THE UNIVERSITY OF DODOMA



COLLEGE OF INFORMATICS AND VIRTUAL EDUCATION(CIVE)
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING(CSE)
COURSE NAME: SOFTWARE DEFINED NETWORKS: CG 311
LAB 2 ASSIGNEMENT

SN	NAMES	REGISTRATION NUMBER	PROGRAMME
2.	AGRIPINA EMMANUEL WILLIAM	T23-03-16504	BSC. CNISE
3.	EVES VITUS SYLVESTER	T23-03-11142	BSC. CNISE

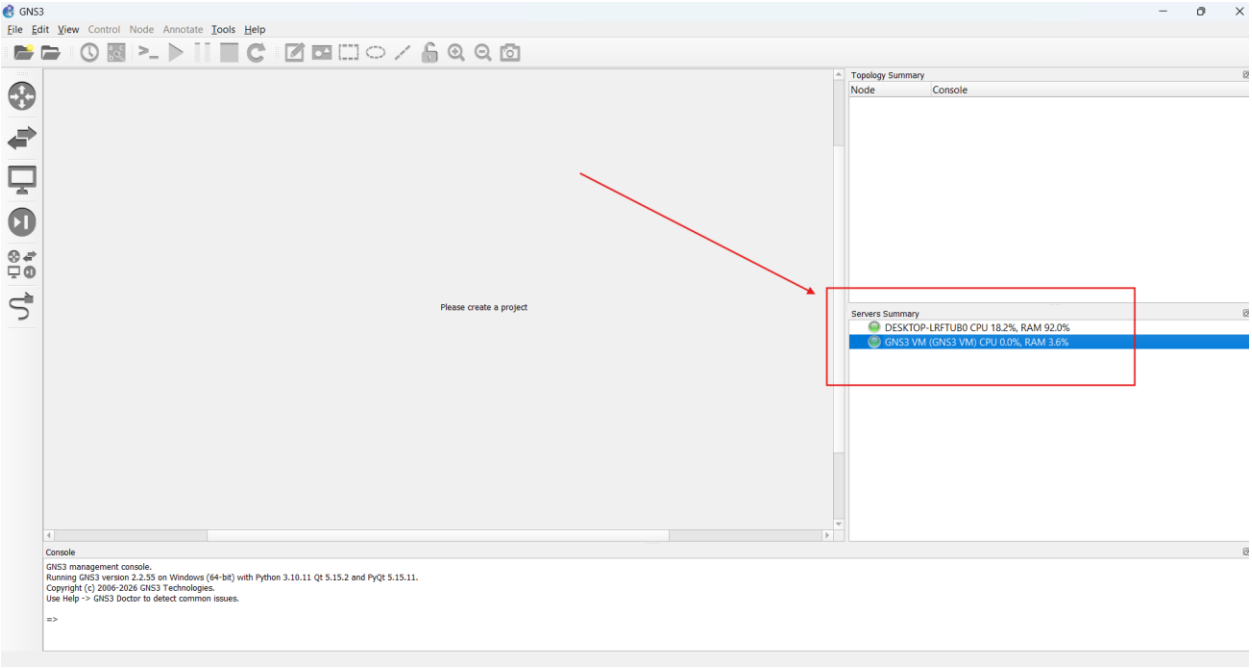
Lab 2: Deploying Open vSwitch (OVS) in GNS3

Objective: To deploy a programmable data plane using Open vSwitch within GNS3, demonstrating its role in an NFV infrastructure and comparing it with a classical Ethernet switch.

Step 2: Technical Background (Introduction)

Perform:

Open GNS3 and ensure GNS3 VM is running.

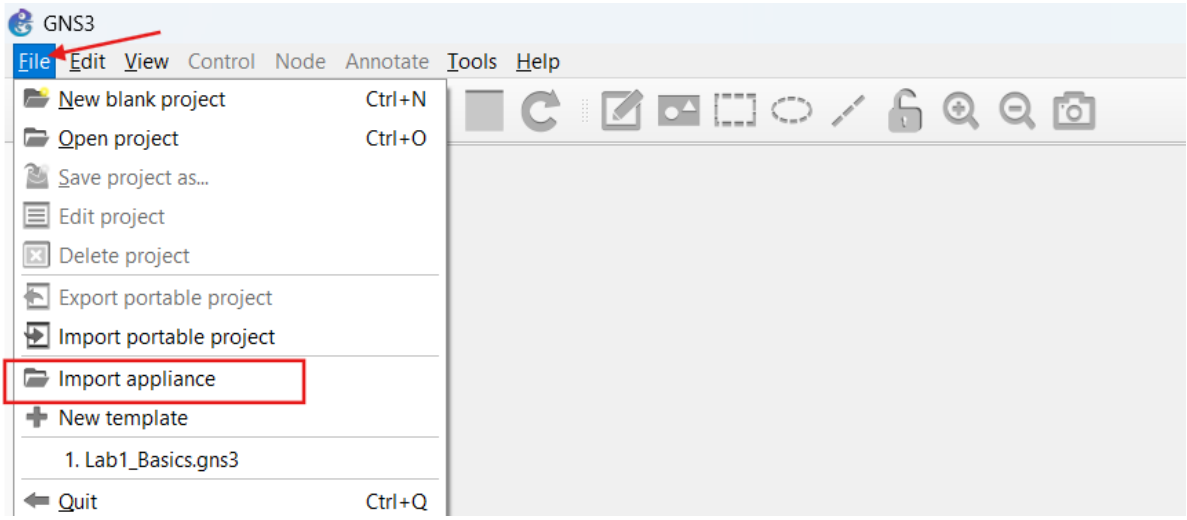


Open vSwitch (OVS) is an open-source multilayer virtual switch used in SDN/NFV environments. Unlike classical hardware switches, OVS allows programmable control of the data plane, supporting automation, network virtualization, and integration with SDN controllers. In the ETSI NFV framework, OVS operates under the Virtual Infrastructure Manager (VIM) layer.

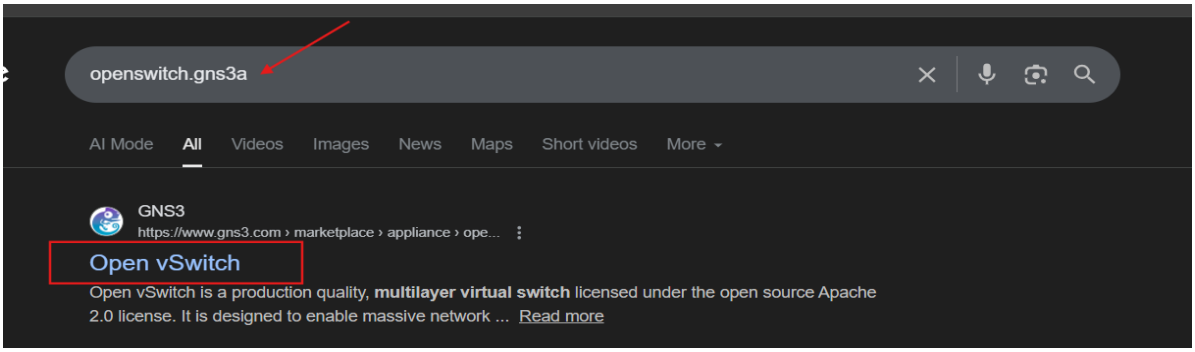
Step 3: Importing the OVS Appliance

Perform:

1. In GNS3, go to:
   File → Import appliance.



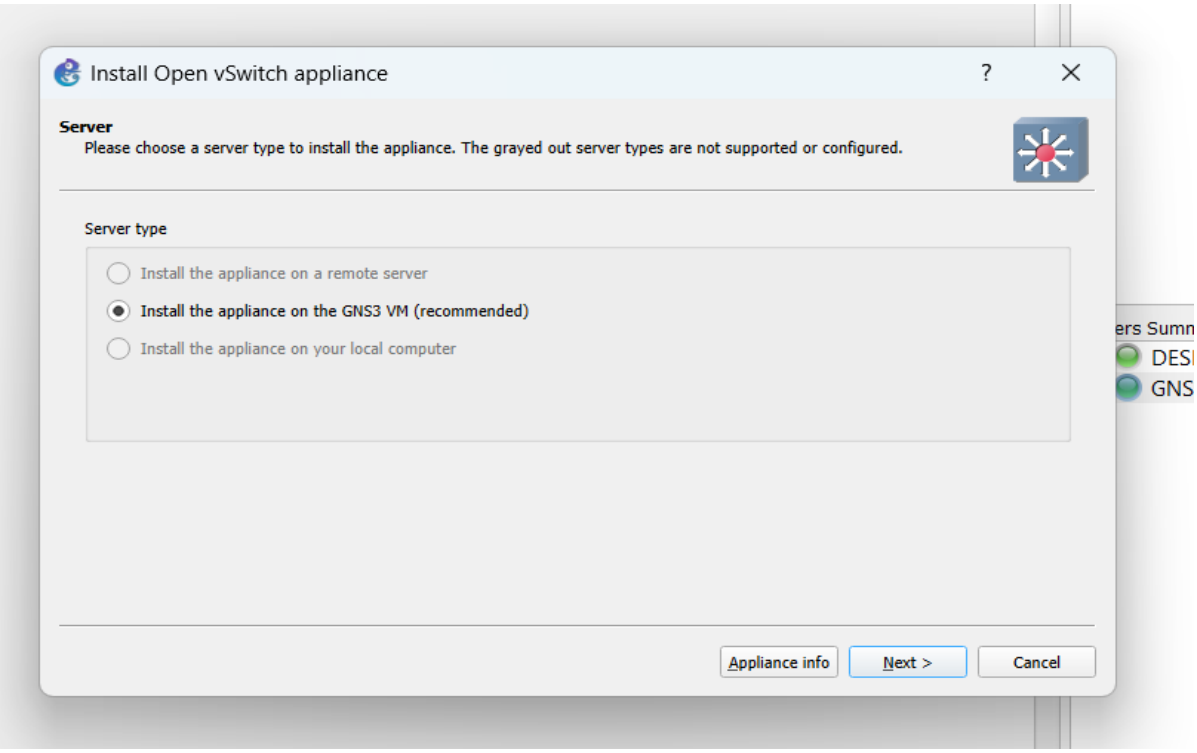
2. Browse to the openswitch.gns3a file (download from GNS3 Marketplace if not available locally).



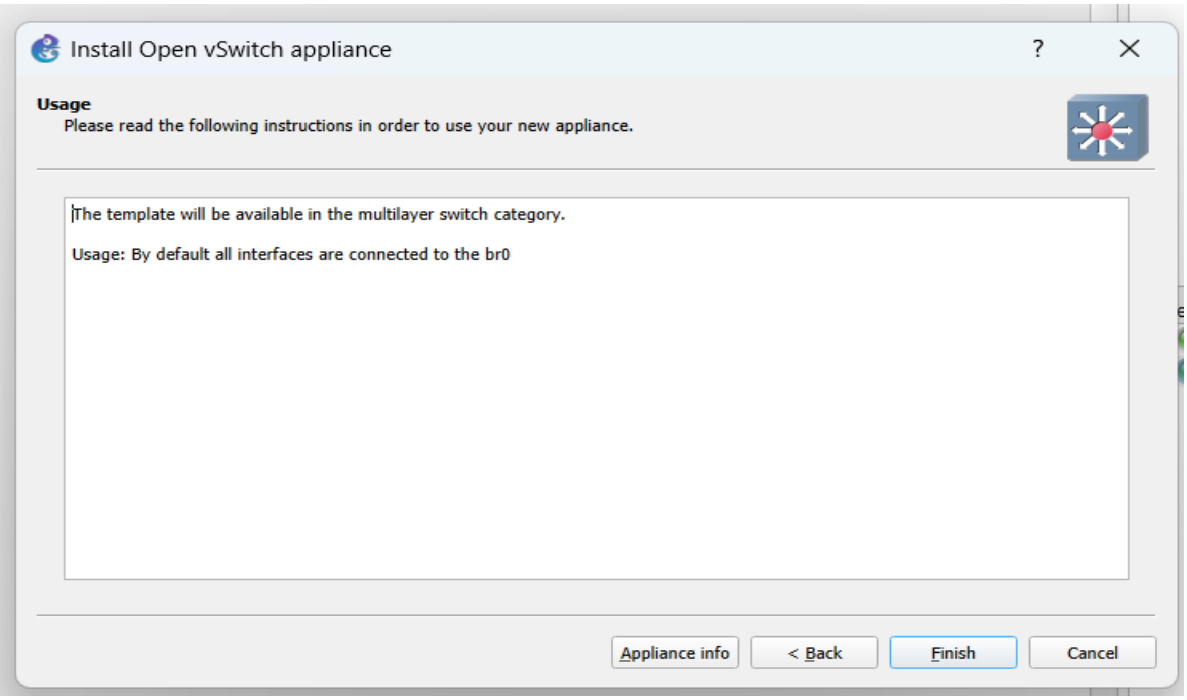
If it does not see. Download via this link. [open-vswitch link]



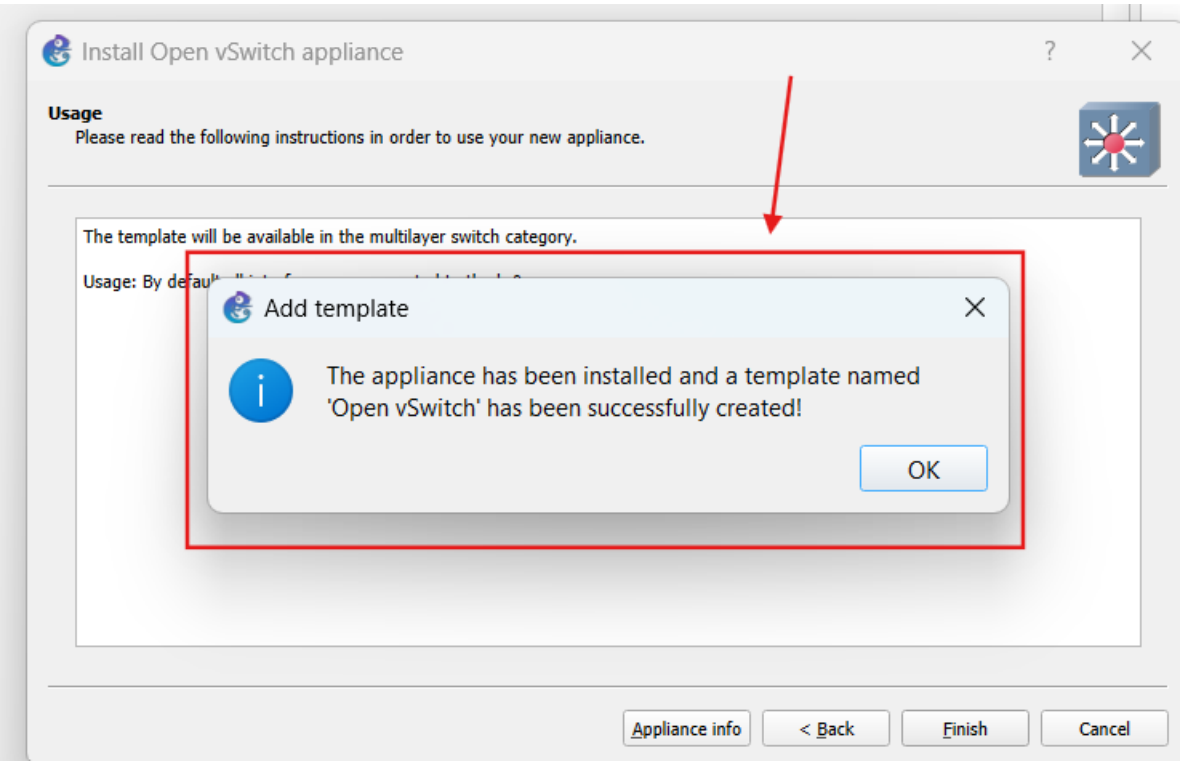
Install following the first step 1.



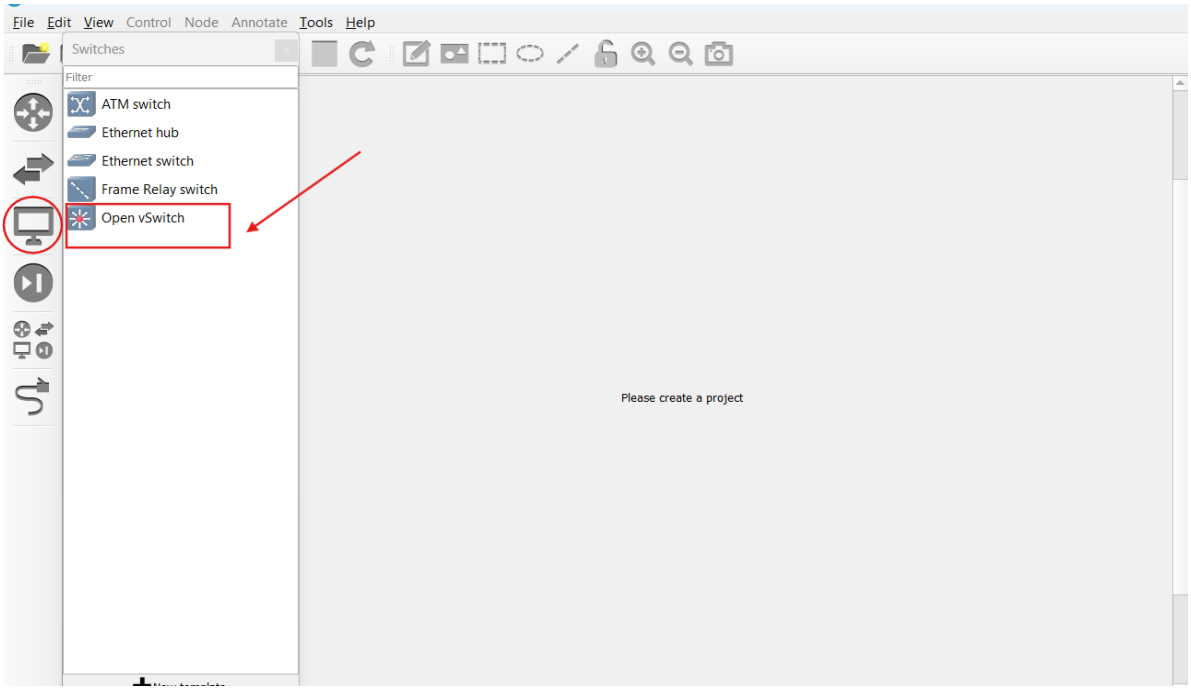
3. Follow prompts to install it on your GNS3 VM. Click next to continue.



Click the “finish” option so as to completed. Then after click “ok”



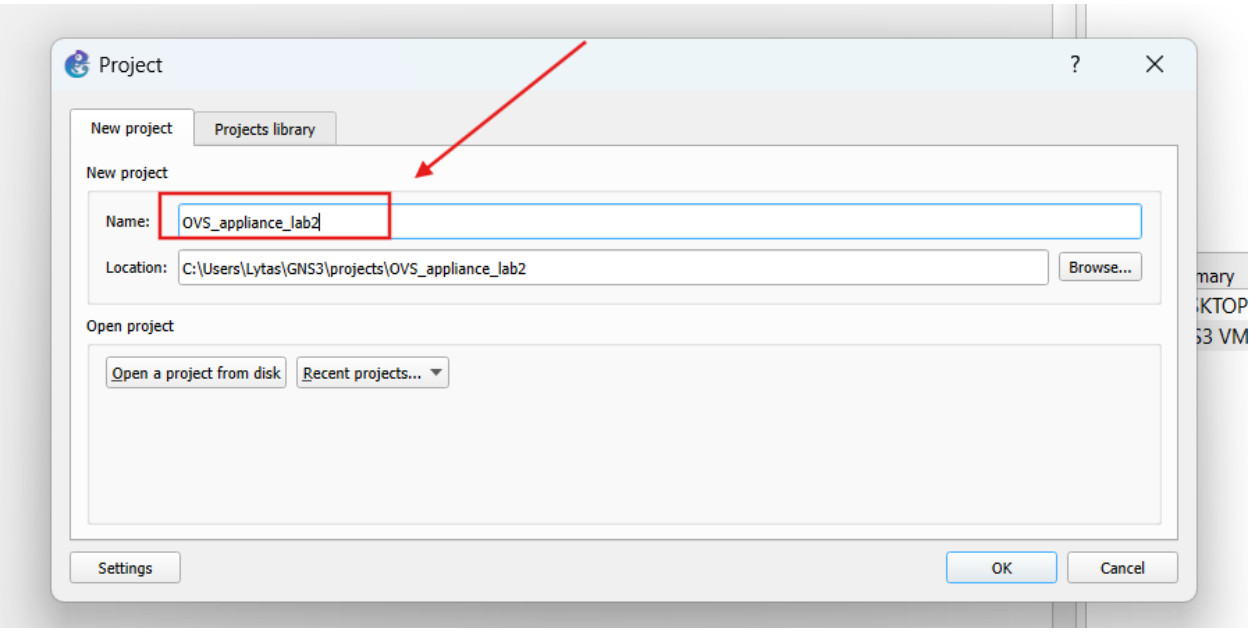
Go to switch then select “open vSwitch” to continue.



The OVS appliance was imported into GNS3 via the Import appliance function. The appliance was installed on the GNS3 VM, which acts as the NFV Infrastructure (NFVI).

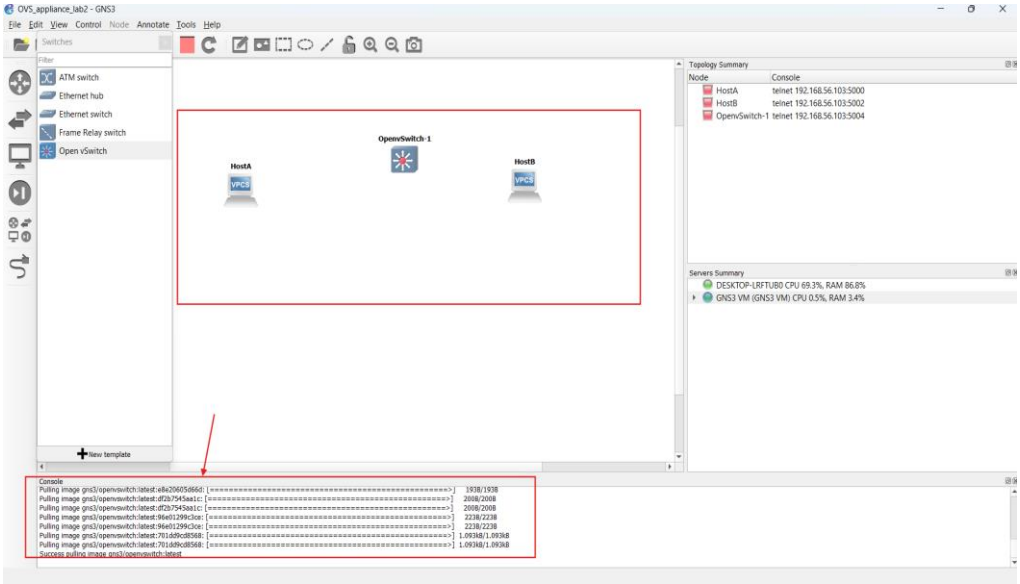
Step 4: Building the Topology

First create a project. To me its called “0VS_appliance_lab2”



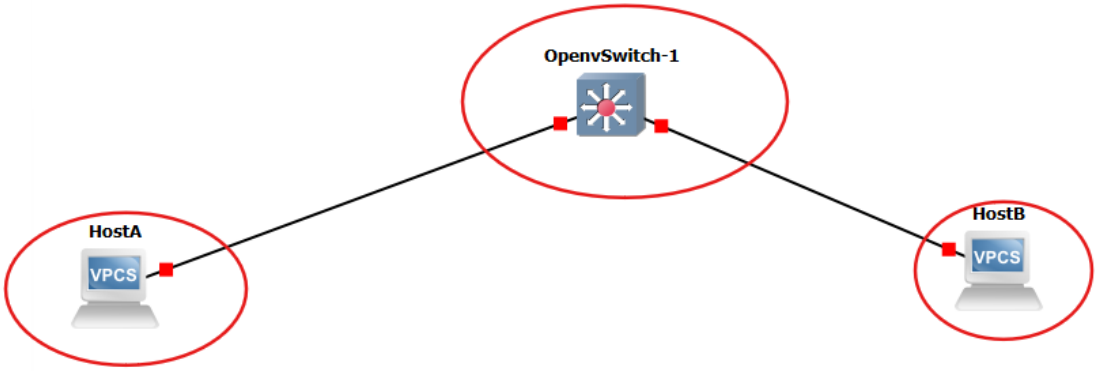
Then Perform:

1. From the devices panel, drag:
    1. 2× VPCS (rename to Host A and Host B)
    2. 1× OVS appliance

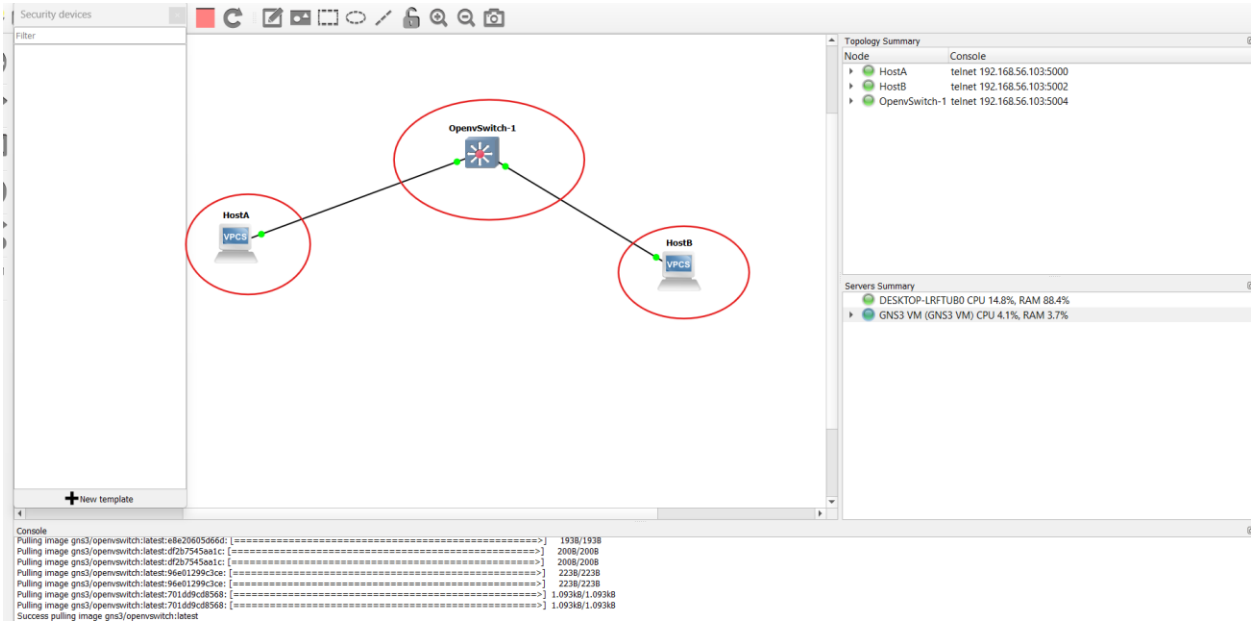


At the time you adding the ovs , it will upgrade to the latest version.

2. Connect:
    1. Host A (Eth0) → OVS Port 1
    2. Host B (Eth0) → OVS Port 2



3. Power on all devices.

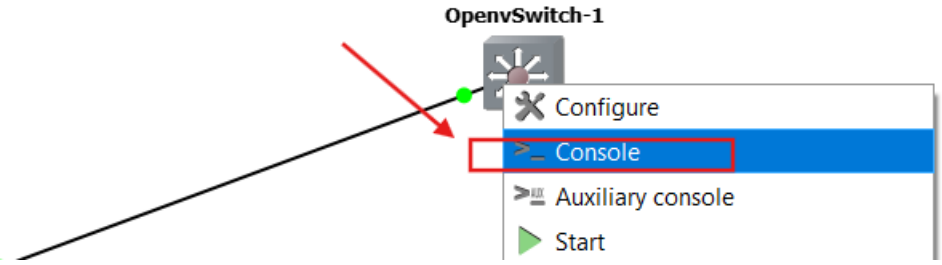


A simple two-host topology was created. Two VPCS nodes (Host A and Host B) were connected to the Open vSwitch appliance via Ethernet links. All nodes were powered on before configuration.

Step 5: Configuring OVS via Console

Perform:

1. Right-click the OVS appliance → Console.



2. Enter the following commands:

Create a Bridge: ovs-vsctl add-br br0

Add Ports to the Bridge: ovs-vsctl add-port br0 eth0

: ovs-vsctl add-port br0 eth1

Verify Configuration: ovs-vsctl show

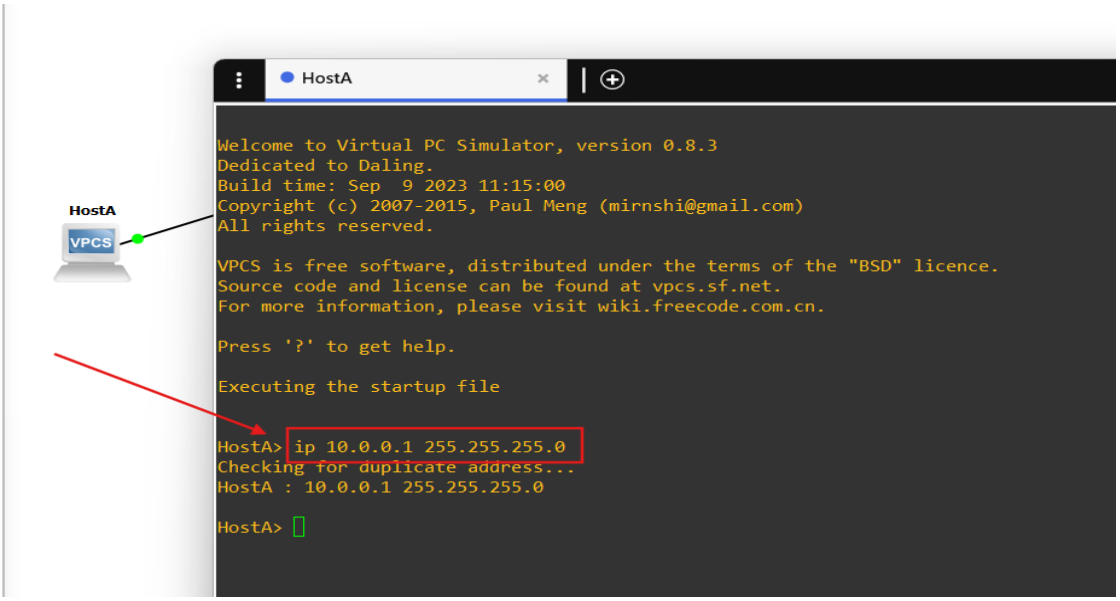
```
2924b
2026-01-27T19:25:08Z|00066|connmgr|INFO|br1: added service controller "punix:/va
r/run/openvswitch/br1.mgmt"
OpenvSwitch-1:/$ ovs-vsctl add-br br0
ovs-vsctl: cannot create a bridge named br0 because a bridge named br0 already exists
OpenvSwitch-1:/$ ovs-vsctl add-port br0 eth0
ovs-vsctl: cannot create a port named eth0 because a port named eth0 already exists on bridge br0
OpenvSwitch-1:/$ ovs-vsctl add-port br0 eth1
ovs-vsctl: cannot create a port named eth1 because a port named eth1 already exists on bridge br0
OpenvSwitch-1:/$ ovs-vsctl show
d7eed97d-552e-4c56-b36e-4730e132a73f
    Bridge br3
        datapath_type: netdev
        Port br3
            Interface br3
                type: internal
```

Upon accessing the OVS console, it was observed that the appliance was pre-configured with multiple bridges (br0, br1, br2, br3). The bridge br0 already contained ports eth0 through eth15. Since eth0 and eth1 were already part of br0, no additional configuration was required for basic switching functionality.

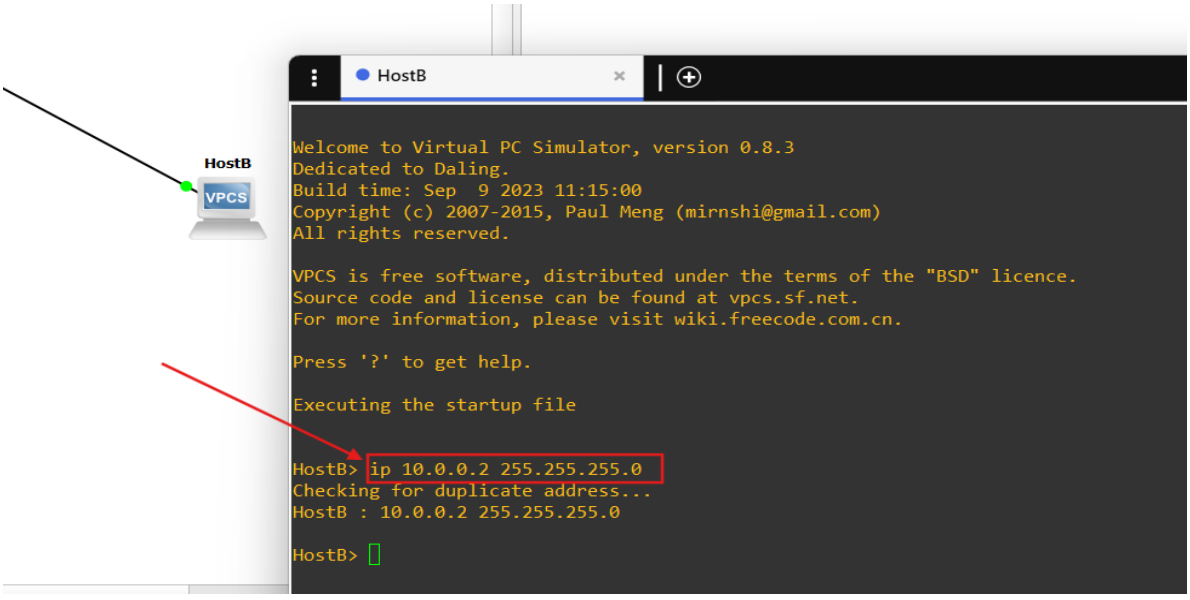
Step 6: Assigning IP Addresses

Perform:

1. Open console for Host A: enter this command: ip 10.0.0.1 255.255.255.0



2. Open console for Host B: ip 10.0.0.2 255.255.255.0



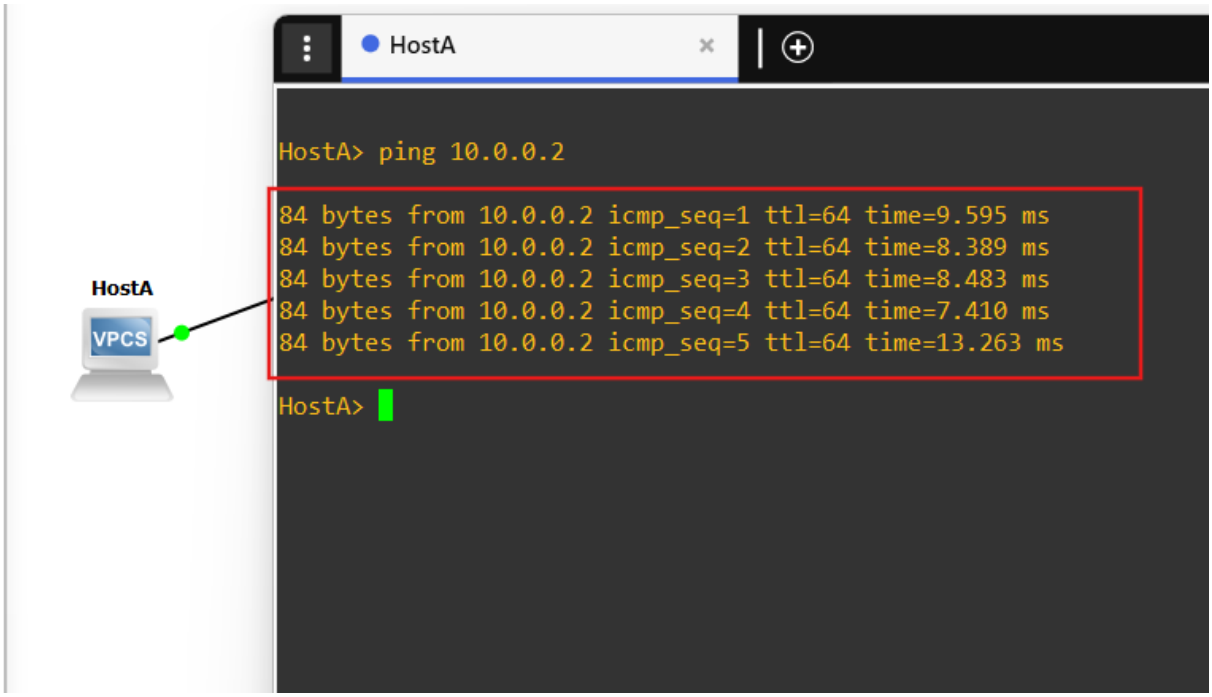
Static IP addresses were assigned to the VPCS nodes:

1. Host A: 10.0.0.1/24
2. Host B: 10.0.0.2/24

Step 7: Testing Connectivity

Perform:

1. In Host A console, type: ping 10.0.0.2



2. Observe replies.

Connectivity was tested using the ping command from Host A to Host B. Successful ICMP replies were received, confirming that the OVS bridge was forwarding traffic correctly between the two hosts.

Step 8: Technical Analysis

In this lab, the OVS bridge functioned as a learning switch in the data plane. Unlike the classical built-in switch from Lab 1, OVS is programmable and can be managed via the OpenFlow protocol when connected to an SDN controller. This aligns with the NFV vision of using Standard High Volume Ethernet Switches-software-based, scalable, and automation-ready.

Step 9: Answer Assessment Questions

Question 1: Compare the "Classical" switch from Lab 1 with the OVS appliance. How does OVS support the concept of Standard High Volume Ethernet Switches described in the NFV vision?

Answer: The classical switch in Lab 1 is a fixed-function, non-programmable device. OVS, however, is software-defined, open-source, and programmable via APIs (e.g., OpenFlow). This supports the NFV vision by decoupling switching functions from proprietary hardware, enabling automation, scalability, and integration with virtualized network functions (VNFs).

Question 2: Based on the ETSI framework, which component is responsible for managing the OVS resources you configured today?

Answer: In the ETSI NFV framework, the Virtual Infrastructure Manager (VIM) is responsible for managing OVS resources, including bridge creation, port mapping, and network resource allocation.

Step 10: Conclusion

This lab successfully demonstrated the deployment and configuration of Open vSwitch within GNS3. OVS provides a flexible, programmable data plane that forms the foundation for advanced SDN/NFV implementations, such as Service Function Chaining (SFC) and dynamic traffic engineering.