

# Scripts de GameCenter

---

Scripts útiles para gestionar el entorno de VMs.

## Scripts Principales

### 1. setup-ansible-env.sh ★ ESENCIAL

Configura el entorno de Ansible con todas las dependencias.

```
bash scripts/setup-ansible-env.sh
```

#### Qué hace:

- Crea entorno virtual Python
- Instala Ansible, pyvmomi, requests
- Instala colección community.vmware
- Configura ansible.cfg
- Verifica que todo funcione

#### Cuándo usarlo:

- Primera vez que configuras el proyecto
- Después de reinstalar el sistema
- Si hay errores de "ModuleNotFoundError: No module named 'pyVim'"

---

### 2. create-vm-interactive.sh

Script interactivo para crear VMs con roles.

```
bash scripts/create-vm-interactive.sh
```

#### Qué hace:

- Menú para seleccionar SO (Ubuntu/Windows)
- Seleccionar rol (admin/auditor/cliente)
- Muestra recursos asignados
- Crea la VM automáticamente

#### Ejemplo de uso:

```
Selecciona el Sistema Operativo:  
1) Ubuntu Desktop 24.04  
2) Windows 11
```

```
Opción [1-2]: 1

Nombre de la VM: ubuntu-cliente01

Selecciona el Rol:
1) Admin    - Acceso total
2) Auditor  - Solo lectura
3) Cliente  - Solo juegos
Opción [1-3]: 3

✓ VM creada: ubuntu-cliente01
```

---

### 3. list-vms.sh

Lista todas las VMs en vSphere y su estado.

```
bash scripts/list-vms.sh
```

#### Qué muestra:

- Nombre de la VM
- Estado (Encendida/Apagada)
- CPU y RAM asignadas
- Dirección IP (o "Sin Tools/IP")
- VMs en inventario Ansible

#### Ejemplo de salida:

NOMBRE	ESTADO	CPU	RAM (MB)	IP
ubuntu-server 2025:db8:10::2	Encendida	2	4096	
ubuntu-cliente01 2025:db8:10::12	Encendida	2	4096	
win11-admin	Apagada	2	4096	N/A

---

### 4. vm-manager.sh

Menú interactivo para gestionar VMs.

```
bash scripts/vm-manager.sh
```

#### Opciones:

1. Listar VMs
2. Encender VM
3. Apagar VM
4. Reiniciar VM
5. Ver estado de VM
6. Salir

**Cuándo usarlo:**

- Encender/apagar VMs sin entrar a vSphere
  - Ver estado rápido de una VM
  - Reiniciar VMs remotamente
- 

## 5. quick-deploy.sh

Despliegue rápido de todo el entorno.

```
bash scripts/quick-deploy.sh
```

**Qué hace:**

1. Crea servidor Ubuntu
2. Configura servicios (DNS, DHCP, Firewall)
3. Crea VMs cliente (Ubuntu/Windows)
4. Crea VM administrador
5. Muestra resumen

**Cuándo usarlo:**

- Despliegue inicial del proyecto
  - Recrear todo el entorno desde cero
- 



## Scripts de Pruebas

## 6. test-govc-connection.sh

Verifica conexión a vCenter con govc.

```
bash scripts/test-govc-connection.sh
```

**Qué verifica:**

- govc instalado
- Credenciales correctas
- Conexión a vCenter

- Acceso al datacenter
- 

## 7. test-ssh-ubpc.sh

Prueba conexión SSH al servidor Ubuntu.

```
bash scripts/test-ssh-ubpc.sh
```

### Qué verifica:

- Servidor accesible por SSH
  - Credenciales correctas
  - Servicios funcionando
- 

## 8. test-network-connectivity.sh

Prueba conectividad de red entre VMs.

```
bash scripts/test-network-connectivity.sh
```

### Qué verifica:

- Ping entre VMs
  - DNS funcionando
  - DHCP asignando IPs
  - Gateway accesible
- 

## Requisitos

Para gestión de VMs (govc):

```
# Instalar govc en WSL
curl -L
https://github.com/vmware/govmomi/releases/latest/download/govc_Linux_x86_64.tar.g
z | tar -xz
sudo mv govc /usr/local/bin/
```

Para Ansible:

```
# Ejecutar setup primero
bash scripts/setup-ansible-env.sh
```

```
# Luego activar entorno  
source ~/.ansible-venv/bin/activate
```

Para jq (parsing JSON):

```
sudo apt install jq -y
```

---

## Flujo de Trabajo Recomendado

Primera vez:

```
# 1. Configurar entorno  
bash scripts/setup-ansible-env.sh  
source ~/.ansible-venv/bin/activate  
  
# 2. Probar conexión  
bash scripts/test-govc-connection.sh  
  
# 3. Desplegar todo  
bash scripts/quick-deploy.sh
```

Uso diario:

```
# Activar entorno  
source ~/.ansible-venv/bin/activate  
  
# Ver estado de VMs  
bash scripts/list-vm.sh  
  
# Crear VM individual  
bash scripts/create-vm-interactive.sh  
  
# Gestionar VMs  
bash scripts/vm-manager.sh
```

---

## Configuración

Los scripts leen credenciales de `group_vars/all.vault.yml`:

```
vault_vcenter_hostname: "168.121.48.254"  
vault_vcenter_port: "10111"
```

```
vault_vcenter_username: "root"  
vault_vcenter_password: "qwe123$"
```

---

## Troubleshooting

Error: govc no encontrado

```
which govc  
# Si no existe, instalar:  
curl -L  
https://github.com/vmware/govmomi/releases/latest/download/govc_Linux_x86_64.tar.g  
z | tar -xz  
sudo mv govc /usr/local/bin/
```

Error: ModuleNotFoundError: No module named 'pyVim'

```
bash scripts/setup-ansible-env.sh  
source ~/.ansible-venv/bin/activate
```

Error: No se puede conectar a vCenter

```
# Verificar credenciales  
cat group_vars/all.vault.yml | grep vcenter  
  
# Probar conexión  
bash scripts/test-govc-connection.sh
```

Scripts no ejecutables (Linux/Mac)

```
chmod +x scripts/*.sh
```

---

## Resumen de Scripts

Script	Para qué sirve	Cuándo usarlo
<b>setup-ansible-env.sh</b>	Configurar entorno	Primera vez / Errores de Python
<b>create-vm-interactive.sh</b>	Crear VMs con menú	Crear VMs individuales
<b>list-vms.sh</b>	Ver estado de VMs	Ver qué VMs hay y su estado

Script	Para qué sirve	Cuándo usarlo
<b>vm-manager.sh</b>	Encender/apagar VMs	Gestión rápida de VMs
<b>quick-deploy.sh</b>	Desplegar todo	Crear entorno completo
<b>test-govc-connection.sh</b>	Probar vCenter	Verificar conexión
<b>test-ssh-ubpc.sh</b>	Probar SSH	Verificar servidor
<b>test-network-connectivity.sh</b>	Probar red	Verificar conectividad

## Seguridad

- Las contraseñas están en `all.vault.yml`
- Encriptar con: `ansible-vault encrypt group_vars/all.vault.yml`
- Los scripts NO muestran contraseñas en pantalla