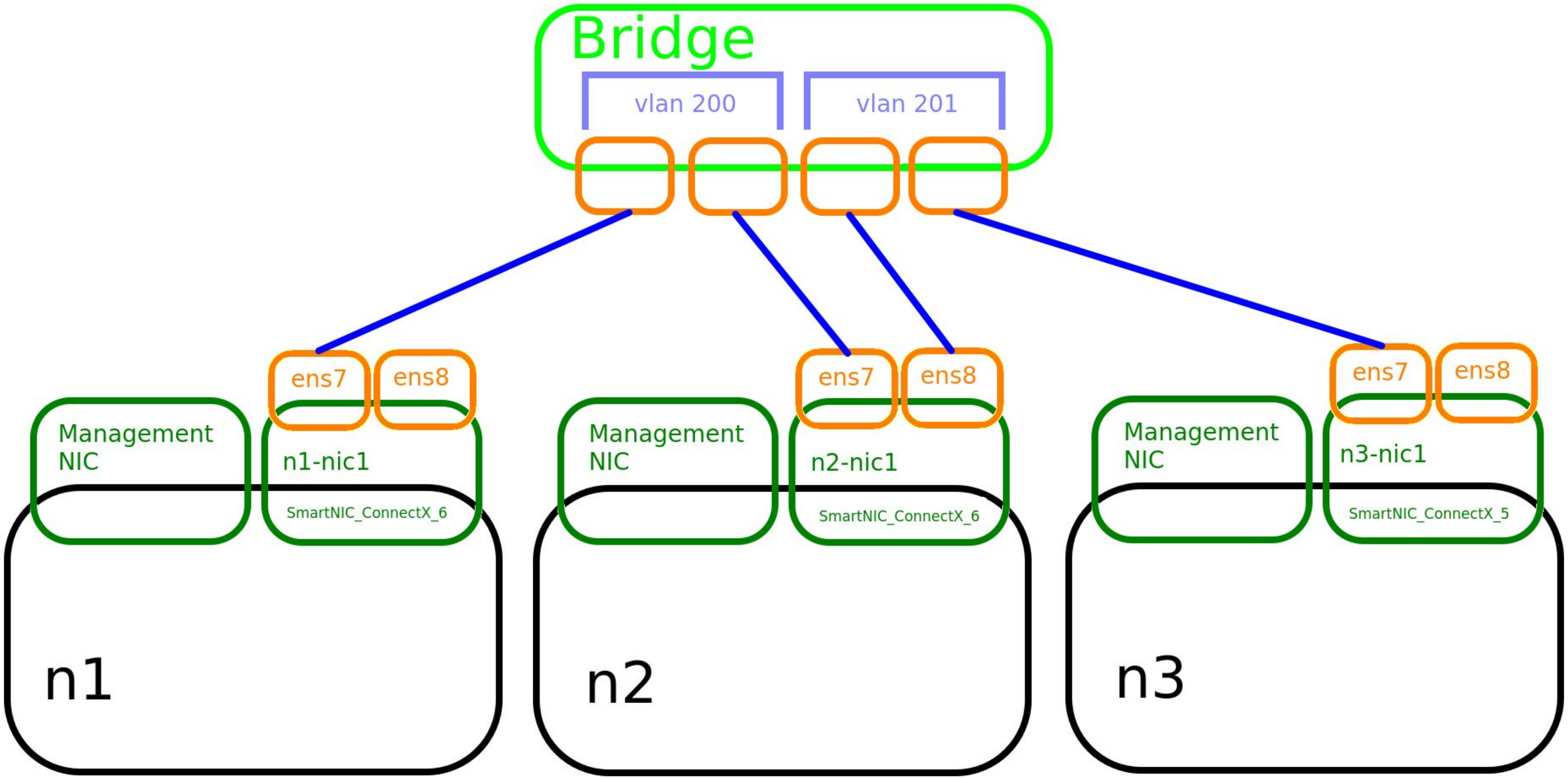
Knit workshop.

We are going to reserve the following topology.


Bridge
vlan 200
vlan 201
ens7
ens8
ens7
ens8
ens7
ens8
Management NIC
n1-nic1
SmartNIC_ConnectX_6
Management NIC
n2-nic1
SmartNIC_ConnectX_6
Management NIC
n3-nic1
SmartNIC_ConnectX_5
n1
n2
n3

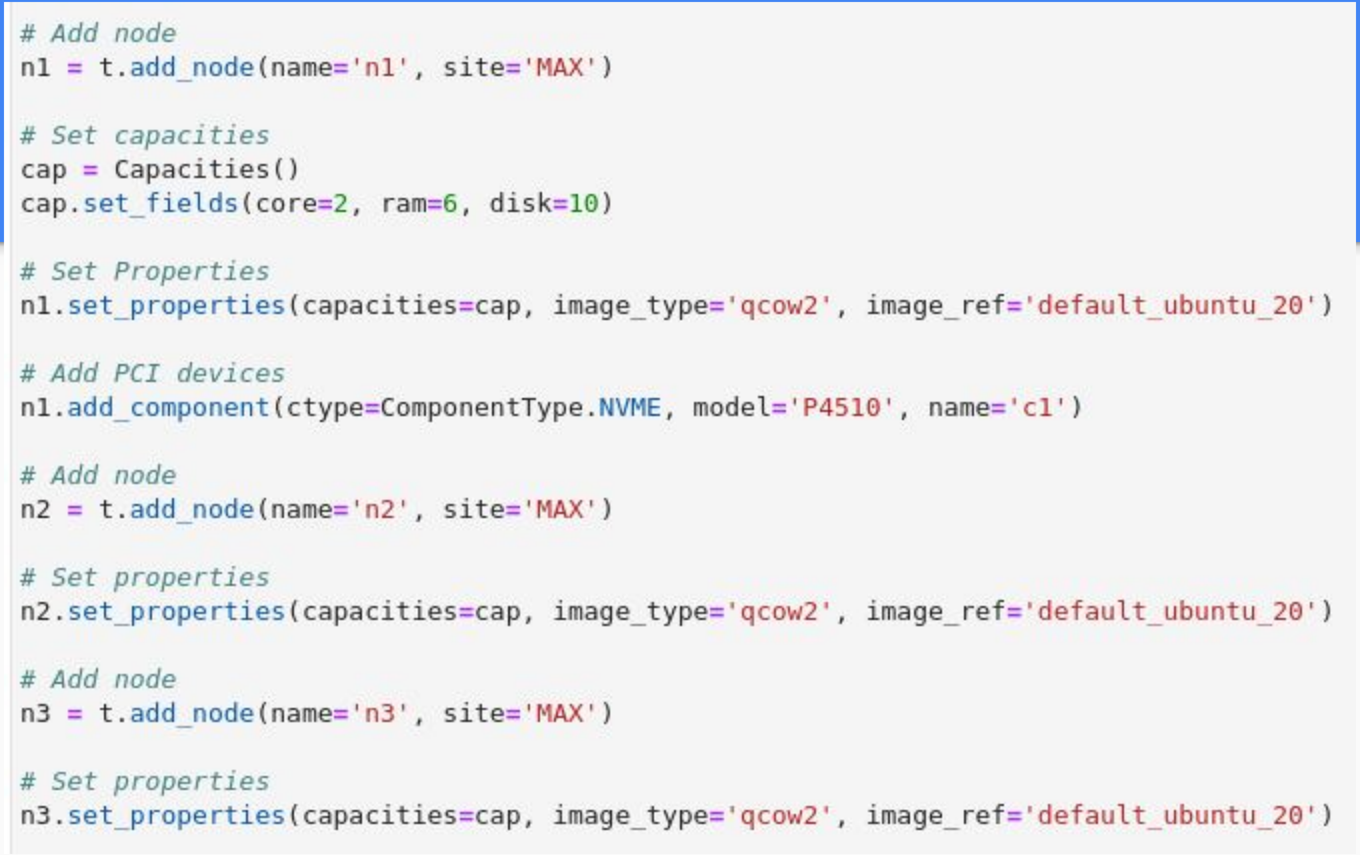
It consists of 3 nodes, each having an extra nic.

The three nics are connected to a bridge.

n1 and n2 nics are isolated with a vlan tag, and n2 and n3 are isolated with another vlan tag.

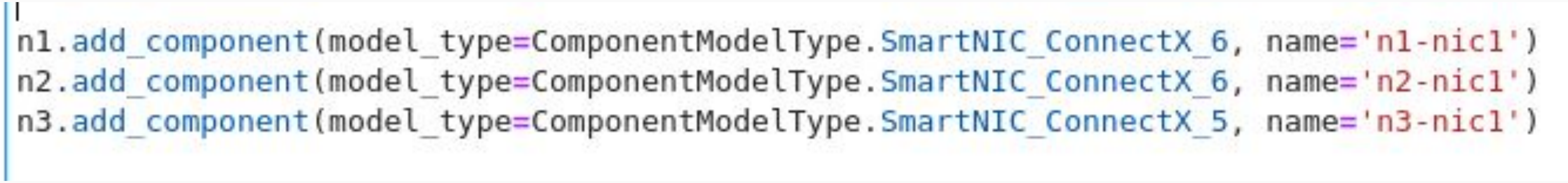
We are going to make n2 act as a router.

First we reserve the nodes.



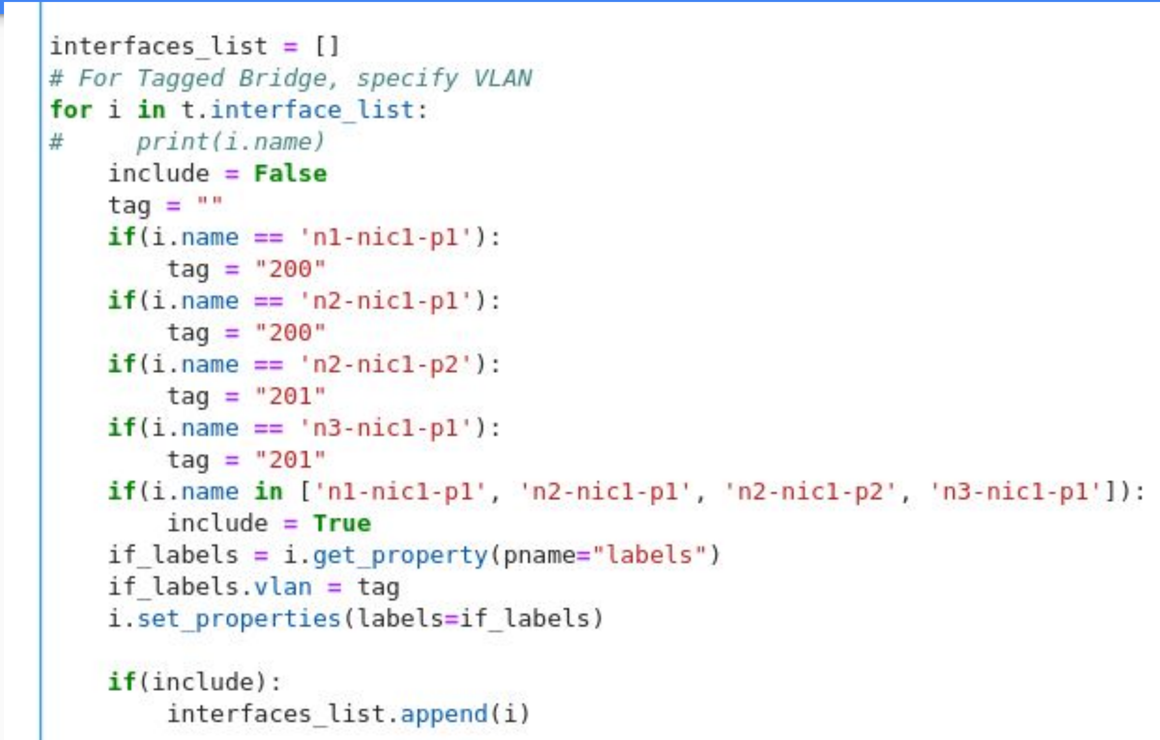
```python
# Add node
n1 = t.add_node(name='n1', site='MAX')

# Set capacities
cap = Capacities()
cap.set_fields(core=2, ram=6, disk=10)

# Set Properties
n1.set_properties(capacities=cap, image_type='qcow2', image_ref='default_ubuntu_20')

# Add PCI devices
n1.add_component(ctype=ComponentType.NVME, model='P4510', name='c1')

# Add node
n2 = t.add_node(name='n2', site='MAX')

# Set properties
n2.set_properties(capacities=cap, image_type='qcow2', image_ref='default_ubuntu_20')

# Add node
n3 = t.add_node(name='n3', site='MAX')

# Set properties
n3.set_properties(capacities=cap, image_type='qcow2', image_ref='default_ubuntu_20')
```

Then we add nics to them.



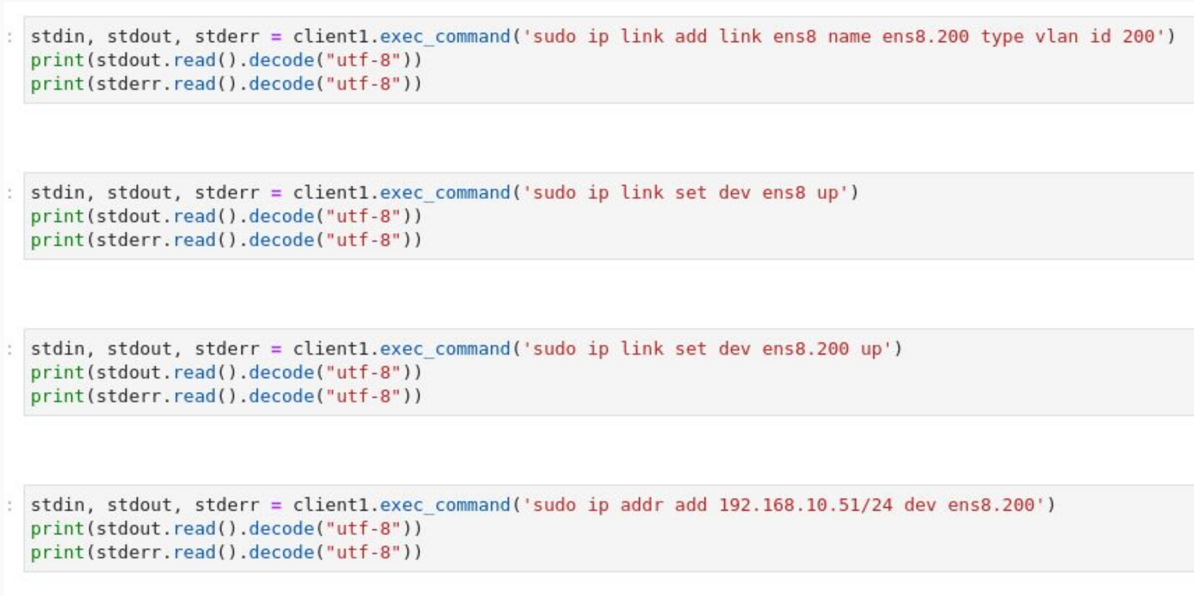
```
n1.add_component(model_type=ComponentModelType.SmartNIC_ConnectX_6, name='n1-nic1')
n2.add_component(model_type=ComponentModelType.SmartNIC_ConnectX_6, name='n2-nic1')
n3.add_component(model_type=ComponentModelType.SmartNIC_ConnectX_5, name='n3-nic1')
```

And we specify the vlan tags



```
interfaces_list = []
# For Tagged Bridge, specify VLAN
for i in t.interface_list:
#     print(i.name)
    include = False
    tag = ""
    if(i.name == 'n1-nic1-p1'):
        tag = "200"
    if(i.name == 'n2-nic1-p1'):
        tag = "200"
    if(i.name == 'n2-nic1-p2'):
        tag = "201"
    if(i.name == 'n3-nic1-p1'):
        tag = "201"
    if(i.name in ['n1-nic1-p1', 'n2-nic1-p1', 'n2-nic1-p2', 'n3-nic1-p1']):
        include = True
    if_labels = i.get_property(pname="labels")
    if_labels.vlan = tag
    i.set_properties(labels=if_labels)

    if(include):
        interfaces_list.append(i)
```

We can then configure the NICs. We give them IPs.



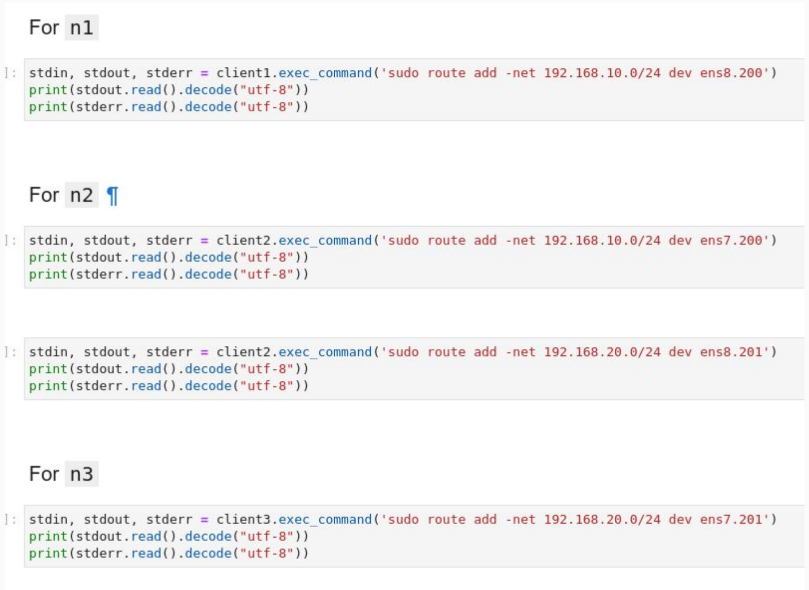
```
stdin, stdout, stderr = client1.exec_command('sudo ip link add link ens8 name ens8.200 type vlan id 200')
print(stdout.read().decode("utf-8"))
print(stderr.read().decode("utf-8"))
```

```
stdin, stdout, stderr = client1.exec_command('sudo ip link set dev ens8 up')
print(stdout.read().decode("utf-8"))
print(stderr.read().decode("utf-8"))
```

```
stdin, stdout, stderr = client1.exec_command('sudo ip link set dev ens8.200 up')
print(stdout.read().decode("utf-8"))
print(stderr.read().decode("utf-8"))
```

```
stdin, stdout, stderr = client1.exec_command('sudo ip addr add 192.168.10.51/24 dev ens8.200')
print(stdout.read().decode("utf-8"))
print(stderr.read().decode("utf-8"))
```

Then we add routes between those IPs we’ve added.



For n1

```
]: stdin, stdout, stderr = client1.exec_command('sudo route add -net 192.168.10.0/24 dev ens8.200')
   print(stdout.read().decode("utf-8"))
   print(stderr.read().decode("utf-8"))
```

For n2 ¶

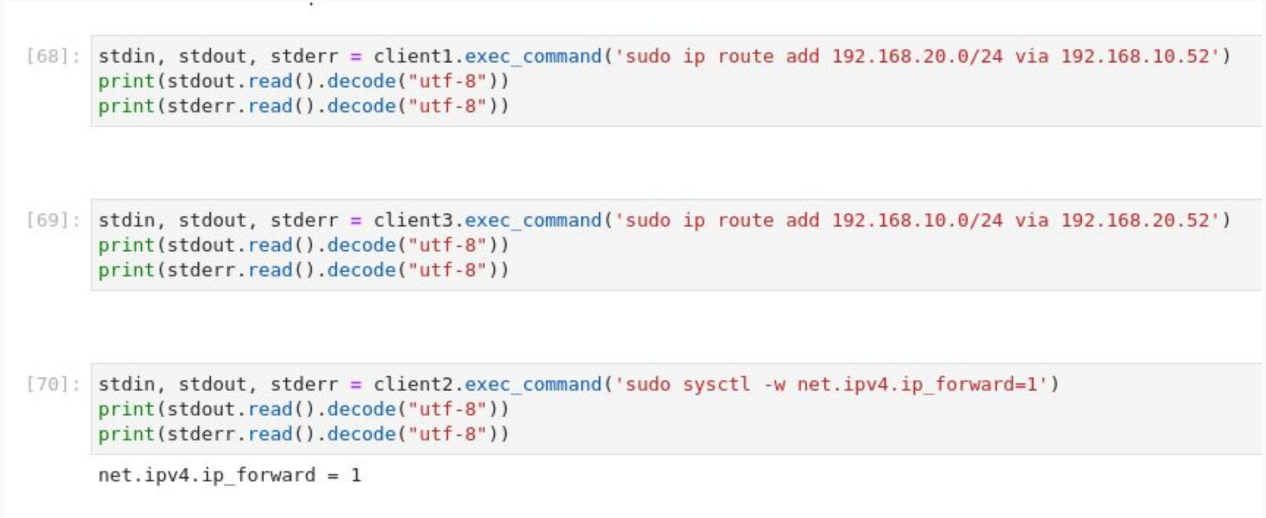
```
]: stdin, stdout, stderr = client2.exec_command('sudo route add -net 192.168.10.0/24 dev ens7.200')
   print(stdout.read().decode("utf-8"))
   print(stderr.read().decode("utf-8"))
```

```
]: stdin, stdout, stderr = client2.exec_command('sudo route add -net 192.168.20.0/24 dev ens8.201')
   print(stdout.read().decode("utf-8"))
   print(stderr.read().decode("utf-8"))
```

For n3

```
]: stdin, stdout, stderr = client3.exec_command('sudo route add -net 192.168.20.0/24 dev ens7.201')
   print(stdout.read().decode("utf-8"))
   print(stderr.read().decode("utf-8"))
```

Then, to make n2 act as a router, we add routes to it that forward traffic back and forth between n1 and n3, and enable packet forwarding.

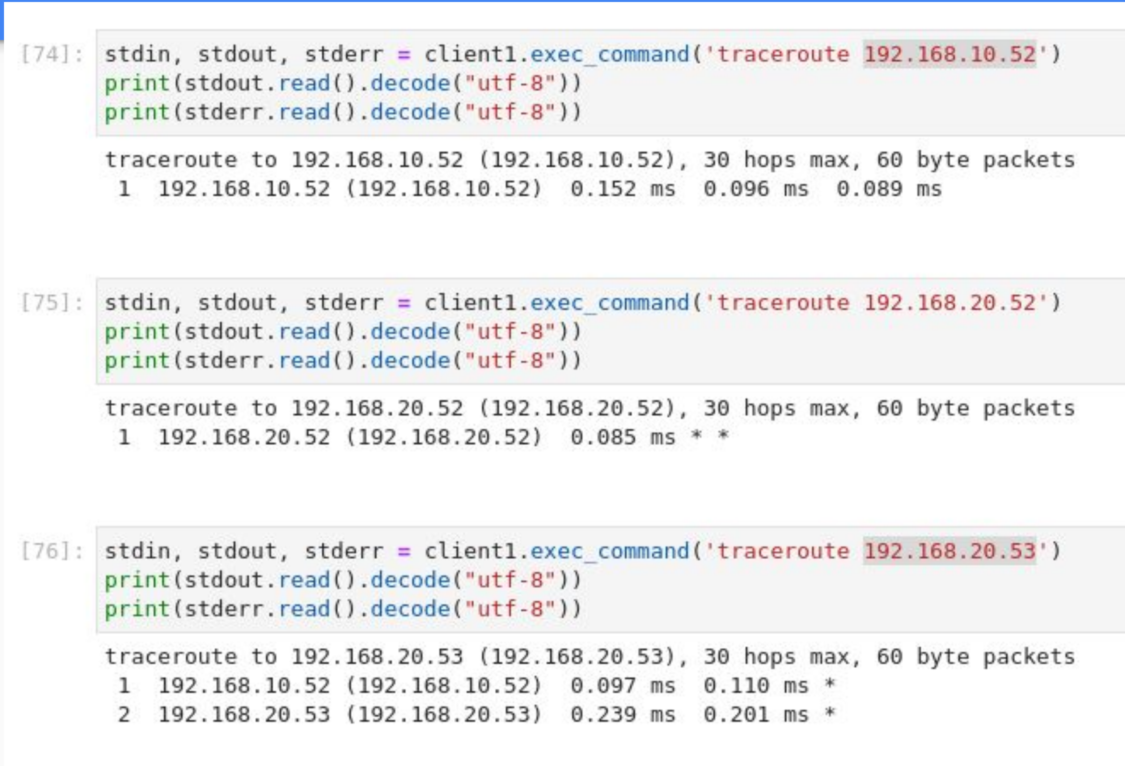


```
[68]: stdin, stdout, stderr = client1.exec_command('sudo ip route add 192.168.20.0/24 via 192.168.10.52')
      print(stdout.read().decode("utf-8"))
      print(stderr.read().decode("utf-8"))
```

```
[69]: stdin, stdout, stderr = client3.exec_command('sudo ip route add 192.168.10.0/24 via 192.168.20.52')
      print(stdout.read().decode("utf-8"))
      print(stderr.read().decode("utf-8"))
```

```
[70]: stdin, stdout, stderr = client2.exec_command('sudo sysctl -w net.ipv4.ip_forward=1')
      print(stdout.read().decode("utf-8"))
      print(stderr.read().decode("utf-8"))

      net.ipv4.ip_forward = 1
```

Now everything is configured. We can do a traceroute to see the paths packets take. Now the nodes can ping each other.

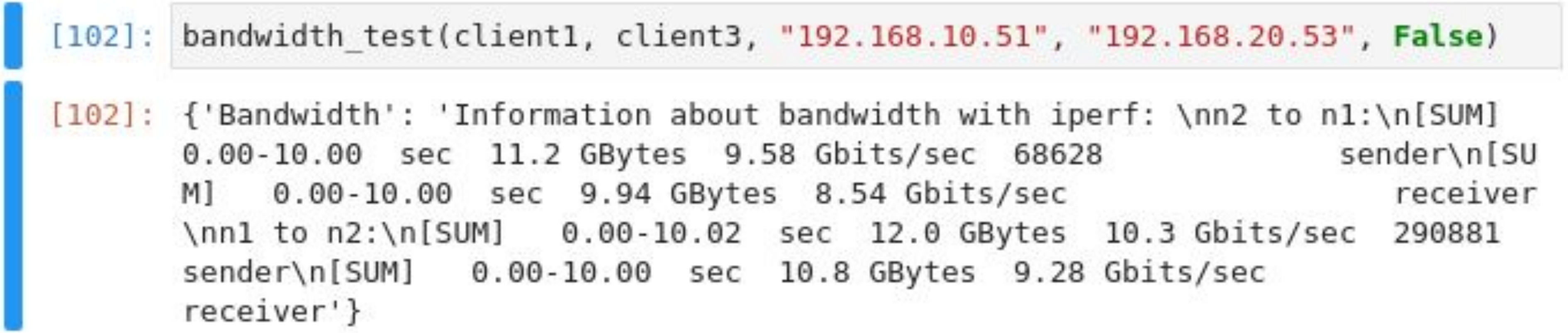


```
[74]: stdin, stdout, stderr = client1.exec_command('traceroute 192.168.10.52')
      print(stdout.read().decode("utf-8"))
      print(stderr.read().decode("utf-8"))

      traceroute to 192.168.10.52 (192.168.10.52), 30 hops max, 60 byte packets
       1  192.168.10.52 (192.168.10.52)  0.152 ms  0.096 ms  0.089 ms
```

```
[75]: stdin, stdout, stderr = client1.exec_command('traceroute 192.168.20.52')
      print(stdout.read().decode("utf-8"))
      print(stderr.read().decode("utf-8"))

      traceroute to 192.168.20.52 (192.168.20.52), 30 hops max, 60 byte packets
       1  192.168.20.52 (192.168.20.52)  0.085 ms * *
```

```
[76]: stdin, stdout, stderr = client1.exec_command('traceroute 192.168.20.53')
      print(stdout.read().decode("utf-8"))
      print(stderr.read().decode("utf-8"))

      traceroute to 192.168.20.53 (192.168.20.53), 30 hops max, 60 byte packets
       1  192.168.10.52 (192.168.10.52)  0.097 ms  0.110 ms *
       2  192.168.20.53 (192.168.20.53)  0.239 ms  0.201 ms *
```

We can also do a bandwidth test using iperf3. We have a code function that we can use.



```
[102]: bandwidth_test(client1, client3, "192.168.10.51", "192.168.20.53", False)

[102]: {'Bandwidth': 'Information about bandwidth with iperf: \nn2 to n1:\n[SUM]
       0.00-10.00  sec  11.2 GBytes  9.58 Gbits/sec  68628             sender\n[SU
       M]   0.00-10.00  sec  9.94 GBytes  8.54 Gbits/sec                  receiver
       \nn1 to n2:\n[SUM]   0.00-10.02  sec  12.0 GBytes  10.3 Gbits/sec  290881
       sender\n[SUM]   0.00-10.00  sec  10.8 GBytes  9.28 Gbits/sec
       receiver'}
```