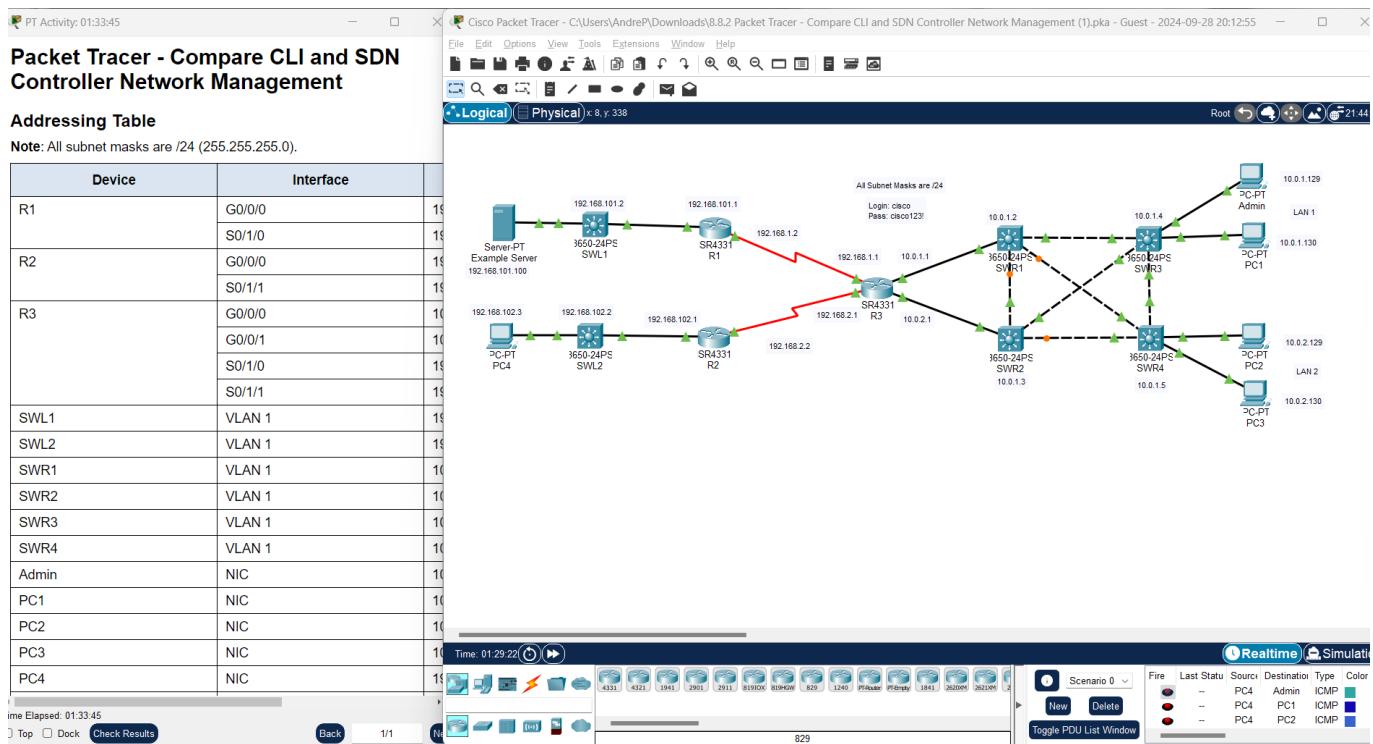
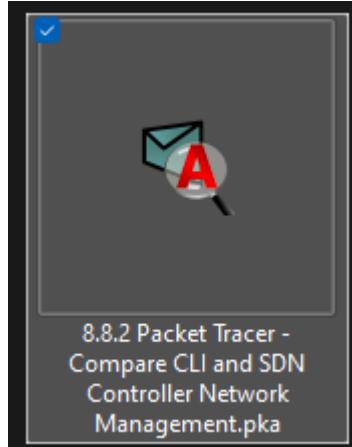


Practice 1 - Network Administration (CC 312)

Part 1: Packet Tracer - Compare CLI and SDN Controller Network Management

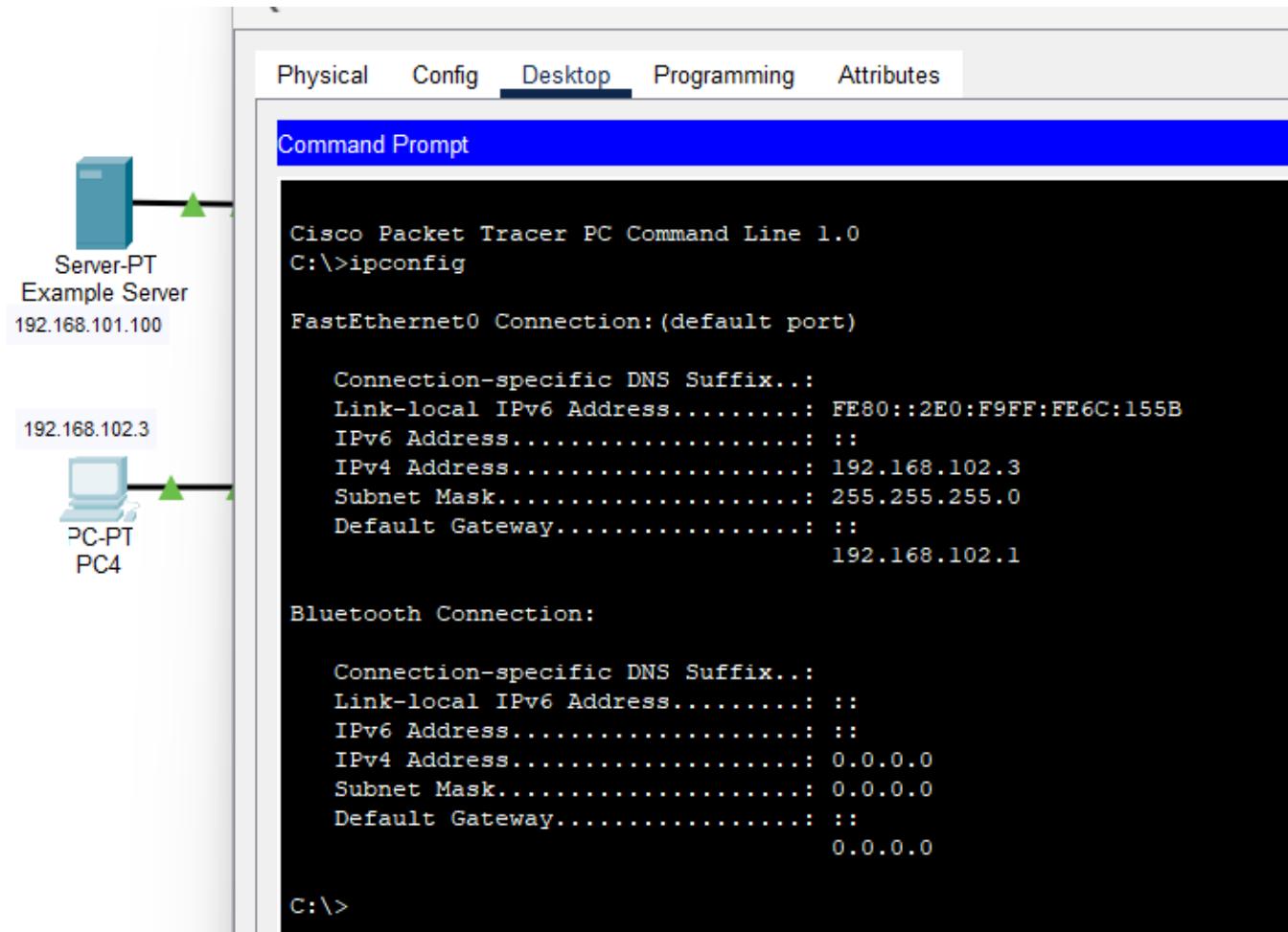
Ingresamos al Packet Tracer. Tuve que buscar el .pka en internet para cargarlo y desarrollar el laboratorio.



Usamos la CLI para recolección de datos de la red:

Nos aseguramos que los dispositivos pueden pingearse:

Ipv4 address de PC4 : 192.168.102.3



Ipv4 address de Example Server: 192.168.101.100

Haciendo el ping desde server a PC4:

The screenshot shows a Cisco Packet Tracer Command Line interface. The tabs at the top are Physical, Config, Services, Desktop (which is selected), Programming, and Attributes. Below the tabs is a blue header bar labeled "Command Prompt". The main area displays the following command-line session:

```
Cisco Packet Tracer SERVER Command Line 1.0
C:\>ipconfig

FastEthernet0 Connection: (default port)

Connection-specific DNS Suffix...:
Link-local IPv6 Address.....: FE80::20A:41FF:FE3D:D793
IPv6 Address.....: ::
IPv4 Address.....: 192.168.101.100
Subnet Mask.....: 255.255.255.0
Default Gateway.....: ::
                           192.168.101.1

C:\>ping 192.168.102.3

Pinging 192.168.102.3 with 32 bytes of data:

Request timed out.
Reply from 192.168.102.3: bytes=32 time=12ms TTL=125
Reply from 192.168.102.3: bytes=32 time=13ms TTL=125
Reply from 192.168.102.3: bytes=32 time=13ms TTL=125

Ping statistics for 192.168.102.3:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 12ms, Maximum = 13ms, Average = 12ms

C:\>
```

Ahora hagamos pings desde los otros dispositivos a PC4:

The screenshot shows a software interface for managing network configurations. The title bar says "PC4". Below it is a menu bar with tabs: Physical, Config, Desktop, Programming, and Attributes. The "Desktop" tab is currently selected. A blue header bar says "Command Prompt". The main area displays the output of several ping commands:

```
Ping statistics for 192.168.101.100:
  Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
  Approximate round trip times in milli-seconds:
    Minimum = 12ms, Maximum = 20ms, Average = 15ms

C:\>ping 10.0.1.130

Pinging 10.0.1.130 with 32 bytes of data:

Request timed out.
Reply from 10.0.1.130: bytes=32 time=13ms TTL=126
Reply from 10.0.1.130: bytes=32 time=4ms TTL=126
Reply from 10.0.1.130: bytes=32 time=11ms TTL=126

Ping statistics for 10.0.1.130:
  Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
  Approximate round trip times in milli-seconds:
    Minimum = 4ms, Maximum = 13ms, Average = 9ms

C:\>ping 10.0.1.129

Pinging 10.0.1.129 with 32 bytes of data:

Request timed out.
Reply from 10.0.1.129: bytes=32 time=9ms TTL=126
Reply from 10.0.1.129: bytes=32 time=10ms TTL=126
Reply from 10.0.1.129: bytes=32 time=9ms TTL=126

Ping statistics for 10.0.1.129:
  Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
  Approximate round trip times in milli-seconds:
    Minimum = 9ms, Maximum = 10ms, Average = 9ms

C:\>ping 10.0.2.129

Pinging 10.0.2.129 with 32 bytes of data:

Request timed out.
Reply from 10.0.2.129: bytes=32 time=3ms TTL=126
Reply from 10.0.2.129: bytes=32 time=8ms TTL=126
Reply from 10.0.2.129: bytes=32 time=6ms TTL=126

Ping statistics for 10.0.2.129:
  Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
  Approximate round trip times in milli-seconds:
```

Command Prompt

```
C:\>ping 10.0.1.129

Pinging 10.0.1.129 with 32 bytes of data:

Request timed out.
Reply from 10.0.1.129: bytes=32 time=9ms TTL=126
Reply from 10.0.1.129: bytes=32 time=10ms TTL=126
Reply from 10.0.1.129: bytes=32 time=9ms TTL=126

Ping statistics for 10.0.1.129:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 9ms, Maximum = 10ms, Average = 9ms

C:\>ping 10.0.2.129

Pinging 10.0.2.129 with 32 bytes of data:

Request timed out.
Reply from 10.0.2.129: bytes=32 time=3ms TTL=126
Reply from 10.0.2.129: bytes=32 time=8ms TTL=126
Reply from 10.0.2.129: bytes=32 time=6ms TTL=126

Ping statistics for 10.0.2.129:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 8ms, Average = 5ms

C:\>ping 10.0.2.13-
Ping request could not find host 10.0.2.13-. Please check the name and try again.

C:\>ping 10.0.2.130

Pinging 10.0.2.130 with 32 bytes of data:

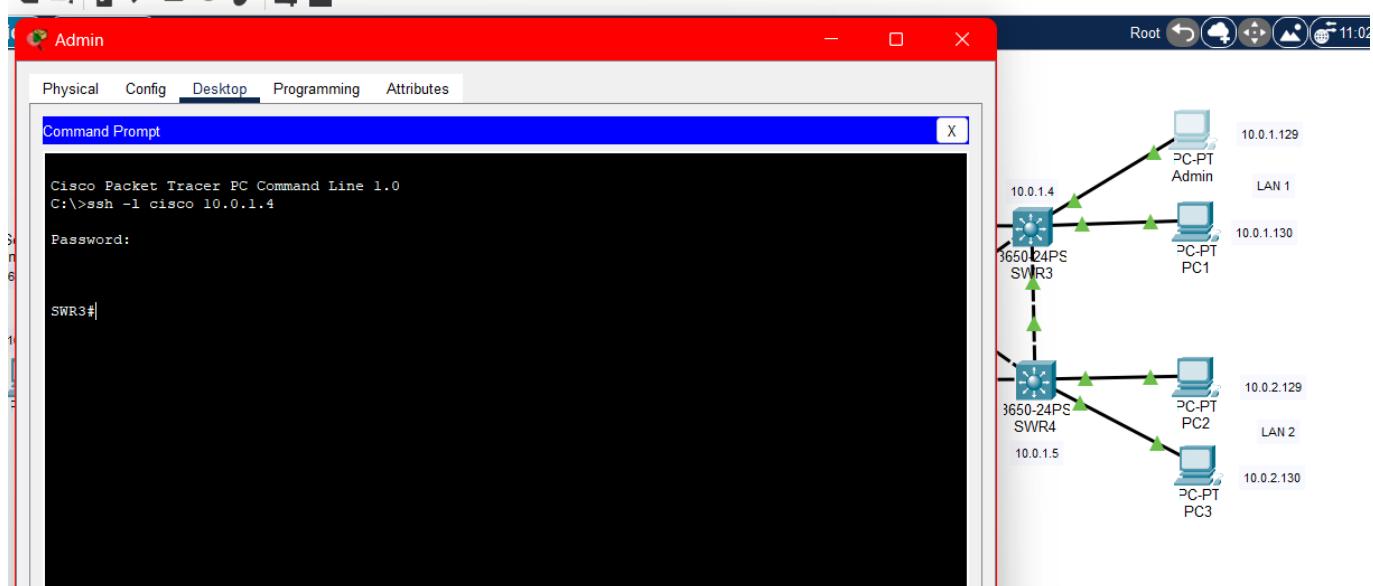
Request timed out.
Reply from 10.0.2.130: bytes=32 time=7ms TTL=126
Reply from 10.0.2.130: bytes=32 time=11ms TTL=126
Reply from 10.0.2.130: bytes=32 time=5ms TTL=126

Ping statistics for 10.0.2.130:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 5ms, Maximum = 11ms, Average = 7ms

C:\>
```

Observamos que PC4 pudo pingear a todos los dispositivos.

Iniciamos sesión en la PC Admin y accedemos al SWR3 Switch:



Lo que arrojó:

```
Cisco IOS Software [Denali], Catalyst L3 Switch Software (CAT3K_CAA-UNIVERSALK9-M), Version 16.3.2, RELEASE SOFTWARE (fc4)
BOOTLDR: CAT3K_CAA Boot Loader (CAT3K_CAA-HBOOT-M) Version 4.26, RELEASE SOFTWARE (P)
```

Hacemos lo mismo con los demás switches:

SWR3:

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ssh -l cisco 10.0.1.4
```

Password:

```
SWR3#show version | include RELEASE
Cisco IOS Software [Denali], Catalyst L3 Switch Software (CAT3K_CAA-UNIVERSALK9-M), Version 16.3.2, RELEASE SOFTWARE (fc4)
BOOTLDR: CAT3K_CAA Boot Loader (CAT3K_CAA-HBOOT-M) Version 4.26, RELEASE SOFTWARE (P)
SWR3#ssh -l cisco 10.0.1.5
```

Password:

```
SWR4#show version | include RELEASE
```

SWR1:

```
SWR1#show version | include RELEASE
Cisco IOS Software [Denali], Catalyst L3 Switch Software (CAT3K_CAA-
UNIVERSALK9-M), Version 16.3.2, RELEASE SOFTWARE (fc4)
BOOTLDR: CAT3K_CAA Boot Loader (CAT3K_CAA-HBOOT-M) Version 4.26,
RELEASE SOFTWARE (P)
SWR1#ssh -l cisco 10.0.1.3
```

Password:

SWR2:

```
SWR2#show version | include RELEASE
Cisco IOS Software [Denali], Catalyst L3 Switch Software (CAT3K_CAA-
UNIVERSALK9-M), Version 16.3.2, RELEASE SOFTWARE (fc4)
BOOTLDR: CAT3K_CAA Boot Loader (CAT3K_CAA-HBOOT-M) Version 4.26,
RELEASE SOFTWARE (P)
SWR2#ssh -l cisco 10.0.1.1
```

Password:

SWR4:

```
SWR4#show version | include RELEASE
Cisco IOS Software [Denali], Catalyst L3 Switch Software (CAT3K_CAA-
UNIVERSALK9-M), Version 16.3.2, RELEASE SOFTWARE (fc4)
BOOTLDR: CAT3K_CAA Boot Loader (CAT3K_CAA-HBOOT-M) Version 4.26,
RELEASE SOFTWARE (P)
SWR4#ssh -l cisco 10.0.1.2
```

I

Password:

R1:

```
R1#show version | include RELEASE
Cisco IOS Software, ISR Software (X86_64_LINUX_IOSD-UNIVERSALK9-M),
Version Version 15.5 (3)S5, RELEASE SOFTWARE (fc2)
```

R2:

```
R2#show version | include RELEASE
Cisco IOS Software, ISR Software (X86_64_LINUX_IOSD-UNIVERSALK9-M),
Version Version 15.5 (3)S5, RELEASE SOFTWARE (fc2)
```

R3:

```
R3#show version | include RELEASE
Cisco IOS Software, ISR Software (X86_64_LINUX_IOSD-UNIVERSALK9-M),
Version Version 15.5 (3)S5, RELEASE SOFTWARE (fc2)
```

SWL1:

```
SWL1#show version | include RELEASE
Cisco IOS Software [Denali], Catalyst L3 Switch Software (CAT3K_CAA-
UNIVERSALK9-M), Version 16.3.2, RELEASE SOFTWARE (fc4)
BOOTLDR: CAT3K_CAA Boot Loader (CAT3K_CAA-HBOOT-M) Version 4.26,
RELEASE SOFTWARE (P)
```

SWL2:

```
SWL2#show version | include RELEASE
Cisco IOS Software [Denali], Catalyst L3 Switch Software (CAT3K_CAA-
UNIVERSALK9-M), Version 16.3.2, RELEASE SOFTWARE (fc4)
BOOTLDR: CAT3K_CAA Boot Loader (CAT3K_CAA-HBOOT-M) Version 4.26,
RELEASE SOFTWARE (P)
SWL2#
```

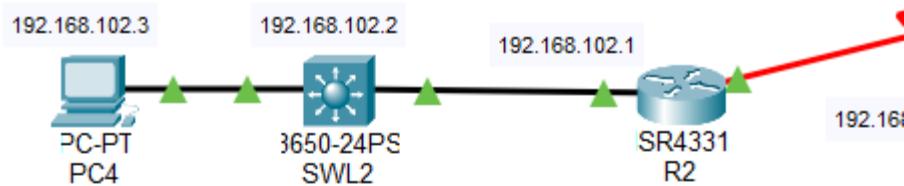
Luego salimos de las sesiones ssh con **exit** en bucle hasta llegar a la PC Admin.

Siguiendo la guía entré a la guía de configuración de controles de red

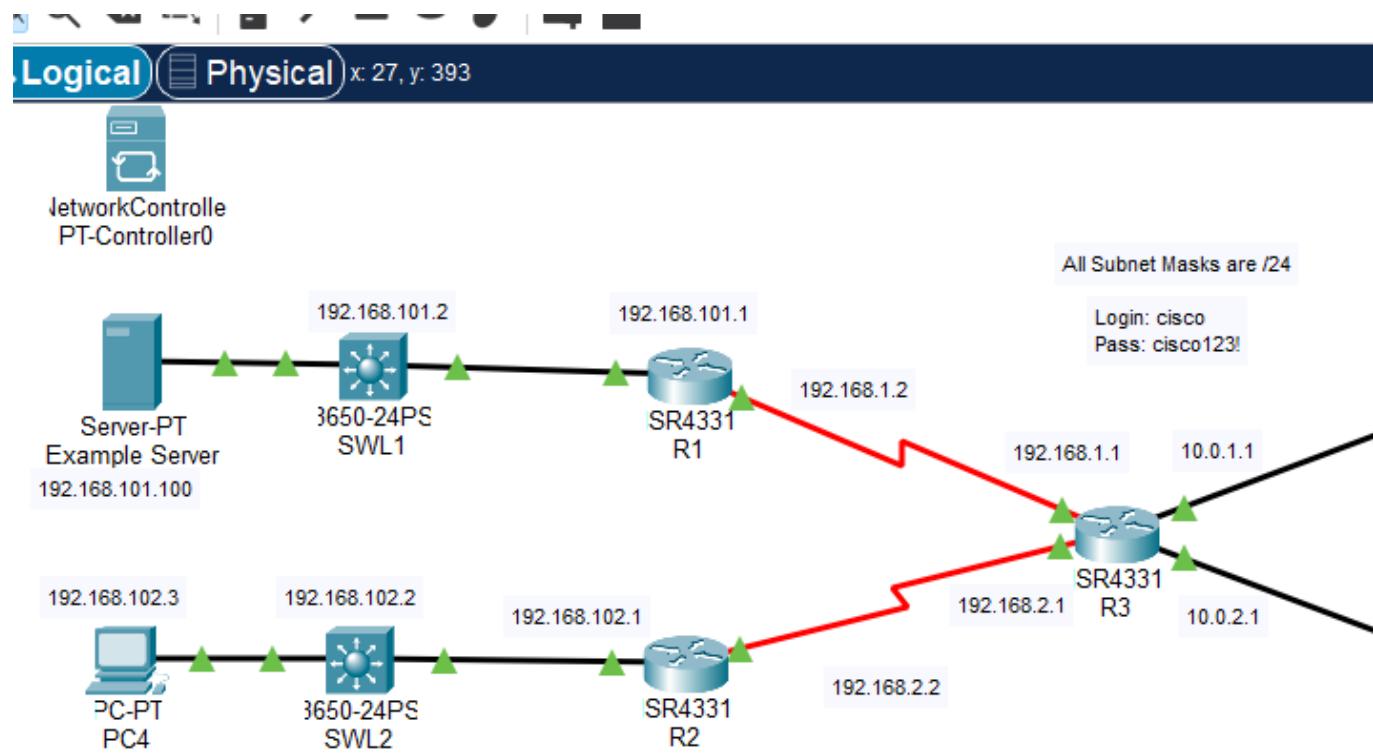
The screenshot shows the Cisco Packet Tracer interface. On the left, there's a sidebar with a tree view of network components. The main area is titled "Configuring Network Controllers". It contains several sections: "Overview", "Where Is It?", "Basic Configuration", and "Connecting It". Under "Where Is It?", it says "Network Controller can be found in the End Devices category on the device palette as shown below." Below this is a screenshot of the device palette, where the "Network Controller" icon is highlighted with a red box. The main window also shows a network diagram with two hosts connected via a logical interface.

Luego seguí con los pasos de la guía:

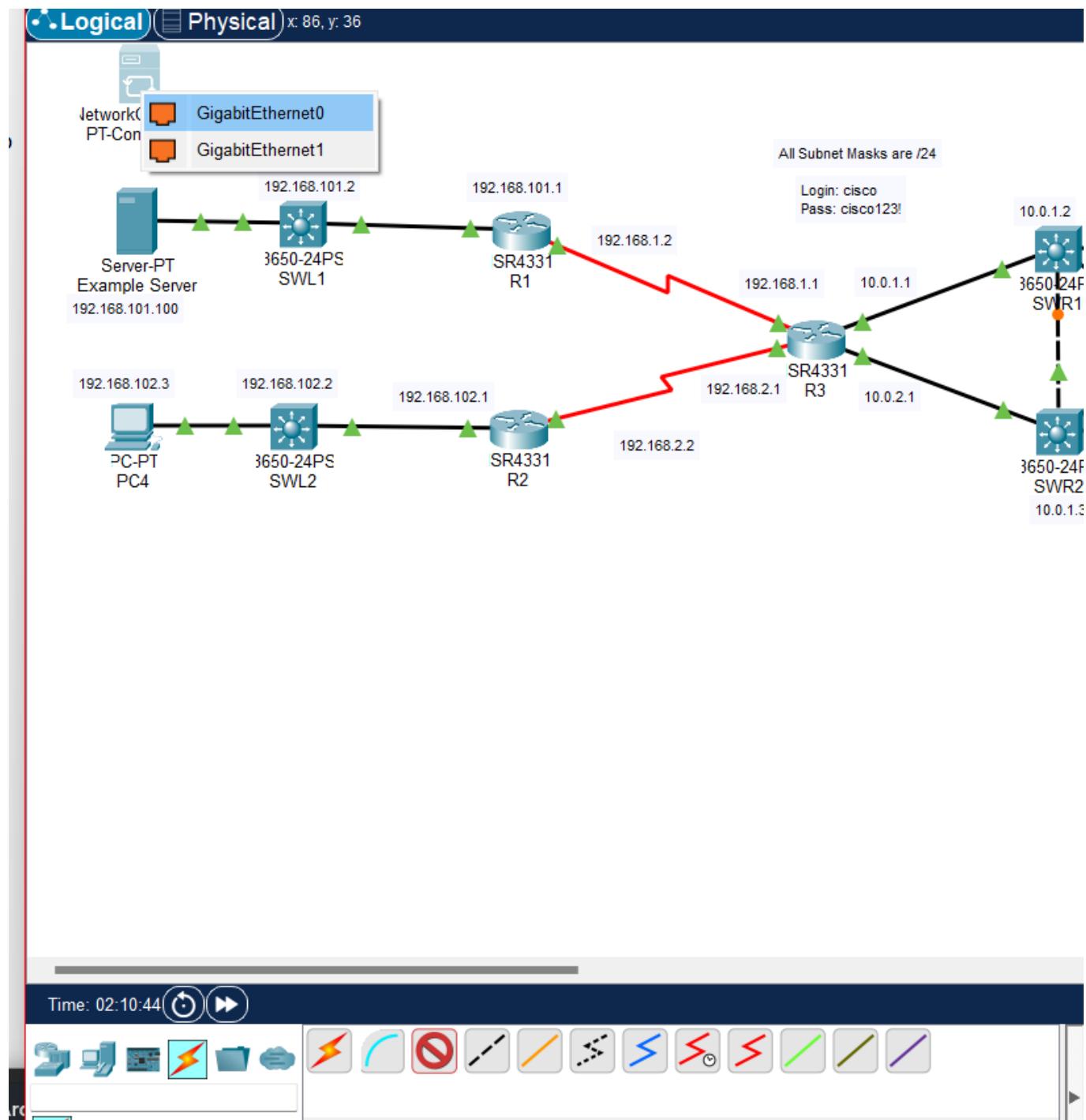
1. Agregar un nuevo control de red:



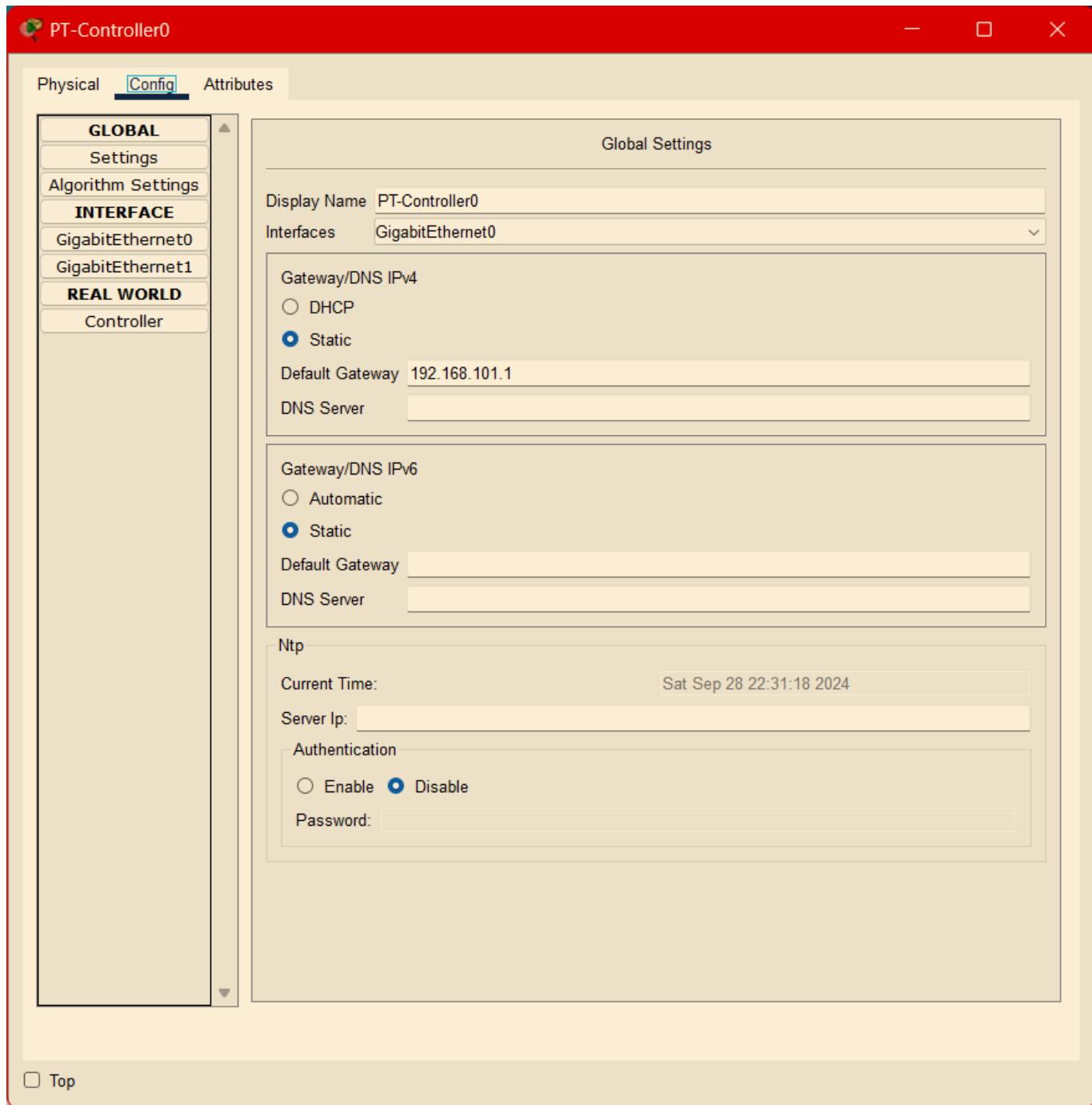
Lo ubiqué donde decía y vi que tiene el nombre que decía en la guía:



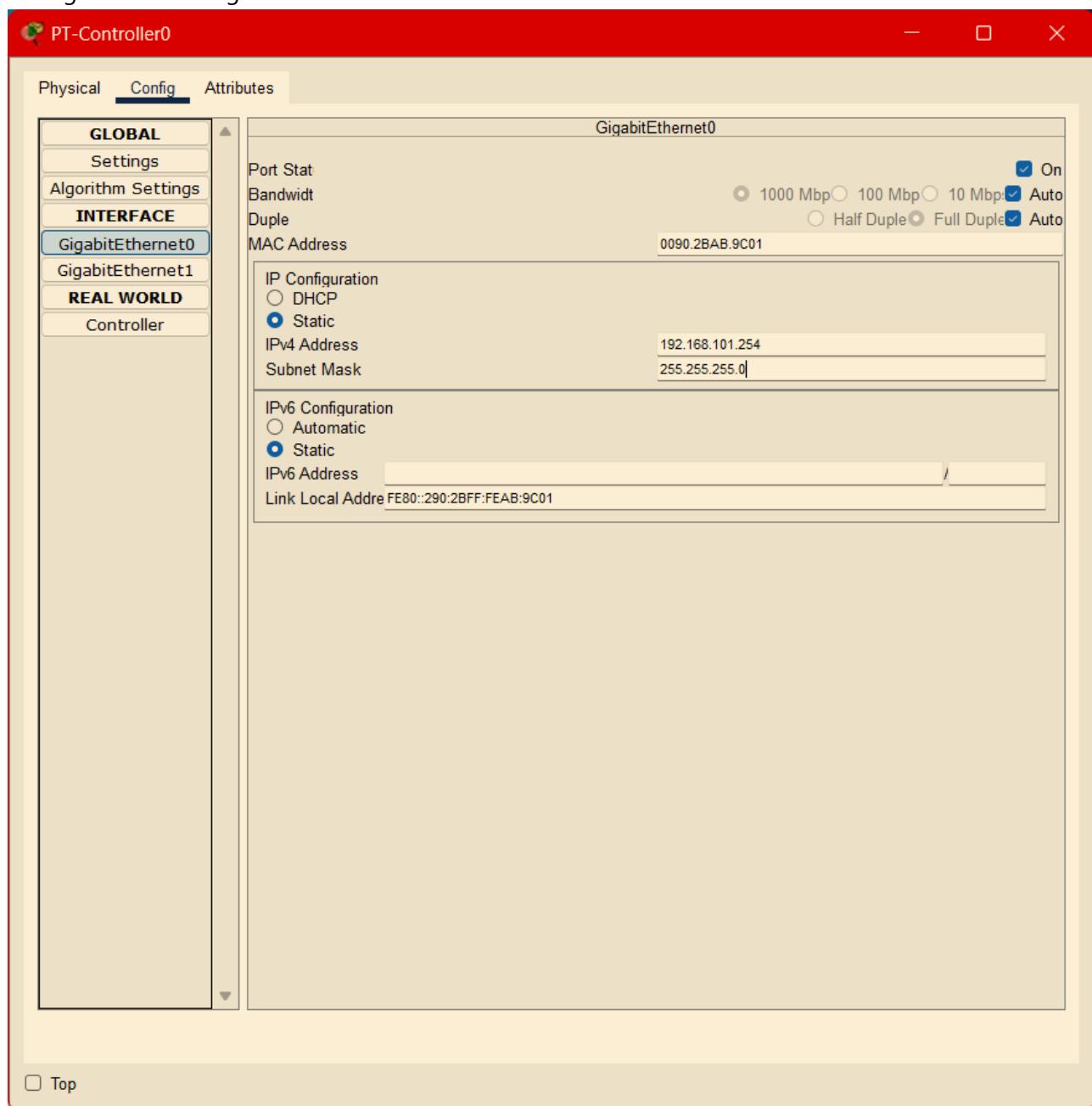
2. Conecto el control de red al switch SWL1



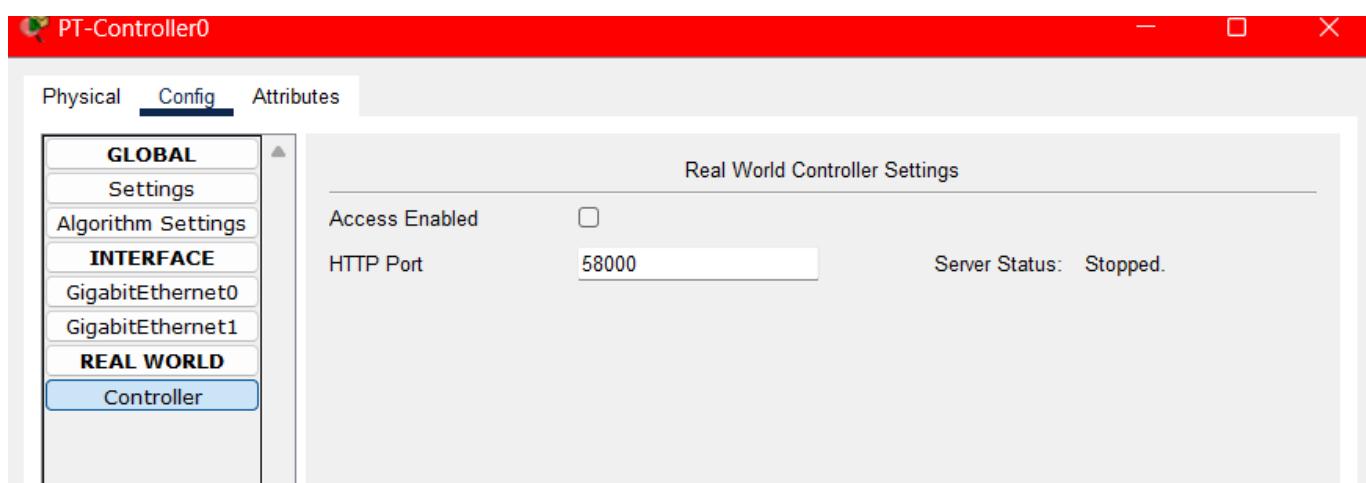
3. Pongo el default gateway en 192.168.101.1



4. Configuro interfaz GigabitEthernet0:



5. En REAL WORLD -> Controller; el server status estaba en Stopped así que proseguí como se ve en la guía:



Verificando conectividad con la PC Admin:

```
C:\>ping 192.168.101.254

Pinging 192.168.101.254 with 32 bytes of data:

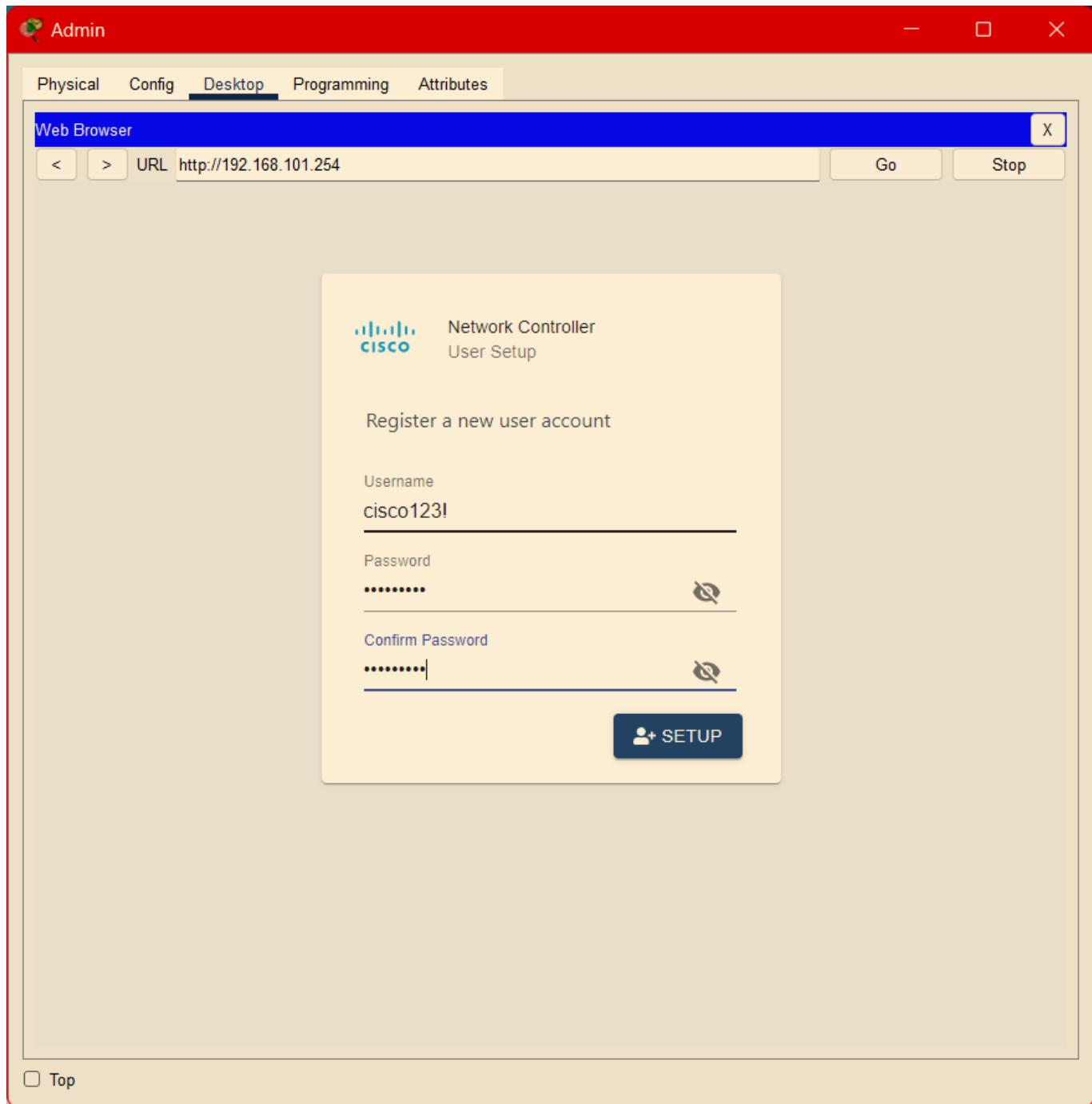
Request timed out.
Reply from 192.168.101.254: bytes=32 time=7ms TTL=126
Reply from 192.168.101.254: bytes=32 time=9ms TTL=126
Reply from 192.168.101.254: bytes=32 time=13ms TTL=126

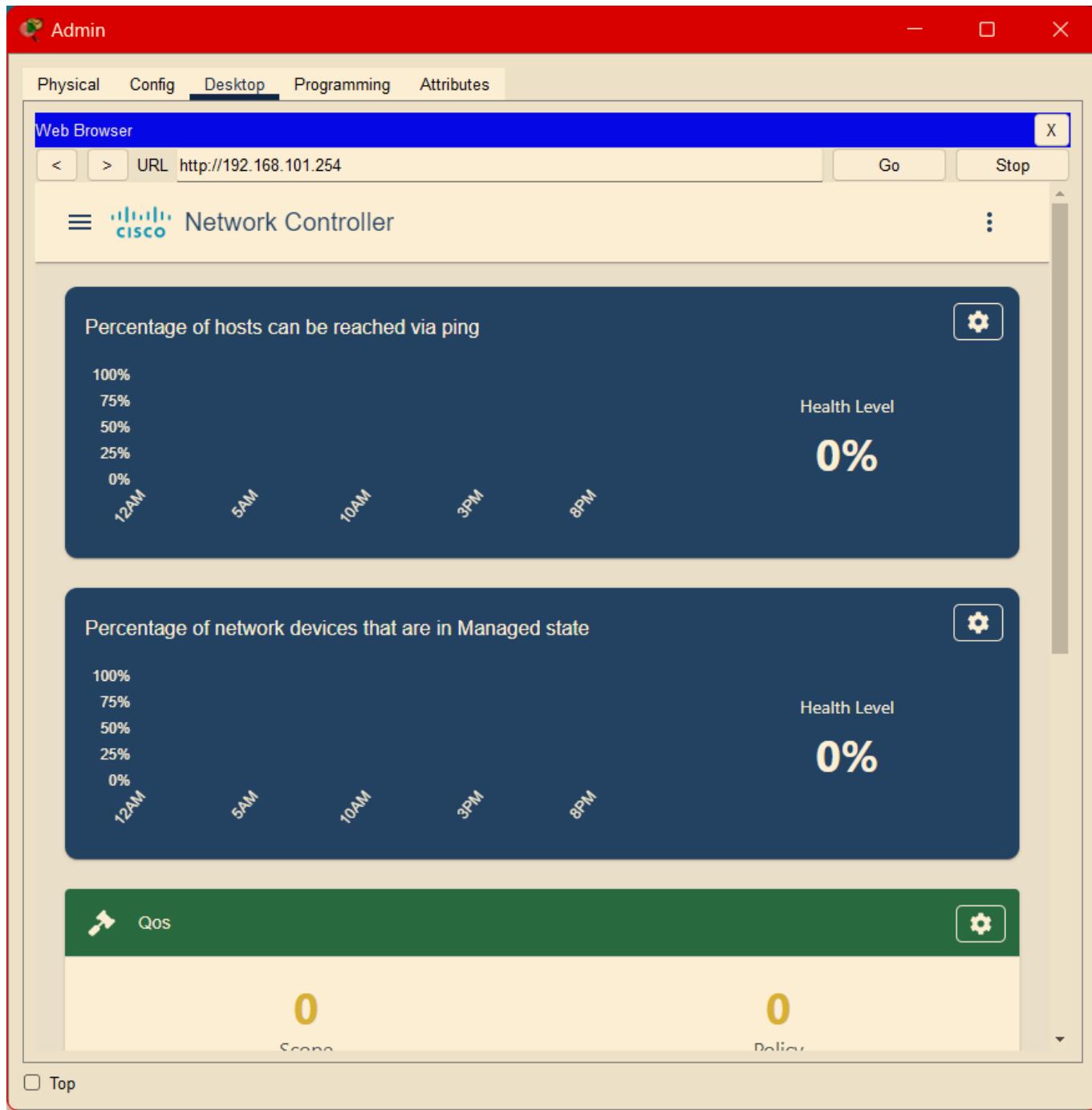
Ping statistics for 192.168.101.254:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 7ms, Maximum = 13ms, Average = 9ms
```

Funcionando!

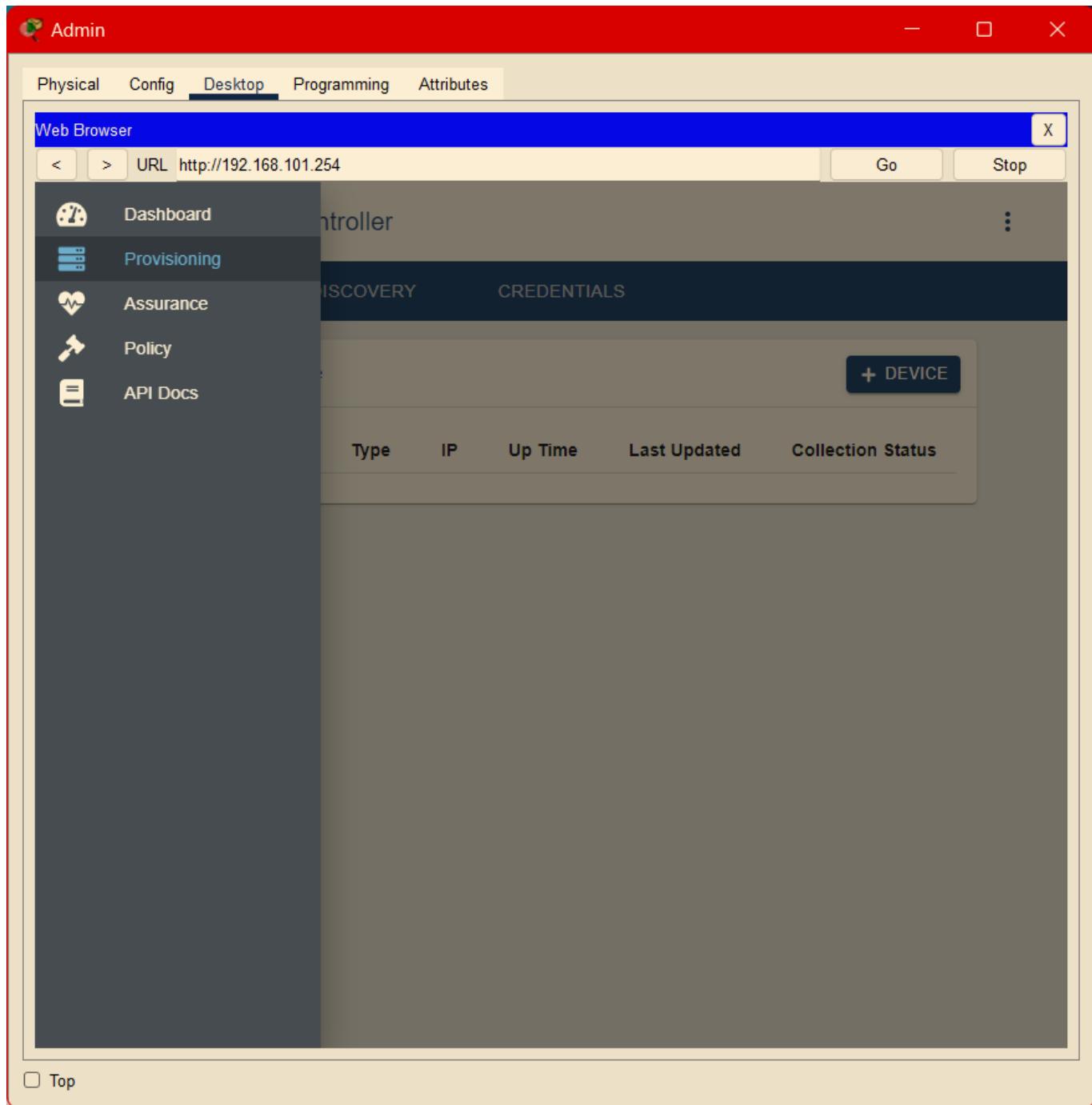
Iniciando sesión a 192.168.101.254 (GigabitEthernet0):

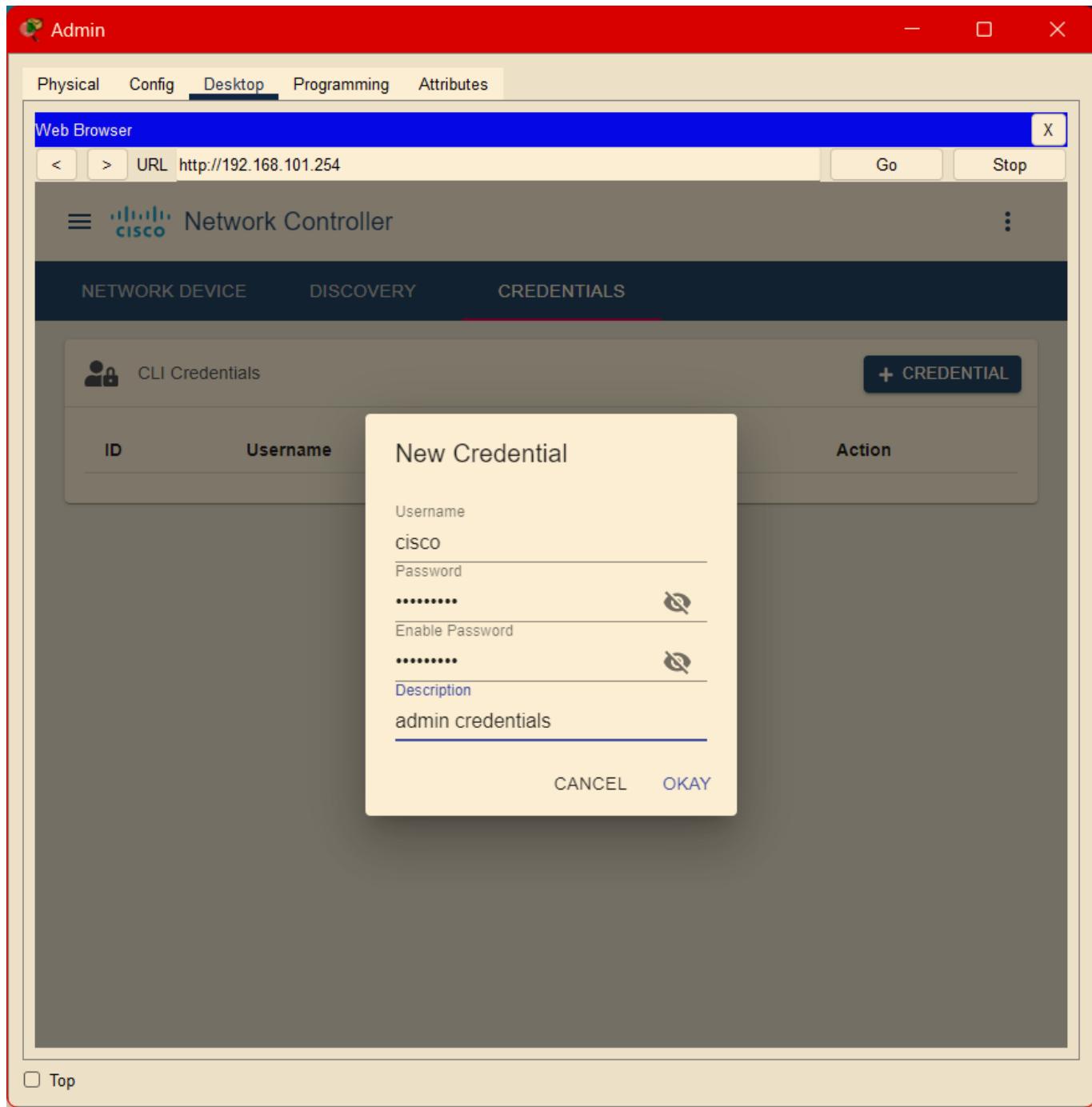
1. Creando un nuevo usuario:



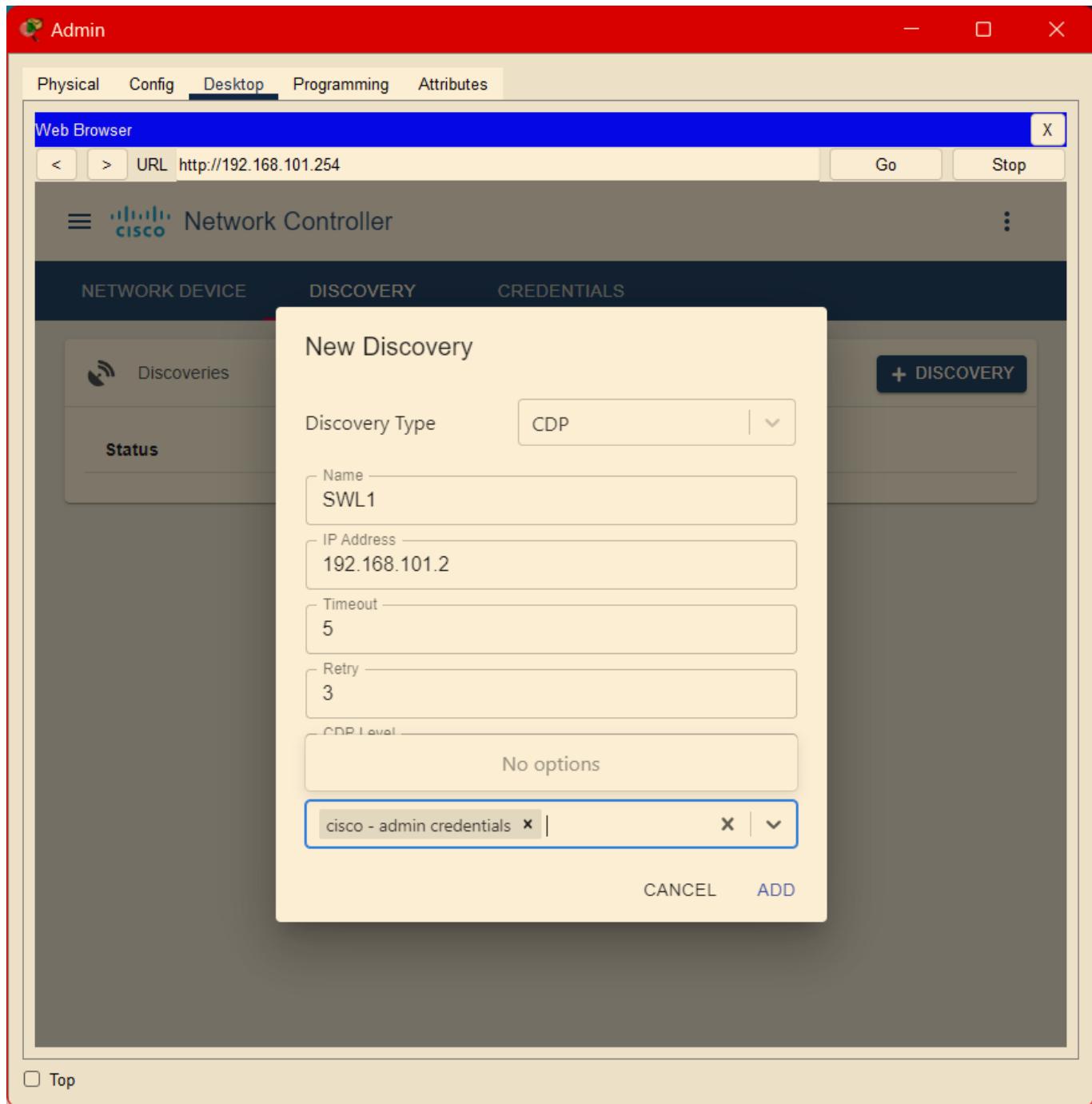


2. Creamos credenciales:





3. Usamos CDP para obtener información de los dispositivos:

 Top

Esperamos a que termine el proceso de discovery:

The screenshot shows a web browser window titled "Admin" with the URL <http://192.168.101.254>. The interface is for a "Network Controller" and includes tabs for Physical, Config, Desktop (selected), Programming, and Attributes. Below the tabs is a toolbar with back, forward, URL input, Go, and Stop buttons. The main content area has tabs for NETWORK DEVICE, DISCOVERY (selected), and CREDENTIALS. Under the DISCOVERY tab, there is a section titled "Discoveries" with a table showing four completed discoveries:

Status	Name
✓ Complete	SWL1
✓ Complete	new_device_detection
✓ Complete	internal_health_check
✓ Complete	internal_health_check

A blue button labeled "+ DISCOVERY" is visible in the top right corner of the discoveries section. At the bottom left of the main content area is a "Top" button.

Devices:

The screenshot shows a web-based network management interface titled "Admin". The top navigation bar includes tabs for "Physical", "Config", "Desktop" (which is selected), "Programming", and "Attributes". Below the navigation is a search bar labeled "Web Browser" with the URL "http://192.168.101.254". The main content area displays a Cisco Network Controller interface. A sub-header "NETWORK DEVICE" is visible above a table. The table lists nine network devices with the following details:

	Hostname	Type	IP	Up Time	Last U...
	SWL1	MultiLayerSwitch	192.168.101.2	2 hours, 24 minutes, 31 seconds	2024-0...
	R1	Router	192.168.1.2	2 hours, 24 minutes, 31 seconds	2024-0...
	R3	Router	192.168.2.1	2 hours, 24 minutes, 31 seconds	2024-0...
	R2	Router	192.168.2.2	2 hours, 24 minutes, 31 seconds	2024-0...
	SWR1	MultiLayerSwitch	10.0.1.2	2 hours, 24 minutes, 31 seconds	2024-0...
	SWR2	MultiLayerSwitch	10.0.1.3	2 hours, 24 minutes, 31 seconds	2024-0...
	SWL2	MultiLayerSwitch	192.168.102.2	2 hours, 24 minutes, 31 seconds	2024-0...
	SWR4	MultiLayerSwitch	10.0.1.5	2 hours, 24 minutes, 31 seconds	2024-0...
	SWR3	MultiLayerSwitch	10.0.1.4	2 hours, 24 minutes, 31 seconds	2024-0...

A "Top" checkbox is located at the bottom left of the interface.

De esta manera es más fácil ver la configuración de cada dispositivo.

The screenshot shows a software application window titled "Admin". The top menu bar includes "Physical", "Config", "Desktop" (which is selected), "Programming", and "Attributes". Below the menu is a "Web Browser" panel with the URL "http://192.168.101.254". The main content area displays "Device Detail" information for a network device named "SWR1". The details are listed in a table:

	Device Detail
Hostname	SWR1
ID	CAT101021Z6-uuid
Interface Count	29
Software Version	16.3.2
MAC Address	00E0.F915.E250
Management IP Address	10.0.1.2
Platform ID	3650
Product ID	3650-24PS
Serial Number	CAT101021Z6-
Type	MultiLayerSwitch
UpTime	2 hours, 25 minutes, 31 seconds
Collection Status	Managed
Connected Interface Name	GigabitEthernet0/0/0 GigabitEthernet1/0/2 GigabitEthernet1/0/3 GigabitEthernet1/0/5

At the bottom left of the browser panel, there is a "Top" button.

Vemos las dashboards del Network Controller:

The screenshot shows the Cisco Network Controller interface. At the top, there's a navigation bar with tabs: Physical, Config, Desktop (which is selected), Programming, and Attributes. Below the navigation bar is a web browser window with the URL <http://192.168.101.254>. The main content area displays several cards:

- Percentage of hosts can be reached via ping**: Shows a green bar chart at 100% (Health Level 100%). The x-axis shows times: 22:46:43, 22:46:49, 22:46:55, 22:47:01, 22:47:07.
- Percentage of network devices that are in Managed state**: Shows a green bar chart at 88% (Health Level 88%). The x-axis shows times: 22:46:43, 22:46:49, 22:46:55, 22:47:01, 22:47:07.
- Qos**: Shows 0 Scope and 0 Policy.
- Host**: Shows 6 Hosts. A pie chart indicates 100% Ping succeeded (green) and 0% Ping failed (red).
- Network Device**: Shows a single green bar at the bottom.

At the bottom left, there's a "Top" checkbox. On the right side, there's a vertical scroll bar.

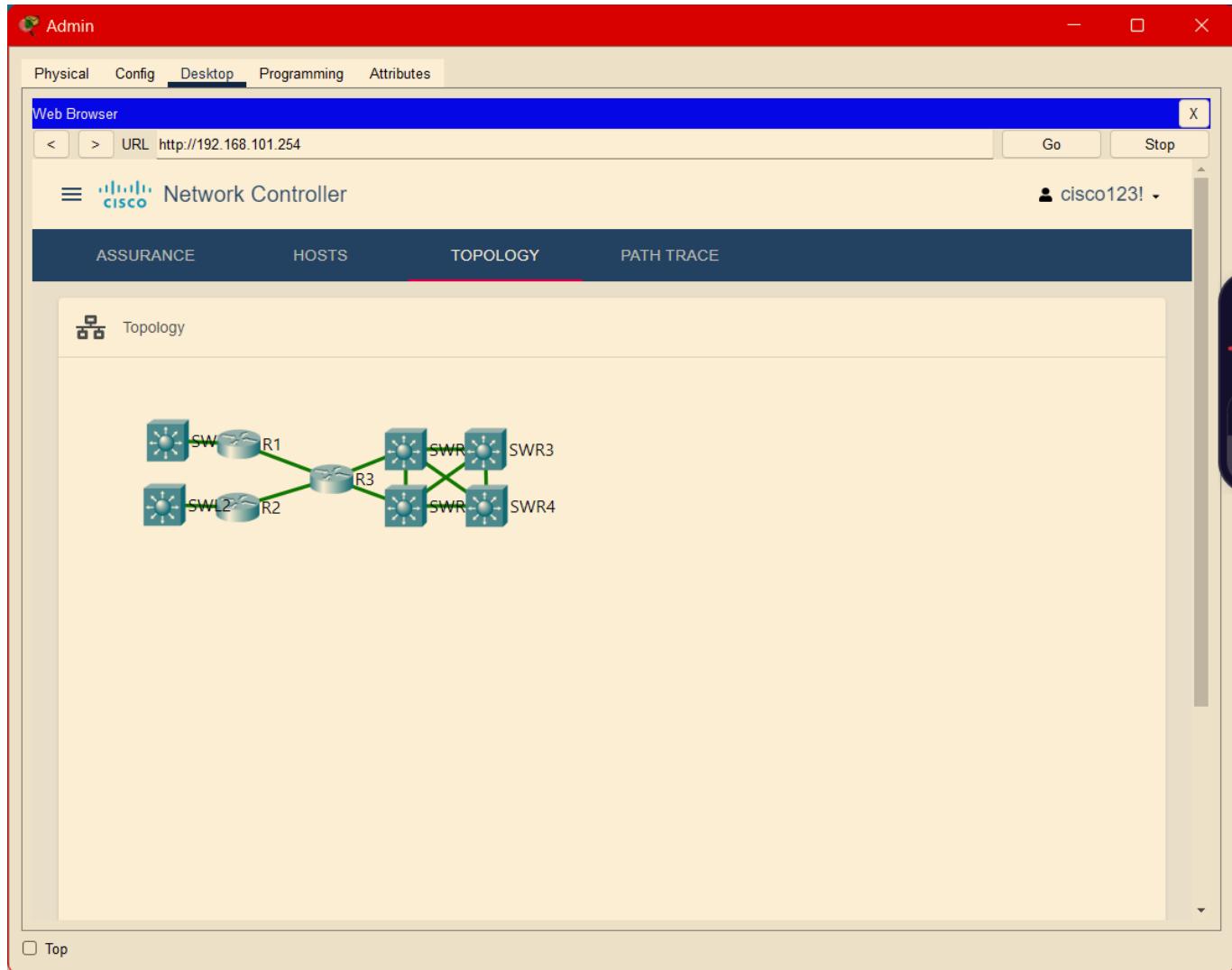
Se ve como una herramienta útil para la administración y monitoreo de la red. Vemos el número de hosts y devices.

The screenshot shows a web browser window titled "Admin" with the URL <http://192.168.101.254>. The page is titled "Network Controller" and features a navigation bar with tabs: ASSURANCE, HOSTS (which is selected), TOPOLOGY, and PATH TRACE. A user profile "cisco123!" is visible in the top right. The main content area is titled "Host" and contains a table with the following data:

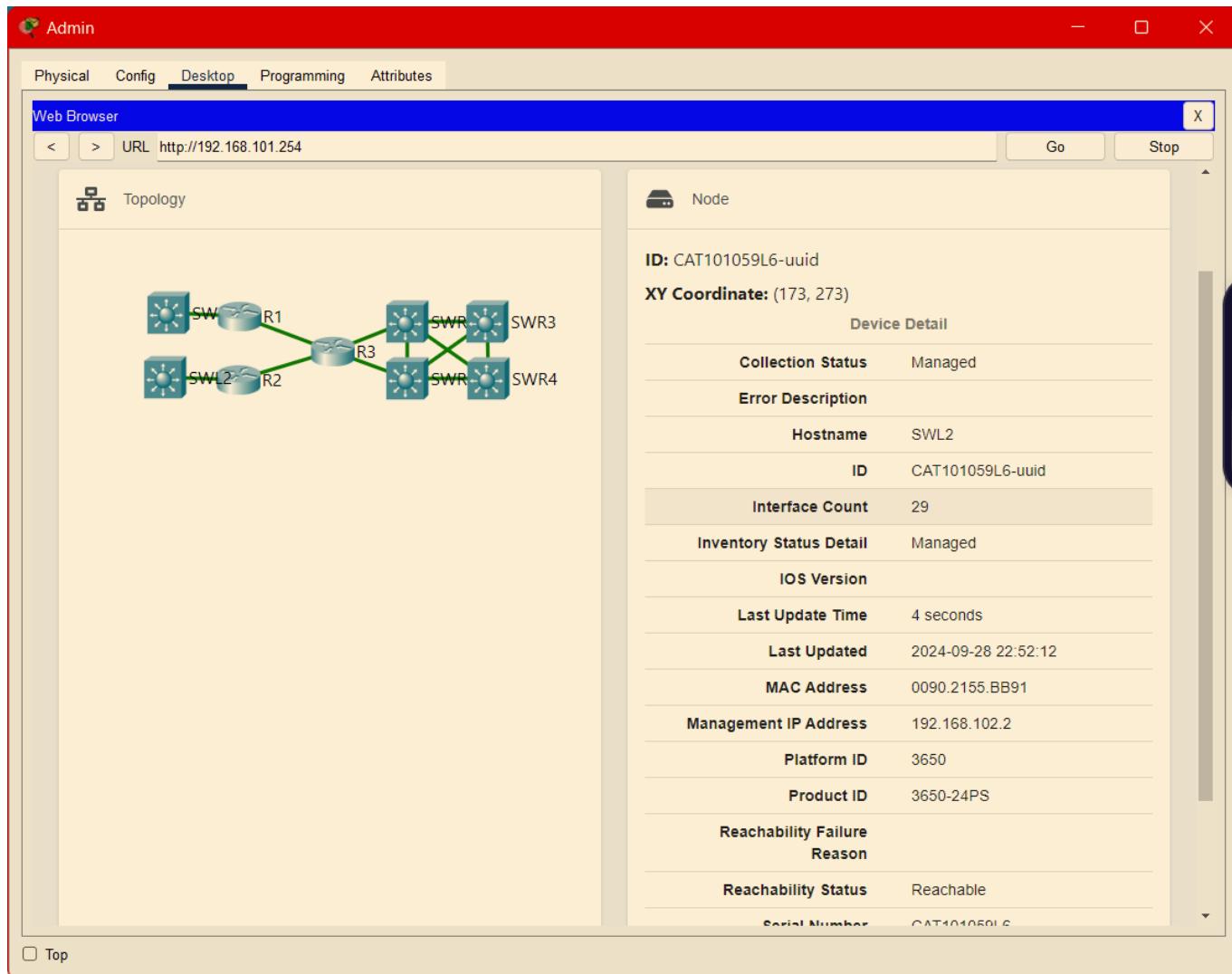
Host Device				Connected Network Device		
MAC	IP	Hostname	Type	IP	Hostname	Port
000A.413D.D793	192.168.101.100	Example Server	Server	192.168.101.2	SWL1	GigabitEthernet1/0/3
00E0.F96C.155B	192.168.102.3	PC4	Pc	192.168.102.2	SWL2	GigabitEthernet1/0/24
0060.47C1.A4DB	10.0.2.130	PC2	Pc	10.0.1.5	SWR4	GigabitEthernet1/0/23
0004.9A42.C245	10.0.2.129	PC3	Pc	10.0.1.5	SWR4	GigabitEthernet1/0/24
0050.0FCE.B095	10.0.1.130	Admin	Pc	10.0.1.4	SWR3	GigabitEthernet1/0/21
00E0.A330.3359	10.0.1.129	PC1	Pc	10.0.1.4	SWR3	GigabitEthernet1/0/22

Top

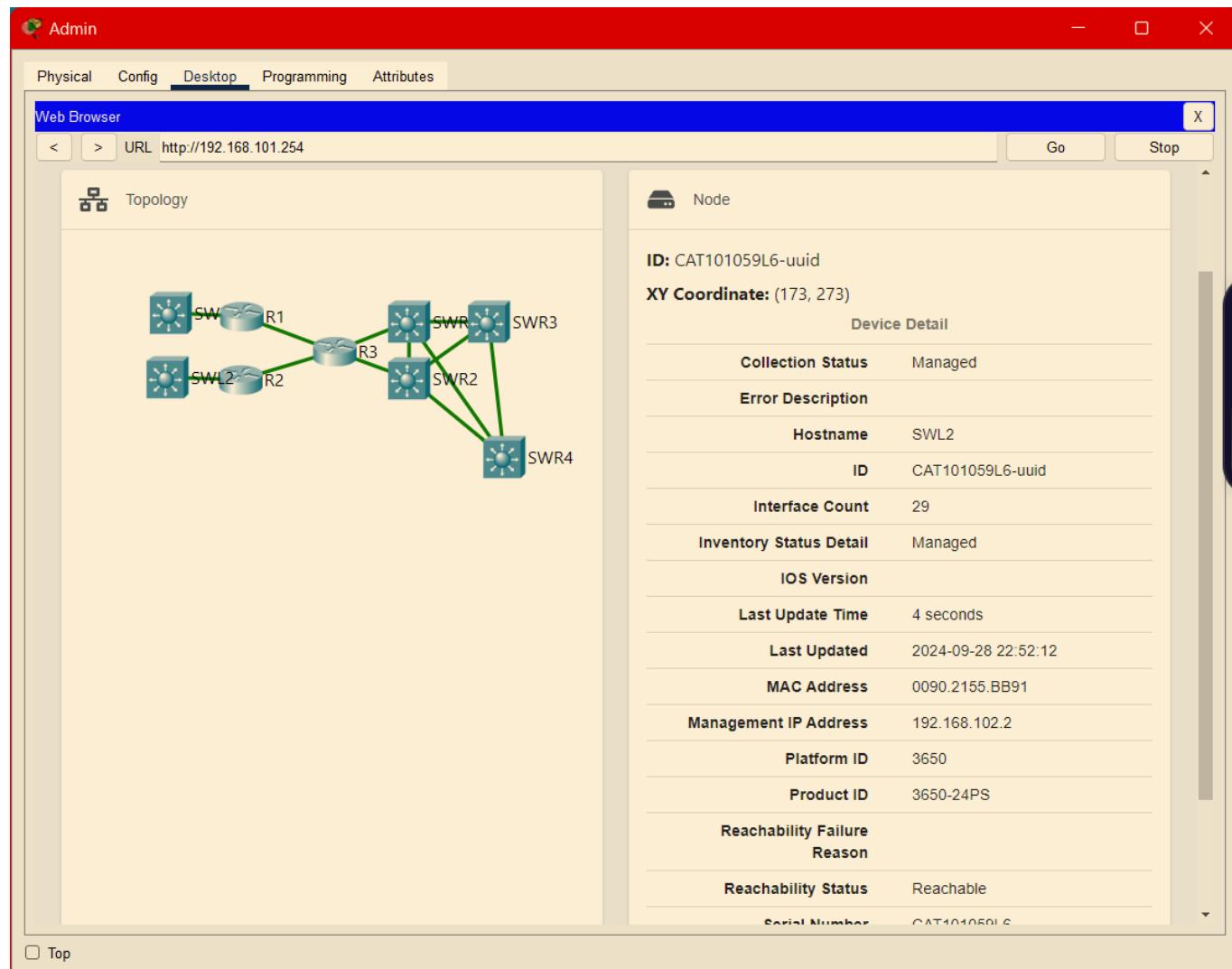
También podemos ver el topology de la red:



Se puede ver el detalle de cada switch:

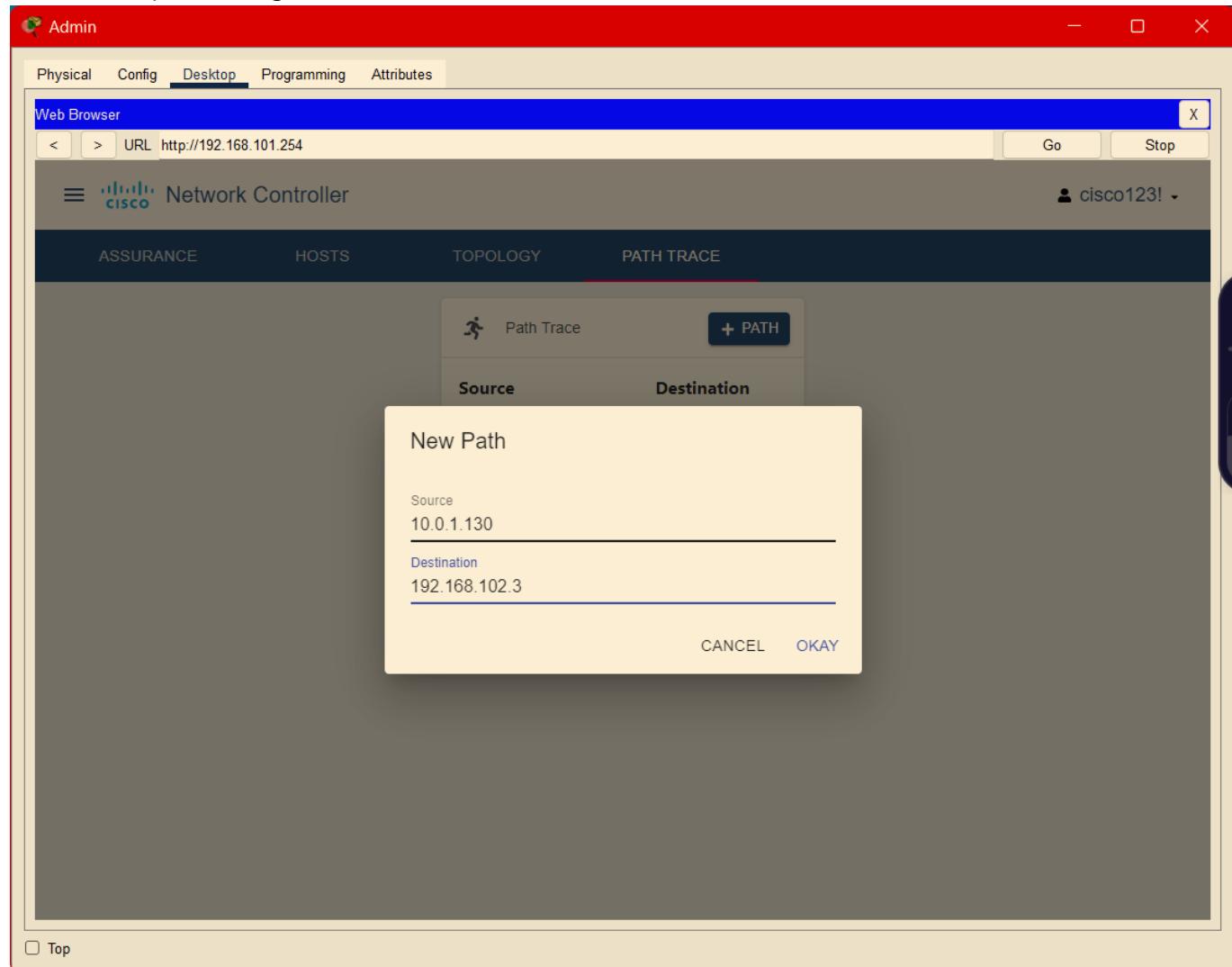


También moverlos, etc.

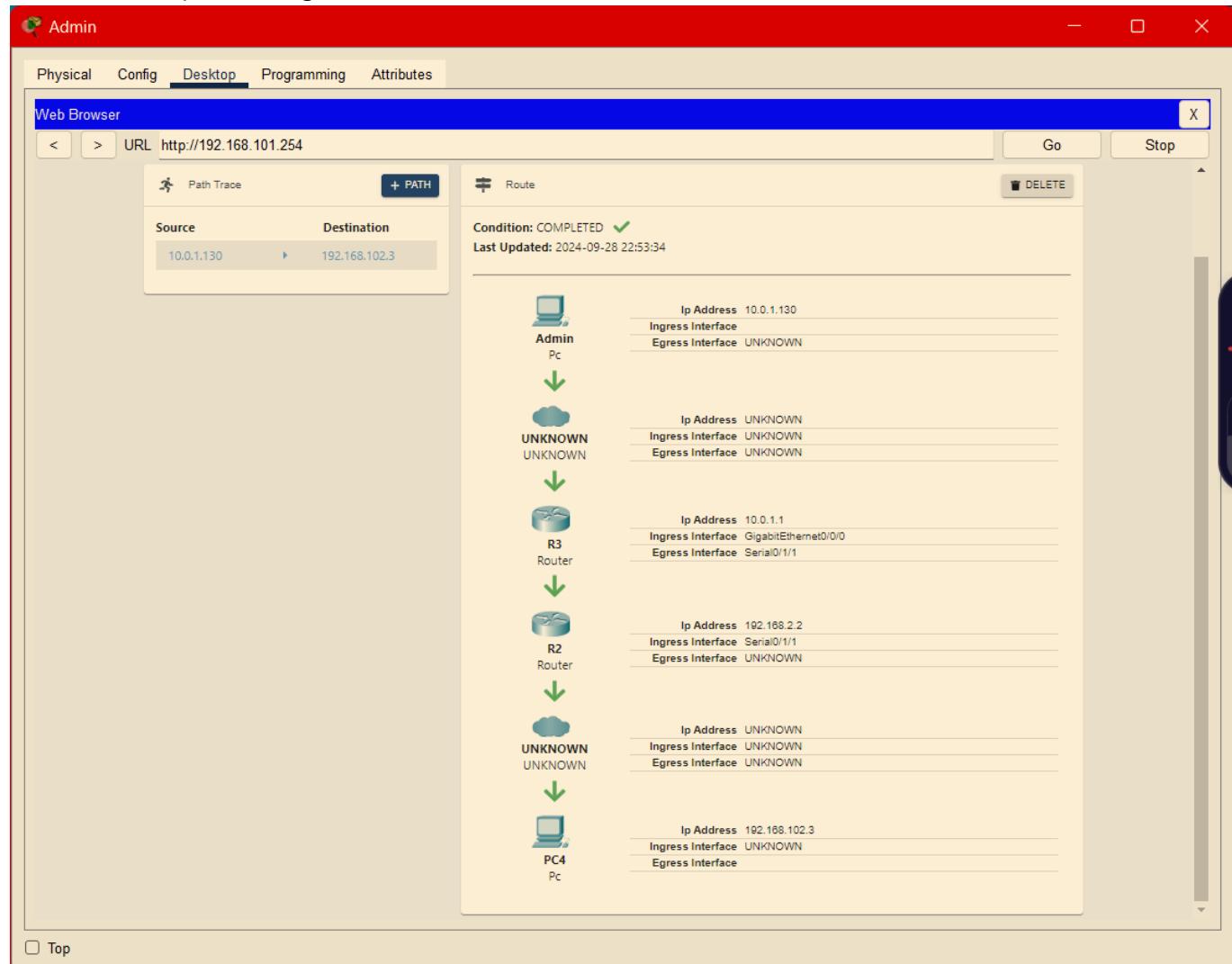


Path tracing

Seteamos el path tracing:



Observamos el path tracing:



Usando un controlador SDN para configurar la red:

Vemos el DNS de Example Server:

The screenshot shows the 'Example Server' application window. The title bar has the text 'Example Server'. The top menu bar includes 'Physical', 'Config', 'Services', 'Desktop', 'Programming', and 'Attributes'. The 'Services' tab is selected, highlighted in blue. On the left, a vertical sidebar lists various services: HTTP, DHCP, DHCPv6, TFTP, DNS (which is selected and highlighted in blue), SYSLOG, AAA, NTP, EMAIL, FTP, IoT, VM Management, and Radius EAP. The main right-hand panel is titled 'DNS'. It contains a section for 'DNS Service' with two radio buttons: 'On' (selected) and 'Off'. Below this is a 'Resource Records' section. A table lists one record: No. 0, Name www.example.com, Type A Record, Detail 192.168.101.100. At the bottom of the panel are buttons for 'Add', 'Save', and 'Remove'. A 'DNS Cache' button is located at the bottom left of the main panel. The bottom left corner of the window has a 'Top' button.

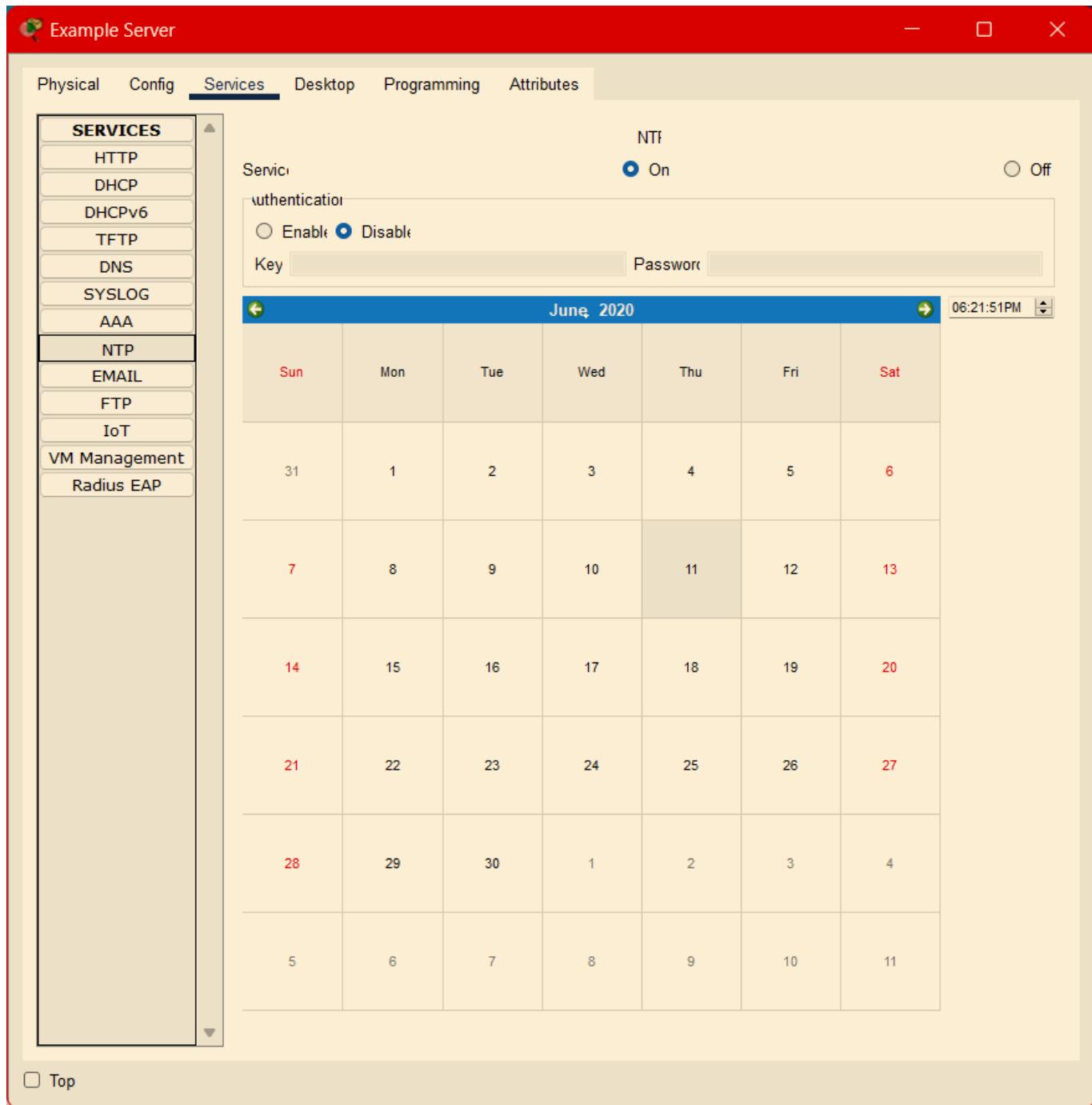
Notamos que Syslog está activo

The screenshot shows the Example Server software interface. The title bar says "Example Server". The top menu bar has tabs: Physical, Config, Services (which is underlined, indicating it's selected), Desktop, Programming, and Attributes. On the left, there's a sidebar with a tree view of services: SERVICES (HTTP, DHCP, DHCPv6, TFTP, DNS, SYSLOG, AAA, NTP, EMAIL, FTP, IoT), VM Management, and Radius EAP. The "SYSLOG" node is currently selected. The main pane displays a log entry for the "Syslog" service. The log table has columns: Time, HostName, and Message. There are two entries:

Time	HostName	Message
1 -	192.168.101.2	%LINK-5-CHANGED: Interface ...
2 -	192.168.101.2	%LINEPROTO-5-UPDOWN: Line protoc...

At the bottom right of the main pane, there's a "Clear Log" button. At the very bottom of the window, there's a toolbar with a "Top" checkbox.

Y NTP también



Configuramos políticas de seguridad:

The screenshot shows a web browser window titled "Admin" with the URL <http://192.168.101.254>. The browser has tabs for Physical, Config, Desktop, Programming, and Attributes, with "Desktop" selected. The main content area is titled "Network Controller". It displays two sections: "QOS" and "NETWORK SETTINGS".

QOS Section:

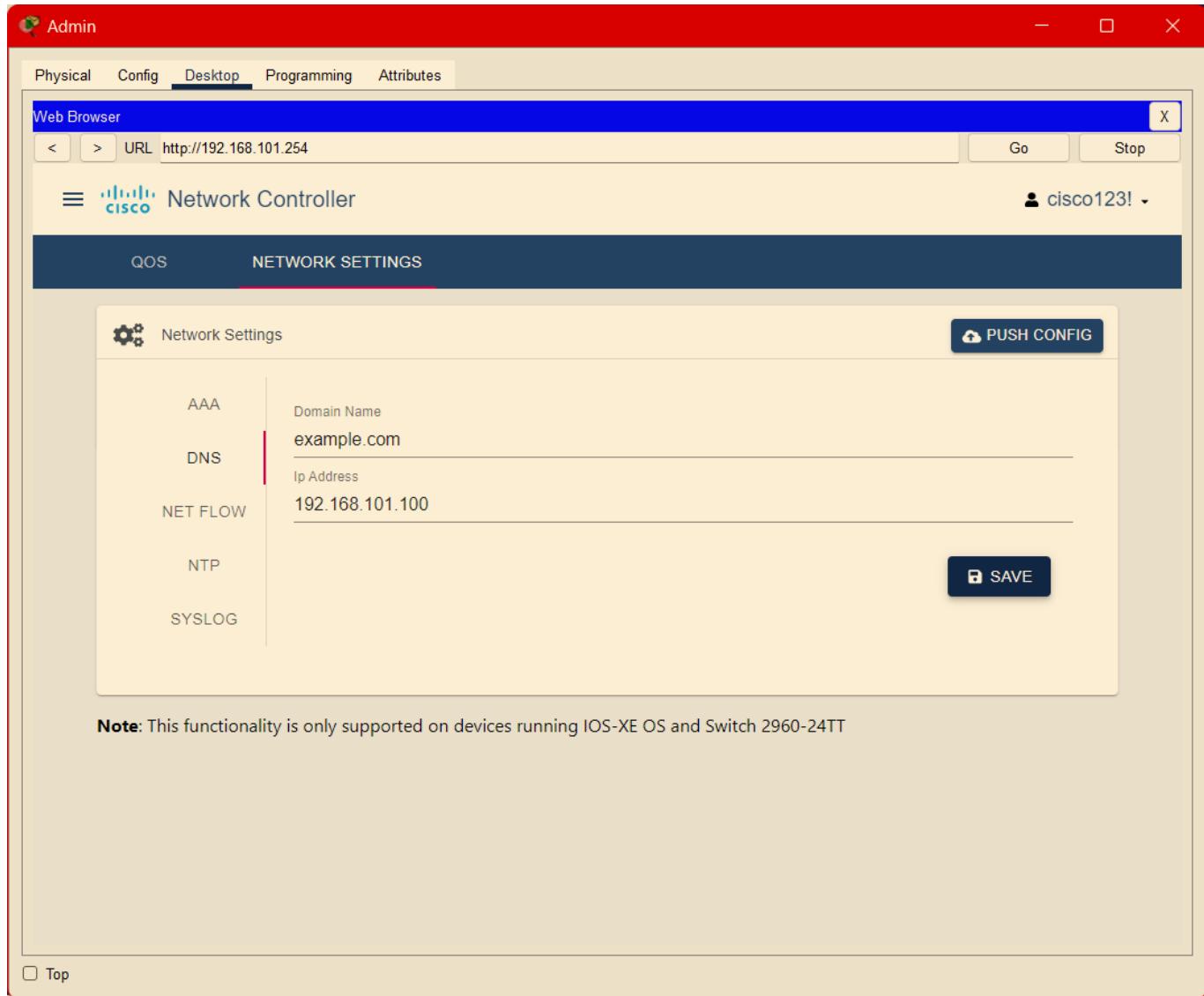
- Header: Scope
- Buttons: + SCOPE
- Table Headers: Scope Name, Device Name, Device IP
- Table Rows: (empty)

Network Settings Section:

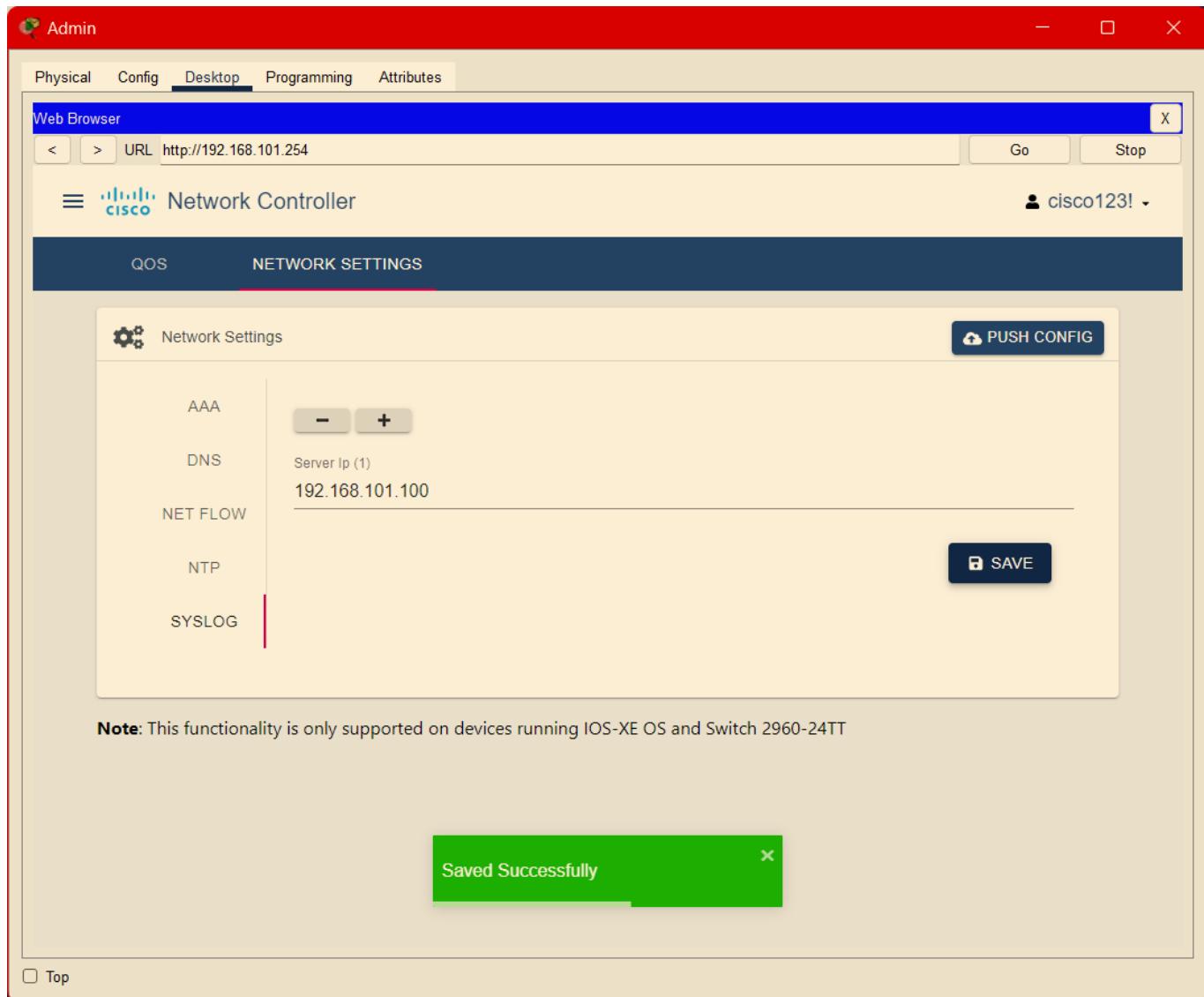
- Header: Policy
- Buttons: + POLICY
- Table Headers: Policy Name, Relevance Level, Protocol, Scope
- Table Rows: (empty)

Note: This functionality is only supported on devices running IOS-XE OS

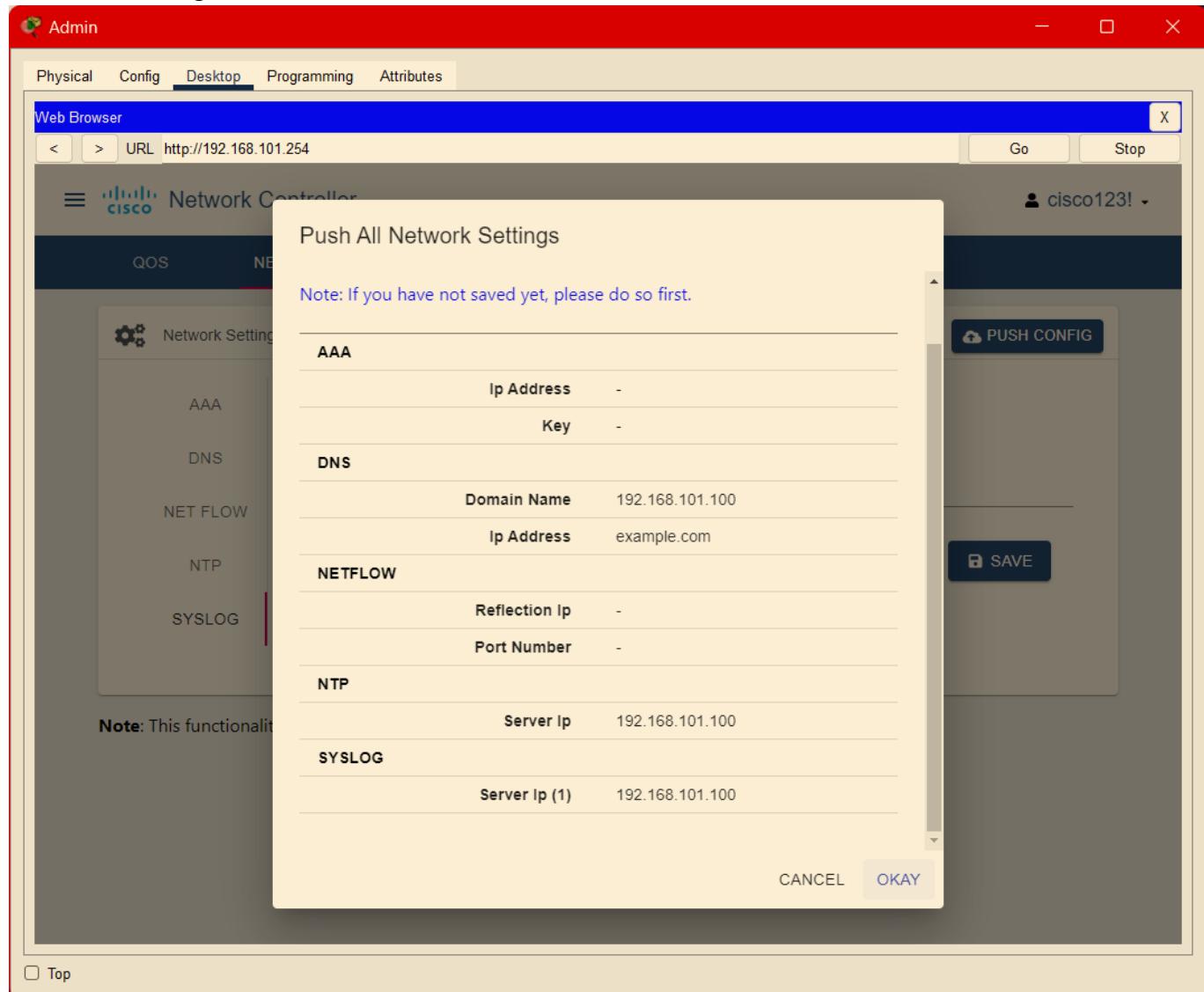
Top



The screenshot shows the Cisco Network Controller interface. At the top, there's a navigation bar with tabs: Physical, Config, Desktop, Programming, and Attributes. The Desktop tab is currently selected. Below the navigation bar is a Web Browser window with the URL <http://192.168.101.254>. The main content area is titled "Network Settings". On the left, there's a sidebar with options: AAA, DNS, NET FLOW, NTP, and SYSLOG. The "AAA" section is active, showing a "Server Ip" field with the value "192.168.101.100". On the right side of the main panel, there are two buttons: "PUSH CONFIG" and "SAVE". A note at the bottom states: "Note: This functionality is only supported on devices running IOS-XE OS and Switch 2960-24TT". A green success message box in the center says "Saved Successfully".



Pusheamos configuraciones:



Verificamos la configuración:

The screenshot shows a window titled 'R1' with a red header bar. Below the header are tabs: 'Physical', 'Config', 'CLI' (which is selected), and 'Attributes'. The main area is labeled 'IOS Command Line Interface'. The terminal output is as follows:

```

3223551K bytes of flash memory at bootflash:.

Press RETURN to get started!

%SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.101.100 port 514 started - CLI initiated

%LINK-5-CHANGED: Interface Serial0/1/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0, changed state to up

23:38:10: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.2.1 on Serial0/1/0 from LOADING to FULL, Loading Done

R1>enable
R1#show run | begin ip domain
ip domain-name example.com
ip name-server 192.168.101.100
!
!
spanning-tree mode pvst
!
!
!
!
!
interface GigabitEthernet0/0/0
ip address 192.168.101.1 255.255.255.0
duplex auto
speed auto
!
interface GigabitEthernet0/0/1
no ip address
duplex auto
speed auto
shutdown
!
--More--

```

At the bottom right of the terminal window are 'Copy' and 'Paste' buttons. At the bottom left is a 'Top' button.

```

R1>enable
R1#show run | begin ip domain
ip domain-name example.com
ip name-server 192.168.101.100

```

Luego vemos las asociaciones de NTP, el clock y el logging, y finalmente cambiamos el estado del interfaz Serial0/1/0 a administratively down y luego a up.

address	ref clock	st	when	poll	reach	delay	offset
disp							
*~192.168.101.100	127.127.1.1	1	1	16	177	0.00	0.00

```
0.12
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured
R1#show clock
18:30:10.789 UTC Thu Jun 11 2020
R1#show run | include logging
logging 192.168.101.100
R1#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#interface s0/1/0
R1(config-if)#shutdown

R1(config-if)#
%LINK-5-CHANGED: Interface Serial0/1/0, changed state to administratively down

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0, changed state to down

18:30:34: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.2.1 on Serial0/1/0 from FULL to
DOWN, Neighbor Down: Interface down or detached

R1(config-if)#no shutdown

R1(config-if)#
%LINK-5-CHANGED: Interface Serial0/1/0, changed state to up

R1(config-if)#end
R1#
%SYS-5-CONFIG_I: Configured from console by console
%SYS-6-LOGGINGHOST_STARTSTOP: Logging to host 192.168.101.100 port 514 started -
CLI initiated
```

Part 2: Build a CI/CD Pipeline Using Jenkins

Part 1: Launch the DEVASC VM



Part 2: Commit the Sample App to Git

Creando repositorio en GitHub:

The screenshot shows the GitHub interface for creating a new repository. At the top, there's a navigation bar with icons for search, issues, pull requests, and user profile. Below it, the main title is "Create a new repository". A note says "A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository." It also mentions that required fields are marked with an asterisk (*).

Repository template: A dropdown menu is set to "No template". A note below it says "Start your repository with a template repository's contents.".

Owner *: A dropdown menu shows "A-PachecoT". **Repository name ***: A text input field contains "sample-app". A note below it says "✓ sample-app is available."

Description (optional): A text area contains "Explore CI/CD with GitHub and Jenkins".

Visibility: Two radio button options are shown: "Public" (unchecked) and "Private" (checked). A note for "Public" says "Anyone on the internet can see this repository. You choose who can commit." A note for "Private" says "You choose who can see and commit to this repository."

Initialize this repository with: An unchecked checkbox for "Add a README file". A note below it says "This is where you can write a long description for your project. [Learn more about READMEs](#)".

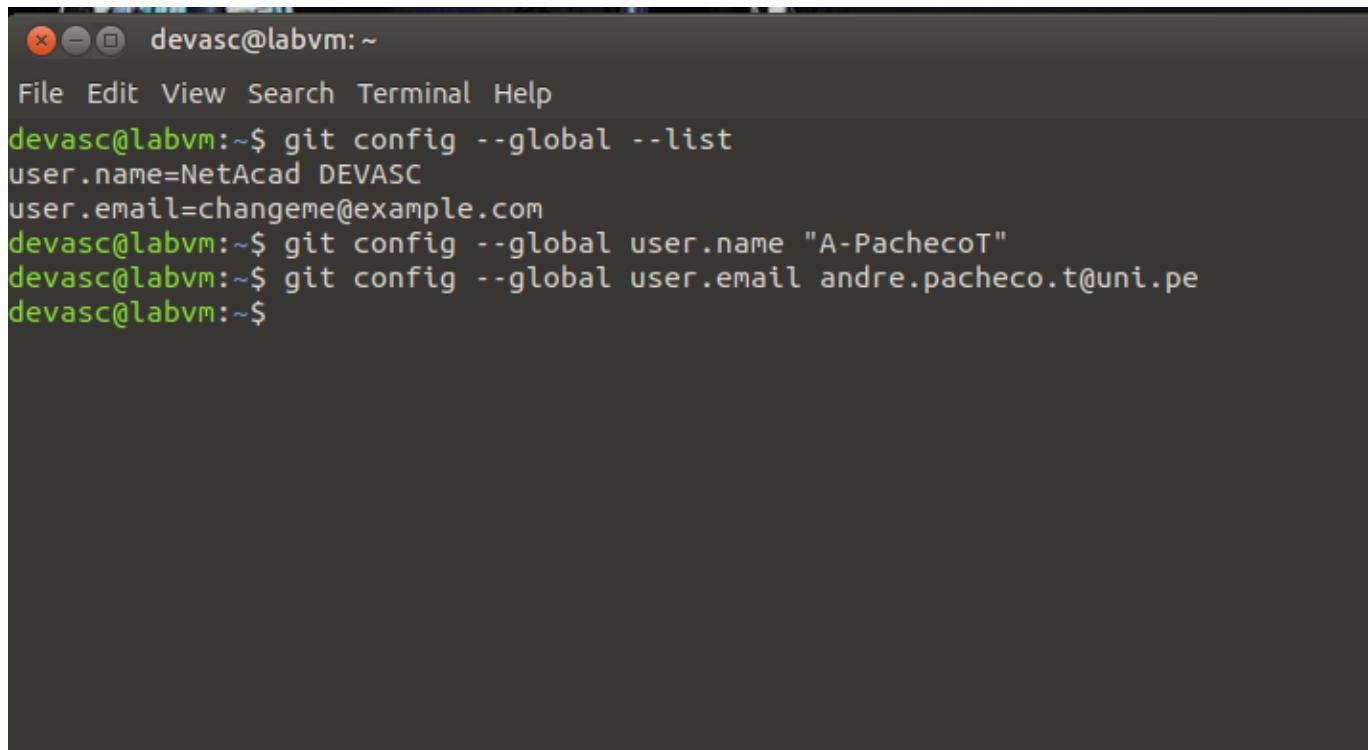
Add .gitignore: A dropdown menu is set to ".gitignore template: None". A note below it says "Choose which files not to track from a list of templates. [Learn more about ignoring files](#)".

Choose a license: A dropdown menu is set to "License: None". A note below it says "A license tells others what they can and can't do with your code. [Learn more about licenses](#)".

Information: A note says "ⓘ You are creating a private repository in your personal account."

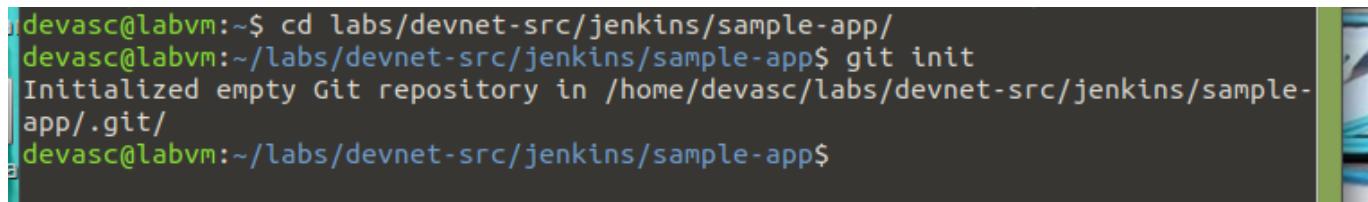
Create repository: A large green button at the bottom right.

Configurando credenciales en la máquina virtual:



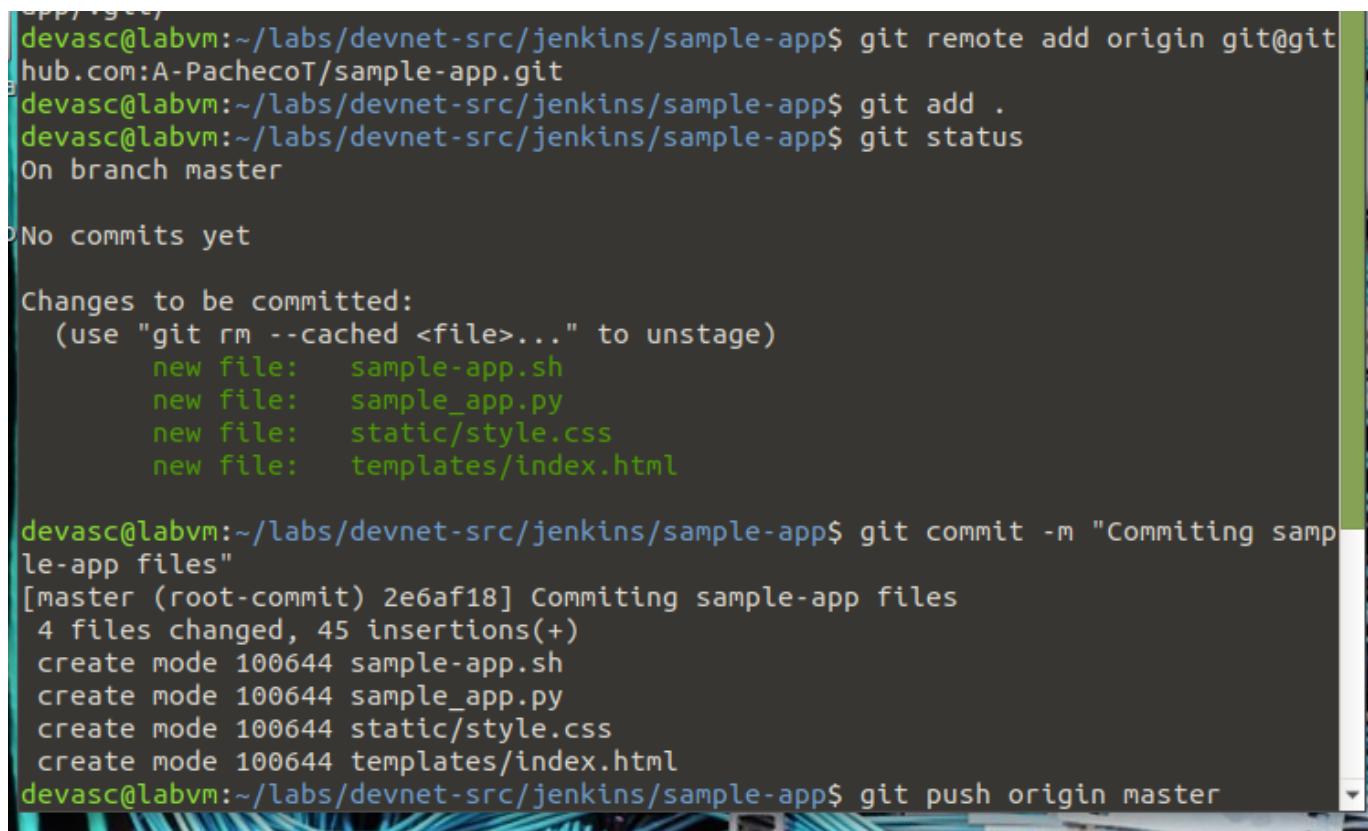
```
devasc@labvm:~$ git config --global --list
user.name=NetAcad DEVASC
user.email=changeme@example.com
devasc@labvm:~$ git config --global user.name "A-PachecoT"
devasc@labvm:~$ git config --global user.email andre.pacheco.t@uni.pe
devasc@labvm:~$
```

Inicializando repositorio local:



```
devasc@labvm:~$ cd labs/devnet-src/jenkins/sample-app/
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git init
Initialized empty Git repository in /home/devasc/labs/devnet-src/jenkins/sample-app/.git/
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$
```

Enlazando con el repositorio remoto y agregando los archivos con add, commiteando y puseando a master:



```
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git remote add origin git@git.hub.com:A-PachecoT/sample-app.git
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git add .
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   sample-app.sh
    new file:   sample_app.py
    new file:   static/style.css
    new file:   templates/index.html

devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git commit -m "Committing sample-app files"
[master (root-commit) 2e6af18] Committing sample-app files
 4 files changed, 45 insertions(+)
 create mode 100644 sample-app.sh
 create mode 100644 sample_app.py
 create mode 100644 static/style.css
 create mode 100644 templates/index.html
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git push origin master
```

El uso de contraseñas para la autenticación es más fácil de implementar, pero Github deprecó el uso de contraseñas para la autenticación por lo que se uso claves ssh. Por ello tengo que crear una clave ssh:

```
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ ssh-keygen -t ed25519 -C "andre.pacheco.t@uni.pe"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/devasc/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/devasc/.ssh/id_ed25519
Your public key has been saved in /home/devasc/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:agvkBEgHfh9R8DNanICpRKpEQ1ZYNVEZnxx6PyleLew andre.pacheco.t@uni.pe
The key's randomart image is:
++-[ED25519 256]---+
|o0=++B=+o.          |
|*+oo *.= o          |
|+oo.. . 0 =          |
|o.... + + o o       |
|. oo S. B .         |
| + ... + o          |
| o o . E            |
| o .                |
| .                  |
+---[SHA256]---+
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ eval "$(ssh-agent -s)"
Agent pid 4488
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ ssh-add ~/.ssh/id_ed25519
Identity added: /home/devasc/.ssh/id_ed25519 (andre.pacheco.t@uni.pe)
```

con

```
cat ~/.ssh/id_rsa.pub
```

vemos la clave. Esta clave se añade a las llaves de mi cuenta de github:

The screenshot shows the GitHub Settings interface. On the left, there's a sidebar with various settings options like Public profile, Account, Appearance, Accessibility, Notifications, Access, Billing and plans, Emails, Password and authentication, Sessions, SSH and GPG keys (which is selected and highlighted in blue), Organizations, Enterprises, and Moderation. Below the sidebar, there are sections for Code, planning, and automation, including Repositories. The main content area is titled "Add new SSH Key". It has fields for "Title" (set to "devasc-vm") and "Key type" (set to "Authentication Key"). The "Key" field contains the following text:
ssh-ed25519 POR SEGURIDAD NO PONGO TODA LA CLASE
AQUI...vJ0 andre.pacheco.t@uni.pe

Finalmente puedo pushear a mi repositorio remoto:

```
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git push origin master
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 2 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (8/8), 1.03 KiB | 1.03 MiB/s, done.
Total 8 (delta 0), reused 0 (delta 0)
To github.com:A-PachecoT/sample-app.git
 * [new branch]      master -> master
```

Part 3: Modify the Sample App and Push Changes to Git

Ahora cambiemos archivos...

Se cambia el puerto en el .py y .hs:

```
x - devasc@labvms: ~/labs/devnet-src/jenkins/sample-app
File Edit View Search Terminal Help
# Add to this file for the sample app lab
from flask import Flask
from flask import request
from flask import render_template

sample = Flask(__name__)

@app.route("/")
def main():
    return render_template("index.html")

if __name__ == "__main__":
    sample.run(host="0.0.0.0", port=5050)
~
```

```
x - devasc@labvms: ~/labs/devnet-src/jenkins/sample-app
File Edit View Search Terminal Help

cp sample_app.py tempdir/.
cp -r templates/* tempdir/templates/.
cp -r static/* tempdir/static/.

echo "FROM python" >> tempdir/Dockerfile
echo "RUN pip install flask" >> tempdir/Dockerfile
echo "COPY ./static /home/myapp/static/" >> tempdir/Dockerfile
echo "COPY ./templates /home/myapp/templates/" >> tempdir/Dockerfile
echo "COPY sample_app.py /home/myapp/" >> tempdir/Dockerfile
echo "EXPOSE 5050" >> tempdir/Dockerfile
echo "CMD python /home/myapp/sample_app.py" >> tempdir/Dockerfile

cd tempdir

docker build -t sampleapp .

docker run -t -d -p 5050:5050 --name samplerunning sampleapp
docker ps -a
~
```

Tenía el contenedor creado así que tuve que eliminar y volver a crearlo:

```
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             NAMES
6e45c30c5ceb      e6554b917808      "/bin/sh -c 'python3..."   41 hours ago    samplerunning
           Exited (137) 40 hours ago
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ docker rm ^C
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ docker rm 6e45c30c5ceb
6e45c30c5ceb
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             NAMES
STATUS              PORTS              NAMES

```

Y vuelvo a ejecutar el .sh:

```
devasc@labvm: ~/labs/devnet-src/jenkins/sample-app
File Edit View Search Terminal Help
---> d4a34b986a46
Step 7/14 : CMD python /home/myapp/sample_app.py
---> Using cache
---> 8f24c207da6b
Step 8/14 : FROM python
---> ea2ebd905ab2
Step 9/14 : RUN pip install flask
---> Using cache
---> 65a193e541eb
Step 10/14 : COPY ./static /home/myapp/static/
---> Using cache
---> 285bad4ca062
Step 11/14 : COPY ./templates /home/myapp/templates/
---> Using cache
---> 413d2b236dff
Step 12/14 : COPY sample_app.py /home/myapp/
---> Using cache
---> 2599f082c490
Step 13/14 : EXPOSE 5050
---> Using cache
---> d4a34b986a46
Step 14/14 : CMD python /home/myapp/sample_app.py
---> Using cache
---> 8f24c207da6b
Successfully built 8f24c207da6b
Successfully tagged sampleapp:latest
0615e1c27e0ffc4c4f921361ce2b2606373e7629e399b6c304d83e598b7c1487
CONTAINER ID        IMAGE               COMMAND                  CREATED             NAMES
STATUS              PORTS              NAMES
0615e1c27e0f      sampleapp          "/bin/sh -c 'python ..."   1 second ago    samplerunning
           Up Less than a second  0.0.0.0:5050->5050/tcp
devasc@labvm: ~/labs/devnet-src/jenkins/sample-app$
```

Listo, ya tenemos el contenedor funcionando con el nuevo puerto.

Por ultimo subimos los cambios a mi repositorio remoto:

```
devasc@labvm:~/labs/devnet-src/jenkins/sample-app
File Edit View Search Terminal Help
Up Less than a second 0.0.0.0:5050->5050/tcp samplerunning
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git add .
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   sample-app.sh
    modified:   sample_app.py
    new file:   tempdir/Dockerfile
    new file:   tempdir/sample_app.py
    new file:   tempdir/static/style.css
    new file:   tempdir/templates/index.html

devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git commit -m "chore: changed port from 8080 to 5050"
[master 9f5a281] chore: changed port from 8080 to 5050
6 files changed, 42 insertions(+), 3 deletions(-)
create mode 100644 tempdir/Dockerfile
create mode 100644 tempdir/sample_app.py
create mode 100644 tempdir/static/style.css
create mode 100644 tempdir/templates/index.html
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ git push origin master
Warning: Permanently added the ECDSA host key for IP address '140.82.112.4' to the list of known hosts.
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 2 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 785 bytes | 98.00 KiB/s, done.
Total 6 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:A-PachecoT/sample-app.git
  2e6af18..9f5a281 master -> master
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$
```

Efectivamente se pusieron los cambios.

A-PachecoT chore: changed port from 8080 to 5050
static Committing sample-app files
tempdir chore: changed port from 8080...
templates Committing sample-app files
sample-app.sh chore: changed port from 8080...
sample_app.py chore: changed port from 8080...

About

Explore CI/CD with GitHub and Jenkins

Activity 0 stars 1 watching 0 forks

Releases

No releases published [Create a new release](#)

Packages

Part 4: Download and Run the Jenkins Docker Image

Descargamos la imagen Its de jenkins con el comando:

```
docker pull jenkins/jenkins:lts
```

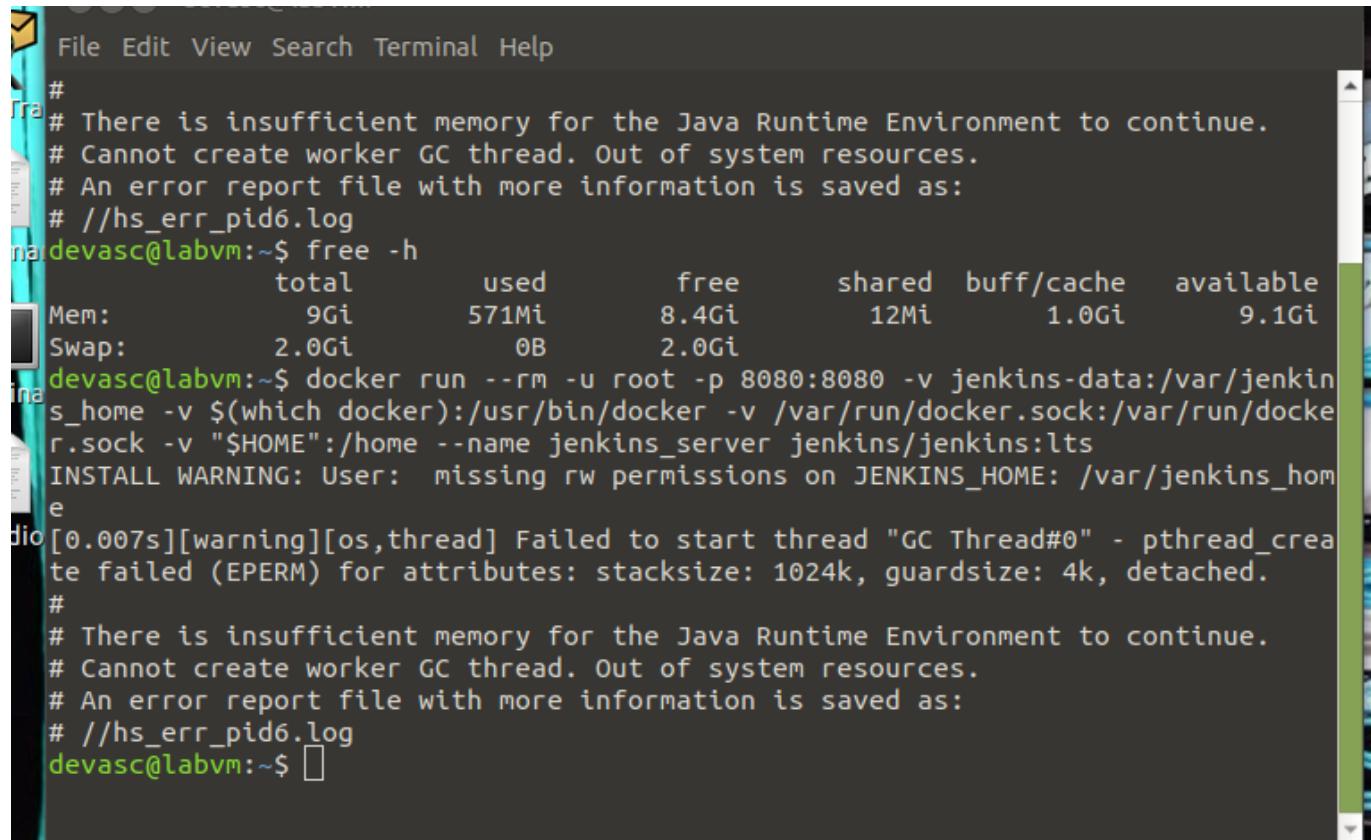
```
2e6af18..9f5a281 master -> master
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ docker pull jenkins/jenkins:lts
lts: Pulling from jenkins/jenkins
903681d87777: Pull complete
f76fc73f1c48: Pull complete
fc1d2482b243: Pull complete
5bae62448211: Pull complete
9020ffff6008: Pull complete
8a9191d56587: Pull complete
da374eff6f05: Pull complete
31aceeb653c9: Pull complete
061dcfd72fbac: Pull complete
79716cc251e4: Pull complete
628c862ab449: Pull complete
d7dec4cb14f6: Pull complete
Digest: sha256:95313257a8cddbef83c74e3d577ea139aeae30c3c014ddcaa83a72b60409bbe1
Status: Downloaded newer image for jenkins/jenkins:lts
docker.io/jenkins/jenkins:lts
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$
```

Empezamos el contenedor con el comando:

```
docker run --rm -u root -p 8080:8080 -v jenkins-data:/var/jenkins_home -v $(which docker):/usr/bin/docker -v /var/run/docker.sock:/var/run/docker.sock -v
```

```
"$HOME":/home --name jenkins_server jenkins/jenkins:lts
```

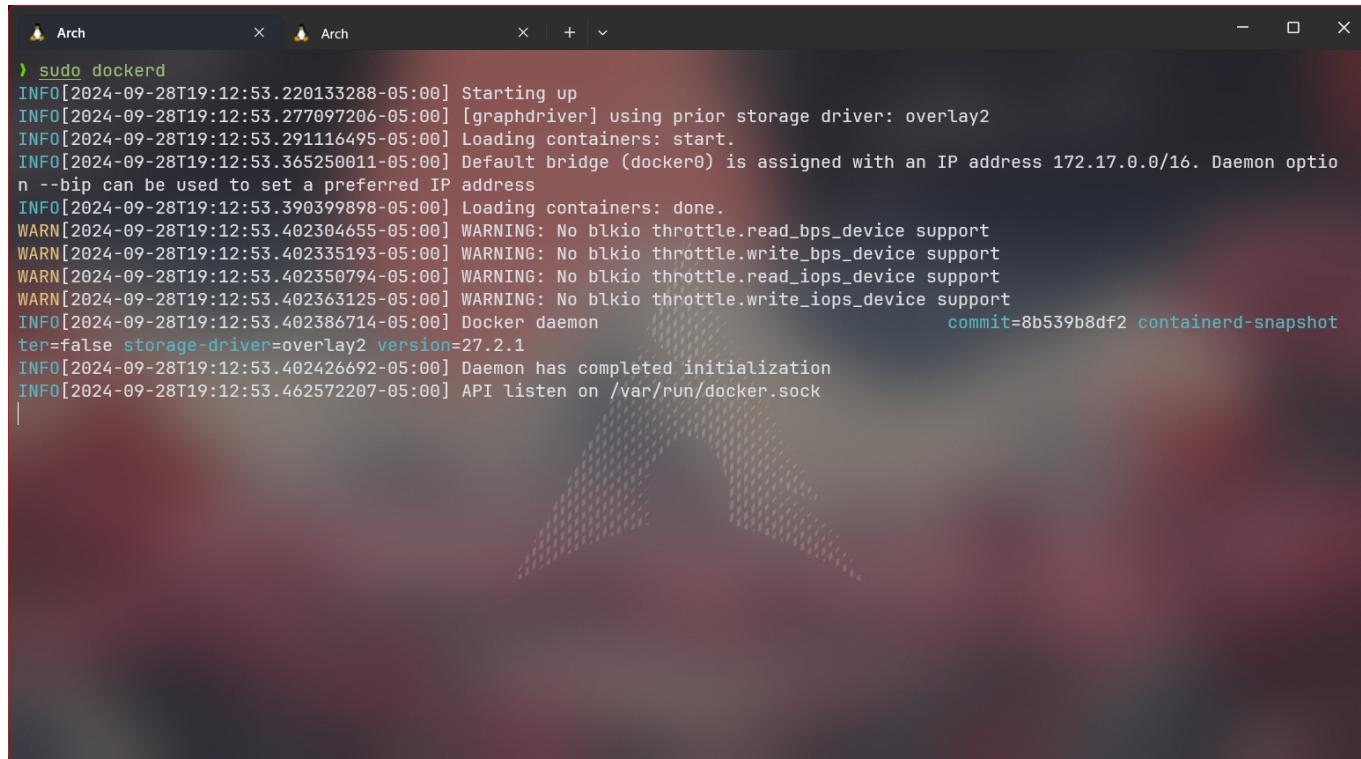
Tuve un error al iniciar el contenedor, por falta de memoria RAM; pero por más que le puse +10GB de RAM no me dejaba iniciar el contenedor.



```
File Edit View Search Terminal Help
#
# There is insufficient memory for the Java Runtime Environment to continue.
# Cannot create worker GC thread. Out of system resources.
# An error report file with more information is saved as:
# //hs_err_pid6.log
devasc@labvm:~$ free -h
              total        used        free      shared  buff/cache   available
Mem:          9Gi       571Mi       8.4Gi       12Mi       1.0Gi       9.1Gi
Swap:        2.0Gi          0B       2.0Gi
devasc@labvm:~$ docker run --rm -u root -p 8080:8080 -v jenkins-data:/var/jenkins_home -v $(which docker):/usr/bin/docker -v /var/run/docker.sock:/var/run/docker.sock -v "$HOME":/home --name jenkins_server jenkins/jenkins:lts
INSTALL WARNING: User: missing rw permissions on JENKINS_HOME: /var/jenkins_home
dio[0.007s][warning][os,thread] Failed to start thread "GC Thread#0" - pthread_create failed (EPERM) for attributes: stacksize: 1024k, guardsize: 4k, detached.
#
# There is insufficient memory for the Java Runtime Environment to continue.
# Cannot create worker GC thread. Out of system resources.
# An error report file with more information is saved as:
# //hs_err_pid6.log
devasc@labvm:~$ 
```

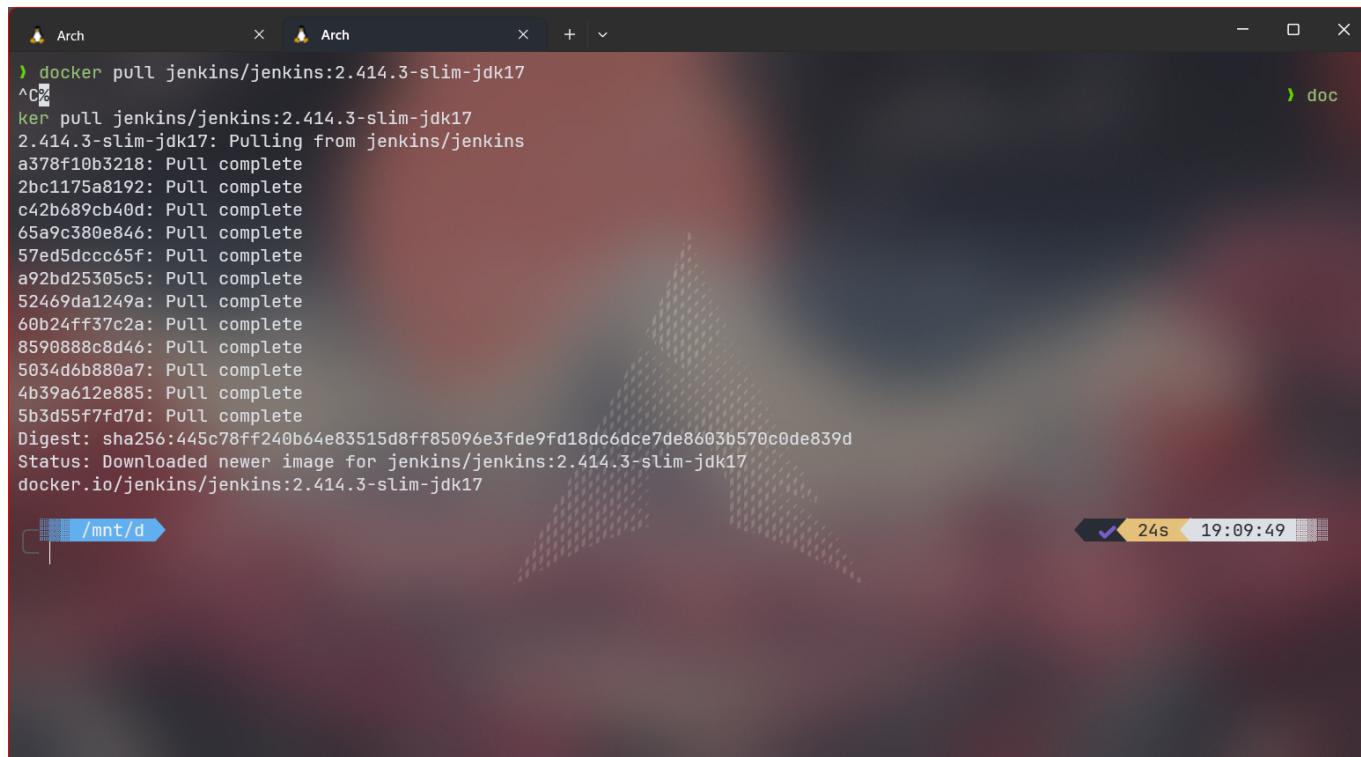
Luego de insistentes errores al seguir con el laboratorio, decidí proseguir usando el Windows Subsystem for Linux (WSL). En particular estoy usando Arch Linux WSL.

Prendo docker daemon



```
> sudo dockerd
INFO[2024-09-28T19:12:53.220133288-05:00] Starting up
INFO[2024-09-28T19:12:53.277097206-05:00] [graphdriver] using prior storage driver: overlay2
INFO[2024-09-28T19:12:53.291116495-05:00] Loading containers: start.
INFO[2024-09-28T19:12:53.365250011-05:00] Default bridge (docker0) is assigned with an IP address 172.17.0.0/16. Daemon option --bip can be used to set a preferred IP address
INFO[2024-09-28T19:12:53.390399898-05:00] Loading containers: done.
WARN[2024-09-28T19:12:53.402304655-05:00] WARNING: No blkio throttle.read_bps_device support
WARN[2024-09-28T19:12:53.402335193-05:00] WARNING: No blkio throttle.write_bps_device support
WARN[2024-09-28T19:12:53.402350794-05:00] WARNING: No blkio throttle.read_iops_device support
WARN[2024-09-28T19:12:53.402363125-05:00] WARNING: No blkio throttle.write_iops_device support
INFO[2024-09-28T19:12:53.402386714-05:00] Docker daemon commit=8b539b8df2 containerd-snapshotter=false storage-driver=overlay2 version=27.2.1
INFO[2024-09-28T19:12:53.402426692-05:00] Daemon has completed initialization
INFO[2024-09-28T19:12:53.462572207-05:00] API listen on /var/run/docker.sock
```

Descargo la imagen de jenkins 2.414.3-slim-jdk17 que es compatible con la clave ssh con la que enlazé mi repositorio remoto:



```
> docker pull jenkins/jenkins:2.414.3-slim-jdk17
^C%
ker pull jenkins/jenkins:2.414.3-slim-jdk17
2.414.3-slim-jdk17: Pulling from jenkins/jenkins
a378f10b3218: Pull complete
2bc1175a8192: Pull complete
c42b689cb40d: Pull complete
65a9c380e846: Pull complete
57ed5dccc65f: Pull complete
a92bd25305c5: Pull complete
52469da1249a: Pull complete
60b24ff37c2a: Pull complete
8590888c8d46: Pull complete
5034d6b880a7: Pull complete
4b39a612e885: Pull complete
5b3d55f7fd7d: Pull complete
Digest: sha256:445c78ff240b64e83515d8ff85096e3fde9fd18dc6dce7de8603b570c0de839d
Status: Downloaded newer image for jenkins/jenkins:2.414.3-slim-jdk17
docker.io/jenkins/jenkins:2.414.3-slim-jdk17
```

Ejecuto el contenedor con el comando:

```
docker run --rm -u root -p 8080:8080 -v jenkins-data:/var/jenkins_home -v $(which docker):/usr/bin/docker -v /var/run/docker.sock:/var/run/docker.sock -v "$HOME":/home --name jenkins_server jenkins/jenkins:2.414.3-slim-jdk17
```

```

❯ docker run --rm -u root -p 8080:8080 -v jenkins-data:/var/jenkins_home -v $(which docker):/usr/bin/docker -v /var/run/docker.sock:/var/run/docker.sock -v "$HOME":/home --name jenkins_server jenkins:2.414.3-slim-jdk17
Running from: /usr/share/jenkins/jenkins.war
webroot: /var/jenkins_home/war
2024-09-29 00:16:28.853+0000 [id=1]     INFO    winstone.Logger#logInternal: Beginning extraction from war file
2024-09-29 00:16:28.925+0000 [id=1]     WARNING o.e.j.s.handler.ContextHandler#setContextPath: Empty contextPath
2024-09-29 00:16:28.985+0000 [id=1]     INFO    org.eclipse.jetty.server.Server#doStart: jetty-10.0.17; built: 2023-10-02T04:04:10.314Z; git: a0f5f05abaa6c3aabb7c3d35f10a6f412ab8b05f; jvm 17.0.8.1+1
2024-09-29 00:16:29.158+0000 [id=1]     INFO    o.e.j.w.StandardDescriptorProcessor#visitServlet: NO JSP Support for /, did not find org.eclipse.jetty.jsp.JettyJspServlet
2024-09-29 00:16:29.196+0000 [id=1]     INFO    o.e.j.s.s.DefaultSessionIdManager#doStart: Session workerName=node0
2024-09-29 00:16:29.571+0000 [id=1]     INFO    hudson.WebAppMain#contextInitialized: Jenkins home directory: /var/jenkins_home found at: EnvVars.masterEnvVars.get("JENKINS_HOME")
2024-09-29 00:16:29.655+0000 [id=1]     INFO    o.e.j.s.handler.ContextHandler#doStart: Started w.@56781d96{Jenkins v2.414.3, /, file:///var/jenkins_home/war/,AVAILABLE}{/var/jenkins_home/war}
2024-09-29 00:16:29.668+0000 [id=1]     INFO    o.e.j.server.AbstractConnector#doStart: Started ServerConnector@1329eff{HTTP/1.1, (http/1.1)}{0.0.0.0:8080}
2024-09-29 00:16:29.679+0000 [id=1]     INFO    org.eclipse.jetty.server.Server#doStart: Started Server@be35cd9{STARTING}[10.0.17,sto=0] @1175ms
2024-09-29 00:16:29.680+0000 [id=27]     INFO    winstone.Logger#logInternal: Winstone Servlet Engine running: controlPort=disabled
2024-09-29 00:16:29.833+0000 [id=34]     INFO    jenkins.InitReactorRunner$1#onAttained: Started initialization
2024-09-29 00:16:29.876+0000 [id=54]     INFO    jenkins.InitReactorRunner$1#onAttained: Listed all plugins
2024-09-29 00:16:30.446+0000 [id=37]     INFO    jenkins.InitReactorRunner$1#onAttained: Prepared all plugins
2024-09-29 00:16:30.450+0000 [id=37]     INFO    jenkins.InitReactorRunner$1#onAttained: Started all plugins
2024-09-29 00:16:30.454+0000 [id=37]     INFO    jenkins.InitReactorRunner$1#onAttained: Augmented all extensions
2024-09-29 00:16:30.643+0000 [id=34]     INFO    jenkins.InitReactorRunner$1#onAttained: System config loaded
2024-09-29 00:16:30.644+0000 [id=43]     INFO    jenkins.InitReactorRunner$1#onAttained: System config adapted
2024-09-29 00:16:30.645+0000 [id=43]     INFO    jenkins.InitReactorRunner$1#onAttained: Loaded all jobs
2024-09-29 00:16:30.646+0000 [id=32]     INFO    jenkins.InitReactorRunner$1#onAttained: Configuration for all jobs updated

```

Ahora sí me salió el mensaje con la contraseña:

```

This may also be found at: /var/jenkins_home/secrets/initialAdminPassword
*****
2024-09-29 00:16:39.372+0000 [id=44]     INFO    jenkins.InitReactorRunner$1#onAttained: Completed initialization
2024-09-29 00:16:39.432+0000 [id=26]     INFO    hudson.lifecycle.Lifecycle#onReady: Jenkins is fully up and running

```

1c50ae0907ff46828edce82287cd93e7

También me cierro que esa es la contraseña al entrar al contenedor:

```

❯ docker exec -it jenkins_servers /bin/bash
Error response from daemon: No such container: jenkins_servers
❯ docker exec -it jenkins_server /bin/bash
root@a0aca5013037:/# cat /var/jenkins_home/secrets/initialAdminPassword
1c50ae0907ff46828edce82287cd93e7
root@a0aca5013037:/# exit
exit

```

Configuramos jenkins:

Entramos con la contraseña que obtuvimos en localhost:8080

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

```
/var/jenkins_home/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

Instalamos los plugins sugeridos.

Skipeamos crear un usuario.

Dejé la config como estaba.

Getting Started

Jenkins is ready!

You have skipped the **setup of an admin user**.

To log in, use the username: "admin" and the administrator password you used to access the setup wizard.

Your Jenkins setup is complete.

[Start using Jenkins](#)

Listo! Ya estamos en el panel de jenkins.

The screenshot shows the Jenkins dashboard interface. On the left, there's a sidebar with various icons and links: 'Dashboard' (selected), 'New Item', 'People' (with 3657 items), 'Build History', 'Manage Jenkins', 'My Views', 'Build Queue' (empty), 'Build Executor Status' (1 Idle, 2 Idle), and 'Add description'. The main content area features a large 'Welcome to Jenkins!' heading, a note about starting a project, a 'Create a job' button, and a 'Set up a distributed build' section.

Dashboard

- + New Item
- People (3657)
- Build History
- Manage Jenkins
- My Views

Build Queue

No builds in the queue.

Build Executor Status

1 Idle
2 Idle

Add description

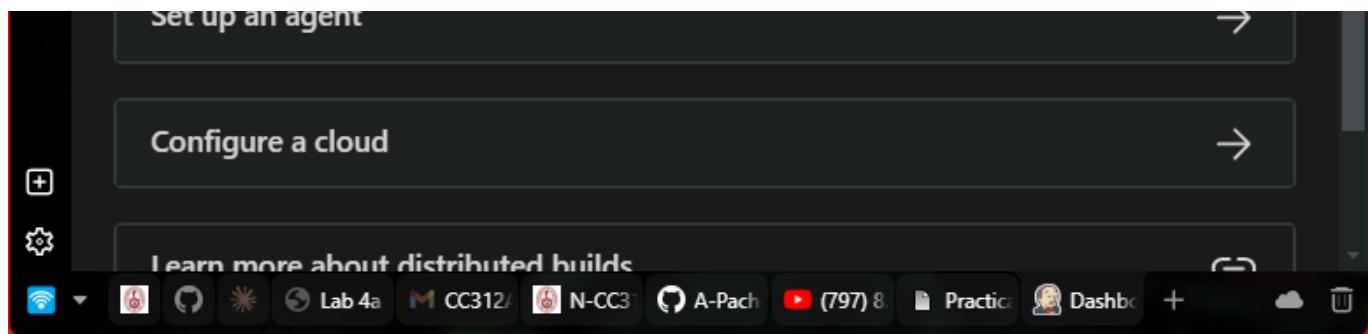
Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job →

Set up a distributed build



Usamos jenkins para construir nuestra app:

Creamos un job BuildAppJob. Se eligió un Freestyle project.

A screenshot of the Jenkins dashboard. At the top, there is a logo of a man holding a coffee cup, followed by the word 'Jenkins'. To the right are icons for notifications, security, user profile, and a refresh button. Below the header, a 'Dashboard' link is visible. The main area features a large input field with the placeholder 'Enter an item name'. Inside this field, the text 'BuildAppJob' is entered and highlighted with a blue border. To the right of the input field is a small icon containing a lock and a person. Below the input field, the text '» Required field' is displayed. The entire interface has a dark theme.

Se establecieron el link del repositorio remoto y las credenciales de github en la configuración del job.

A screenshot of the Jenkins job configuration page for 'BuildAppJob'. The top navigation bar shows 'Dashboard > BuildAppJob > Configuration'. The main section is titled 'General'. On the right, there is a 'Enabled' switch with a checked status. Below the title, there is a 'Description' field containing the text 'My first Jenkins job'. Underneath the description is a 'Plain text' link and a 'Preview' link. There are also two configuration options: 'Discard old builds' (unchecked) and 'GitHub project' (checked). The 'GitHub project' option has a 'Project url' field below it. The entire page has a light gray background with dark text and form elements.

<https://github.com/A-PachecoT/sample-app>

Advanced ▾

This project is parameterized ?

Throttle builds ?

Execute concurrent builds if necessary ?

Advanced ▾

Source Code Management

None

Git ?

Repositories ?

Repository URL ?

<https://github.com/A-PachecoT/sample-app.git>



Credentials ?

A-PachecoT/*****



Add ▾

Advanced ▾

Se agregó un Build Step de tipo Execute shell con el comando de construcción del contenedor:

Build Steps

The screenshot shows the 'Execute shell' build step configuration in Jenkins. It includes a command input field containing 'bash ./sample-app.sh', an 'Advanced' dropdown, and an 'Add build step' button.

```
bash ./sample-app.sh
```

Advanced ▾

Add build step ▾

Se guardó la configuración.

Usamos jenkins para testear la construcción de nuestra app:

Le di a Build Now. Salida de consola:

Console Output

Progress:  X

```
Started by user admin
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/BuildAppJob
The recommended git tool is: NONE
using credential 1dc7f1d7-da96-4caa-ab09-c19131bd89e3
Cloning the remote Git repository
Cloning repository https://github.com/A-PachecoT/sample-app.git
> git init /var/jenkins_home/workspace/BuildAppJob # timeout=10
Fetching upstream changes from https://github.com/A-PachecoT/sample-app.git
> git --version # timeout=10
> git --version # 'git version 2.39.2'
using GIT_ASKPASS to set credentials
> git fetch --tags --force --progress -- https://github.com/A-PachecoT/sample-app.git
+refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/A-PachecoT/sample-app.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 9f5a281a2e5c0780bf4bbfb65ff11176fdbba3cb (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 9f5a281a2e5c0780bf4bbfb65ff11176fdbba3cb # timeout=10
Commit message: "chore: changed port from 8080 to 5050"
First time build. Skipping changelog.
[BuildAppJob] $ /bin/sh -xe /tmp/jenkins5486945177744948488.sh
+ bash ./sample-app.sh
mkdir: cannot create directory 'tempdir': File exists
mkdir: cannot create directory 'tempdir/templates': File exists
mkdir: cannot create directory 'tempdir/static': File exists
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
    Install the buildx component to build images with BuildKit:
    https://docs.docker.com/go/buildx/

Sending build context to Docker daemon 6.656kB

Step 1/21 : FROM python
latest: Pulling from library/python
cdd62bf39133: Pulling fs layer
a47cff7f31e9: Pulling fs layer
a173f2aee8e9: Pulling fs layer
01272fe8adba: Pulling fs layer
ca852fa4243b: Pulling fs layer
8e0f50619958: Pulling fs layer
fc5ef6d7bb7c: Pulling fs layer
ca852fa4243b: Waiting
01272fe8adba: Waiting
fc5ef6d7bb7c: Waiting
```

```

--> bdb9fe7ff969
Successfully built bdb9fe7ff969
Successfully tagged sampleapp:latest
3986465cee017e53f37a6a48edb6c48d21b0bc6c268f4805de799f6545fc6c73
CONTAINER ID   IMAGE          COMMAND                  CREATED             STATUS              NAMES
STATUS          PORTS          NAMES
3986465cee01   sampleapp      "/bin/sh -c 'python ..."   Less than a second ago
Up Less than a second   0.0.0.0:5050->5050/tcp, :::5050->5050/tcp   samplerunning
a0aca5013037   jenkins/jenkins:2.414.3-slim-jdk17  "/usr/bin/tini -- /u..."  31 minutes ago
Up 31 minutes    0.0.0.0:8080->8080/tcp, :::8080->8080/tcp, 50000/tcp  jenkins_server
Finished: SUCCESS

```

Además observamos que se creó el contenedor `sampleapp` con el build. Se ve en la consola y también en mi WSL:

```

$ docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED             STATUS              PORTS
NAMES
3986465cee01   sampleapp      "/bin/sh -c 'python ..."   46 seconds ago   Up 46 seconds   0.0.0.0:5050->5
050/tcp, :::5050->5050/tcp   samplerunning
a0aca5013037   jenkins/jenkins:2.414.3-slim-jdk17  "/usr/bin/tini -- /u..."  32 minutes ago   Up 32 minutes   0.0.0.0:8080->8
080/tcp, :::8080->8080/tcp, 50000/tcp  jenkins_server

```



Y se puede acceder a la app:



Usamos jenkins para testear nuestra app

Se para y remueve el contenedor:

```

$ docker stop samplerunning
samplerunning
$ docker rm samplerunning
samplerunning

```

Creamos nuevo job para testing TestAppJob. Como Freestyle project.

Enter an item name

TestAppJob
» *Required field*

 **Freestyle project**
This is the central feature of Jenkins. Jenkins will build your project, combining system, and this can be even used for something other than software build.

 **Pipeline**

Configuramos el job:

Configure General

Enabled

General

Description
My first Jenkins test

Plain text [Preview](#)

Discard old builds ?

GitHub project

This project is parameterized ?

Throttle builds ?

Execute concurrent builds if necessary ?

Advanced ▾

Source Code Management

None

Git ?

Build Triggers

Trigger builds remotely (e.g., from scripts) ?

Build after other projects are built ?

Projects to watch
BuildAppJob

Trigger only if build is stable

Trigger even if the build is unstable

Trigger even if the build fails

Always trigger, even if the build is aborted

Build Steps

Execute shell

Command

See [the list of available environment variables](#)

```
if curl http://172.17.0.1:5050/ | grep "You are calling me from 172.17.0.1"; then
    exit 0
else
    exit 1
fi
```

Advanced ▾

Add build step ▾

Post-build Actions

Add post-build action ▾

Save Apply

Corremos el BuildAppJob:

S	W	Name ↓	Last Success	Last Failure	Last Duration	Schedule a Build for BuildAppJob
✓	☀️	BuildAppJob	12 min #1	N/A	53 sec	
...	☀️	TestAppJob	N/A	N/A	N/A	

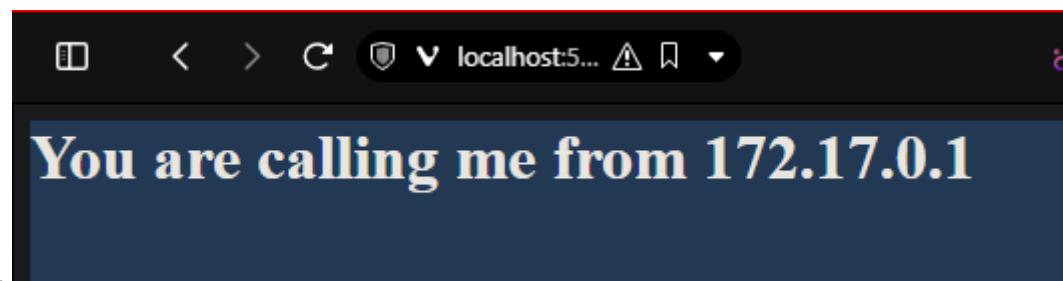
Como vemos se ejecutaron ambos jobs con éxito:

S	W	Name ↓	Last Success	Last Failure	Last Duration	
✓	☀️	BuildAppJob	39 sec #2	N/A	1.8 sec	
✓	☀️	TestAppJob	31 sec #1	N/A	28 ms	

Log del testing:

✓ Console Output

```
Started by user admin
Running as SYSTEM
Building in workspace /var/jenkins_home/workspace/TestAppJob
[TestAppJob] $ /bin/sh -xe /tmp/jenkins1881255256384345467.sh
+ curl http://172.17.0.1:5050/
+ grep You are calling me from 172.17.0.1
% Total    % Received % Xferd  Average Speed   Time     Time     Time  Current
                                         Dload  Upload   Total   Spent    Left  Speed
0      0      0      0      0      0      0 --::-- --::-- --::-- 0
100  177  100  177      0      0  96721      0 --::-- --::-- --::-- 172k
<h1>You are calling me from 172.17.0.1</h1>
+ exit 0
Finished: SUCCESS
```



Funcionando la app!

Creamos pipeline en jenkins:

Creamos el item SamplePipeline de tipo Pipeline.

Enter an item name

SamplePipeline
» Required field

 **Freestyle project**
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

 **Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

 **Multi-configuration project**
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

 **Folder**
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

 **Multibranch Pipeline**
Creates a set of Pipeline projects according to detected branches in one SCM repository.

 **Organization Folder**
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

 **Copy from**
Type to autocomplete

OK

Configuramos la pipeline:

Agregamos Script:

Pipeline

Definition

Pipeline script



Script ?

```
1 ▾ node {  
2 ▾   stage('Preparation') {  
3 ▾     catchError(buildResult: 'SUCCESS') {  
4       sh 'docker stop samplerunning'  
5       sh 'docker rm samplerunning'  
6     }  
7   }  
8 ▾   stage('Build') {  
9     build 'BuildAppJob'  
10  }  
11 ▾   stage('Results') {  
12     build 'TestAppJob'  
13   }  
14 }  
15 |
```

try sample Pipeline... ▾

Use Groovy Sandbox ?

Pipeline Syntax

Save

Apply

Ejecutamos el pipeline:

Status

Pipeline SamplePipeline

</> Changes Build scheduled

▷ Build Now

Add description

Configure

Delete Pipeline

Full Stage View

Rename

Pipeline Syntax

Build History trend ▾

Average stage times:

Preparation 499ms

#1 Sep 28 20:10 No Changes

499ms

No builds

Atom feed for all Atom feed for failures

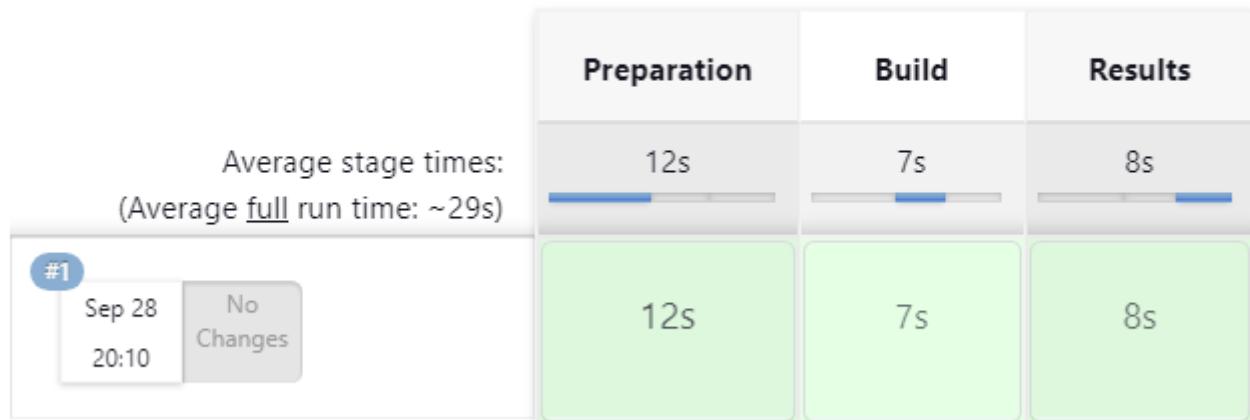
Permalinks

This screenshot shows the Jenkins pipeline dashboard for 'SamplePipeline'. It includes a sidebar with various project management options like 'Status', 'Changes', 'Build Now', 'Configure', etc. The main area displays the 'Stage View' for the latest build (#1), which completed on Sep 28 at 20:10 with 'No Changes'. The preparation stage took 499ms. Below this, there's a summary of average stage times: Preparation (499ms), Build (499ms), and Results (499ms). To the left, it shows 'No builds' and links to atom feeds. On the right, there's a 'Permalinks' section with navigation arrows.

Resultados:

Stage View:

Stage View



Logs:



Console Output

```
Started by user admin
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/jenkins_home/workspace/SamplePipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Preparation)
[Pipeline] catchError
[Pipeline] {
[Pipeline] sh
+ docker stop samplerunning
samplerunning
[Pipeline] sh
+ docker rm samplerunning
samplerunning
[Pipeline] }
[Pipeline] // catchError
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Build)
[Pipeline] build (Building BuildAppJob)
Scheduling project: BuildAppJob
Starting building: BuildAppJob #3
Build BuildAppJob #3 completed: SUCCESS
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Results)
[Pipeline] build (Building TestAppJob)
Scheduling project: TestAppJob
Starting building: TestAppJob #4
Build TestAppJob #4 completed: SUCCESS
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Además se comprobó que la app funcionaba.

Cuestionario

1. ¿Qué es una pipeline de Jenkins? ¿Cuáles son los tipos de pipelines de Jenkins?

Una pipeline de Jenkins es una serie automatizada de pasos para CI/CD. Los tipos principales son declarativas (estructura predefinida, sintaxis simple) y scripted (flexibles, usan código Groovy).

Las pipelines permiten definir el ciclo de vida completo de una aplicación, desde la obtención del código fuente hasta el despliegue.

- Las declarativas usan bloques predefinidos como 'pipeline', 'stage', y 'steps', facilitando la creación de flujos de trabajo estandarizados.
- Las scripted ofrecen mayor control y personalización, permitiendo lógica compleja mediante Groovy. Ambos tipos soportan control de versiones, lo que facilita la gestión y evolución de los procesos de CI/CD.

2. ¿Cómo se pueden activar (triggered)/ detener / controlar los trabajos de Jenkins mediante programación?

Los trabajos de Jenkins se pueden controlar programáticamente de varias formas:

- API REST: Permite interactuar con Jenkins mediante solicitudes HTTP.
 - Iniciar trabajos: POST /job/{nombre_trabajo}/build
 - Detener trabajos: POST /job/{nombre_trabajo}/stop
 - Obtener estado: GET /job/{nombre_trabajo}/lastBuild/api/json
- CLI de Jenkins: Ofrece comandos para control desde la línea de comandos.
 - java -jar jenkins-cli.jar -s http://localhost:8080/ build {nombre_trabajo}
 - java -jar jenkins-cli.jar -s http://localhost:8080/ stop-builds {nombre_trabajo}
- Bibliotecas de lenguajes: Wrappers que simplifican la interacción con Jenkins.
 - Python: python-jenkins
 - Java: jenkins-client-api
 - Node.js: jenkins-api

3. ¿Cómo se logra la integración continua con Jenkins?

Jenkins logra la integración continua mediante los siguientes pasos:

1. Conexión con control de versiones:
 - Se integra con sistemas como Git, SVN para detectar cambios en el código.
 - Configura webhooks o polling para iniciar automáticamente el proceso CI.
2. Compilación automática:
 - Inicia la compilación del proyecto cuando se detectan cambios.
 - Utiliza herramientas como Maven, Gradle, o npm según el tipo de proyecto.

3. Ejecución de pruebas:

- Ejecuta conjuntos de pruebas unitarias, de integración y funcionales.
- Genera informes de cobertura y calidad del código.

4. Análisis estático de código:

- Integra herramientas como SonarQube para evaluar la calidad del código.

5. Notificaciones:

- Informa a los desarrolladores sobre el estado de las compilaciones y pruebas.
- Utiliza correos electrónicos, Slack, o plugins de notificación.

6. Despliegue automático:

- Despliega la aplicación en entornos de prueba o producción si todas las etapas anteriores son exitosas.

Este proceso permite detectar y corregir problemas rápidamente, mejorando la calidad del software y acelerando el desarrollo.

4. Análisis del texto:

Lo que entendí del texto es que la aplicación de CI/CD a las redes es un proceso complejo pero necesario para mejorar la gestión y confiabilidad de las infraestructuras de red modernas. El autor explica que para implementar CI/CD en redes, primero necesitamos crear un modelo preciso de la infraestructura existente, lo cual no es tarea fácil debido a la complejidad de las redes actuales y el uso de tecnologías de virtualización como VRF y EVPN.

Me pareció interesante el concepto de "gemelo digital de red" que se menciona. Entiendo que es una réplica virtual de la red real que nos permite probar cambios de configuración antes de aplicarlos en producción. Esto me recuerda a cómo los desarrolladores de software usan entornos de prueba, pero aplicado a redes.

El texto enfatiza que el proceso de CI/CD para redes debe ser similar al usado en desarrollo de software: desarrollar, probar en un entorno controlado, y si todo está bien, implementar en producción. Sin embargo, reconoce que muchas organizaciones querrán verificar manualmente los cambios antes de aplicarlos, lo cual me parece prudente.

Lo que más me llamó la atención fue el enfoque gradual que sugiere para adoptar CI/CD en redes. Propone comenzar con tareas simples y de bajo impacto, e ir escalando poco a poco. Esto me parece muy sensato, ya que permite a los equipos adaptarse y aprender del proceso sin riesgos mayores.

En resumen, entiendo que aunque implementar CI/CD en redes requiere un cambio significativo en la forma de trabajar y una inversión inicial en herramientas y procesos, los beneficios a largo plazo en términos de reducción de errores y mayor eficiencia hacen que valga la pena el esfuerzo. El objetivo final es llevar las mejores prácticas del desarrollo de software al mundo de las redes, lo cual tiene sentido en un entorno cada vez más dependiente de la tecnología y la automatización.

5. Explica qué es virtual routing and forwarding (VRF) y Ethernet VPN (EVPN).

Ambas son tecnologías de virtualización de red que permiten la separación lógica de recursos de red:

VRF (Virtual Routing and Forwarding):

- Es una tecnología que permite múltiples instancias de una tabla de enrutamiento coexistir en el mismo router simultáneamente.
- Cada instancia de VRF tiene su propia tabla de enrutamiento independiente, proporcionando aislamiento de tráfico y permitiendo el uso de espacios de direcciones superpuestos.
- Principales características:
- Segmentación de red: Separa lógicamente diferentes redes en un mismo dispositivo físico.
- Reutilización de direcciones IP: Permite usar los mismos rangos de IP en diferentes VRFs sin conflictos.
- Mejora la seguridad: Aísla el tráfico entre diferentes VRFs.

EVPN (Ethernet VPN):

- Es una solución de capa 2 y capa 3 para redes de centros de datos y campus que utiliza BGP para distribuir información de direcciones MAC e IP.
- Proporciona una forma eficiente de interconectar sitios a través de una red IP/MPLS.
- Características principales:
- Soporte multi-tenancy: Permite a múltiples clientes compartir la misma infraestructura física.
- Control plane basado en BGP: Utiliza BGP para el intercambio de información de enrutamiento y reenvío.
- Reducción de inundación (flooding): Mejora la eficiencia al limitar la propagación innecesaria de tráfico broadcast y multicast.
- Movilidad de MAC: Facilita la movilidad de máquinas virtuales entre diferentes sitios.

Ambas tecnologías son fundamentales en la creación de redes virtuales y en la implementación de soluciones de nube y centros de datos modernos, proporcionando flexibilidad, escalabilidad y seguridad mejoradas.