

**UNIVERSIDAD PRIVADA DE TACNA  
FACULTAD DE INGENIERÍA  
ESCUELA DE INGENIERÍA DE SISTEMAS**



# **EXAMEN DE UNIDAD III**

## **“Plan de Auditoría de Tecnologías de la Información para DevIA360”**

Enlace Github: [https://github.com/JosueChUPT/AS\\_U3\\_EXAMEN\\_PRACTICO](https://github.com/JosueChUPT/AS_U3_EXAMEN_PRACTICO)

**Que se presenta para el curso:**

“Auditoría de sistemas”

**Integrante(s):**

Chambilla Zuñiga, Josue

**Docente:**

Dr. Oscar Juan Jimenez Flores

TACNA – PERÚ

2025

## Índice

1. Origen .....	3
2. Información de la Entidad .....	4
3. Denominación de la Materia de Control.....	4
4. Alcance.....	5
5. Objetivo.....	6
- Objetivos Específicos: .....	6
6. Procedimientos de Auditoría .....	7
7. Plazo de la Auditoría y Cronograma.....	8
8. Criterios de Auditoría.....	9
9. Información Administrativa .....	10
10. Documento a Emitir .....	10
11. ANEXOS.....	13
11.1. Preparación del Entorno .....	13
11.2. Visualizar WordPress .....	15
11.3. Examen y Hallazgos.....	15
Exposición de Credenciales en Código Fuente.....	15
SSL/TLS Inseguro .....	16
Sin Control de Versiones .....	17
Logs No Centralizados.....	17
Sin Separación de Ambientes .....	18
Puertos Expuestos .....	18
Sin Gestión de Secretos .....	19
11.4. Matriz de Riesgos .....	20

## 1. Origen

La presente auditoría surge a solicitud de la gerencia de DevIA360, empresa especializada en soluciones tecnológicas avanzadas, ante la identificación preliminar de riesgos críticos de seguridad y deficiencias operativas en su sistema automatizado de despliegue de aplicaciones web, específicamente en el entorno basado en Chef, Vagrant y Wordpress (repositorio: [Repositorio Chef\\_Vagrant\\_Wp](#)).

Este sistema, clave para el proceso de integración y entrega continua (CI/CD), es utilizado para garantizar entornos estandarizados y eficientes en el despliegue de servicios para clientes. No obstante, durante una revisión interna, se detectaron problemas como la exposición de credenciales, configuraciones de red no seguras, falta de registros de auditoría, uso de versiones obsoletas y ausencia de separación entre ambientes de desarrollo y producción.

Dado que estos elementos comprometen la seguridad, confiabilidad y cumplimiento normativo del sistema, se ha decidido llevar a cabo una auditoría exhaustiva, con el objetivo de evaluar, documentar y mitigar los riesgos asociados al despliegue automatizado, proponiendo mejoras y buenas prácticas alineadas con los estándares de seguridad de TI y DevOps.

## **2. Información de la Entidad**

- Razón Social: DevIA360 S.A.C.
- Rubro: Servicios de tecnologías de la información (TI) y automatización de despliegue de aplicaciones web.
- Tipo de organización: Empresa privada especializada en soluciones tecnológicas avanzadas.
- Ubicación: Tacna, Perú.
- Alcance de operaciones: Nacional, con proyección a brindar servicios de despliegue automatizado de aplicaciones web a empresas del sector privado y público a nivel nacional.

## **3. Denominación de la Materia de Control**

Evaluación de la Seguridad, Configuración y Gestión del Sistema de Despliegue Automatizado Chef\_Vagrant\_Wp para entornos WordPress en DevIA360.

Esta materia de control se enfoca en analizar los componentes técnicos, operativos y de seguridad del sistema automatizado de despliegue de entornos WordPress que utiliza la empresa DevIA360. El objetivo principal es verificar la correcta implementación de prácticas seguras de configuración, protección de credenciales, segmentación de ambientes, control de versiones, y trazabilidad de procesos dentro del flujo continuo de integración y entrega (CI/CD)

#### 4. Alcance

La presente auditoría de Tecnologías de la Información está enfocada en evaluar el sistema de despliegue automatizado de entornos WordPress implementado por DevIA360 mediante la solución Chef\_Vagrant\_Wp. El análisis abarca aspectos técnicos, operativos y de seguridad, centrados en los siguientes elementos:

- Revisión del código fuente del repositorio Chef\_Vagrant\_Wp para identificar prácticas inseguras, configuraciones deficientes o elementos obsoletos.
- Verificación del proceso de despliegue utilizando Vagrant y Chef, desde la creación del entorno virtual hasta la puesta en funcionamiento de WordPress.
- Análisis de la configuración de red, puertos y servicios expuestos dentro de la máquina virtual generada.
- Inspección del manejo de credenciales y archivos sensibles durante la automatización del entorno.
- Evaluación de la existencia de mecanismos de auditoría y trazabilidad durante el proceso de automatización (logs, registros, validaciones).
- Revisión de la segmentación de ambientes (desarrollo, prueba y producción) y las políticas aplicadas para su separación.

- Identificación y documentación de riesgos que afecten la confidencialidad, integridad, disponibilidad y cumplimiento normativo del sistema.

.

## 5. Objetivo

Evaluar la seguridad, eficiencia y cumplimiento del sistema automatizado de despliegue de entornos WordPress implementado por DevIA360 mediante la herramienta Chef\_Vagrant\_Wp, con el fin de identificar riesgos críticos, vulnerabilidades técnicas y oportunidades de mejora que garanticen entornos confiables y seguros.

### - **Objetivos Específicos:**

- Verificar la correcta configuración del entorno virtualizado y su proceso de automatización utilizando Vagrant y Chef.
- Identificar y documentar posibles vulnerabilidades de seguridad en el manejo de credenciales y exposición de servicios.
- Analizar el nivel de trazabilidad y auditoría presente durante el proceso de despliegue automatizado.
- Evaluar la segmentación y control de ambientes (desarrollo, prueba, producción) en el sistema analizado.
- Elaborar una matriz de riesgos que permita valorar el impacto y la probabilidad de ocurrencia de las principales amenazas detectadas.

## 6. Procedimientos de Auditoría

Objetivo Específico	Procedimientos Técnicos Aplicados
Verificar la correcta configuración del entorno virtualizado y su proceso de automatización	<ul style="list-style-type: none"><li>- Clonar el repositorio y desplegar el entorno según el README desde GitHub</li><li>- Verificar si WordPress funciona correctamente dentro de la VM.</li><li>- Registrar capturas de pantalla del despliegue.</li></ul>
Identificar y documentar vulnerabilidades de seguridad	<ul style="list-style-type: none"><li>- Analizar archivos sensibles (Vagrantfile, recipes, attributes) en busca de credenciales expuestas.</li><li>- Validar si los puertos expuestos son seguros o innecesarios.</li><li>- Verificar versiones obsoletas del software desplegado.</li></ul>
Analizar el nivel de trazabilidad y auditoría	<ul style="list-style-type: none"><li>- Comprobar si existen logs, registros o mecanismos de seguimiento del proceso de despliegue.</li><li>- Verificar si Chef genera registros de ejecución o errores.</li></ul>
Evaluar la segmentación y control de ambientes.	<ul style="list-style-type: none"><li>- Revisar si el sistema permite separar ambientes (desarrollo, prueba, producción).</li><li>- Comprobar si hay variables o archivos que distingan los entornos o si todo se maneja como uno solo.</li></ul>
Elaborar una matriz de riesgos	<ul style="list-style-type: none"><li>- Identificar al menos cinco riesgos técnicos a partir del análisis.</li><li>- Clasificar los riesgos por nivel de impacto y probabilidad.</li><li>- Asociar cada riesgo con evidencia técnica documentada (fragmento de código, captura, etc.)</li></ul>

## 7. Plazo de la Auditoría y Cronograma

<b>Etapas</b>	<b>Fecha de Inicio</b>	<b>Fecha de Finalización</b>	<b>Duración (días)</b>	<b>Actividades Detalladas</b>
<b>Planificación</b>	27/06/2025	28/06/2025	2	Definición del alcance, objetivos, criterios y metodología de auditoría.
<b>Preparación del entorno</b>	29/06/2025	01/07/2025	2	Clonar el repositorio, configurar Vagrant y Chef, levantar la VM, validar el entorno.
<b>Revisión técnica</b>	01/07/2025	04/07/2025	4	Análisis del código fuente, configuraciones, credenciales, puertos, versiones
<b>Evaluación de seguridad</b>	05/07/2025	06/07/2025	2	Identificación de vulnerabilidades, trazabilidad y segmentación de ambientes.
<b>Identificación de riesgos</b>	07/07/2025	08/07/2025	2	Redacción de la matriz de riesgos con evidencia técnica
<b>Elaboración del informe</b>	09/07/2025	10/07/2025	2	Redacción del informe de auditoría con hallazgos, conclusiones y recomendaciones.



## 8. Criterios de Auditoría

Los criterios de auditoría son los principios, normas técnicas y buenas prácticas que sirven como referencia para evaluar el sistema de despliegue automatizado de DevIA360. A continuación, se detallan los criterios asociados a cada objetivo específico:

Objetivo Específico	Criterios de Auditoría	Fuente / Referencia
Verificar la configuración del entorno automatizado.	La configuración del entorno debe ser clara, reproducible, segura y documentada.	Infrastructure as Code Best Practices / DevOps Handbook
	El Vagrantfile debe restringir el acceso a puertos innecesarios o inseguros.	OWASP Deployment Best Practices / CIS Benchmarks
Identificar vulnerabilidades de seguridad.	Las credenciales no deben estar en texto plano; deben gestionarse mediante herramientas seguras (Chef Vault, HashiCorp Vault, etc.).	OWASP Secure Configuration / ISO/IEC 27001 A.9 (Access Control)
	El software instalado debe estar actualizado y libre de vulnerabilidades conocidas (CVE).	NIST SP 800-40 / OWASP Dependency Management
Analizar trazabilidad y auditoría.	Todo proceso automatizado debe generar registros persistentes de ejecución y errores.	ISO/IEC 27001 A.12.4 (Logging and Monitoring)
	Debe ser posible rastrear los cambios realizados en cada despliegue.	ITIL – Control de Cambios / DevSecOps Practices
Evaluar la separación de ambientes.	El entorno debe permitir una separación clara entre desarrollo, pruebas y producción.	OWASP Deployment Controls / DevOps Handbook

	La configuración por entorno debe ser gestionada a través de variables, perfiles o entornos virtuales independientes.	Chef Environments Best Practices
Elaborar una matriz de riesgos.	Se deben identificar riesgos con base en evidencias técnicas, y evaluarlos por impacto y probabilidad.	ISO 31000 (Gestión de Riesgos) / NIST Risk Management Framework

## 9. Información Administrativa

Cargo	Nombre y Apellidos	Rol / Especialidad	Participación
Auditor Líder	Josue Chambilla	Seguridad de la Información / DevOps	Coordinación general de la auditoría, validación de hallazgos, redacción del informe final.
Auditor Técnico	Josue Chambilla	Automatización de Infraestructura / Chef	Análisis del código fuente, ejecución de entorno, revisión de configuraciones.
Auditor Asistente	Josue Chambilla	Soporte Técnico / Evidencia Documental	Captura de evidencias gráficas, soporte en documentación y elaboración de anexos.

## 10. Documento a Emitir

Como resultado del proceso de auditoría realizado al sistema de despliegue automatizado Chef\_Vagrant\_Wp utilizado por DevIA360, se emitirá el siguiente documento:

### Nombre del Documento a Emitir

Informe Técnico de Auditoría de Tecnologías de la Información sobre el Sistema Automatizado de Despliegue Chef\_Vagrant\_Wp en DevIA360

### Descripción del Documento

El documento a emitir es un informe técnico que consolidará todos los hallazgos, análisis, evidencias y conclusiones obtenidas durante la auditoría realizada al sistema de despliegue automatizado utilizado por DevIA360. Este informe tiene como objetivo principal

proporcionar a la gerencia una visión clara y detallada del estado actual del sistema, en cuanto a su seguridad, eficiencia, configuración, trazabilidad y cumplimiento de buenas prácticas.

### **Contenido Estructurado del Informe:**

El informe contendrá las siguientes secciones clave:

1. **Carátula y Datos Generales**

Información institucional, datos de la auditoría, fecha y participantes.

2. **Origen y Justificación de la Auditoría**

Motivo que originó la revisión técnica, antecedentes y contexto del sistema evaluado.

3. **Objetivos de la Auditoría**

Objetivo general y específicos, alineados al enfoque técnico y de seguridad.

4. **Alcance de la Auditoría**

Límites de la auditoría, recursos evaluados y restricciones.

5. **Criterios de Evaluación**

Normas, estándares y buenas prácticas utilizadas como referencia técnica.

6. **Procedimientos de Auditoría**

Actividades desarrolladas, herramientas utilizadas y técnicas aplicadas.

7. **Cronograma de Ejecución**

Fechas, etapas y duración de cada fase del trabajo de auditoría.

8. **Matriz de Riesgos y Evidencias Técnicas**

Riesgos detectados, causas, niveles de impacto y anexos que los sustentan.

9. **Análisis y Resultados Detallados**

Evaluación técnica de configuraciones, credenciales, versiones de software, segmentación de entornos y registros.

10. **Recomendaciones de Mejora**

Propuestas de solución concretas, viables y alineadas a buenas prácticas de DevSecOps y gestión de infraestructura.

## **11. Conclusiones Finales**

Síntesis de los hallazgos más relevantes y su impacto en la operación segura del sistema.

## **12. Anexos Técnicos**

Capturas de pantalla, fragmentos de código, archivos de configuración, comandos ejecutados y cualquier evidencia relevante.

### **Finalidad del Documento**

Este informe técnico tiene como propósito:

- Proveer a DevIA360 de un diagnóstico preciso sobre el estado actual de su sistema automatizado de despliegue.
- Permitir la identificación y mitigación oportuna de riesgos técnicos y de seguridad.
- Fomentar la adopción de mejores prácticas en automatización, gestión de configuraciones y control de versiones.
- Servir como evidencia documental para auditorías futuras, procesos de mejora continua, certificaciones o revisiones internas.
- Facilitar la toma de decisiones estratégicas por parte de la alta dirección en cuanto a seguridad de la infraestructura y cumplimiento.

## 11. ANEXOS

### 11.1. Preparación del Entorno

Primero tenemos que ir a:

# Ubicación: `vmagrant.d\gems\3.3.8\gems\dotenv-0.11.1\lib\dotenv.rb`

donde tenemos que cambiar la palabra `"exists"` a `"exist"` como se muestra en la imagen

```
EXPLORADOR  ...  README.md  dotenv.rb x
C:\Users\HP> .\vmagrant.d\gems\3.3.8\gems\dotenv-0.11.1\lib\dotenv.rb
1  require 'dotenv/parser'
2  require 'dotenv/environment'
3
4  module Dotenv
5  def self.load(*filenames)
6  | with(*filenames) { |f| Environment.new(f).apply if File.exists?(f) }
7  end
8
9  # same as 'load', but raises Errno::ENOENT if any files don't exist
10 def self.load!(*filenames)
11 | with(*filenames) { |f| Environment.new(f).apply }
12 end
13
14 # same as 'load', but will override existing values in 'ENV'
15 def self.overload(*filenames)
16 | with(*filenames) { |f| Environment.new(f).apply if File.exists?(f) }
17 end
18
19 protected
20
21 def self.with(*filenames, &block)
22 | filenames << '.env' if filenames.empty?
23
24 | filenames.inject({}) do |hash, filename|
25 | | filename = File.expand_path filename
26 | | hash.merge(block.call(filename) || {})
27 | end
28 end
29 end
30
```

Después ejecutamos instalamos el plugin `vagrant-env` para poder cargar variables de ambiente desde el archivo `.env` con el comando:

`vagrant plugin install vagrant-env`

También debes instalar la gema `serverspec` para poder ejecutar las pruebas de integración e infraestructura:

`gem install serverspec`

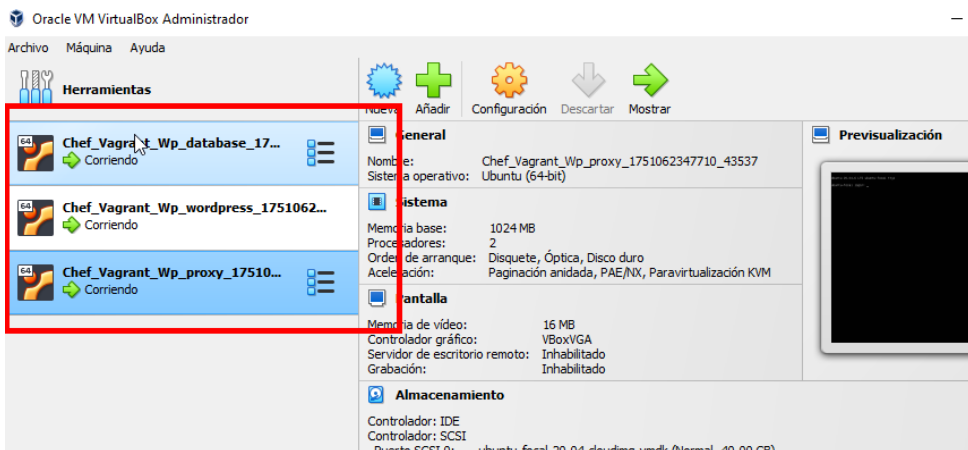
Para levantar las dos máquinas virtuales con Ubuntu 20.04 ejecuta el comando:

`vagrant up`

```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACION  TERMINAL  PUERTOS

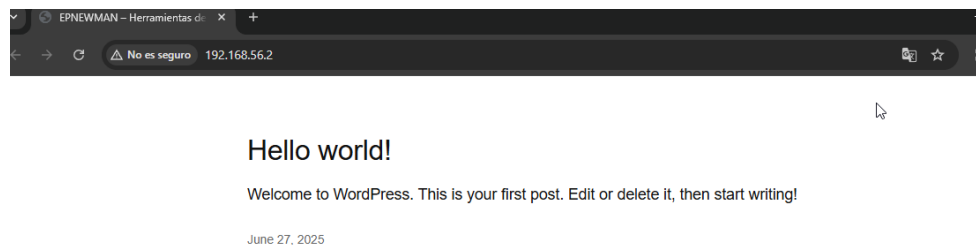
==> proxy:
==> proxy: -# server {
==> proxy:
==> proxy: -# listen localhost:143;
==> proxy: -# protocol imap;
==> proxy: -# proxy on;
==> proxy: -# }
==> proxy: -#
==> proxy: [2025-06-27T22:13:17+00:00] INFO: template[/etc/nginx/nginx.conf] sending restart action to service[nginx] (immediate)
==> proxy: * service[nginx] action restart
==> proxy: [2025-06-27T22:13:17+00:00] INFO: service[nginx] restarted
==> proxy:
==> proxy: [2025-06-27T22:13:17+00:00] INFO: service[nginx] restarted
==> proxy: [2025-06-27T22:13:17+00:00] INFO: service[nginx] restarted
==> proxy: [2025-06-27T22:13:17+00:00] INFO: service[nginx] restarted
==> proxy: [2025-06-27T22:13:17+00:00] INFO: service[nginx] restarted
==> proxy:
==> proxy: - restart service service[nginx]
==> proxy: [2025-06-27T22:13:17+00:00] INFO: service[nginx] restarted
==> proxy: [2025-06-27T22:13:17+00:00] INFO: service[nginx] restarted
==> proxy: [2025-06-27T22:13:17+00:00] INFO: service[nginx] restarted
==> proxy:
==> proxy: - restart service service[nginx]
==> proxy: [2025-06-27T22:13:17+00:00] INFO: Chef Infra Client Run complete in 7.428748062 seconds==> proxy:
==> proxy: Running handlers:
==> proxy: [2025-06-27T22:13:17+00:00] INFO: Running report handlers
==> proxy: Running handlers complete
==> proxy: [2025-06-27T22:13:17+00:00] INFO: Report handlers complete
==> proxy: Infra Phase complete, 4/6 resources updated in 08 seconds
PS C:\Users\WP\Desktop\Chef_Vagrant_Wp>
```

También se puede visualizar en el Virtual Box que las máquinas virtuales se levantaron correctamente



## 11.2. Visualizar WordPress

Una vez que se hayan levantado todas las VMs podrás acceder a Wordpress en la página: <http://192.168.56.2/>



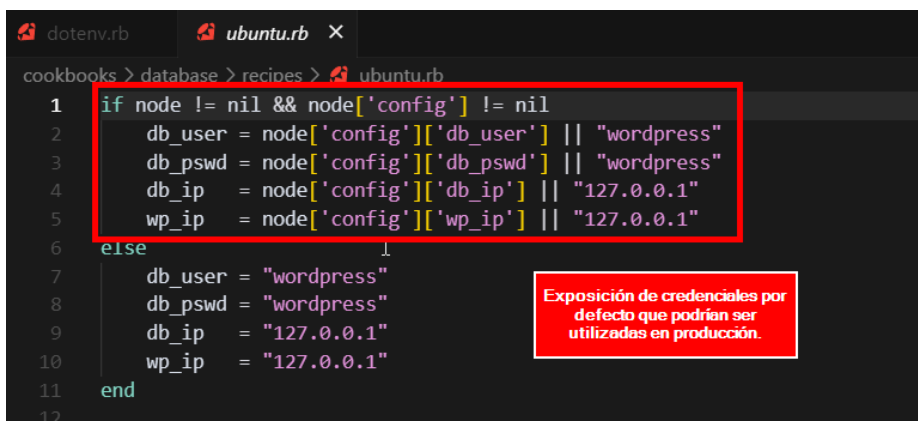
### EPNEWMAN – Herramientas de automatización de despliegues

- Blog
- About
- FAQs
- Authors
- Events
- Shop
- Patterns
- Themes

## 11.3. Examen y Hallazgos

### Exposición de Credenciales en Código Fuente

# Ubicación: *cookbooks/database/recipes/ubuntu.rb*



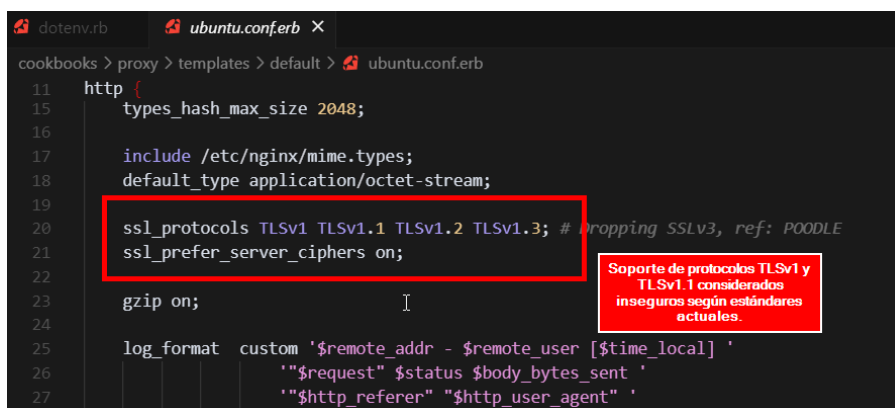
### Impacto Detallado:

- **Seguridad:** Compromiso potencial de la base de datos por credenciales predecibles y expuestas

- **Datos:** Posible acceso no autorizado a información sensible de WordPress y usuarios
- **Cumplimiento:** Violación de estándares de seguridad (ISO 27001, GDPR)
- **Reputacional:** Pérdida de confianza si se comprometen datos de usuarios
- **Financiero:** Costos asociados a la respuesta a incidentes y posibles multas por incumplimiento
- **Operacional:** Necesidad de cambio de credenciales en todos los sistemas si son comprometidas

### SSL/TLS Inseguro

# Ubicación: *cookbooks/proxy/templates/default/ubuntu.conf.erb*



```
cookbooks > proxy > templates > default > ubuntu.conf.erb
11 http {
15     types_hash_max_size 2048;
16
17     include /etc/nginx/mime.types;
18     default_type application/octet-stream;
19
20     ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3; # dropping SSLv3, ref: POODLE
21     ssl_prefer_server_ciphers on;
22
23     gzip on;
24
25     log_format custom '$remote_addr - $remote_user [$time_local] '
26         '"$request" $status $body_bytes_sent '
27         '"$http_referer" "$http_user_agent" '
28         '"$http_accept_encoding" ';
```

Soporte de protocolos TLSv1 y TLSv1.1 considerados inseguros según estándares actuales.

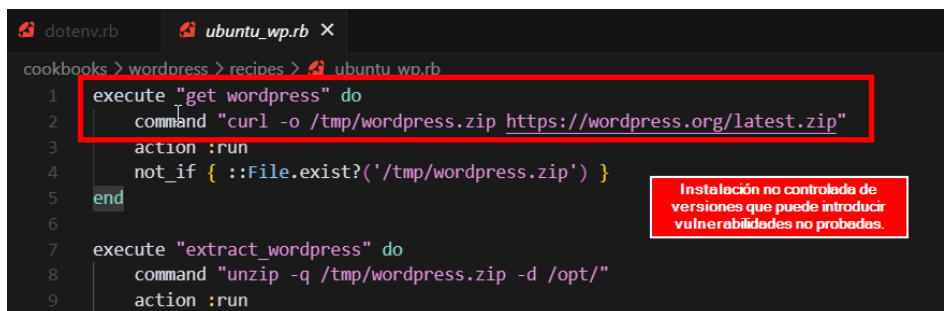
### Impacto Detallado:

- **Seguridad:** Vulnerabilidad a ataques Man-in-the-Middle y downgrade attacks
- **Cumplimiento:** Incumplimiento de requisitos PCI-DSS que prohíben TLS 1.0/1.1
- **Rendimiento:** Uso innecesario de protocolos obsoletos que pueden ralentizar conexiones
- **Compatibilidad:** Posibles problemas con navegadores modernos que rechazan protocolos antiguos
- **Reputacional:** Alertas de seguridad en navegadores modernos
- **Legal:** Posible incumplimiento de regulaciones de protección de datos



## Sin Control de Versiones

# Ubicación: cookbooks/wordpress/recipes/ubuntu\_wp.rb



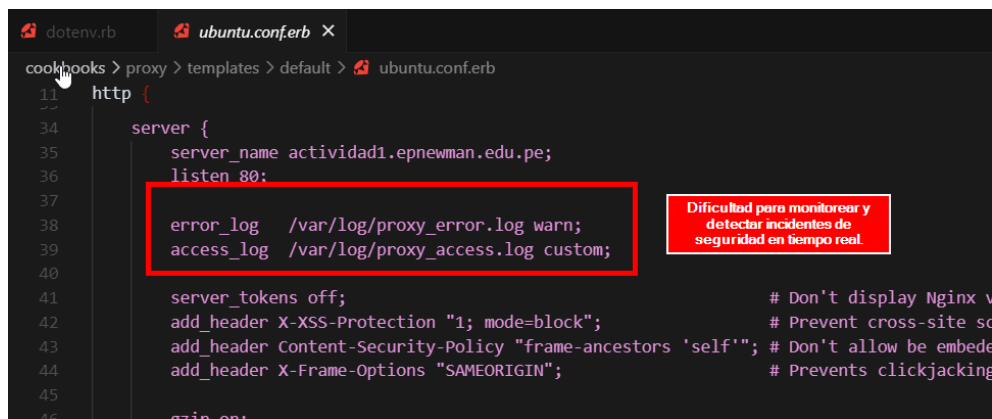
```
dotenv.rb  ubuntu_wp.rb X
cookbooks > wordpress > recipes > ubuntu_wp.rb
1  execute "get_wordpress" do
2    command "curl -o /tmp/wordpress.zip https://wordpress.org/latest.zip"
3    action :run
4    not_if { ::File.exist?('/tmp/wordpress.zip') }
5  end
6
7  execute "extract_wordpress" do
8    command "unzip -q /tmp/wordpress.zip -d /opt/"
9    action :run
```

### Impacto Detallado:

- **Estabilidad:** Riesgo de incompatibilidad entre versiones de plugins y temas
- **Seguridad:** Posible introducción de vulnerabilidades no probadas
- **Operacional:** Dificultad para realizar rollbacks en caso de problemas
- **Mantenimiento:** Imposibilidad de reproducir ambientes exactos
- **Auditoría:** Pérdida de trazabilidad de versiones desplegadas
- **Rendimiento:** Posibles problemas de rendimiento con versiones no probadas

## Logs No Centralizados

# Ubicación: cookbooks/proxy/templates/default/ubuntu.conf.erb



```
dotenv.rb  ubuntu.conf.erb X
cookbooks > proxy > templates > default > ubuntu.conf.erb
11 http {
12
34   server {
35     server_name actividad1.epnewman.edu.pe;
36     listen 80;
37
38     error_log /var/log/proxy_error.log warn;
39     access_log /var/log/proxy_access.log custom;
40
41     server_tokens off;
42     add_header X-XSS-Protection "1; mode=block";
43     add_header Content-Security-Policy "frame-ancestors 'self'";
44     add_header X-Frame-Options "SAMEORIGIN";
45
46     gzip on;
```

### Impacto Detallado:

- **Monitoreo:** Dificultad para detectar patrones de ataque en tiempo real
- **Forense:** Complejidad en investigaciones post-incidente
- **Operacional:** Mayor tiempo de respuesta ante incidentes

- **Cumplimiento:** Dificultad para demostrar controles de auditoría
- **Eficiencia:** Tiempo excesivo en análisis de logs distribuidos
- **Costos:** Incremento en recursos necesarios para gestión de logs

## Sin Separación de Ambientes

# Ubicación: *Vagrantfile*

```
1 Vagrant.configure("2") do |config|
4   if ENV['TESTS'] == 'true'
5     config.vm.define "test" do |testing|
20       SHELL
21     end
22   else
23     config.vm.define "database" do |db|
24       db.vm.box = ENV["BOX_NAME"] || "ubuntu/focal64" # Utilizamos una imagen de Ubuntu 20.04 por
25       db.vm.hostname = "db.epnewman.edu.pe"
26       db.vm.network "private_network", ip: ENV["DB_IP"]
27     end
28     db.vm.provision "chef_solo" do |chef|
29       chef.install = "true"
30       chef.arguments = "chef-license-accept"
```

Mismo código y configuración para todos los ambientes sin distinción.

## Impacto Detallado:

- **Seguridad:** Riesgo de exponer configuraciones de producción en desarrollo
- **Calidad:** Imposibilidad de garantizar pruebas en ambientes idénticos
- **Operacional:** Riesgo de despliegues incorrectos entre ambientes
- **Desarrollo:** Conflictos entre equipos trabajando en diferentes features
- **Costos:** Ineficiencia en uso de recursos por falta de optimización por ambiente
- **Mantenimiento:** Complejidad en gestión de configuraciones específicas

## Puertos Expuestos

# Ubicación: *cookbooks/proxy/recipes/default.rb*

```
21 when 'rhel', 'fedora'
36 end
37
38 execute 'firewall-cmd --zone=public --add-port=80/tcp --permanent' do
39   action :run
40 end
41
42 execute 'firewall-cmd --reload' do
43   action :run
44 end
45 end
```

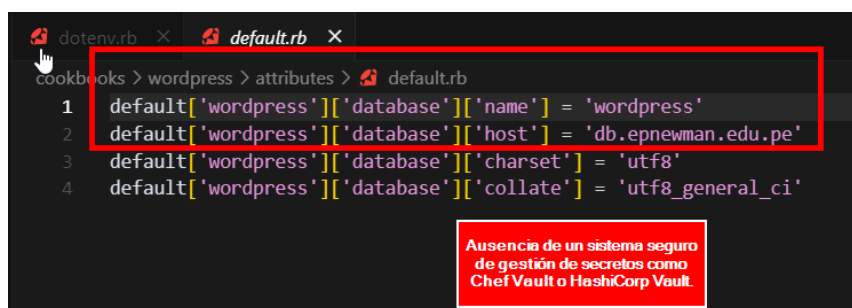
Configuración permisiva de firewall sin restricciones específicas por ambiente.

**Impacto Detallado:**

- **Seguridad:** Exposición innecesaria de servicios a Internet
- **Superficie de Ataque:** Incremento en puntos de entrada potenciales
- **Control:** Falta de granularidad en accesos por ambiente
- **Monitoreo:** Dificultad para detectar accesos no autorizados
- **Cumplimiento:** Violación de principio de mínimo privilegio
- **Rendimiento:** Sobrecarga potencial por tráfico no filtrado

**Sin Gestión de Secretos**

# Ubicación: *cookbooks/wordpress/attributes/default.rb*



```
dotenv.rb x default.rb x
cookbooks > wordpress > attributes > default.rb
1 default['wordpress']['database']['name'] = 'wordpress'
2 default['wordpress']['database']['host'] = 'db.epnewman.edu.pe'
3 default['wordpress']['database']['charset'] = 'utf8'
4 default['wordpress']['database']['collate'] = 'utf8_general_ci'
```

Ausencia de un sistema seguro de gestión de secretos como Chef Vault o HashiCorp Vault.

**Impacto Detallado:**

- **Seguridad:** Exposición de información sensible en control de versiones
- **Gestión:** Dificultad para rotar credenciales de forma segura
- **Auditoría:** Falta de control sobre acceso a secretos
- **Cumplimiento:** Violación de políticas de gestión de accesos
- **Operacional:** Riesgo de fugas de información en logs y backups
- **Desarrollo:** Exposición de secretos en entornos de desarrollo
- **Legal:** Incumplimiento de regulaciones de protección de datos

## 11.4. Matriz de Riesgos

Riesgo Identificado	Causa Específica (Evidencia Técnica / Anexo)	Nivel de Riesgo	Probabilidad de Ocurrencia (%)	Impacto
Exposición de Credenciales en Código Fuente	Exposición de credenciales por defecto que podrían ser utilizadas en producción. Ubicación: `cookbooks/database/recipes/ubuntu.rb`	Alto	90%	Alto
Configuración Insegura de SSL/TLS	Soporte de protocolos TLSv1 y TLSv1.1 considerados inseguros según estándares actuales. Ubicación: `cookbooks/proxy/templates/default/ubuntu.conf.erb`	Medio	75%	Medio
Ausencia de Control de Versiones de Software	Instalación no controlada de versiones que puede introducir vulnerabilidades no probadas. Ubicación: `cookbooks/wordpress/recipes/ubuntu_wp.rb`	Medio	80%	Medio
Falta de Centralización de Logs	Dificultad para monitorear y detectar incidentes de seguridad en tiempo real. Ubicación: `cookbooks/proxy/templates/default/ubuntu.conf.erb`	Bajo	60%	Bajo
Ausencia de Separación de Ambientes	Mismo código y configuración para todos los ambientes sin distinción. Ubicación: `Vagrantfile`	Alto	85%	Alto
Exposición de Puertos sin Restricción	Configuración permisiva de firewall sin restricciones específicas por ambiente. Ubicación: `cookbooks/proxy/recipes/default.rb`	Medio	70%	Medio
Ausencia de Gestión Segura de Secretos	Ausencia de un sistema seguro de gestión de secretos como Chef Vault o HashiCorp Vault. Ubicación: `cookbooks/wordpress/attributes/default.rb`	Alto	95%	Alto

