PROXMOX VE · SESIÓN 4 — CONFIGURACIÓN ESPECÍFICA DE PROXMOX VE EN EL IES GONZALO NAZARENO

# Virtualización con Proxmox VE

## Sesión 4 · Configuración específica de Proxmox VE en el IES Gonzalo Nazareno


📧 José Domingo Muñoz

🏫 IES Gonzalo Nazareno · Dos Hermanas


🌐 josedom24.github.io/curso_proxmox_2026

💻 github.com/josedom24/curso_proxmox_2026

01

# Configuración específica en nuestro centro

Objetivos y principios

# Conceptos clave (I)

## POOLS DE RECURSOS

Agrupación lógica de:

- **Máquinas virtuales (MV)**
- **Contenedores (LXC)**
- **Almacenamiento**

Para facilitar su gestión y asignación de cuotas.

Cada usuario o departamento puede tener su propio pool.

Permite **organizar y limitar** los recursos disponibles.

## USUARIOS Y GRUPOS

**Usuarios**: cuentas individuales que acceden a Proxmox (integrados con LDAP del centro).

**Grupos**: agrupaciones de usuarios para facilitar la asignación de permisos comunes.

Simplifica la **administración de permisos a escala**.

# Conceptos clave (II)

### ROL

Conjunto predefinido de **permisos** asociados a una función específica.

Ejemplos: *Administrator*, *PVEAdmin*, *PVEUser*, *PVEVMUser*

Cada rol define un **conjunto de capacidades**.

### PERMISOS

Reglas que definen **qué acciones puede hacer** cada usuario sobre cada recurso.

Se asignan combinando: usuario/grupo + rol + recurso (nodo, VM, pool, etc.)

Control **granular y flexible**.

# ¿Qué queríamos conseguir? (I)

## CONTROL DE RECURSOS

Usuarios que **controlen sus propios recursos** en Proxmox:

- Máquinas virtuales (MV)
- Contenedores LXC
- Almacenamiento asignado

Cada usuario gestiona su espacio de forma **independiente y segura**.

## LIMITACIONES DE SEGURIDAD

Las **redes no pueden ser controladas** por usuarios — es una restricción de seguridad.

Solo el administrador configura la topología de red, bridges y VLANs.

**Garantiza la integridad** de la infraestructura de red.

# ¿Qué queríamos conseguir? (II)

## CREACIÓN RÁPIDA DE MÁQUINAS

Aunque los usuarios pueden crear MV desde una ISO, queremos que lo hagan de forma **rápida y ágil**.

**Solución**: **clonar plantillas predefinidas** que ya tenemos preparadas.

**Ventajas:**

- Ahorra tiempo de instalación
- Garantiza configuraciones **consistentes**
- Todos los alumnos parten del mismo estado
- Reduce errores de instalación

# Roles creados en nuestro Proxmox VE (I)

En el IES hemos definido **cuatro roles** para separar responsabilidades y permitir que cada usuario tenga solo los permisos necesarios:

`IESGN`

### Rol de usuario estándar

Usuario completo que puede **crear, modificar y gestionar** sus propias máquinas virtuales con amplia autonomía.

Sin acceso de administrador del clúster.

`IESGN-RED`

### Rol de redes

Acceso especializado para la **administración de redes SDN**.

Crear y gestionar redes virtuales sin tocar VMs ni almacenamiento.

# Roles creados en nuestro Proxmox VE (II)

`IESGN-TEMPLATE-CLONE`

### Rol para clonar plantillas

Usuario que **solo puede crear VMs** clonando plantillas ya existentes.

No puede modificar ni crear plantillas nuevas.

`IESGN-TEMPLATE-CREATE`

### Rol para crear plantillas

Usuario autorizado a **crear nuevas máquinas** que se convertirán en plantillas.

Complemento del rol anterior en el flujo de trabajo de plantillas.

# Permisos detallados por rol

| ROL | PERMISOS |
|---|---|
| **iesgn** | `Datastore.AllocateSpace` `Datastore.Audit` `Permissions.Modify` `Pool.Audit` `SDN.Use` `Sys.Audit` `Sys.Console` `Sys.Modify` `Sys.Syslog` `VM.Allocate` `VM.Audit` `VM.Console` `VM.PowerMgmt` `VM.Backup` `VM.Clone` `VM.Migrate` `VM.Snapshot` `VM.Snapshot.Rollback` `VM.Config.*` `VM.GuestAgent.*` |
| **iesgn-red** | `SDN.Allocate` `SDN.Audit` `SDN.Use` `Sys.Modify` |
| **iesgn-template-clone** | `Pool.Audit` `VM.Audit` `VM.Clone` |
| **iesgn-template-create** | `Pool.Allocate` `VM.Allocate` |

| | | |
|---|---|---|
| No | iesgn | Datastore.AllocateSpace Datastore.Audit Permissions.Modify Pool.Audit SDN.Use Sys.Audit Sys.Console Sys.Modify Sys.Syslog VM.Allocate VM.Audit VM.Backup VM.Clone VM.Config.CDROM VM.Config.CPU VM.Config.Cloudinit VM.Config.Disk VM.Config.HWType VM.Config.Memory VM.Config.Network VM.Config.Options VM.Console VM.GuestAgent.Audit VM.GuestAgent.FileRead VM.GuestAgent.FileSystemMgmt VM.GuestAgent.FileWrite VM.GuestAgent.Unrestricted VM.Migrate VM.PowerMgmt VM.Snapshot VM.Snapshot.Rollback |
| No | iesgn-red | SDN.Allocate SDN.Audit SDN.Use Sys.AccessNetwork Sys.Modify |
| No | iesgn-template-clone | Pool.Audit VM.Audit VM.Clone |
| No | iesgn-template-create | Pool.Allocate VM.Allocate |

# ¿Cómo podemos conseguirlo? (I)

### 1. Grupos de usuarios

Los usuarios se agrupan por **curso o categoría**:

- `@asir1-iesgn` — alumnos de 1º ASIR
- `@smr2-iesgn` — alumnos de 2º SMR
- `@profesores-iesgn` — profesores

Facilita la **asignación masiva de permisos**.

# ¿Cómo podemos conseguirlo? (II)

### 2. Pools de recursos

- **Pool "Proyecto de usuario"**
  - Asignado a cada usuario individual o grupo
  - Cada usuario crea sus propios recursos en su pool
  - Proporciona **aislamiento y seguridad**
- **Pool "Imágenes"**
  - Repositorio centralizado de plantillas
  - Los usuarios pueden **clonar plantillas** de aquí
  - Solo profesores pueden crear/modificar plantillas

# Asignación de permisos

## CUATRO ÁMBITOS DEL CONTROL DE ACCESO

1. **Administración global** → solo `admin@pve` en la raíz.
2. **Plantillas** → profesores las producen, todos las consumen.
3. **Proyectos de alumnos** → cada alumno (o grupo reducido) gestiona únicamente su propio pool, sin ver ni tocar los de los demás.
4. **Red SDN** → todos los colectivos pueden trabajar con redes virtuales dentro de la zona `localnetwork`, pero sin afectar la red física del clúster.

## PRINCIPIO DE DISEÑO

**Mínimo privilegio**: cada usuario recibe solo los permisos necesarios para su función, con **herencia automática** hacia los recursos dentro de su ámbito.

# Tabla de ACLs — Ejemplo práctico

| Path ↑ | User/Group/API Token | Role |
|---|---|---|
| / | admin@pve | Administrator |
| /pool/Imagenes | @smr2-iesgn | iesgn-template-clone |
| /pool/Imagenes | @profesores-iesgn | iesgn-template-create |
| /pool/Imagenes | @profesores-iesgn | iesgn-template-clone |
| /pool/Imagenes | @asir1-iesgn | iesgn-template-clone |
| /pool/Proyecto1 | [redacted]@iesgn | iesgn |
| /pool/Proyecto1 | [redacted]@iesgn | iesgn |
| /pool/Proyecto1 | [redacted]@iesgn | iesgn |
| /pool/Proyecto_[redacted] | [redacted]@iesgn | iesgn |
| /sdn/zones/localnetwork | @smr2-iesgn | iesgn-red |
| /sdn/zones/localnetwork | @asir1-iesgn | iesgn-red |
| /sdn/zones/localnetwork | @profesores-iesgn | iesgn-red |

# Posibles mejoras

### 1. ROL `IESGN-PROFESOR`

Crear un rol específico para profesores que incluya `Pool.Allocate`, permitiendo:

- Crear plantillas en su proyecto personal (sin exposición al alumnado)
- Reasignarlas a `/pool/Imagenes` cuando estén listas
- Actualmente el profesor puede hacer plantillas directamente en el `/pool/Imagenes`

### 2. ACCESO SUPERVISADO A POOLS DEL ALUMNADO

Otorgar al grupo `profesores-iesgn` acceso de auditoría o intervención sobre los pools del alumnado:

- Supervisar y evaluar el trabajo en curso
- Diagnosticar incidencias técnicas
- Intervenir en situaciones bloqueantes sin credenciales admin

02

# DEMO 1: Clonación de MV de un usuario

Perfil alumno

02

# DEMO 2: Configuración de máquinas virtuales usando cloud-init

Automatización de la configuración inicial

# Concepto de cloud-init

## ¿QUÉ ES CLOUD-INIT?

**cloud-init** es un estándar de inicialización que permite **configurar máquinas virtuales automáticamente** en el primer arranque sin intervención manual.

## ¿POR QUÉ ES NECESARIO?

- **Ahorra tiempo**: no hay que entrar manualmente en cada máquina
- **Consistencia**: todas las máquinas se configuran igual
- **Escalabilidad**: permite provisionar cientos de máquinas rápidamente
- **Flexibilidad**: cada máquina puede tener configuración única sin crear plantillas distintas

# Configuración con cloud-init

## PARÁMETROS CONFIGURABLES

Los templates con cloud-init permiten configurar:

- **Hostname** de la máquina
- **Usuario y contraseña** de acceso
- **Clave pública SSH** para acceso remoto
- **Configuración de red** (IP, DNS, rutas)
- **DNS** y otros parámetros de red

## MECANISMO DE ENTREGA

cloud-init obtiene la configuración desde un:

- **Dispositivo CDROM virtual** en el hardware de la máquina
- Contiene un archivo de configuración YAML
- Se ejecuta **solo en el primer arranque**
- Luego se desactiva automáticamente

# Templates con cloud-init en nuestro centro

## MÁQUINAS VIRTUALES BASE

Los **templates del pool "Imágenes"** son máquinas virtuales con **cloud-init ya instalado**, listos para ser clonados.

## VENTAJAS DE NUESTRO ENFOQUE

- Los usuarios **clonan una plantilla** existente
- La máquina nueva se **configura automáticamente** en el primer arranque
- Cada usuario puede personalizar:
  - Nombre de máquina
  - Credenciales de acceso
  - Configuración de red
  - Claves SSH para acceso remoto

# Soporte para otros sistemas operativos

## LINUX (CLOUD-INIT)

- Todas las distribuciones modernas incluyen cloud-init
- Compatible con Debian, Ubuntu, CentOS, etc.
- Recurso: Configuración automática de una máquina virtual de Proxmox con cloud-init

## WINDOWS (CLOUDBASE-INIT O SYSPREP)

- **cloudbase-init**: equivalente a cloud-init para Windows
- **Sysprep**: herramienta de Microsoft para generalizar la instalación. Cuando clonamos a partir del tamplate comienza la última fase de configuración (**hay que asigna nueva contraseña**).

03

# Scripts de administración

Instrucciones de línea de comandos para automatización

# La API de Proxmox VE

## ARQUITECTURA GENERAL

Proxmox expone **toda su funcionalidad** a través de una **API REST** en el puerto **8006** (HTTPS).

**URL base:** `https://<servidor>:8006/api2/json/<ruta>`

La API está **organizada jerárquicamente**:

- `/nodes/<nodo>` — operaciones sobre un nodo
- `/nodes/<nodo>/qemu/<vmid>` — gestión de VMs KVM
- `/nodes/<nodo>/lxc/<vmid>` — gestión de contenedores LXC
- `/cluster/...` — operaciones a nivel de clúster
- `/access/...` — autenticación y permisos

# Clientes de línea de comandos

| CLIENTE | DESCRIPCIÓN Y USO | ACCESO |
|---|---|---|
| `pveum` | Gestión de usuarios, grupos, roles, ACLs y tokens | Local (nodo) |
| `qm` | Gestión de máquinas virtuales KVM (crear, clonar, snapshots, templates) | Local (nodo) |
| `pct` | Gestión de contenedores LXC (crear, clonar, snapshots, templates) | Local (nodo) |
| `pvesm` | Gestión de almacenamientos y volúmenes | Local (nodo) |
| `pvesh` | **Cliente universal** — acceso a cualquier endpoint de la API sin construir HTTP manualmente | Local (nodo) |
| `pveclient` | Cliente remoto basado en red — administración desde fuera del clúster | Remoto (red) |

# Ejemplos con `pvesh` — Cliente universal

```
pvesh get /nodes                              # Listar nodos
pvesh get /pools/Imagenes                     # Ver miembros de un pool
pvesh get /access/users --output-format json  # Listar usuarios en JSON
pvesh create /nodes/proxmox1/qemu --vmid 999 \
  --name nueva --memory 2048                  # Crear VM

pvesh set /nodes/proxmox1/qemu/321/config \
  --description "Plantilla Debian 12"         # Modificar configuración

pvesh delete /pools/Antiguo                   # Eliminar pool
```

# Scripts de administración

### PYTHON (LIBRERÍA PROXMOXER)

`proxmoxer` es la librería más usada para Python. Abstrae la API a una sintaxis muy natural.

- **Repositorio de scripts**: Ejemplos avanzados de administración con proxmoxer

### AUTOMATIZACIÓN CON SHELL SCRIPTS

Bash es ideal para **scripts rápidos y ligeros** usando `curl` o `pvesh` directamente:

- **Repositorio de scripts**: Colección de scripts bash para Proxmox del profesor Manuel Domínguez

# Otros métodos de automatización

## TERRAFORM

Infraestructura como código (IaC) para definir máquinas virtuales y recursos de forma declarativa y reproducible.

Provider oficial: `bpg/proxmox`

## ANSIBLE

Orquestación y configuración idempotente mediante la colección `community.general.proxmox`.

Facilita tareas repetitivas y gestión de máquinas a escala.

# Aplicación práctica en vuestro centro

## CASOS DE USO PRINCIPALES

1. **Aprovisionamiento de alumnos**: Scripts con `pveum` + `pvesh` para crear usuarios, grupos, pools y ACLs a principio de curso desde un CSV.
2. **Automatización de plantillas**: Scripts que construyen plantillas de forma reproducible, garantizando que todas se preparan igual y pueden regenerarse ante actualizaciones.
3. **Limpieza periódica**: Detectar y eliminar VMs antiguas, snapshots olvidados, alumnos que ya no están matriculados.
4. **Monitorización de consumo**: Consultar la API periódicamente para avisar cuando un pool sobrepase ciertos umbrales de recursos.

04

# Ampliación y escalabilidad del sistema

Evolución de la infraestructura

# Ampliación y escalabilidad del sistema

## ALMACENAMIENTO REMOTO SAN/NAS

- Centralización del almacenamiento
- Mayor capacidad y flexibilidad
- Independencia del servidor físico

## CLÚSTER DE ALTA DISPONIBILIDAD

- Múltiples nodos Proxmox
- Migración en vivo de máquinas
- Redundancia y tolerancia a fallos
- Escalabilidad horizontal

# ¡Gracias!

📧 José Domingo Muñoz 🏫 IES Gonzalo Nazareno · Dos Hermanas 🌐 https://josedom24.github.io/curso_proxmox_2026