

PE02 - Infraestructura WordPress Multi-máquina

Peso: 35% de la nota

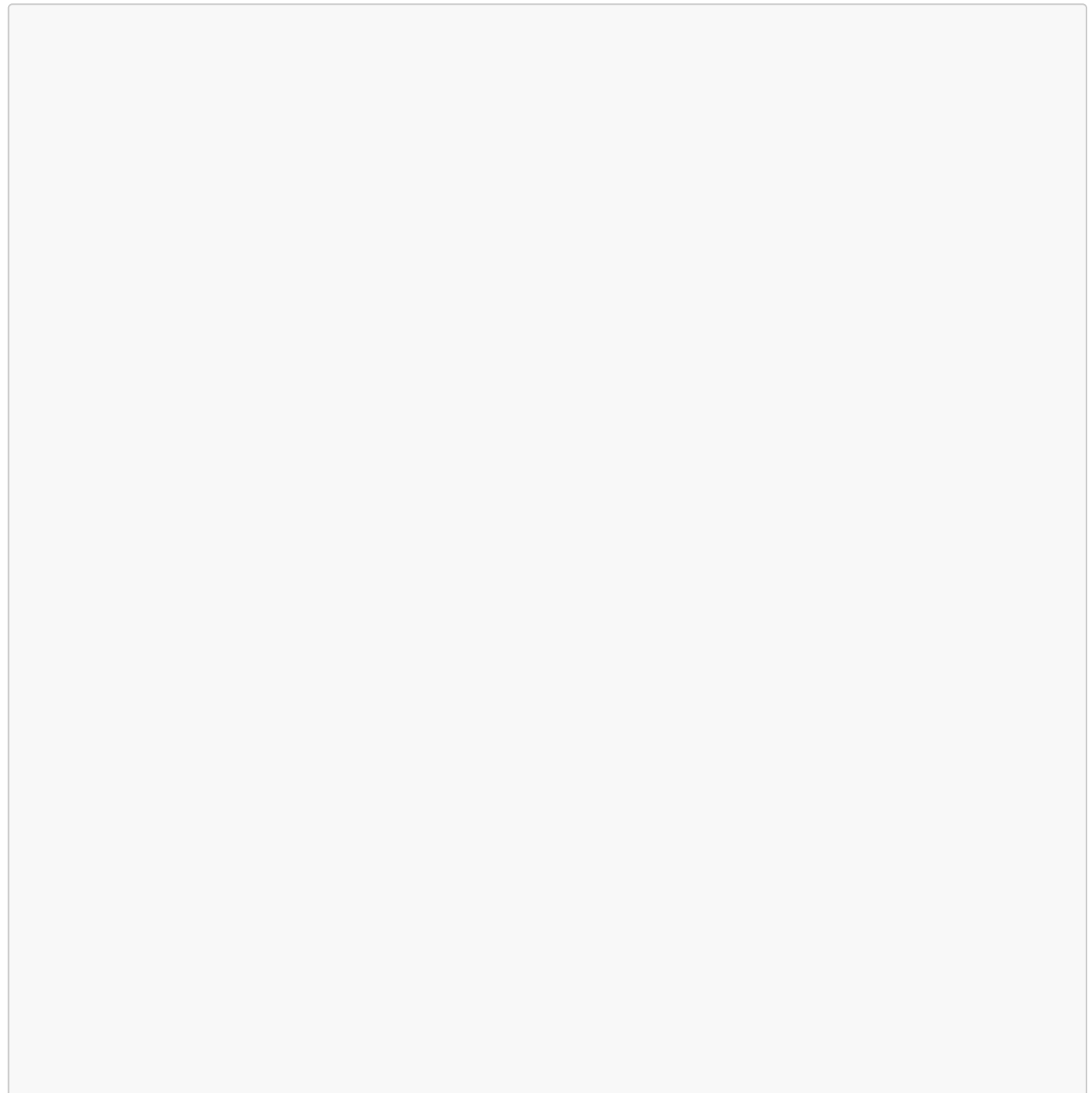
Duración: 3-4 horas

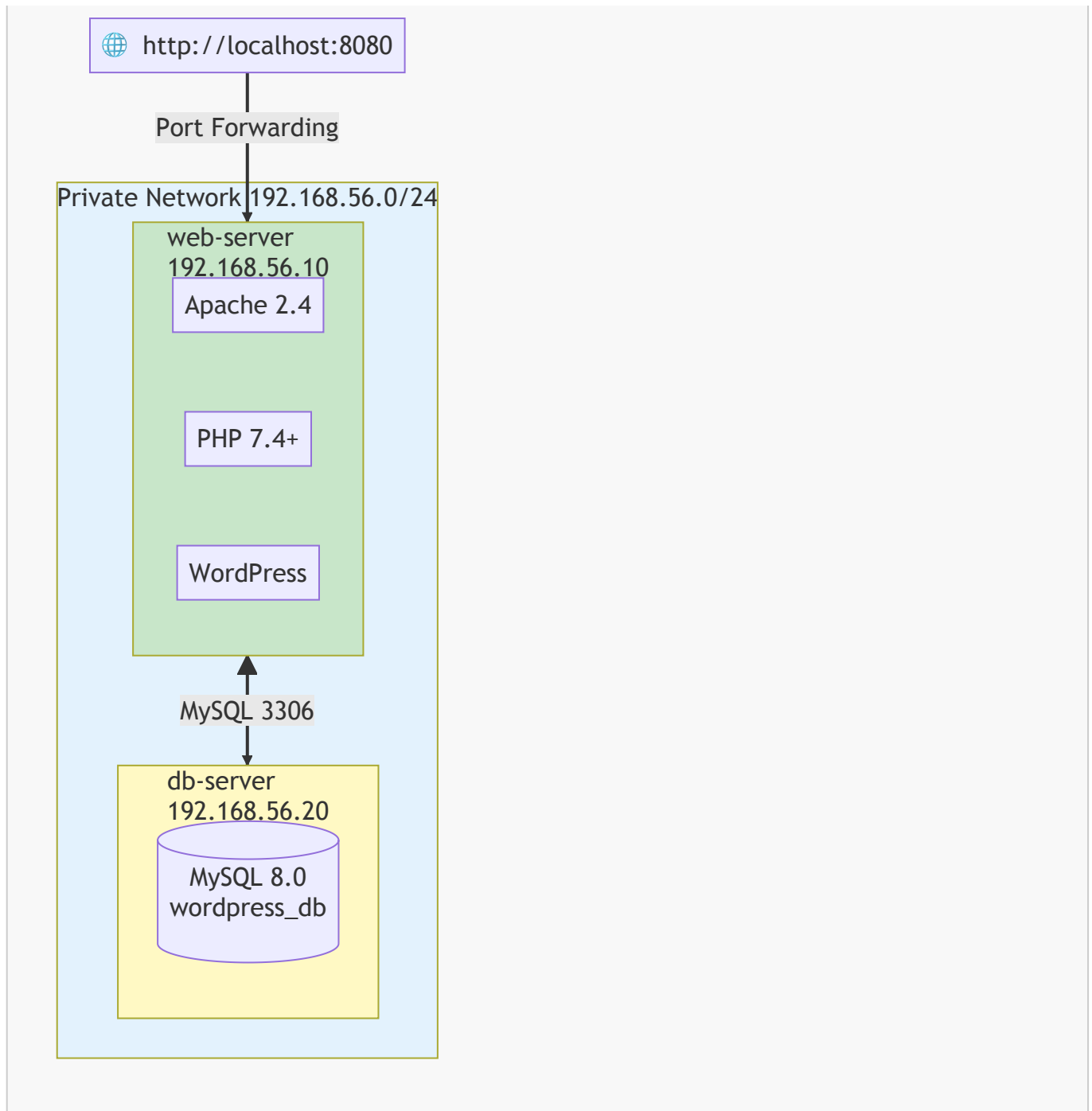
Dificultad: ★★☆☆ Media-Alta

Objetivo

Crear una infraestructura WordPress profesional con separación de servicios: un servidor web con Apache y PHP, y un servidor de base de datos MySQL separado. Ambos comunicándose a través de red privada.

Arquitectura requerida





Requisitos mínimos (6 puntos)

1. Dos VMs funcionando (1 punto)

- web-server con Apache + PHP
- db-server con MySQL

2. Networking correcto (1 punto)

- Red privada entre VMs
- Port forwarding del host al web-server

3. MySQL configurado remotamente (1.5 puntos)

- Escucha en 0.0.0.0

- Usuario con permisos desde red privada
- Base de datos creada

4. **WordPress instalado** (1.5 puntos)

- Descargado e instalado
- Conectado a BD remota
- Accesible desde navegador

5. **Provisioning automatizado** (1 punto)

- Sin intervención manual

Requisitos avanzados (3 puntos)

6. **Resolución de nombres** (0.5 puntos)

- `/etc/hosts` configurado en ambas VMs

7. **wp-config.php autoconfigurable** (1 punto)

- Generado automáticamente
- Credenciales correctas

8. **Permisos correctos** (0.5 puntos)

- `www-data` propietario de WordPress

9. **Scripts organizados** (1 punto)

- Scripts separados para cada servicio

Requisitos extras (1 punto)

10. **Cache de opcodes** (0.25 puntos)

- PHP OPcache habilitado

11. **Seguridad MySQL** (0.25 puntos)

- Solo usuario necesario, sin root remoto

12. **README profesional** (0.5 puntos)

- Documentación completa con diagramas

Especificaciones técnicas

VM web-server

```
Hostname: web-server
IP: 192.168.56.10
RAM: 1024 MB
Puerto: 8080 → 80
Disco: Configuración por defecto

Software:
- Apache 2.4
- PHP 7.4+ con extensiones WordPress
- WordPress (última versión)
```

VM db-server

```
Hostname: db-server
IP: 192.168.56.20
RAM: 2048 MB
Disco: Configuración por defecto

Software:
- MySQL 8.0

Base de datos:
- Nombre: wordpress_db
- Usuario: wp_user
- Password: wp_secure_pass
- Host permitido: 192.168.56.%
```

Estructura del proyecto

```
PE02-WordPress/
├── Vagrantfile
├── README.md
├── scripts/
│   ├── common.sh
│   ├── install-web.sh
│   ├── install-db.sh
│   └── configure-wordpress.sh
├── config/
│   └── wordpress.conf
```

Entregables

1. **Vagrantfile** completo y funcional
2. **Scripts de provisioning** organizados

3. README.md con:

- Diagrama de arquitectura
- Instrucciones paso a paso
- Credenciales de acceso
- Capturas de pantalla

4. Configuración de virtual host (opcional)

Criterios de evaluación

Aspecto	Puntos	Descripción
Funcionalidad	4	Todo funciona correctamente
Multi-máquina	2	Comunicación correcta entre VMs
Automatización	2	Sin intervención manual
Organización	1	Código limpio y modular
Documentación	1	README completo

Guía de implementación

1. Vagrantfile estructura

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

# Variables
SUBNET = "192.168.56"
WEB_IP = "#{SUBNET}.10"
DB_IP = "#{SUBNET}.20"

Vagrant.configure("2") do |config|
  config.vm.box = "ubuntu/focal64"

  # === DB SERVER (primero para que esté listo) ===
  config.vm.define "db" do |db|
    db.vm.hostname = "db-server"
    db.vm.network "private_network", ip: DB_IP

    db.vm.provider "virtualbox" do |vb|
      vb.name = "WP-Database"
      vb.memory = "2048"
    end

    db.vm.provision "shell", path: "scripts/common.sh"
    db.vm.provision "shell", path: "scripts/install-db.sh"
  end

  # === WEB SERVER ===
```

```

config.vm.define "web", primary: true do |web|
  web.vm.hostname = "web-server"
  web.vm.network "private_network", ip: WEB_IP
  web.vm.network "forwarded_port", guest: 80, host: 8080

  web.vm.provider "virtualbox" do |vb|
    vb.name = "WP-WebServer"
    vb.memory = "1024"
  end

  web.vm.provision "shell", path: "scripts/common.sh"
  web.vm.provision "shell", path: "scripts/install-web.sh"
  web.vm.provision "shell", path: "scripts/configure-wordpress.sh", env:
  {
    "DB_HOST" => DB_IP,
    "DB_NAME" => "wordpress_db",
    "DB_USER" => "wp_user",
    "DB_PASS" => "wp_secure_pass"
  }
end
end

```

2. Script common.sh

```

#!/bin/bash
set -e

echo "=== Configuración común ==="

# Actualizar sistema
apt-get update

# Instalar utilidades
apt-get install -y vim curl wget net-tools

# Configurar /etc/hosts
cat >> /etc/hosts <<EOF
192.168.56.10 web-server
192.168.56.20 db-server
EOF

# Configurar timezone
timedatectl set-timezone Europe/Madrid

echo "=== Configuración común completada ==="

```

3. Script install-db.sh

```
#!/bin/bash
set -e
export DEBIAN_FRONTEND=noninteractive

echo "=== Instalando MySQL ==="
apt-get install -y mysql-server

echo "=== Configurando MySQL para acceso remoto ==="
# Cambiar bind-address
sed -i 's/bind-address.*/bind-address = 0.0.0.0/'
/etc/mysql/mysql.conf.d/mysqld.cnf

systemctl restart mysql

echo "=== Creando base de datos y usuario ==="
mysql <<EOF
-- Crear base de datos
CREATE DATABASE IF NOT EXISTS wordpress_db DEFAULT CHARACTER SET utf8mb4
COLLATE utf8mb4_unicode_ci;

-- Crear usuario con acceso desde red privada
CREATE USER IF NOT EXISTS 'wp_user'@'192.168.56.%' IDENTIFIED BY
'wp_secure_pass';

-- Otorgar permisos
GRANT ALL PRIVILEGES ON wordpress_db.* TO 'wp_user'@'192.168.56.%';
FLUSH PRIVILEGES;

-- Verificar
SHOW DATABASES;
SELECT User, Host FROM mysql.user;
EOF

echo "=== MySQL configurado correctamente ==="
echo "Base de datos: wordpress_db"
echo "Usuario: wp_user (acceso desde 192.168.56.%)"
```

4. Script install-web.sh

```
#!/bin/bash
set -e
export DEBIAN_FRONTEND=noninteractive

echo "=== Instalando Apache ==="
apt-get install -y apache2

echo "=== Instalando PHP y extensiones ==="
apt-get install -y php libapache2-mod-php php-mysql php-curl php-gd \
    php-mbstring php-xml php-xmlrpc php-zip php-intl php-opcache
```

```

echo "=== Habilitando módulos Apache ==="
a2enmod rewrite

echo "=== Descargando WordPress ==="
cd /tmp
wget -q https://wordpress.org/latest.tar.gz
tar -xzf latest.tar.gz

echo "=== Instalando WordPress ==="
rm -rf /var/www/html/*
cp -r wordpress/* /var/www/html/

echo "=== Configurando permisos ==="
chown -R www-data:www-data /var/www/html
find /var/www/html/ -type d -exec chmod 755 {} \;
find /var/www/html/ -type f -exec chmod 644 {} \;

echo "=== Configurando Apache VirtualHost ==="
cat > /etc/apache2/sites-available/wordpress.conf <<'EOF'
<VirtualHost *:80>
    DocumentRoot /var/www/html

    <Directory /var/www/html>
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/wordpress_error.log
    CustomLog ${APACHE_LOG_DIR}/wordpress_access.log combined
</VirtualHost>
EOF

a2dissite 000-default.conf
a2ensite wordpress.conf

systemctl restart apache2

echo "=== Apache y PHP instalados ==="

```

5. Script configure-wordpress.sh

```

#!/bin/bash
set -e

echo "=== Configurando WordPress ==="

cd /var/www/html

# Crear wp-config.php
cp wp-config-sample.php wp-config.php

```



```

# Configurar credenciales de base de datos
sed -i "s/database_name_here/$DB_NAME/" wp-config.php
sed -i "s/username_here/$DB_USER/" wp-config.php
sed -i "s/password_here/$DB_PASS/" wp-config.php
sed -i "s/localhost/$DB_HOST/" wp-config.php

# Generar salt keys
SALT_KEYS=$(curl -s https://api.wordpress.org/secret-key/1.1/salt/)

# Reemplazar las claves de ejemplo con las generadas
php <<ENDPHP
<?php
\${config} = file_get_contents('/var/www/html/wp-config.php');

// Definir las claves a reemplazar
\${keys} = array(
    'AUTH_KEY',
    'SECURE_AUTH_KEY',
    'LOGGED_IN_KEY',
    'NONCE_KEY',
    'AUTH_SALT',
    'SECURE_AUTH_SALT',
    'LOGGED_IN_SALT',
    'NONCE_SALT'
);

// Reemplazar cada clave con una vacía primero
foreach (\${keys} as \${key}) {
    \${config} = preg_replace(
        "/define\(\s*'\${key}',\s*'[^']*'\s*\);/",
        "define('\${key}', '$(openssl rand -base64 32)');",
        \${config}
    );
}

file_put_contents('/var/www/html/wp-config.php', \${config});
?>
ENDPHP

# Añadir configuraciones adicionales
cat >> wp-config.php <<'EOF'

/* Configuración adicional */
define('WP_DEBUG', false);
define('WP_AUTO_UPDATE_CORE', false);
define('DISALLOW_FILE_EDIT', true);

/* Dirección del sitio */
define('WP_SITEURL', 'http://192.168.56.10');
define('WP_HOME', 'http://192.168.56.10');
EOF

echo "=== Verificando conexión a base de datos ==="
php -r "

```

```

try {
    \ $pdo = new PDO('mysql:host=$DB_HOST;dbname=$DB_NAME', '$DB_USER',
'$DB_PASS');
    echo 'Conexión a BD exitosa!';
} catch (PDOException \ $e) {
    echo 'Error: ' . \ $e->getMessage();
    exit(1);
}
"

chown www-data:www-data wp-config.php

echo "=== WordPress configurado correctamente ==="
echo ""
echo "Accede a: http://localhost:8080"
echo "O directamente: http://192.168.56.10"

```

Pruebas a realizar

```

# 1. Levantar infraestructura
vagrant up

# 2. Verificar que ambas VMs corren
vagrant status

# 3. Verificar conectividad entre VMs
vagrant ssh web -c "ping -c 3 192.168.56.20"
vagrant ssh web -c "ping -c 3 db-server"

# 4. Verificar MySQL desde web
vagrant ssh web -c "mysql -h 192.168.56.20 -u wp_user -pwp_secure_pass -e
'SHOW DATABASES;'"

# 5. Verificar WordPress en navegador
# http://localhost:8080 → Debería mostrar instalador de WordPress

# 6. Completar instalación de WordPress
# - Título del sitio
# - Usuario admin
# - Contraseña
# - Email

# 7. Verificar funcionamiento
# http://localhost:8080/wp-admin

```

Troubleshooting

MySQL rechaza conexiones remotas

```
vagrant ssh db
sudo mysql -e "SELECT User, Host FROM mysql.user WHERE User='wp_user';"
sudo netstat -tlnp | grep 3306
sudo tail -f /var/log/mysql/error.log
```

WordPress no conecta a BD

```
vagrant ssh web
cat /var/www/html/wp-config.php | grep DB_
php -r "new PDO('mysql:host=192.168.56.20;dbname=wordpress_db', 'wp_user', 'wp_secure_pass');"

```

Rúbrica

Criterio	Suspenso	Aprobado	Notable	Sobresaliente
Multi-VM	No funcionan	1 funciona	Ambas funcionan	Comunicación perfecta
Networking	Sin red privada	Red parcial	Red funcional	Resolución nombres
MySQL	No instala	Instala local	Acceso remoto	Seguridad aplicada
WordPress	No instala	Instalación manual	Automático parcial	Totalmente automático
Documentación	Sin README	README básico	README completo	README ejemplar

Fecha de entrega

[Definir según calendario del curso]

Siguiente práctica: [PE03 - Proyecto Final](#)