# GNS3 Laboratory 101

## Download

To properly configure your GNS3 installation, you need to download these resources. Do not proceed until all downloads are complete!

- Cisco C7200 (~45 MB)
- Cisco C3745 (~38 MB)
- Ubuntu Cloud Host (~580 MB)
- Ubuntu Cloud Init Data (~128 KB)
- C3745_startup-config (~1.5K)
- c7200_startup-config (~1.6K)

## Installation

You should have a native *nix-based host OS. If not, it is strongly recommended that you set one up before using GNS3, as Windows OS often encounters various issues with this software. Additionally, some upcoming labs require a native Linux installation.

Therefore, I will provide instructions only for Ubuntu and Arch Linux. For other systems, please refer to the project's official doc.

Let's begin the GNS3 installation!

### Ubuntu and Ubuntu like distro

```
$ sudo add-apt-repository ppa:gns3/ppa
$ sudo apt update

### As I write this tutorial, a GNS3 dependency is broken so I have to install it manually.
$ wget http://launchpadlibrarian.net/722482020/vinagre_3.22.0-8ubuntu5_amd64.deb
$ sudo dpkg -i vinagre_3.22.0-8ubuntu5_amd64.deb
$ sudo apt-get -f install

$ sudo apt install qemu-system ubridge gns3-gui gns3-server # yes and no
$ sudo usermod -aG ubridge,libvirt,kvm $(whoami) # relogin to make this change effective
```

Finally, run it using the following command.

```
$ gns3
```

### Archlinux and archlinux like distro

For the Arch installation, we need to use an AUR package manager, such as YAY. To install it from a binary file, follow these steps.

```
$ sudo pacman -S --needed git base-devel
$ git clone https://aur.archlinux.org/yay-bin.git
$ cd yay-bin
$ makepkg -si
```

```
$ yay -S qemu vpcs dynamips libvirt ubridge inetutils python-sqlalchemy gns3-server gns3-gui
$ sudo usermod -aG ubridge,libvirt,kvm $(whoami) # Log out and log in back to apply the
changes.
```

# First configuration

Below are the images of the initial GNS3 configuration. Follow the steps shown in the images to set it up correctly.

# Template creation

Now, with a fresh installation of the GNS3 server and GUI, we can start creating two templates that will allow us to use these devices in all projects.

The templates we will create are:

- Cisco C3745 - Switch
- Cisco C7200 - Router
- Ubuntu Cloud Server

## Switch: Cisco C3745

-
-
-
-
-
-
-
-
-
-
-
-

-

Now that the template is created, we need to make a few customizations.

1. Open the configuration panel by right-clicking on the template and selecting **"Configure Template"**.

2. Navigate to the **Slot** tab in the C3745 configuration panel and ensure your entries match the image below.

3. The device we just installed is a router. However, since we will be using it as a switch, and to avoid confusion with the C7200 device (which will serve as a router), I recommend modifying the template as shown below:

4. The final step before the switch can be used correctly is linking the appropriate configuration file.
    - Download the file **c3745_startup-config.cfg** and set it in the **General** tab of the template configuration.
    - This configuration file does not depend on the host machine, so you likely won't need to make any changes.

## TIPS:

When connecting this device to others in your topology, follow these guidelines:

- **FastEthernet0/0..1** → These ports are for the routing interface and should not be used.
- **FastEthernet1/0..15** → These are switch ports; you should connect your other devices to these ports.

## Router: Cisco C7200

-
-
-
-
-
-
-
-
-
-

Make sure to adjust your slot setup according to the image below:

-

The final step before the switch can function correctly is to link the appropriate configuration file.

- Download **c7200_startup-config.cfg** and set it in the **General** tab of the template configuration.
-

If you have already configured the external connection, you can test it by pinging an external resource, such as Google DNS.

-

## TIPS:

Pay close attention to the IP address and routing rules in the configuration file. The configuration provided is standard, but if you have (intentionally or unintentionally) made changes to your host machine (e.g., if `virbr0` is not set to `192.168.122.1`), you must modify the configuration accordingly to align with your local setup.

## Ubuntu Cloud Server

For this virtual machine you need a virtual machine management system (a hypervisor). The most well-known opensource is definitely virtualbox. Over the years we have noticed that virtualbox has quite a few compatibility problems with GNS3 (e.g. projects that have stopped working out of nowhere). I strongly recommend using qemu instead of virtualbox, which has proven to be more stable.

-
-
-
-
-

### Network setup

The latest version of Ubuntu Server uses Netplan to manage networking.

Follow these steps to properly configure your machine with a static IP:

1. **Create a `yaml` file** in `/etc/netplan/`.
    - The filename should consist of two decimal digits followed by an alphanumeric string, separated by a hyphen (e.g., `00-eth.yaml`).
2. **Use the following template** and modify it as needed:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    ens3:
      dhcp4: no
      addresses:
 - 10.0.0.20/24
      routes:
 - to: default
          via: 10.0.0.1
      nameservers:
        addresses: [8.8.8.8, 8.8.4.4]
```

- Make sure to set the correct read/write permissions for the file.

```
$ sudo chmod 600 /etc/netplan/00-eth.yaml
```

- Apply the changes using the following command

```
$ sudo netplan apply
```

# NAT: Connect to external

In order to connect to external you need to use a special interface "Cloud1".

If you don't see `virbr0` need to install it with:

```
$ sudo apt install libvirt0 libvirt-clients libvirt-daemon libvirt-dev
```

or if you are on ArchLinux.

```
$ sudo pacman -S libvirt
```

Then start both of these service

```
$ sudo systemctl start libvirtd.service
$ sudo systemctl start virtlogd.service
```

Enable packet forwarding:

```
$ sudo sysctl -w net.ipv4.ip_forward=1
```

Take a look `ip addr` command to find name of your physical network interface.
Maybe this command help you

```
$ ip addr | grep -B3 192 | awk '{ print $2 }' | head -n 1 | sed 's/://'
enp4s0
```

For me, my NIC have the name `enp4s0`. The last step to get Internet access within the GNS3 labs is to configure the NAT. Set the following iptables rules.

```
$ sudo iptables -t nat -A POSTROUTING -o enp4s0 -j MASQUERADE
$ sudo iptables -A FORWARD -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
$ sudo iptables -A FORWARD -i virbr0 -o enp4s0 -j ACCEPT
```

# Exercise: Start a base project

If you have followed all the previous steps correctly, you should now be able to create an example project like this:

## Components in this topology:

- **1 External Link** (Cloud1)
- **1 Switch** (C3745)
- **1 Router** (C7200)
- **2 Ubuntu Servers Victims** (Alice and Bob)
- **1 Ubuntu Servers Attacker** (Darth)

## Configuration Goals:

To complete the setup, ensure the following:
□ All machines (**Ubuntu**) should have internet access.
□ All machines (**Ubuntu**) should be able to **ping** each other within the topology.

If you achieve all these objectives, **congratulations—you're done!** □

**That's all, folks!** □

# Troubleshoot

## I cannot add any resources to any project

If you try to add any resource to your project and encounter a **500 - INTERNAL SERVER ERROR**, it may be due to low disk space.

If your free disk space is **less than 10%**, and you attempt to start `gns3server` in a dedicated shell to monitor the logs while adding a resource, you may see a warning like this:

```
2025-01-12 14:05:06 WARNING gns3server.compute.project_manager:98 Only 5.80% or less of free
disk space detected in "/home/$USER/GNS3/projects/$PROJECTID$" on "$HOST"
```

Then, you may receive an exception from `uvicorn`, indicating that it **cannot find the event loop**.
If you encounter this issue, try the following:

1. **Edit the file** `/usr/lib/python3.13/site-packages/gns3server/compute/project_manager.py`
2. **Comment out line 99** in the file.

This may help resolve the issue and allow you to add resources to your project successfully.

```
    def _check_available_disk_space(self, project):
        """
Sends a warning notification if disk space is getting low.

:param project: project instance
"""

        try:
used_disk_space = psutil.disk_usage(project.path).percent
        except FileNotFoundError:
log.warning(f"Could not find '{project.path}' when checking for used disk space")
            return
        # send a warning if used disk space is >= 90%
        if used_disk_space >= 90:
message = 'Only {:.2f}% or less of free disk space detected in "{}" on "{}"'.format(
                100 - used_disk_space, project.path, platform.node()
)

log.warning(message)
            ### Comment ONLY this line!
            # project.emit("log.warning", {"message": message})
```

save and run again gns3 and gns3server.
I have test this fix on GNS3 v3.0.2 on ArchLinux.

## Libvirtd is installed and running but I can't see virbr0 in GNS3

I found this solution on the GNS3 page in the ArchWiki.

If you have correctly installed **libvirt** and started the **libvirtd** daemon but still do not see the virtual interface in GNS3, it means it has not started correctly. Try starting it manually with:

```
$ sudo virsh net-start default
```

If you want it to start automatically at boot time, run:

```
$ sudo virsh net-autostart default
```

I have tested this fix on GNS3 v3.0.2 on ArchLinux.

## No left space in ubuntu machine

Close all Ubuntu instances and run the following command in a shell.

```
$ qemu-img resize /home/<USER>/GNS3/projects/<PROJECT_UUID>/project-
files/qemu/<MACHINE_UUID>/hda_disk.qcow2 +1G
```

In this case, we have increased the size by only 1G. You can adjust this value according to your needs.
The full path with all variables correctly set can be found by **right-clicking** on the machine you want to expand and selecting **"Show in file manager"**. This will open a window in your default file manager, directly in the folder of interest.