

## FACULTAD DE INGENIERÍA MOCHIS.



**Carrera:** Ingeniería en Software

**Materia:** Administración de Sistemas

**Práctica:** No. 1 - Configuración de Entorno de Virtualización

**Docente:** Dr. Herman Geovany Ayala

**Autor:** María Leticia Muñoz Carlon

**Grupo:** 3-01

**Fecha:** 3 de febrero de 2026

**Repositorio:** [https://github.com/marialequa05-hue/Tarea1\\_Entorno-de-Virtualizacion](https://github.com/marialequa05-hue/Tarea1_Entorno-de-Virtualizacion)

## Contenido

1.	Historial de Cambios: .....	2
2.	Introducción y Arquitectura.....	2
	Objetivo .....	2
	Diagrama de Topología .....	2
3.	Guía de Uso de los Scripts (Manual de Usuario) .....	3
	A) Script para Linux (tarea1_diagnostico.sh).....	3
	B) Script para Windows (tarea1_diagnostico.ps1).....	3
4.	Bitácora de Desarrollo y Configuración .....	4
	Tabla de Direcccionamiento IP Propuesta .....	4
	Explicación del Script (Lógica): .....	4
5.	Pruebas de Funcionamiento (Validación) .....	5
	Protocolo de Pruebas.....	5
	Prueba 01: Prueba de Conectividad Linux Server -> Windows Server .....	6
	Prueba 02: Prueba de Conectividad Linux Server-> Cliente Windows.....	6
	Prueba 03: Prueba de Conectividad Linux Server -> Internet (Google) .....	6
	Prueba 04: Prueba de Conectividad Windows Server -> Linux Server .....	7
	Prueba 05: Prueba de Conectividad Windows Server -> Windows Cliente .....	7
	Prueba 06: Prueba de conectividad Windows Server -> Internet (Google).....	8
	Prueba 07: Conexión de cliente a Linux Server .....	8
	Prueba 08: Conexión de cliente a Windows Server .....	8
	Prueba 09: Conectividad de cliente a Internet (Google).....	9
	Prueba 10: Ejecución de script de Linux.....	9
	Prueba 11: Ejecución de script de Windows.....	10
6.	Optimización de Recursos y Hardware Virtual.....	12
	Sistemas "Headless" (Sin Interfaz Gráfica):.....	12
7.	Conclusiones y Referencias .....	13
	Lecciones Aprendidas.....	13
	Bibliografía y Recursos .....	13

## 1. Historial de Cambios:

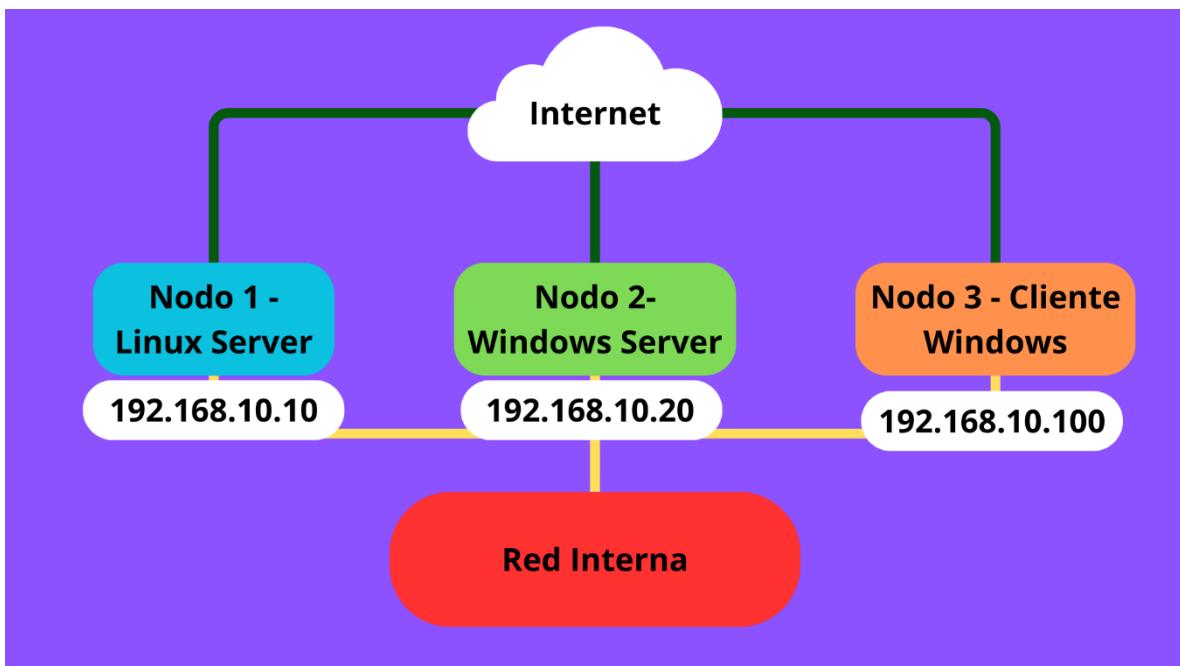
Versión	Fecha	Autor	Descripción del Cambio / Hito
v1.0	03/02/2026	Maria Leticia	Creación de VMs e instalación de SO (AlmaLinux y WinServer).
v1.1	03/02/2026	Maria Leticia	Configuración de Red Interna (192.168.10.x) y Hostnames.
v1.2	03/02/2026	Maria Leticia	Creación de scripts de diagnóstico (.sh y .ps1).
v1.3	03/02/2026	Maria Leticia	Carga final al repositorio GitHub y documentación.

## 2. Introducción y Arquitectura

### Objetivo

Implementar un laboratorio virtualizado compuesto por tres nodos (Linux Server, Windows Server Core y Cliente) para simular un entorno de red empresarial aislado. El objetivo principal es garantizar la conectividad interna mediante direccionamiento IP estático, asegurar la salida a internet mediante NAT y preparar los sistemas para futuras automatizaciones mediante scripts de diagnóstico.

### Diagrama de Topología



### 3. Guía de Uso de los Scripts (Manual de Usuario)

Esta sección detalla cómo ejecutar las herramientas de diagnóstico creadas para validar el estado de los servidores.

#### A) Script para Linux (tarea1\_diagnostico.sh)

##### Requisitos Previos

- Tener permisos de ejecución sobre el archivo (chmod +x).
- Estar logueado en la terminal de AlmaLinux (Nodo 1).

##### Instrucciones de Ejecución

`./tarea1_diagnostico.sh`

##### Flujo de Interacción

El script no requiere intervención del usuario. Al ejecutarse, limpiará la pantalla y mostrará automáticamente: Fecha, Hostname, IPs activas y Espacio en disco.

#### B) Script para Windows (tarea1\_diagnostico.ps1)

##### Requisitos Previos:

- PowerShell 5.1 o superior.
- Permisos de ejecución de scripts habilitados (si aplica).

##### Instrucciones de Ejecución

`.\tarea1_diagnostico.ps1`

##### Flujo de Interacción

El sistema mostrará una tabla con colores indicando el estado del servidor. Los datos de almacenamiento se convierten automáticamente a GB para facilitar la lectura.

## 4. Bitácora de Desarrollo y Configuración

### Tabla de Direccionamiento IP Propuesta

Esta tabla cubre el requisito del documento PDF. He definido una red clase C estándar (192.168.10.x) para la red interna.

Nodo	Rol / Hostname	Sistema Operativo	Adaptador 1 (NAT)	Adaptador 2 (Red Interna: red_sistemas)	Máscara de Subred
Nodo 1	nodo1-SrvLinux	Alma Linux 10.1	DHCP (Automática)	192.168.10.10	255.255.255.0 (/24)
Nodo 2	Nodo2SrvWin	Windows Server 2022	DHCP (Automática)	192.168.10.20	255.255.255.0 (/24)
Nodo 3	Cliente-01	Windows 10/11 o Linux	DHCP (Automática)	192.168.10.100	255.255.255.0 (/24)

### Explicación del Script (Lógica):

#### 1. Obtención de IP:

- En Linux, se utilizó el comando ip a filtrando con grep para excluir la interfaz local (127.0.0.1) y awk para limpiar la salida y mostrar solo la dirección numérica.
- En Windows, se usó el cmdlet Get-NetIPAddress filtrando por familia IPv4 y excluyendo la interfaz Loopback para obtener la IP real de la red interna.

#### 2. Cálculo de Espacio en Disco:

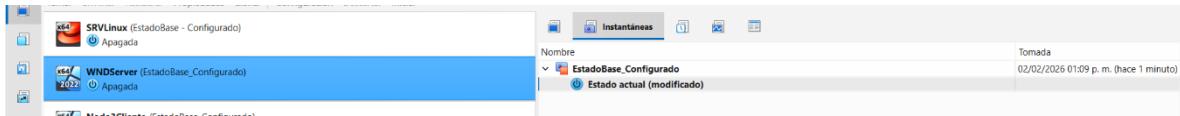
- En **Windows**, el comando Get-PSDrive devuelve el valor en Bytes. Se implementó una expresión matemática Expression={"{0:N2}" -f (\$\_.Used/1GB)} dentro del Select-Object para convertir esos bytes a Gigabytes con solo dos decimales, haciendo el reporte legible para humanos.

### Evidencias de Configuración:

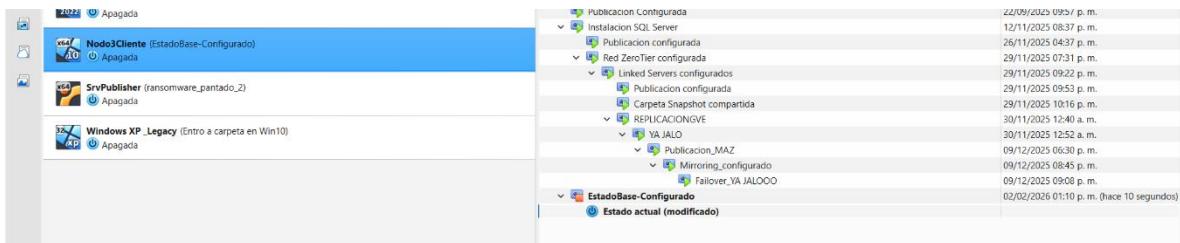
#### Nodo 1 – Alma Linux Server



#### Nodo 2 – Windows Server 2022



### Nodo 3 - Cliente



## 5. Pruebas de Funcionamiento (Validación)

### Protocolo de Pruebas

ID	Prueba	Entrada (Acción)	Salida Esperada	Resultado
01	Conectividad Linux Server -> Windows Server	ping 192.168.10.20	Respuesta (TTL=128 aprox)	Éxito
02	Conectividad Linux Server-> Cliente Windows	ping 192.168.10.100	Respuesta (TTL=64 aprox)	Éxito
03	Conectividad Linux Server -> Internet (Google)	ping -c 4 google.com	Respuesta (TTL=64 aprox)	Éxito
04	Conectividad Windows Server -> Linux Server	ping 192.168.10.10	Respuesta (TTL=128 aprox)	Éxito
05	Conectividad Windows Server -> Windows Cliente	ping 192.168.10.100	Respuesta (TTL=64 aprox)	Éxito
06	Conectividad Windows Server -> Internet (Google)	ping google.com	Respuesta (TTL=64 aprox)	Éxito
07	Conectividad Windows Cliente -> Linux Server	ping 192.168.10.10	Respuesta (TTL=128 aprox)	Éxito
08	Conectividad Windows Cliente -> Windows Server	ping 192.168.10.20	Respuesta (TTL=64 aprox)	Éxito
09	Conectividad Windows Cliente -> Internet (Google)	ping google.com	Respuesta (TTL=64 aprox)	Éxito
10	Ejecución Script Linux	./tarea1_diagnostico.sh	Reporte con Hostname: Srv-Linux	Éxito
11	Ejecución Script Windows	./tarea1_diagnostico.ps1	Reporte con Hostname: Srv-Win	Éxito

## Prueba 01: Prueba de Conectividad Linux Server -> Windows Server

```
[root@nodo1-SrvLinux ~]# ping 192.168.10.100
PING 192.168.10.100 (192.168.10.100) 56(84) bytes of data.
64 bytes from 192.168.10.100: icmp_seq=1 ttl=128 time=2.27 ms
64 bytes from 192.168.10.100: icmp_seq=2 ttl=128 time=0.746 ms
64 bytes from 192.168.10.100: icmp_seq=3 ttl=128 time=0.942 ms
64 bytes from 192.168.10.100: icmp_seq=4 ttl=128 time=0.905 ms
64 bytes from 192.168.10.100: icmp_seq=5 ttl=128 time=1.02 ms
64 bytes from 192.168.10.100: icmp_seq=6 ttl=128 time=0.904 ms
64 bytes from 192.168.10.100: icmp_seq=7 ttl=128 time=0.952 ms
64 bytes from 192.168.10.100: icmp_seq=8 ttl=128 time=0.886 ms
64 bytes from 192.168.10.100: icmp_seq=9 ttl=128 time=0.918 ms

64 bytes from 192.168.10.100: icmp_seq=10 ttl=128 time=0.835 ms
64 bytes from 192.168.10.100: icmp_seq=11 ttl=128 time=1.01 ms

64 bytes from 192.168.10.100: icmp_seq=12 ttl=128 time=0.874 ms
64 bytes from 192.168.10.100: icmp_seq=13 ttl=128 time=0.862 ms
C
--- 192.168.10.100 ping statistics ---
3 packets transmitted, 13 received, 0% packet loss, time 12029ms
rtt min/avg/max/mdev = 0.746/1.009/2.268/0.369 ms
[root@nodo1-SrvLinux ~]# ping -c 4 192.168.10.100
PING 192.168.10.100 (192.168.10.100) 56(84) bytes of data.
64 bytes from 192.168.10.100: icmp_seq=1 ttl=128 time=1.98 ms
64 bytes from 192.168.10.100: icmp_seq=2 ttl=128 time=1.21 ms
64 bytes from 192.168.10.100: icmp_seq=3 ttl=128 time=1.31 ms
64 bytes from 192.168.10.100: icmp_seq=4 ttl=128 time=1.43 ms

--- 192.168.10.100 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3009ms
rtt min/avg/max/mdev = 1.208/1.483/1.980/0.297 ms
[root@nodo1-SrvLinux ~]#
```

## Prueba 02: Prueba de Conectividad Linux Server-> Cliente Windows

```
[root@nodo1-SrvLinux ~]# ping -c 4 192.168.10.20
PING 192.168.10.20 (192.168.10.20) 56(84) bytes of data.
64 bytes from 192.168.10.20: icmp_seq=1 ttl=128 time=4.38 ms
64 bytes from 192.168.10.20: icmp_seq=2 ttl=128 time=1.19 ms
64 bytes from 192.168.10.20: icmp_seq=3 ttl=128 time=1.02 ms
64 bytes from 192.168.10.20: icmp_seq=4 ttl=128 time=1.54 ms

--- 192.168.10.20 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3018ms
rtt min/avg/max/mdev = 1.019/2.029/4.375/1.366 ms
[root@nodo1-SrvLinux ~]#
```

## Prueba 03: Prueba de Conectividad Linux Server -> Internet (Google)

```
[root@nodo1-SrvLinux ~]# ping -c 4 google.com
PING google.com (192.178.52.174) 56(84) bytes of data.
64 bytes from tzqroa-ab-in-f14.1e100.net (192.178.52.174): icmp_seq=1 ttl=255 time=31.4 ms
64 bytes from tzqroa-ab-in-f14.1e100.net (192.178.52.174): icmp_seq=2 ttl=255 time=31.0 ms
64 bytes from tzqroa-ab-in-f14.1e100.net (192.178.52.174): icmp_seq=3 ttl=255 time=30.6 ms
64 bytes from tzqroa-ab-in-f14.1e100.net (192.178.52.174): icmp_seq=4 ttl=255 time=30.7 ms

--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3008ms
rtt min/avg/max/mdev = 30.589/30.909/31.362/0.306 ms
[root@nodo1-SrvLinux ~]#
```

## Prueba 04: Prueba de Conectividad Windows Server -> Linux Server

```
PS C:\Users\Administrador> ping 192.168.10.10

Haciendo ping a 192.168.10.10 con 32 bytes de datos:
Respuesta desde 192.168.10.10: bytes=32 tiempo=1ms TTL=64
Respuesta desde 192.168.10.10: bytes=32 tiempo=1ms TTL=64
Respuesta desde 192.168.10.10: bytes=32 tiempo<1m TTL=64
Respuesta desde 192.168.10.10: bytes=32 tiempo=1ms TTL=64

Estadísticas de ping para 192.168.10.10:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
                (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 0ms, Máximo = 1ms, Media = 0ms
PS C:\Users\Administrador>
```

## Prueba 05: Prueba de Conectividad Windows Server -> Windows Cliente

```
PS C:\Users\Administrador> ping 192.168.10.100

Haciendo ping a 192.168.10.100 con 32 bytes de datos:
Respuesta desde 192.168.10.100: bytes=32 tiempo=1ms TTL=128
Respuesta desde 192.168.10.100: bytes=32 tiempo=1ms TTL=128
Respuesta desde 192.168.10.100: bytes=32 tiempo=3ms TTL=128
Respuesta desde 192.168.10.100: bytes=32 tiempo=1ms TTL=128

Estadísticas de ping para 192.168.10.100:
    Paquetes: enviados = 4, recibidos = 4, perdidos = 0
                (0% perdidos),
    Tiempos aproximados de ida y vuelta en milisegundos:
        Mínimo = 1ms, Máximo = 3ms, Media = 1ms
```

## Prueba 06: Prueba de conectividad Windows Server -> Internet (Google)

```
PS C:\Users\Administrador> ping google.com

Haciendo ping a google.com [192.178.52.174] con 32 bytes de datos:
Respuesta desde 192.178.52.174: bytes=32 tiempo=37ms TTL=255
Respuesta desde 192.178.52.174: bytes=32 tiempo=31ms TTL=255
Respuesta desde 192.178.52.174: bytes=32 tiempo=32ms TTL=255
Respuesta desde 192.178.52.174: bytes=32 tiempo=31ms TTL=255

Estadísticas de ping para 192.178.52.174:
  Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
  Mínimo = 31ms, Máximo = 37ms, Media = 32ms
PS C:\Users\Administrador>
```

## Prueba 07: Conexión de cliente a Linux Server

```
PS C:\Users\ServidorMaz> ping 192.168.10.10

Haciendo ping a 192.168.10.10 con 32 bytes de datos:
Respuesta desde 192.168.10.10: bytes=32 tiempo=3ms TTL=64
Respuesta desde 192.168.10.10: bytes=32 tiempo<1ms TTL=64
Respuesta desde 192.168.10.10: bytes=32 tiempo=2ms TTL=64
Respuesta desde 192.168.10.10: bytes=32 tiempo=1ms TTL=64

Estadísticas de ping para 192.168.10.10:
  Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
  Mínimo = 0ms, Máximo = 3ms, Media = 1ms
```

## Prueba 08: Conexión de cliente a Windows Server

```
PS C:\Users\ServidorMaz> ping 192.168.10.20

Haciendo ping a 192.168.10.20 con 32 bytes de datos:
Respuesta desde 192.168.10.20: bytes=32 tiempo=3ms TTL=128
Respuesta desde 192.168.10.20: bytes=32 tiempo=1ms TTL=128
Respuesta desde 192.168.10.20: bytes=32 tiempo<1ms TTL=128
Respuesta desde 192.168.10.20: bytes=32 tiempo=1ms TTL=128

Estadísticas de ping para 192.168.10.20:
  Paquetes: enviados = 4, recibidos = 4, perdidos = 0
    (0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
  Mínimo = 0ms, Máximo = 3ms, Media = 1ms
PS C:\Users\ServidorMaz>
```

## Prueba 09: Conectividad de cliente a Internet (Google)

```
PS C:\Users\ServidorMaz> ping google.com

Haciendo ping a google.com [192.178.52.174] con 32 bytes de datos:
Respuesta desde 192.178.52.174: bytes=32 tiempo=31ms TTL=255

Estadísticas de ping para 192.178.52.174:
  Paquetes: enviados = 4, recibidos = 4, perdidos = 0
              (0% perdidos),
  Tiempos aproximados de ida y vuelta en milisegundos:
    Mínimo = 31ms, Máximo = 31ms, Media = 31ms
```

## Prueba 10: Ejecución de script de Linux

### Instrucciones de Ejecución

./tarea1\_diagnostico.sh

### Script

```
#!/bin/bash

echo "-----"
echo "  Reporte de estado del sistema   "
echo "-----"
echo ""
echo "Fecha: $(date)"
echo ""
echo "1. NOMBRE DEL EQUIPO: "
hostname
echo ""
echo "2. DIRECCIONES IP: "
ip a | grep inet | grep -v 127.0.0.1 | awk '{print $2}'
echo ""
echo "3. ESPACIO EN DISCO: "
df -h | grep -E '^/dev/'
echo ""
echo "-----"
```

## Ejecución

```
"tarea_diagnostico1.sh" 18L, 431B written
[root@nodo1-SrvLinux ~]# ./tarea_diagnostico1.sh
-----
Reporte de estado del sistema
-----

Fecha: Mon Feb  2 19:28:12 MST 2026

1. NOMBRE DEL EQUIPO:
nodo1-SrvLinux

2. DIRECCIONES IP:
::1/128
10.0.2.15/24
fd17:625c:f037:2:a00:27ff:fed:edaf/64
fe80::a00:27ff:fed:edaf/64
192.168.10.10/24
fe80::a00:27ff:fedb:a31f/64

3. ESPACIO EN DISCO:
/dev/mapper/almalinux-root   17G  1.4G   16G   8% /
/dev/sda2                      960M 331M  630M  35% /boot
-----
[root@nodo1-SrvLinux ~]#
```

## Prueba 11: Ejecución de script de Windows

### Instrucciones de Ejecución

.\tarea1\_diagnostico.ps1

### Script

```
Clear-Host

Write-Host "-----" -ForegroundColor Magenta
Write-Host "    Reporte del estado del sistema " -ForegroundColor Cyan
Write-Host "-----" -ForegroundColor Magenta

Write-Host ""

Write-Host "Fecha: $(Get-Date)"

Write-Host ""
```

```
Write-Host "A) Nombre del equipo: " -ForegroundColor DarkCyan
Hostname
Write-Host ""

Write-Host "B) Direcciones IP (Red Interna):" -ForegroundColor DarkCyan
Get-NetIPAddress | Where-Object {$_.AddressFamily -eq "IPv4" -and
$_.InterfaceAlias -ne "Loopback Pseudo-Interface 1"} | Select-Object
IPAddress, InterfaceAlias
Write-Host ""

Write-Host "C) Espacio en Disco: " -ForegroundColor DarkCyan
Get-PSDrive C | Select-Object @{"Name="Usado
(GB)" ; Expression = "{$_.Used/1GB}"}, @ {"Name="Libre
(GB)" ; Expression = "{$_.Free/1GB}"}, @ {"Name="Total
(GB)" ; Expression = "{$_.Used + $_.Free}/1GB"} } | Format-
List
Write-Host ""
Write-Host "-----"
```

Ejecución

```
c:\ Administrador: C:\Windows\system32\cmd.exe
-----
Reporte del estado del sistema
-----
Fecha: 02/03/2026 06:54:33

A) Nombre del equipo:
nodo2SrvWin

B) Direcciones IP (Red Interna):

C) Espacio en Disco:
IPAddress      InterfaceAlias
-----
192.168.10.20  Ethernet 2
10.0.2.15      Ethernet

Usado (GB) : 7.52
Libre (GB) : 16.83
Total (GB) : 24.34
```

## 6. Optimización de Recursos y Hardware Virtual

Para garantizar la estabilidad del entorno virtualizado y evitar el desbordamiento de memoria en el equipo anfitrión (*Host*), se implementó una estrategia de asignación de recursos basada en el principio de "mínimo privilegio y mínimo consumo".

### Sistemas "Headless" (Sin Interfaz Gráfica):

- En el Nodo 1, se implementó AlmaLinux 9 Minimal. Al prescindir del entorno de escritorio (GNOME/KDE), se redujo el consumo base de RAM de aprox. 1.5 GB a menos de 400 MB en reposo, liberando recursos para servicios críticos.
- En el Nodo 2, se optó por Windows Server Core, administrado vía PowerShell. Esto elimina la sobrecarga de la "Desktop Experience", reduciendo la superficie de ataque y mejorando el rendimiento de I/O.

2. **Asignación de Memoria (RAM):** Se configuró un límite estricto 2048MB por nodo servidor, suficiente para las tareas de enrutamiento y servicios de red, sin comprometer la fluidez del sistema operativo anfitrión.
3. **Almacenamiento Dinámico:** Los discos virtuales (VDI) se configuraron como "reservados dinámicamente" (*Thin Provisioning*), ocupando en el disco físico solo el espacio realmente escrito por el sistema invitado, optimizando el almacenamiento local.

## 7. Conclusiones y Referencias

### Lecciones Aprendidas

Durante el desarrollo de la práctica, el principal desafío fue la configuración del Firewall en Windows Server Core. Aunque la red estaba bien configurada, los pings fallaban con "Tiempo de espera agotado". Se solucionó aplicando una regla de entrada ICMPv4 mediante PowerShell (`New-NetFirewallRule`). También se reforzó el uso de Git en línea de comandos, aprendiendo a configurar la identidad del usuario (`user.name` y `user.email`) para poder realizar los commits correctamente.

### Bibliografía y Recursos

1. AlmaLinux OS Foundation. (2024). *AlmaLinux 9 Documentation*. Recuperado de [wiki.almalinux.org](https://wiki.almalinux.org).
2. Microsoft. (2024). *New-NetFirewallRule (NetSecurity)*. Microsoft Learn.
3. Asistencia Técnica con IA:
  - IA Utilizada: Google Gemini.
  - Prompts Clave: "Cómo configurar IP estática en AlmaLinux CLI", "Solucionar error de git author identity unknown".