IoT network (digital twin) security testbed using GNS3

This tutorial outlines the step-by-step approach for developing a network security testbed using GNS3.

Digital Twin IoT Network Testbed

Files and Scripts are available in the repository:

https://github.com/Siong23/iot-digital-twin

OS: [Ubuntu 22.04.4 LTS](https://releases.ubuntu.com/22.04/?_gl=1*1r2uc59*_gcl_au*MTc1Nzg2NDA4MS4xNzIxNjE4MTUx&_ga=2.157068985.348104353.1721968430-1698206738.1721618148)

Dependencies:

* Have [KVM Support](https://www.cyberciti.biz/faq/linux-xen-vmware-kvm-intel-vt-amd-v-support/)
* [make](https://www.geeksforgeeks.org/how-to-install-make-on-ubuntu/)
* [wget](https://www.cyberciti.biz/faq/how-to-install-wget-togetrid-of-error-bash-wget-command-not-found/)
* [konsole](https://www.ubuntumint.com/konsole-terminal-emulator/)
* [Python 3](https://www.cherryservers.com/blog/install-python-on-ubuntu), as well as [Python3-venv](https://www.cherryservers.com/blog/install-python-on-ubuntu#step-2-set-up-a-virtual-environment-for-python3--optional-) package
* [Docker](https://docs.docker.com/engine/install/)
* [GNS3](https://www.gns3.com/) (better to install it in Linux).
* [Cisco 7200 router GNS3 image](https://www.gns3.com/marketplace/appliances/cisco-7200).
* Open vSwitch (Fixed version).
* [Ubuntu Server](https://releases.ubuntu.com/22.04/?_ga=2.149898549.2084151835.1707729318-1126754318.1683186906&_gl=1*144ytao*_gcl_au*NzEyNTM5MzIuMTc0NzM5ODc3Mg..).
* Ubuntu Guest Additions.
* [Kali Linux for GNS3](https://gns3.com/marketplace/appliances/kali-linux-2).
* TightVNC.
* [MediaMTX](https://github.com/bluenviron/mediamtx)

References:

* [MQTT Broker Setup](https://medium.com/gravio-edge-iot-platform/how-to-set-up-a-mosquitto-mqtt-broker-securely-using-client-certificates-82b2aaaef9c8).

Table of Contents

[Installing GNS3 Linux 1](#_Toc206665313)

[Installing Docker CE 2](#_Toc206665314)

[Installing Fixed Open vSwitch Appliances 3](#_Toc206665315)

[Installing Cisco 7200 router image 6](#_Toc206665316)

[Installing Ubuntu Server appliance 11](#_Toc206665317)

[Installing Ubuntu Desktop Guest appliance 18](#_Toc206665318)

[Creating the IoT Physical Twin Topology 22](#_Toc206665319)

[Creating IoT Physical Twin Scenarios (Physical IoTBroker/Server) 27](#_Toc206665320)

[Creating IoT Physical Twin Scenarios (Physical TempHumidSensor) 30](#_Toc206665321)

[Creating the IoT Digital Twin Topology 31](#_Toc206665322)

[Creating IoT Digital Twin Scenarios (ProxyBridge) 39](#_Toc206665323)

[Creating IoT Digital Twin Scenarios (DigitalIoTBroker/Server) 44](#_Toc206665324)

[Creating IoT Digital Twin Scenarios (DigitalIPCam) 47](#_Toc206665325)

[Creating IoT Digital Twin Scenarios (TemperatureHumiditySensor) 49](#_Toc206665326)

[Creating IoT Digital Twin Scenarios (IoTConsumer) 50](#_Toc206665327)

[Creating Automatic Router Configuration Sync (Physical to Digital) 52](#_Toc206665328)

# Installing GNS3 Linux

1. To install GNS3, we need to **add GNS3 apt repository**. In the terminal, run the command:
   * $ sudo add-apt-repository ppa:gns3/ppa

* Then, **update the apt repository** by using the command:
  + $ sudo apt update
* Run the command to **install the GNS3** user interface as well as GNS3 Server:
  + $ sudo apt install gns3-gui gns3-server
* Select “**Yes**” both times when prompted whether non-root users should be allowed to use **wireshark** and **ubridge**.

1. To **include IOU Support**, run these commands:
   * $ sudo dpkg --add-architecture i386
   * $ sudo apt update
   * $ sudo apt install gns3-iou

# Installing Docker CE

1. To install Docker CE, first we need to **remove any old versions** by using the command:
   * $ sudo apt remove docker docker-engine docker.io
   * $ sudo snap remove docker
2. Then, **install the following packages** using the command:
   * $ sudo apt-get install apt-transport-https ca-certificates curl software-properties-common
3. **Import the official Docker GPG key** using the command:
   * $ curl -fsSL <https://download.docker.com/linux/ubuntu/gpg> | sudo apt-key add –
4. **Add the appropriate repository** using the command (after each “\”, press **enter**):
   * $ sudo add-apt-repository \

"deb [arch=amd64] [https://download.docker.com/linux/ubuntu \](https://download.docker.com/linux/ubuntu%20\)

$(lsb\_release -cs) stable"

1. Then, **install the Docker-CE** using the command:
   * $ sudo apt update
   * $ sudo apt install docker-ce
2. Finally, **add your user to the following groups** (ubridge, libvirt, kvm, wireshark, docker) using the command and restart your user session by logging out and back in, or restarting the system:
   * $ sudo usermod -aG ubridge,libvirt,kvm,wireshark,docker $(whoami)

# Installing Fixed Open vSwitch Appliances

1. First, you will need to install the fixed Open vSwitch Appliances for GNS3 using this link: <https://gitlab.com/Fumeaux/openvswitch/-/raw/master/openvswitch-fixed.gns3a?ref_type=heads&inline=false>.
2. After we have finished installing the appliances file, we need to **fix the source code** of the new switch as below:

* In the /openvswitch-fixed.gns3a, we need to **remove** the **“,”** in the source code in **line 17** and save the file:

A close up of a text

Description automatically generated

Line 17: Before

A close up of a text

Description automatically generated

Line 17: After

1. Then, in GNS3, click this icon to get to the switch window.

A screenshot of a computer

AI-generated content may be incorrect.

1. Click the “**+ New template**" button at the bottom of the window.

A white background with a black and white text

AI-generated content may be incorrect.

1. Make sure to select “**Import an appliance file (.gns3a extension)**” and click **Next**.

A screenshot of a computer

AI-generated content may be incorrect.

1. Find the openvswitch-fixed.gns3a file that you have downloaded and click **Open**.

A screenshot of a computer

AI-generated content may be incorrect.

1. Select as below and click **Next**.

A screenshot of a computer

AI-generated content may be incorrect.

1. Click **Finish**.

A screenshot of a computer

AI-generated content may be incorrect.

# Installing Cisco 7200 router image

1. Click the download button on Cisco 7200 website that has been attached to the hyperlink in the prerequisite.

A screenshot of a phone

Description automatically generated

1. Download the Cisco 7200 image from [this link](https://mega.nz/file/RZtA0SwD#XBjqI5Dkrienz7tHaYg601Dwq-ypAqWZv8Ut3mFuKoI).

A white light switch on a black background

Description automatically generated

1. After finishing installing the file, extract the Cisco 7200 zip file.

A screenshot of a computer

Description automatically generated

Output of the extraction.

1. In GNS3, go to **File > Import Appliance** and **import** the cisco-7200.gns3a file.

A screenshot of a computer code

Description automatically generated

1. Select “**Install the appliance on your local computer**” and click **Next**.

A screenshot of a computer

Description automatically generated

1. Just click **Next** until you reach this page, usually GNS3 will automatically detect the appliance file if the name of the file is correct. If the **7200 version 124-24.T5** is **ready to install**, **select the appliance version** and click **Next**. If the status is missing files, just follow the next step.

A screenshot of a computer error

Description automatically generated

1. If the **status is missing files**, click the **dropdown** of the version you want to install and click **Import**.

A screenshot of a computer

Description automatically generated

1. Go to the output file of the extraction that you did earlier, select the .image file.

A red and black rectangular object with numbers

Description automatically generated

1. After finishing uploading the .image file, click **the version** and **Next**.

A screenshot of a computer error

Description automatically generated

1. Click **Yes** to proceed with the installation.

A screenshot of a computer

Description automatically generated

1. Click **Finish** when you reach this page.

A screenshot of a computer

Description automatically generated

1. The router image now be able to be used by going to the router category.

A screenshot of a computer

Description automatically generated

# Installing Ubuntu Server appliance

1. Go to the Ubuntu Server website that has been attached as hyperlinks in the prerequisite or just [click here](https://releases.ubuntu.com/22.04/?_ga=2.149898549.2084151835.1707729318-1126754318.1683186906&_gl=1*144ytao*_gcl_au*NzEyNTM5MzIuMTc0NzM5ODc3Mg..). Make sure to download the **Server install image** version.

A screenshot of a computer

AI-generated content may be incorrect.

1. After finishing downloading the iso image, in GNS3, create a new template for the Ubuntu Server.

A screenshot of a computer

AI-generated content may be incorrect.

1. Select as below and click **Next**.

A screenshot of a computer

AI-generated content may be incorrect.

1. Click “**QEMU VMs**”.

A screenshot of a computer

AI-generated content may be incorrect.

1. Click “**New**” to create new templates.

A grey rectangular object with a white border

AI-generated content may be incorrect.

1. Choose any name for the template and click **Next**.

A screenshot of a computer

AI-generated content may be incorrect.

1. We will put **2048 MB RAM** for this Ubuntu Server.

**A screenshot of a computer program

AI-generated content may be incorrect.**

1. For console type, choose “**vnc**”.

A screenshot of a computer program

AI-generated content may be incorrect.

1. As for the disk image, we will need to “**Create New Image**”.

A screenshot of a computer

AI-generated content may be incorrect.

1. Select as below and click **Next** and **Finish**.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. Make sure to click **Apply** and **OK**.

A screenshot of a computer

AI-generated content may be incorrect.

1. So, the next step will just need to continue with the installation of the OS.

# Installing Ubuntu Desktop Guest appliance

1. Make sure to download the **Ubuntu Desktop image** first by visiting the GNS3 Appliance Marketplace or just [click here](https://gns3.com/marketplace/appliances/ubuntu-with-gui). Download the “**22.04**” version.

A screenshot of a computer

AI-generated content may be incorrect.

1. Create a new template in GNS3 and select as below:

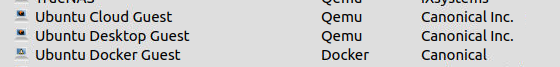
A screenshot of a computer

AI-generated content may be incorrect.

1. Click “**Guest**”, find “**Ubuntu Desktop Guest**” and click **Install**.

A screenshot of a computer

AI-generated content may be incorrect.



1. Click **Next**.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

1. You will see the downloaded images are found by GNS3, click the **version 22.04** and click **Next.** Click **Yes** if the popup appears asking to install the appliance and make sure to remember the username and password for the appliance.

A screenshot of a computer

AI-generated content may be incorrect.

1. So, the next step will just need to continue with the installation of the OS.

# Creating the IoT Physical Twin Topology

A computer on a desk

AI-generated content may be incorrect.

1. First, configure the physical router based on the command below:
   1. Enable and open the configuration terminal

R1> enable

R1# configure terminal

* 1. Configure the GigabitEthernet 0/1 (Connected to the PoE Switch)

R1(config)# interface GigabitEthernet 0/1

R1(config-if)# ip address 192.168.254.1 255.255.255.0

R1(config-if)# no shutdown

R1(config-if)# exit

* 1. Configure the GigabitEthernet 0/2 (Connected to the Digital Twin PC)

R1(config)# interface GigabitEthernet 0/2

R1(config-if)# ip address 192.168.10.1 255.255.255.0

R1(config-if)# no shutdown

R1(config-if)# exit

* 1. Configure the GigabitEthernet 0/3 (Connected to the Physical Twin PC)

R1(config)# interface GigabitEthernet 0/3

R1(config-if)# ip address 192.168.20.1 255.255.255.0

R1(config-if)# no shutdown

R1(config-if)# exit

* 1. Configure the IP Address that will be excluded from the DHCP

R1(config)# ip dhcp excluded-address 192.168.254.1 192.168.254.9

* 1. Setting up the DHCP

R1(config)# ip dhcp pool physical-twin

R1(config-dhcp)# network 192.168.254.0 255.255.255.0

R1(config-dhcp)# default-router 192.168.254.1

R1(config-dhcp)# exit

* 1. Setting up the Default IP Route (Route to PhysicalRouter for allowing traffic to go through)

R1(config)# ip route 0.0.0.0 0.0.0.0 192.168.10.254

* 1. Saving the configuration to memory and boot file (It will run the configuration every time the router boots up)

R1# copy run start

R1# write memory

1. Connect the rest of the appliances by referencing them to the table below:

|  |  |  |  |
| --- | --- | --- | --- |
| **Device** | **Connected From** | **Connected to** | **IPv4 Address** |
| **Network Layer** | | | |
| Physical Router | g0/1 | PoE Switch  Port 5 (uplink) | 192.168.254.1 |
| g0/2 | Digital Twin PC  enp3s0 | 192.168.10.1 |
| g0/3 | Physical Twin PC  enp5s0 | 192.168.20.1 |
| Digital Twin PC | enp3s0 | Physical Router  g0/2 | Not Set  (Make sure to turn off the Ethernet Interface) |
| **Cloud Layer** | | | |
| Physical Twin PC | enp5s0 | Physical Router | 192.168.20.2 |
| **Edge Layer** | | | |
| Physical IPCamera |  | PoE Switch  Port 2 | 192.168.254.3  (Static IP) |
| Raspberry Pi | eth0 | PoE Switch  Port 1 | DHCP  (192.168.254.x) |

1. In Physical PC network configuration, make sure to set the default gateway as below:

|  |  |  |
| --- | --- | --- |
| **IP Routes** | | |
| **Address** | **Netmask** | **Gateway** |
| 192.168.254.0 | 255.255.255.0 | 192.168.20.1 |
| 192.168.10.0 | 255.255.255.0 | 192.168.20.1 |

1. For creating SNMP Server in Physical Router:
   1. Enable and open the configuration terminal

R1> enable

R1# configure terminal

* 1. Creating SNMP-Server configuration:

R1(config)# snmp-server view mib2 mib-2 included

R1(config)# snmp-server community public view ROUTERVIEW RO

R1(config)# snmp-server community private RW

R1(config)# snmp-server location PhysicalTwinRouter

R1(config)# snmp-server contact PhysicalTwinAdmin

* 1. Configure the IP Flow configuration:

R1(config)# ip flow-cache timeout active 1

R1(config)# ip flow-export source GigatbitEthernet0/3

R1(config)# ip flow-export version 9

R1(config)# ip flow-export destination 192.168.20.2 9996

* 1. Configure the Flow Exporter configuration:

R1(config)# flow exporter PHYSExporter

R1(config-flow-exporter)# destination 192.168.20.2

R1(config-flow-exporter)# source GigabitEthernet0/3

R1(config-flow-exporter)# transport udp 9996

R1(config-flow-exporter)# template data timeout 60

R1(config-flow-exporter)# exit

* 1. Configure the Flow Monitor configuration:

R1(config)# flow monitor PHYSMonitor

R1(config-flow-monitor)# record netflow-original

R1(config-flow-monitor)# exporter PHYSExporter

R1(config-flow-monitor)# cache timeout active 60

R1(config-flow-monitor)# exit

* 1. Setting up the PHYSMonitor for interface GigabitEthernet0/1:

R1(config)# interface GigabitEthernet0/1

R1(config-if)# description Edge Layer – 192.168.254.1

R1(config-if)# ip flow monitor PHYSMonitor input

R1(config-if)# ip flow monitor PHYSMonitor output

R1(config-if)# exit

* 1. Setting up the PHYSMonitor for interface GigabitEthernet0/2:

R1(config)# interface GigabitEthernet0/2

R1(config-if)# description Network Layer – 192.168.10.1

R1(config-if)# ip flow monitor PHYSMonitor input

R1(config-if)# ip flow monitor PHYSMonitor output

R1(config-if)# exit

* 1. Setting up the PHYSMonitor for interface GigabitEthernet0/3:

R1(config)# interface GigabitEthernet0/3

R1(config-if)# description Cloud Layer – 192.168.20.1

R1(config-if)# ip flow monitor PHYSMonitor input

R1(config-if)# ip flow monitor PHYSMonitor output

R1(config-if)# exit

* 1. Saving the configuration to memory and boot file (It will run the configuration every time the router boots up):

R1# copy run start

R1# write memory

# Creating IoT Physical Twin Scenarios (Physical IoTBroker/Server)

1. Then, to create the scenarios, in Physical Twin PC, make sure to download [MQTT Broker](https://medium.com/gravio-edge-iot-platform/how-to-set-up-a-mosquitto-mqtt-broker-securely-using-client-certificates-82b2aaaef9c8) and [MediaMTX binary file](https://github.com/bluenviron/mediamtx).
2. First, for MQTT Broker, install the MQTT using the command:
   1. Update the repo:

$ sudo apt update

* 1. Install MQTT:

$ sudo apt install mosquitto mosquitto-clients -y

* 1. Configure MQTT like below:

$ sudo nano /etc/mosquitto/mosquitto.confA computer screen with white text

AI-generated content may be incorrect.

* 1. Setting up User Authentication:

$ sudo mosquitto\_passwd -c /etc/mosquitto/passwd admin

<Insert Password as admin123>

* 1. Restart Mosquitto:

$ sudo systemctl restart mosquitto

* 1. Test the MQTT broker:

$ mosquitto\_sub -h localhost -t test -u “admin” -P “admin123”

$ mosquitto\_pub -h localhost -t test -m “Hello World” -u “admin” -P “admin123”

* 1. Now, get script from GitHub (https://github.com/Siong23/iot-digital-twin/IoTDevice/MQTTScenarios/mqttbroker.py) to view any published data to the broker:

$ python3 mqttbroker.py

1. Now, for MediaMTX server:
   1. Download the MediaMTX binary:

$ wget <https://github.com/bluenviron/mediamtx/releases/download/v1.12.2/mediamtx_v1.12.2_linux_amd64.tar.gz>

* 1. Extract the file:

$ sudo tar -xvf \*.tar.gz

* 1. Make sure to set the mediamtx.yml file as below:

$ nano mediamtx.yml

A screenshot of a computer program

AI-generated content may be incorrect.

* 1. In Path settings, set the source based on the Physical IPCamera link to view stream.

A screenshot of a computer

AI-generated content may be incorrect.

* 1. Then, run the mediamtx binary:

$ ./mediamtx

# Creating IoT Physical Twin Scenarios (Physical TempHumidSensor)

1. For Raspberry Pi, get the script from GitHub (<https://github.com/Siong23/iot-digital-twin/IoTDevice/MQTTScenarios/sensor.py>)
2. Make sure to create the mqtt\_credentials.txt file first:

$ nano mqtt\_credentials.txt

A screen shot of a computer

AI-generated content may be incorrect.

1. Then, run the script:

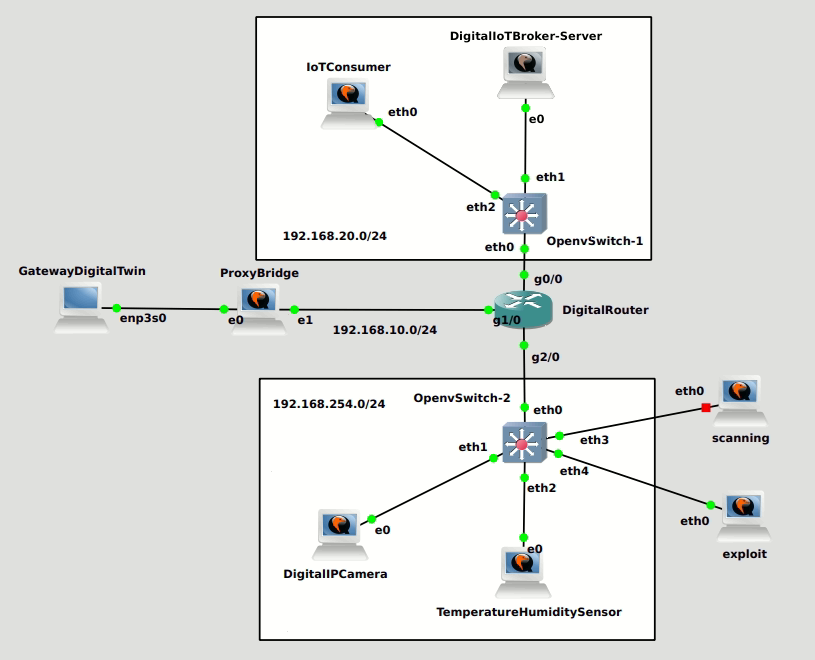
$ python3 sensor.py

1. Now, the Raspberry Pi and MQTT Broker are connected.

A screenshot of a computer

AI-generated content may be incorrect.

# Creating the IoT Digital Twin Topology



1. Before creating the project, make sure to disable the network interface that we have connected to the Physical Router.
2. In GNS3, go to **File > Create new blank project** and name the project as you want and click **OK**.

A screenshot of a computer

AI-generated content may be incorrect.

1. After creating the project, add a Cloud appliance (GatewayDigitalTwin), 1 Cisco 7200 router, 2 Open vSwitch, 5 Ubuntu Server, 3 Ubuntu Guest (IoTConsumer, scanning and exploit) into the project as above.
2. You can change the symbol for Cloud1 and name for each device. Name the devices as below for better understanding.

A computer diagram of a network

AI-generated content may be incorrect.

1. We need to configure the router, right click on the router appliance and configure the adapter slots of the router as below.

A screenshot of a computer

Description automatically generated

1. Then, connect the appliances by using the **Add a link** button.

A close up of a computer

Description automatically generated

1. Connect the GatewayDigitalTwin to DigitalRouter.

A computer diagram of a computer network

AI-generated content may be incorrect.

1. Connect the rest of the appliances by referencing to the table below:

|  |  |  |  |
| --- | --- | --- | --- |
| **Device** | **Connected From** | **Connected to** | **IPv4 Address** |
| **Network Layer** | | | |
| GatewayDigitalTwin | Your Connected GNS3 PC’s Network Interface to Physical Router | ProxyBridge  e0 | - |
| DigitalRouter | g0/0 | OpenvSwitch-1  eth0 | 192.168.20.1 |
| g1/0 | ProxyBridge  e1 | 192.168.10.1 |
| g2/0 | OpenvSwitch-2  eth2 | 192.169.254.1 |
| ProxyBridge | e0 | GatewatyDigitalTwin | 192.168.10.254 |
| e1 | DigitalRouter  g1/0 | 192.168.10.254 |
| **Cloud Layer (10.10.10.0/24)** | | | |
| IoTConsumer | eth0 | Open vSwitch-1  eth2 | 192.168.20.3 |
| DigitalIoTBroker-Server | e0 | Open vSwitch-1  eth1 | 192.168.20.2 |
| **Edge Layer (11.10.10.0/24)** | | | |
| DigitalIPCamera | e0 | Open vSwitch-1  eth1 | 192.168.254.3 |
| TemperatureHumiditySensor | e0 | Open vSwitch-1  eth2 | DHCP  (192.168.254.x) |
| **Attack Device (Connected in Edge Layer)** | | | |
| exploit | eth0 | Open vSwitch-1  eth4 | DHCP  (192.168.254.x) |
| scanning | eth0 | Open vSwitch-1  eth3 | DHCP  (192.168.254.x) |

1. After connecting all the appliances, we will start with router configuration, but before that start all the devices by using this button.

A close up of a logo

Description automatically generated

1. Right click the router to open the console and wait for the router to finish booting up.

A screenshot of a computer program

AI-generated content may be incorrect.

1. Configure the router using this configuration:
   1. Enable and open the configuration terminal

R1> enable

R1# configure terminal

* 1. Configure the GigabitEthernet 0/0 (Connected to the ProxyBridge)

R1(config)# interface GigabitEthernet 0/0

R1(config-if)# ip address 192.168.20.1 255.255.255.0

R1(config-if)# no shutdown

R1(config-if)# exit

* 1. Configure the GigabitEthernet 1/0 (Connected to the Agent)

R1(config)# interface GigabitEthernet 1/0

R1(config-if)# ip address 192.168.10.1 255.255.255.0

R1(config-if)# no shutdown

R1(config-if)# exit

* 1. Configure the GigabitEthernet 2/0 (Connected to the Edge Layer’s Switch)

R1(config)# interface GigabitEthernet 2/0

R1(config-if)# ip address 192.168.254.1 255.255.255.0

R1(config-if)# no shutdown

R1(config-if)# exit

* 1. Configure the IP Address that will be excluded from the DHCP

R1(config)# ip dhcp excluded-address 192.168.254.1 192.168.254.9

* 1. Setting up the DHCP

R1(config)# ip dhcp pool physical-twin

R1(config-dhcp)# network 192.168.254.0 255.255.255.0

R1(config-dhcp)# default-router 192.168.254.1

R1(config-dhcp)# exit

* 1. Setting up the Default IP Route (Route to ProxyBridge for allowing traffic to go through)

R1(config)# ip route 0.0.0.0 0.0.0.0 192.168.10.254

* 1. Saving the configuration to memory and boot file (It will run the configuration every time the router boots up)

R1# copy run start

R1# write memory

1. For creating SNMP Server in Digital Router:
   1. Enable and open the configuration terminal

R1> enable

R1# configure terminal

* 1. Creating SNMP-Server configuration:

R1(config)# snmp-server view mib2 mib-2 included

R1(config)# snmp-server community public view ROUTERVIEW RO

R1(config)# snmp-server community private RW

R1(config)# snmp-server location PhysicalTwinRouter

R1(config)# snmp-server contact PhysicalTwinAdmin

* 1. Configure the IP Flow configuration:

R1(config)# ip flow-cache timeout active 1

R1(config)# ip flow-export source GigatbitEthernet0/0

R1(config)# ip flow-export version 9

R1(config)# ip flow-export destination 192.168.20.2 9996

* 1. Configure the Flow Exporter configuration:

R1(config)# flow exporter PHYSExporter

R1(config-flow-exporter)# destination 192.168.20.2

R1(config-flow-exporter)# source GigabitEthernet0/0

R1(config-flow-exporter)# transport udp 9996

R1(config-flow-exporter)# template data timeout 60

R1(config-flow-exporter)# exit

* 1. Configure the Flow Monitor configuration:

R1(config)# flow monitor PHYSMonitor

R1(config-flow-monitor)# record netflow-original

R1(config-flow-monitor)# exporter PHYSExporter

R1(config-flow-monitor)# cache timeout active 60

R1(config-flow-monitor)# exit

* 1. Setting up the PHYSMonitor for interface GigabitEthernet2/0:

R1(config)# interface GigabitEthernet2/0

R1(config-if)# description Edge Layer – 192.168.254.1

R1(config-if)# ip flow monitor PHYSMonitor input

R1(config-if)# ip flow monitor PHYSMonitor output

R1(config-if)# exit

* 1. Setting up the PHYSMonitor for interface GigabitEthernet1/0:

R1(config)# interface GigabitEthernet1/0

R1(config-if)# description Network Layer – 192.168.10.1

R1(config-if)# ip flow monitor PHYSMonitor input

R1(config-if)# ip flow monitor PHYSMonitor output

R1(config-if)# exit

* 1. Setting up the PHYSMonitor for interface GigabitEthernet0/0:

R1(config)# interface GigabitEthernet0/0

R1(config-if)# description Cloud Layer – 192.168.20.1

R1(config-if)# ip flow monitor PHYSMonitor input

R1(config-if)# ip flow monitor PHYSMonitor output

R1(config-if)# exit

* 1. Saving the configuration to memory and boot file (It will run the configuration every time the router boots up):

R1# copy run start

R1# write memory

# Creating IoT Digital Twin Scenarios (ProxyBridge)

1. First we will configure the ProxyBridge, to open multiple terminals, I use X-Server which can be downloaded with simple command:
   1. Update the repo:$ sudo apt update
   2. Install X server:$ sudo apt install xserver-xorg-core xinit xterm
   3. Run the X windows:$ startx

A screenshot of a computer

AI-generated content may be incorrect.

1. You can open the terminal by clicking anywhere.

A screen shot of a computer

AI-generated content may be incorrect.

1. Write the routing setup to make sure traffic flows correctly.

* $ sudo nano /etc/networkd-dispatcher/routingsetup
* Paste below:

#!bin/bash

# Policy-based routing rules

ip addr flush dev ens4

ip addr flush dev ens5

ip addr add 192.168.10.254/24 dev ens4

ip addr add 192.168.10.254/24 dev ens5

ip addr add 192.168.10.241/24 dev ens4

ip addr add 192.168.10.242/24 dev ens5

#Physical

ip route flush table phys

ip route add 192.168.10.0/24 dev ens4 table phys

ip route add default via 192.168.10.1 dev ens4 table phys

ip rule add from 192.168.10.254 table phys

#Digital

ip route flush table digi

ip route add 192.168.10.0/24 dev ens5 table digi

ip route add default via 192.168.10.1 dev ens5 table digi

ip rule add from 192.168.10.254 table digi

ip route add 192.168.254.0/24 dev ens5

ip rule add to 192.168.20.2 lookup phys

ip rule add from 192.168.10.241 lookup phys

ip rule add from 192.168.10.242 lookup digi

* Save and run $ sudo nano /etc/systemd/system/routing-setup.service

A screenshot of a computer program

AI-generated content may be incorrect.

Make sure everything is same as above

* $ sudo systemctl enable routing-setup
* $ sudo reboot

1. Now all the rules are applied, now we need to have the MediaMTX server and MQTT running which will retrieve and publish the data from Physical Broker server to Digital Edge. For receiving the streams, we need to download the MediaMTX binary file from the GitHub and unzip the file by using the command:

* $ wget <https://github.com/bluenviron/mediamtx/releases/download/v1.12.2/mediamtx_v1.12.2_linux_amd64.tar.gz>
* $ sudo tar -xvf \*.tar.gz

1. Configure the mediamtx.yml file as below:

* $ sudo nano mediamtx.yml

A screenshot of a computer

AI-generated content may be incorrect.

Make sure to check rtsp server as yes.

A screenshot of a computer error

AI-generated content may be incorrect.

Physical IoT Broker IP Address

1. To run the MediaMTX server to receive the streams, use the command “./mediamtx”.

A screenshot of a computer program

AI-generated content may be incorrect.

1. For the MQTT to retrieve and publish sensors data, we need to install the required packages:

$ sudo apt install mosquito mosquito-clients -y

1. In the GitHub repo we have the scripts for running the mqttproxy (https://github.com/Siong23/iot-digital-twin/blob/main/TwinningAgent/mqttproxy.py).

* $ sudo nano /etc/mosquitto/mosquitto.conf

A screenshot of a computer

AI-generated content may be incorrect.

Configuration of the mosquito.conf (See References).

* $ python3 mqttproxy.py (Make sure to check if the credentials, bind address host and title are correct.)

# Creating IoT Digital Twin Scenarios (DigitalIoTBroker/Server)

1. Now we will configure DigitalIoTBroker/Server to make sure the Digital IoT devices can publish all the physical IoT devices data towards it.
2. For Digital IoT Broker/Server, we need to have the MediaMTX server and MQTT server running which will act as the broker for the digital IoT devices. For receiving the DigitalIPCam streams, we need to download the MediaMTX binary file from the GitHub and unzip the file by using the command:

* $ wget <https://github.com/bluenviron/mediamtx/releases/download/v1.12.2/mediamtx_v1.12.2_linux_amd64.tar.gz>
* $ sudo tar -xvf \*.tar.gz

1. Configure the mediamtx.yml file as below:

* $ sudo nano mediamtx.yml

A screenshot of a computer

AI-generated content may be incorrect.

Make sure to check rtsp server as yes.

A screenshot of a computer

AI-generated content may be incorrect.

Make sure to comment all the path settings so that we not retrieving the streams but we accept for devices to publish to its (At the bottom of the file)

1. To run the MediaMTX server to receive the DigitalIPCam streams, use the command “./mediamtx”.

A screenshot of a computer program

AI-generated content may be incorrect.

1. For the MQTT server, in the GitHub repo also have the scripts for running the MQTT broker (https://github.com/Siong23/iot-digital-twin/IoTDevice/MQTTScenarios/mqttbroker.py).

* $ sudo nano /etc/mosquito/mosquito.conf

A screenshot of a computer

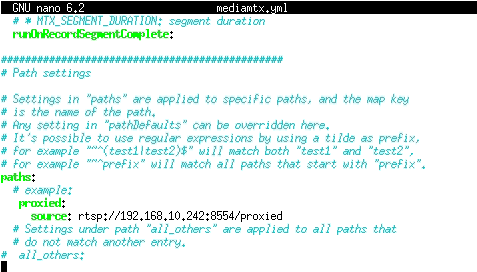
AI-generated content may be incorrect.

Configuration of the mosquito.conf (See References).

* $ python3 mqttbroker.py (To start the broker and listen any publish, make sure to check if the credentials, host and title is correct.)

# Creating IoT Digital Twin Scenarios (DigitalIPCam)

1. For Digital IPCam, same as before, we need to run the MediaMTX server to retrieve the streams from ProxyBridge:



ProxyBridge Secondary IP Address

* $ ./mediamtx (To run the MediaMTX server)

1. After running the MediaMTX server, just use ffmpeg command to publish the stream to the Digital IoT Broker/Server:

* $ sudo apt install ffmpeg
* $ ffmpeg -re -stream\_loop -1 -i rtsp://{DigitalIPCam IP}:8554/ip\_camera -c copy -f rtsp rtsp://{DigitalIoTBroker/Server IP}:8554/ip\_camera (Make sure the Broker is running before)

1. I have created a script file for the ffmpeg command, to run just type:

* $./relayrtsp.sh

A screenshot of a computer screen

AI-generated content may be incorrect.

# Creating IoT Digital Twin Scenarios (TemperatureHumiditySensor)

1. For Digital IoT TempHumid Sensor, we need to install the required packages:

* $ sudo apt install mosquito mosquitto-clients -y

1. Run the broker using our created scripts:

* $ python3 mqttbroker.py

1. Then we need to have a script to publish the data from the self-hosted broker inside TemperatureHumiditySensor (Digital) to be published to the Digital IoT Broker/Server. The script is also ready in the GitHub Repo named as publish.py. Then run the script.

* $ python3 publish.py

A screenshot of a computer program

AI-generated content may be incorrect.

# Creating IoT Digital Twin Scenarios (IoTConsumer)

1. As we did earlier, for IoTConsumer we will use Ubuntu Desktop Guest for better viewing of the streams and retrieving the IoT sensors data.

* $ sudo apt install mosquito mosquitto-clients -y
* $ sudo apt install ffmpeg
* $ ffplay rtsp://{DigitalIoTServer/Broker IP}:8554/ip\_camera (To play the stream from Broker)

1. I also have created the script to run the ffplay command, to run just type:

* $ ./autoplay.sh

A screenshot of a computer

AI-generated content may be incorrect.

1. To retrieve Temperature Humidity Sensor data from Digital IoT Broker, run this command

* $ mosquitto\_sub -h {DigitalIoTServer/Broker IP} -u admin -P admin123 -t sensors/digital/data (To retrieve IoT sensor data from Broker)

1. I also have created the scripts to run the subscribe command, just run:

* $ ./subscribe\_mqtt.sh

# Creating Automatic Router Configuration Sync (Physical to Digital)

1. Configure the physical and digital router so they can use ssh.
   1. Enable and open the configuration terminal

R1> enable

R1# configure terminal

* 1. Set domain, generate RSA key, force SSH v2 and local login user:

R1(config)# ip domain-name lab.local

R1(config)# crypto key generate rsa modulus 1024 R1(config)# ip ssh version 2

R1(config)# username {YourUserName} privilege 15 secret {YourPassword}

* 1. Configure the VTY line:

R1(config)# line vty 0 4

R1(config-line)# login local

R1(config-line)# transport input telnet ssh

R1(config-line)# end

* 1. Saving the configuration to memory and boot file (It will run the configuration every time the router boots up):

R1# copy run start

R1# write memory

1. Now we go back to ProxyBridge and install Netmiko

$ python3 pip install netmiko

1. After that, get the script to sync the config in the GitHub repo (<https://github.com/Siong23/iot-digital-twin/blob/main/TwinningAgent/config_sync_ssh.py>)

* Try running $ python3 config\_sync\_ssh.py (remember to change the username, password and secret to what you set before)



1. To make it run automatically every hour, we use Cron to do it.

- $ crontab -e

1. Add this line to the bottom of the file

0 \* \* \* \* /usr/bin/python3 /location of config sync python file/config\_sync\_ssh.py

>> /location of configs log file/cron.log 2>&1

A screenshot of a computer code

AI-generated content may be incorrect.

1. Save it and the config sync will now run automatically every hour.