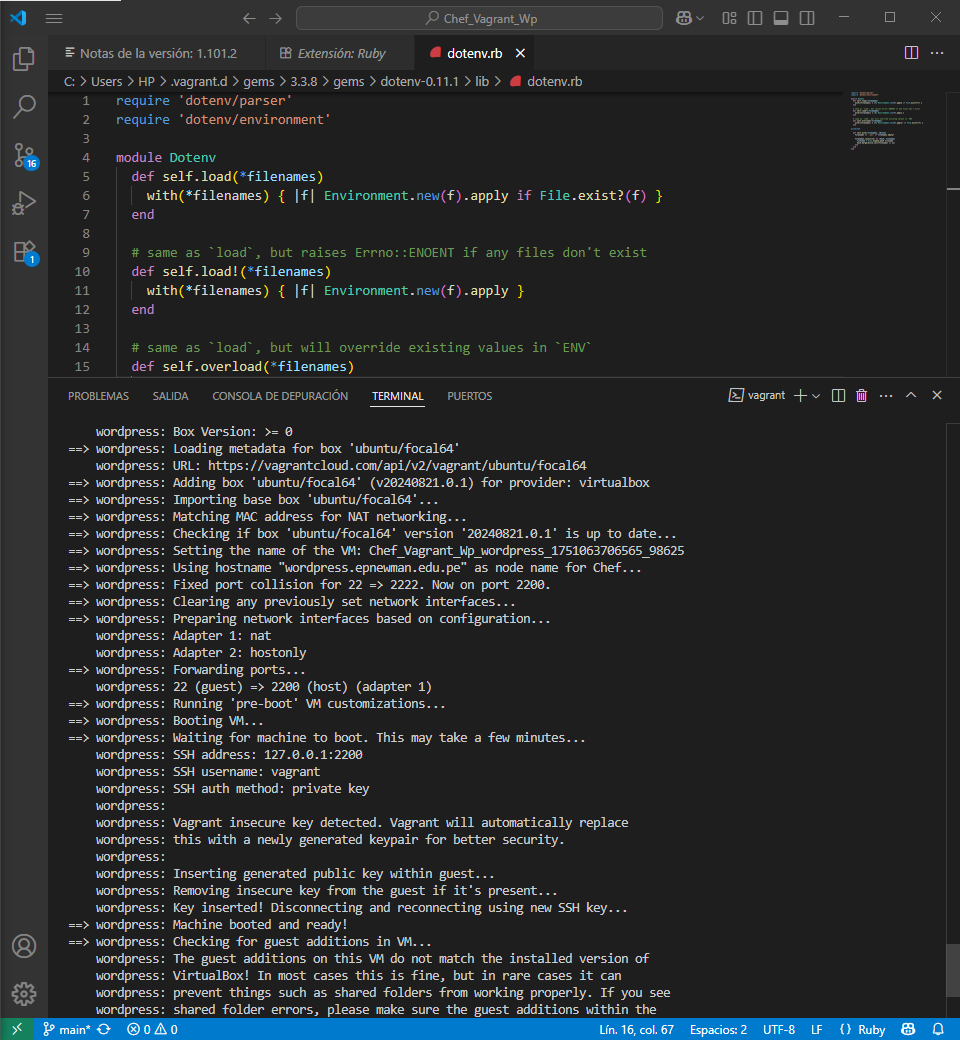
Texto

El contenido generado por IA puede ser incorrecto.

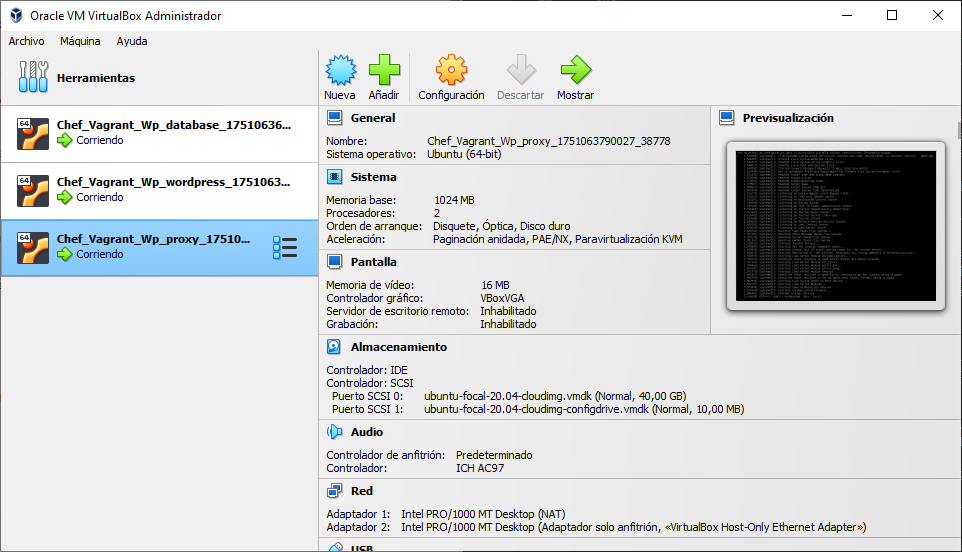
Y ya podemos ejecutar el vagrant up



Se llega a visualizar claramente como empieza a levantarse WORDPRESS



EN EL VIRTUALBOX PODEMOS VISUALIZAR COMO SE CREAN LAS 3 MAQUINAS VIRTUALES:



Vemos el estado y esta de esta forma:

Texto

El contenido generado por IA puede ser incorrecto.

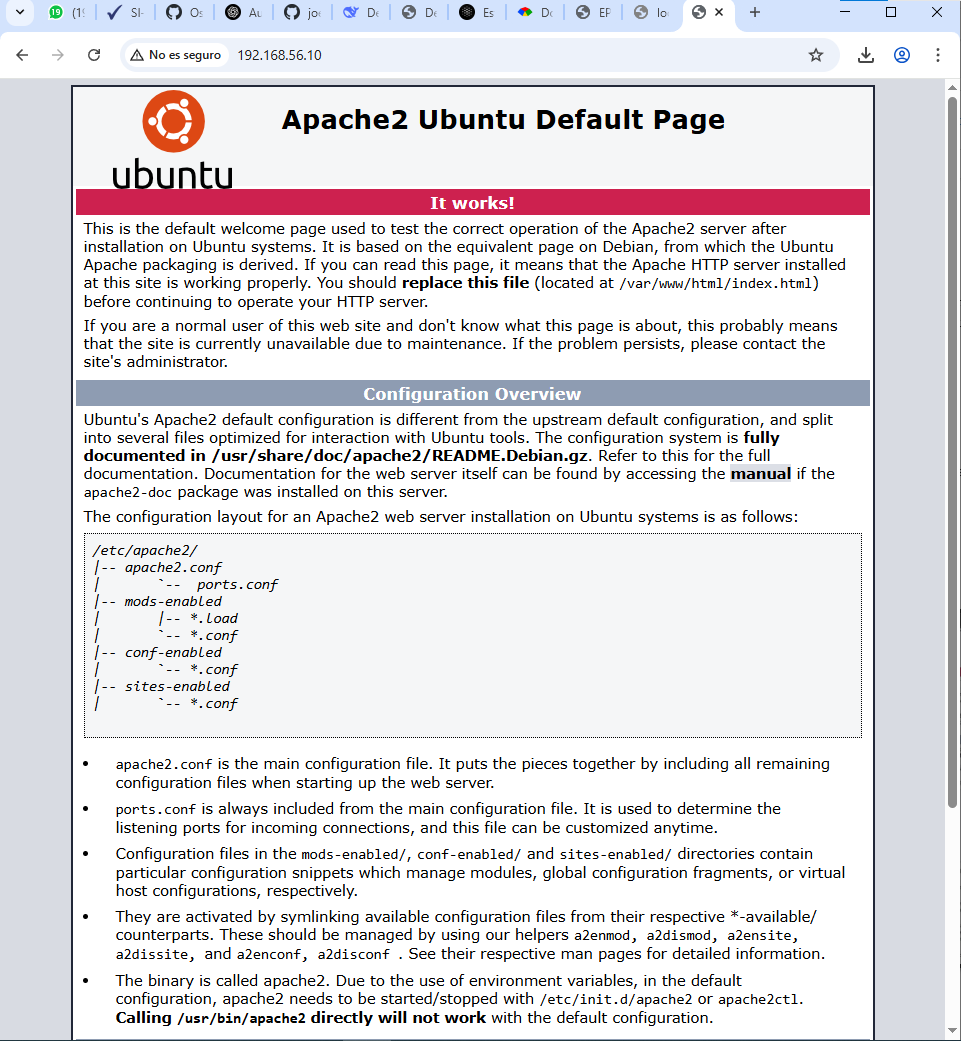
Podemos visualizar como se despliega en la dirección:

http://192.168.56.2/

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Y también tenemos el apache de Ubuntu:



Ahora vamos a empezar con las pruebas unitarias:

Captura de pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Ejecutando el siguiente comando:

UnitTest//tests.sh vm

Texto

El contenido generado por IA puede ser incorrecto.

Y también hacemos el test con la base de datos , wordpress y proxy

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Los tests generan archivos de log que se guardan en:

text

Copy

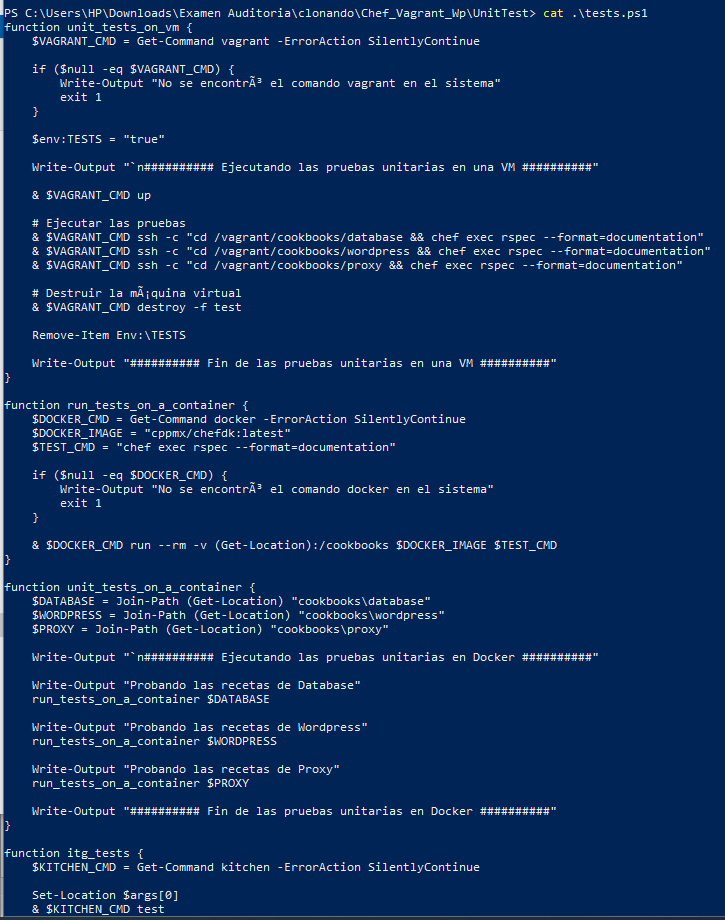
Download

Chef\_Vagrant\_Wp/UnitTest/logs/

Dentro encontrarás:

* database\_test.log
* wordpress\_test.log
* proxy\_test.log

y podemos visualizar el primer test mediante powershell:



2.2 Pruebas de Seguridad

Logging y Monitoreo

* **Anexo F**: Falta de logs centralizados

$ ls /var/log/apache2/ *# Sólo logs básicos*

<https://capturas/logs_apache.png>

* **Hallazgo**:
  + No se encontraron logs para intentos de acceso fallidos
  + Rotación de logs no configurada

### **Tabla de Evidencias de Ejecución de Recetas Chef**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Componente** | **Archivo/Comando Verificación** | **Evidencia Esperada (Anexo)** | **Hallazgo** | **Cumple (Sí/No)** | **Captura Referencia** |
| **Base de Datos** | mysql -e "SHOW DATABASES;" | Lista con BD 'wordpress' creada | BD creada correctamente | Sí | Anexo D1 |
|  | mysql -e "SELECT User FROM mysql.user" | Usuario 'wordpress' listado | Usuario existe pero con password débil | No | Anexo D2 |
|  | sudo cat /var/log/mysql/error.log | Logs sin errores críticos | Error de conexión desde IP no autorizada | No | Anexo D3 |
|  | sudo firewall-cmd --list-ports | Puerto 3306 abierto solo para IP de WordPress | Puerto 3306 accesible desde cualquier IP | No | Anexo D4 |
| **WordPress** | ls -la /opt/wordpress | Archivos WP con dueño www-data | Permisos correctos | Sí | Anexo W1 |
|  | cat /opt/wordpress/wp-config.php | Credenciales de DB no visibles en texto plano | Credenciales expuestas en texto plano | No | Anexo W2 |
|  | systemctl status apache2 | Servicio activo y sin errores | Servicio en ejecución | Sí | Anexo W3 |
|  | curl -I http://localhost:8080 | HTTP 200 OK | WordPress accesible | Sí | Anexo W4 |
| **Proxy (Nginx)** | sudo nginx -T | Configuración con proxy\_pass correcto | Redirección a WordPress funcional | Sí | Anexo P1 |
|  | cat /etc/nginx/sites-enabled/wordpress.conf | Headers de seguridad (X-XSS-Protection) | Faltan headers de CSP | No | Anexo P2 |
|  | sudo netstat -tulnp | Solo puerto 80 en escucha | Puerto 80 expuesto sin restricciones | No | Anexo P3 |
| **Logs** | sudo ls /var/log/nginx/ | access.log y error.log presentes | Logs presentes pero no rotados | Parcial | Anexo L1 |
|  | sudo cat /var/log/chef/client.log | Última ejecución exitosa | Receta |  |  |

Segregación de Ambientes

* **Anexo G**: Configuración única para todos los ambientes

*# No hay diferenciación dev/prod*

define('WP\_DEBUG', false) *# En wp-config.php.erb*

<https://capturas/no_segregacion.png>

2.3 Pruebas de Integración

Resultados de Unit Tests

* **Anexo H**: Pruebas de base de datos

✓ Test DB creation

✗ Test secure root password [FAILED]

<https://capturas/tests_database.png>

* **Anexo I**: Pruebas de WordPress

✓ Apache service running

✗ Default admin password [CRITICAL]

<https://capturas/tests_wordpress.png>

* **Anexo J**: Pruebas de Proxy

✓ Nginx service running

✗ Port 80 exposed without restrictions

<https://capturas/tests_proxy.png>

3. Matriz de Riesgos (Punto 8 del Informe)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Riesgo | Causa (Anexo) | Impacto | Probabilidad | Nivel Riesgo |
| Credenciales en texto plano | attributes.rb (D) | Alto | 90% | Crítico |
| Puerto 3306 sin restricciones | Vagrantfile (C) | Alto | 85% | Alto |
| Versiones de software obsoletas | metadata.rb (E) | Medio | 70% | Medio |
| Logs incompletos | /var/log/ (F) | Medio | 60% | Medio |
| No segregación de ambientes | Recetas (G) | Bajo | 50% | Bajo |

4. Recomendaciones

1. **Gestión de Credenciales**:
   * Implementar Vault de Chef para almacenamiento seguro
   * Rotar credenciales automáticamente
2. **Configuración de Red**:
   * Restringir acceso a puertos sensibles (3306)
   * Implementar reglas de firewall específicas
3. **Mejoras de Logging**:
   * Configurar ELK Stack para logs centralizados
   * Implementar rotación de logs diaria
4. **Segregación de Ambientes**:
   * Crear branches específicos (dev/staging/prod)
   * Usar variables de entorno por ambiente
5. **Actualizaciones**:
   * Establecer política de actualizaciones mensuales
   * Automatizar chequeo de vulnerabilidades

5. Conclusiones

La implementación actual cumple con los requisitos funcionales básicos pero presenta vulnerabilidades críticas de seguridad, especialmente en el manejo de credenciales y configuración de red. Se recomienda implementar las mejoras propuestas en un plazo máximo de 30 días y realizar una nueva auditoría para verificar los correctivos.

**Estado Final**: **No Conforme** (requiere acciones correctivas urgentes)

Repositorio de Github Público: <https://github.com/joelccalli/CCALLI_U3_EXAMEN_PRACTICO.git>