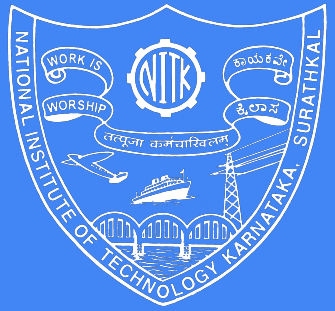


Emulating CoAP with Network Namespaces and NeST

Mohit P. Tahiliani

Wireless Information Networking Group (WiNG),

Dept. of CSE, National Institute of Technology Karnataka, Surathkal

tahiliani@nitk.edu.in

7th April 2022

Credits: Leslie Monis and Gautam Ramakrishnan, NITK Surathkal

Agenda

1. Installation of libcoap library in Ubuntu 20.04
2. Basic example with CoAP Client and Server
3. Create network namespaces
4. Create virtual ethernet (veth) pair to connect network namespaces
5. Attach ‘veth’ endpoints to network namespaces
6. Assign IP addresses to ‘veth’ interfaces
7. Ping from one network namespace to another
8. Run an iperf test between two network namespaces
9. Configure an asymmetric link using the traffic control (tc) subsystem of Linux
10. Example with CoAP Client and Server

Installation of libcoap library in Ubuntu 20.04

Install the following packages:

```
$ sudo apt install git build-essential net-tools asciidoc docbook doxygen
libssl-dev gnutls-bin openssl autoconf automake pkg-config libtool
```

Download libcoap via git

```
$ git clone https://github.com/obgm/libcoap
```

Install libcoap (run the following commands one-by-one)

```
$ cd libcoap, ./autogen.sh, ./configure, make, sudo make install
```

Basic example with CoAP Client and Server

Open a terminal, go to libcoap/examples/ directory and run the Server

```
$ ./coap-server
```

Open a new terminal, go to libcoap/examples/ directory and run the Client

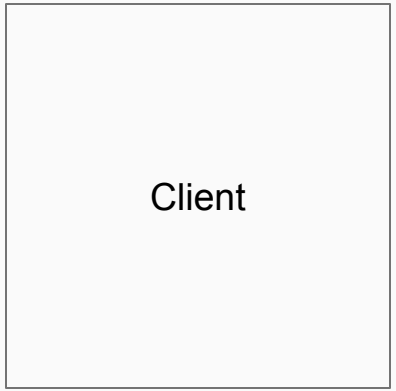
```
$ ./coap-client -m get coap://[::1]/.well-known/core
```

Expected Output

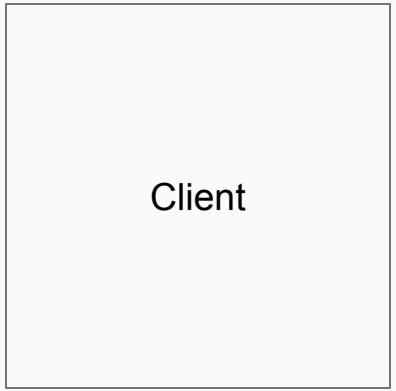
```
$ </>;title="General
Info";ct=0,</time>;if="clock";rt="ticks";title="Internal
Clock";ct=0;obs,</async>;ct=0,</example_data>;title="Example
Data";ct=0;obs
```

Topology setup with network namespaces: Create two nodes connected via a direct link

```
# ip netns add client
```



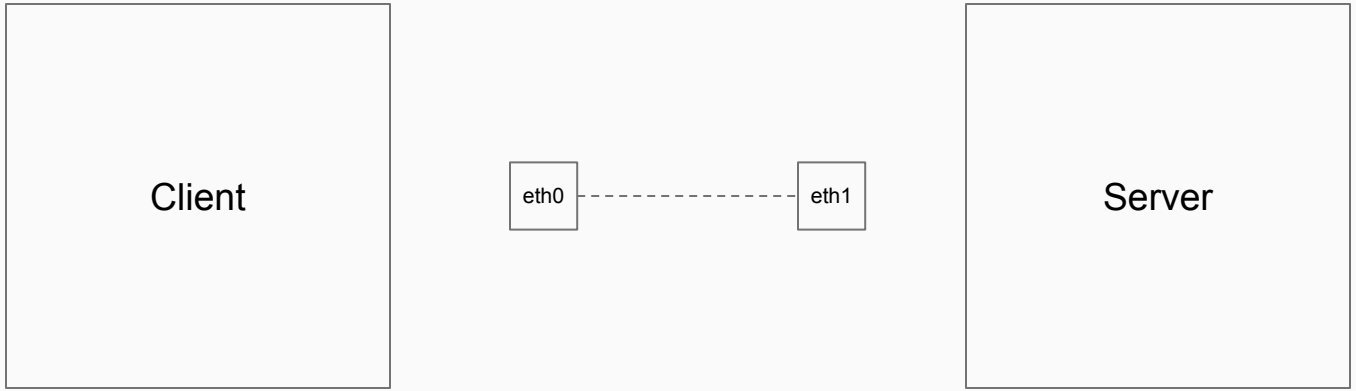

```
# ip netns add client
#
```



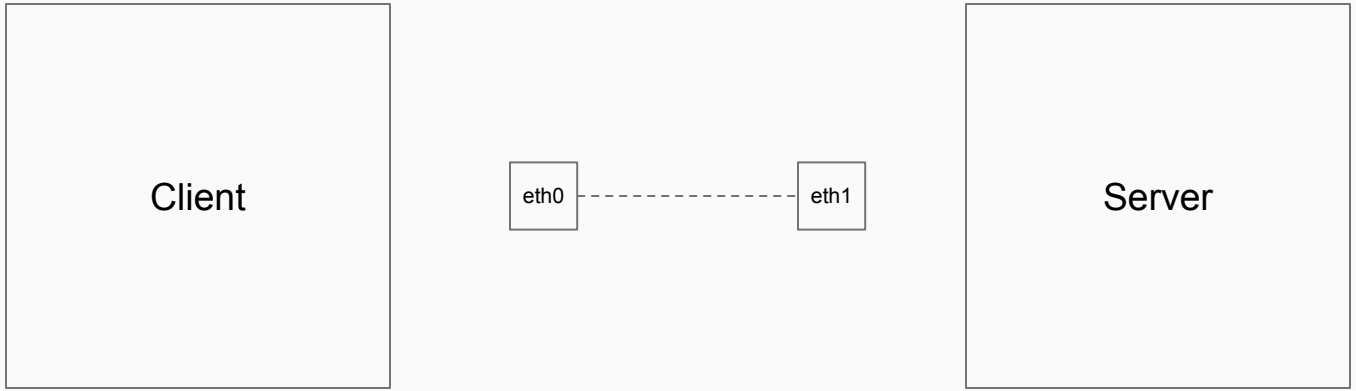

```
# ip netns add client
# ip netns add server
```


```
# ip netns add client
# ip netns add server
#
```

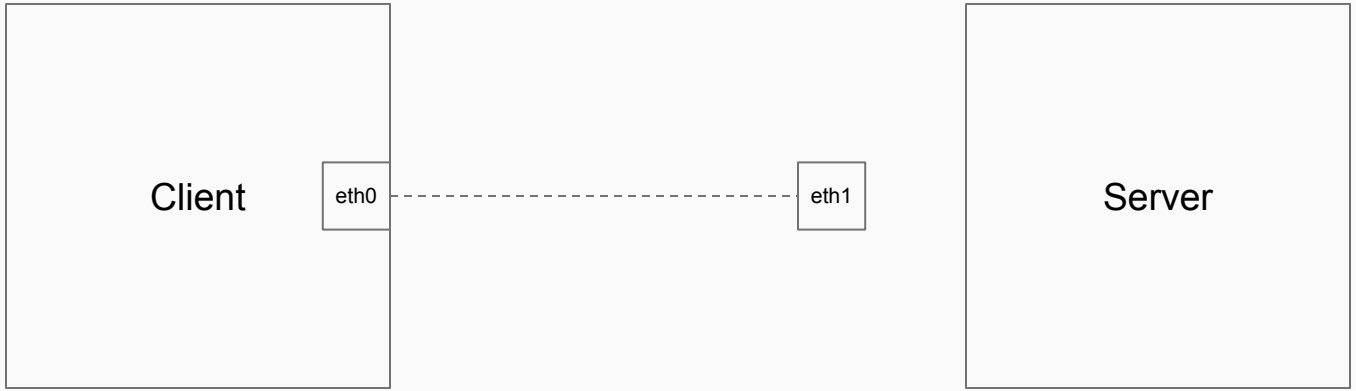

```
# ip netns add client
# ip netns add server
# ip link add eth0 type veth peer name eth1
```



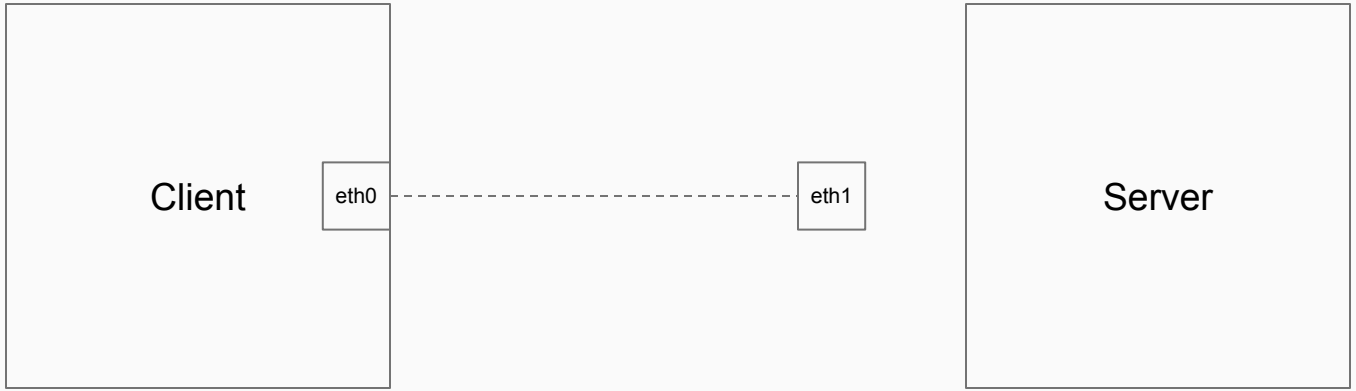

```
# ip netns add client
# ip netns add server
# ip link add eth0 type veth peer name eth1
#
```



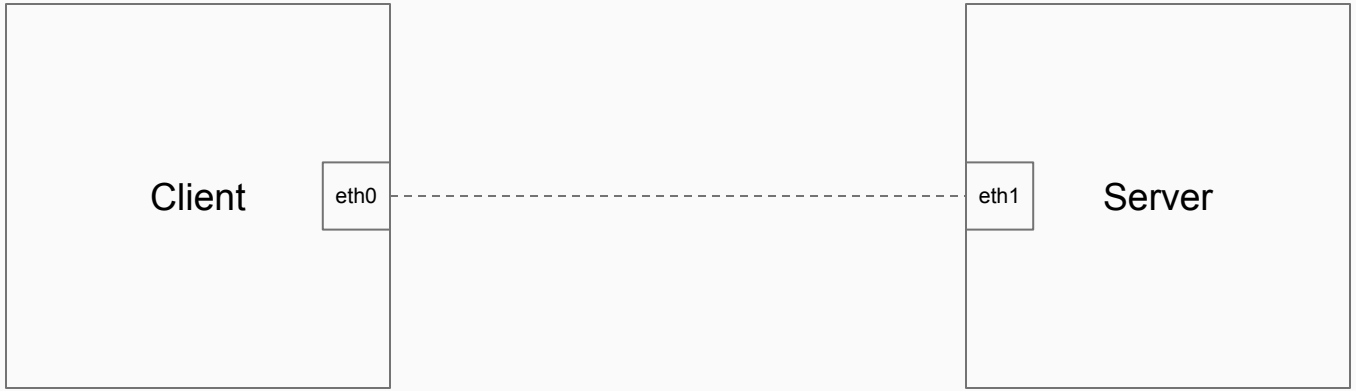

```
# ip netns add client
# ip netns add server
# ip link add eth0 type veth peer name eth1
# ip link set eth0 netns client
```



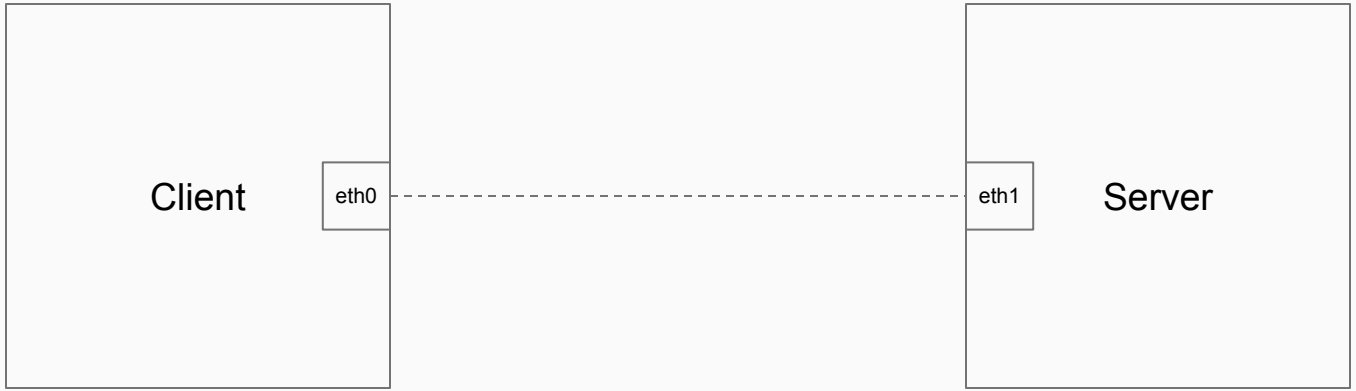

```
# ip netns add client
# ip netns add server
# ip link add eth0 type veth peer name eth1
# ip link set eth0 netns client
#
```



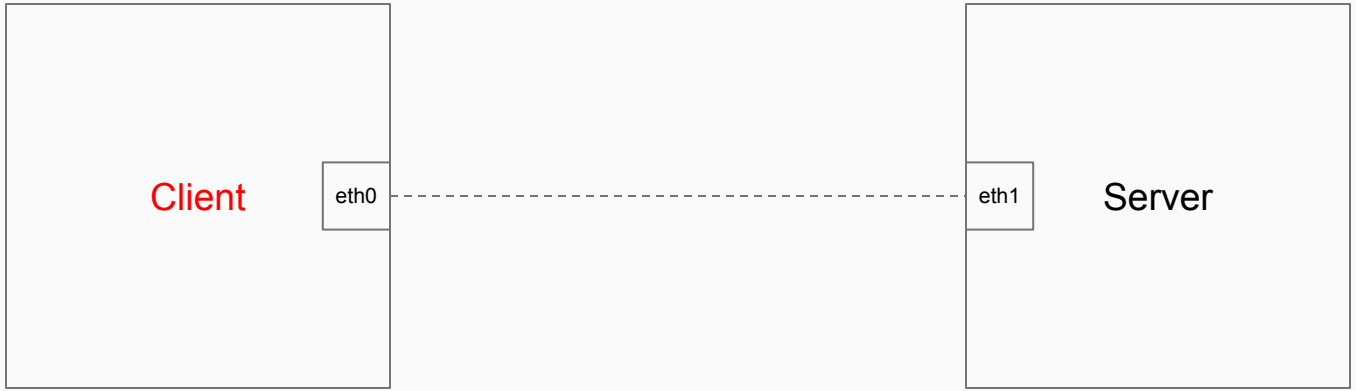

```
# ip netns add client
# ip netns add server
# ip link add eth0 type veth peer name eth1
# ip link set eth0 netns client
# ip link set eth1 netns server
```



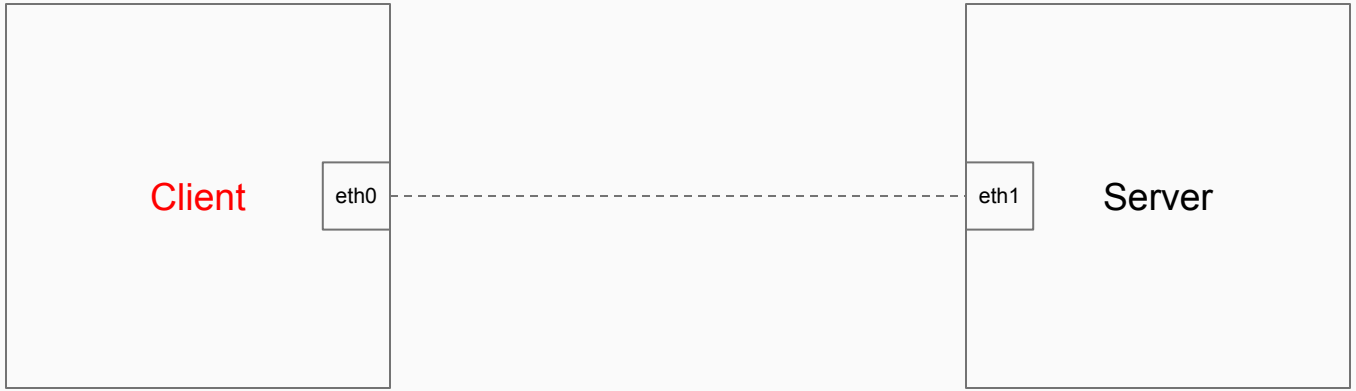

```
# ip netns add client
# ip netns add server
# ip link add eth0 type veth peer name eth1
# ip link set eth0 netns client
# ip link set eth1 netns server
#
```



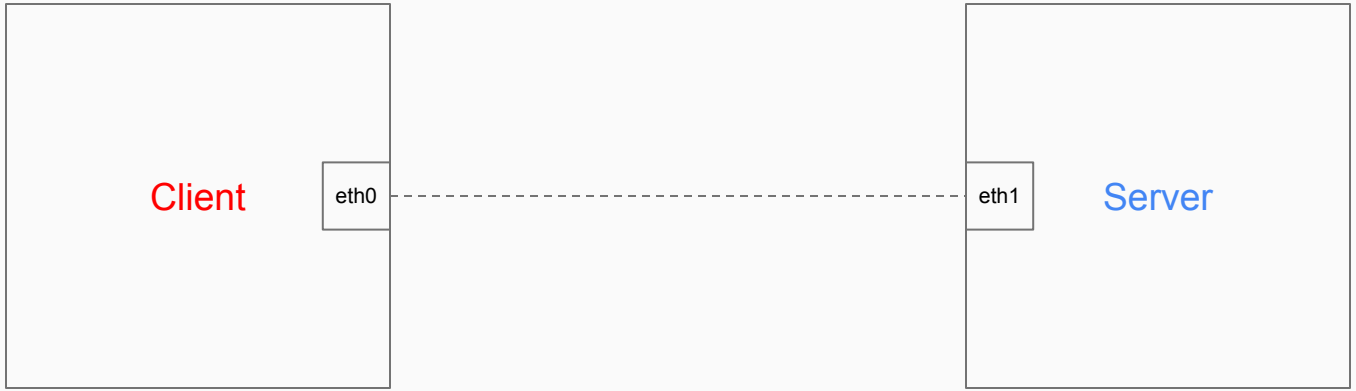

```
# ip netns add client
# ip netns add server
# ip link add eth0 type veth peer name eth1
# ip link set eth0 netns client
# ip link set eth1 netns server
# ip netns exec client ip link set lo up
```



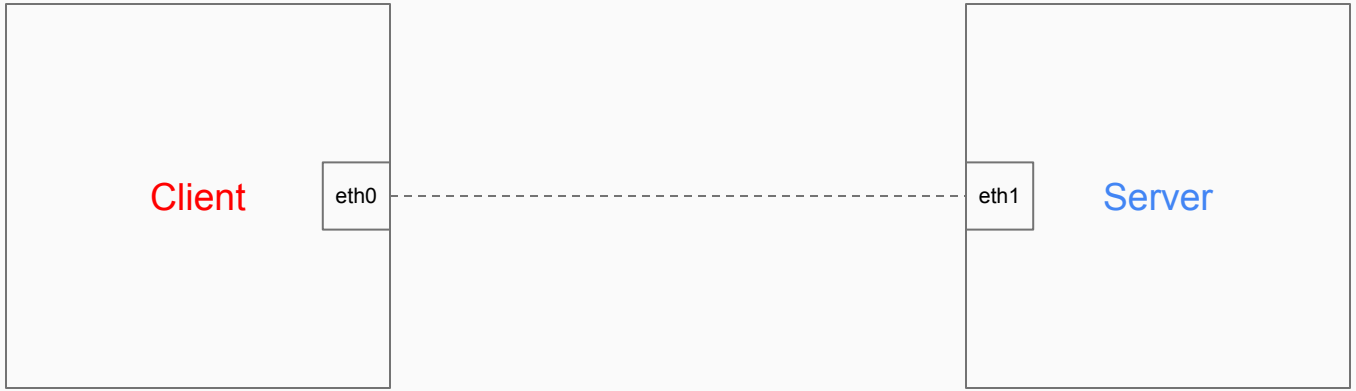

```
# ip netns add client
# ip netns add server
# ip link add eth0 type veth peer name eth1
# ip link set eth0 netns client
# ip link set eth1 netns server
# ip netns exec client ip link set lo up
#
```



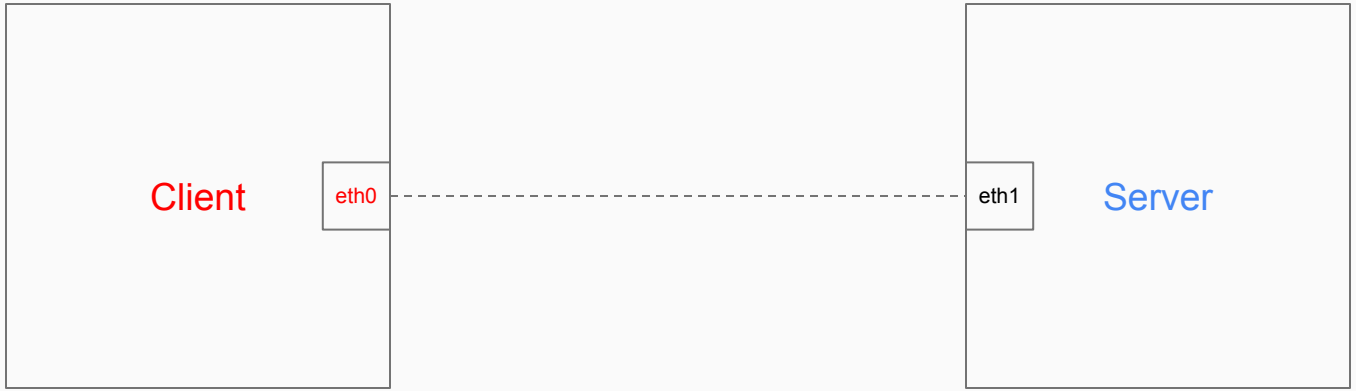

```
# ip netns add client
# ip netns add server
# ip link add eth0 type veth peer name eth1
# ip link set eth0 netns client
# ip link set eth1 netns server
# ip netns exec client ip link set lo up
# ip netns exec server ip link set lo up
```



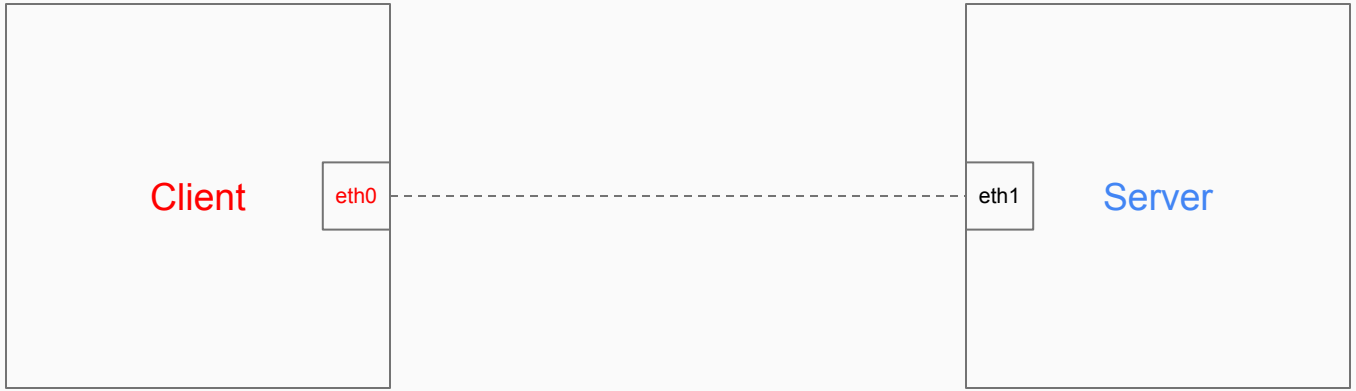

```
# ip netns add client
# ip netns add server
# ip link add eth0 type veth peer name eth1
# ip link set eth0 netns client
# ip link set eth1 netns server
# ip netns exec client ip link set lo up
# ip netns exec server ip link set lo up
#
```



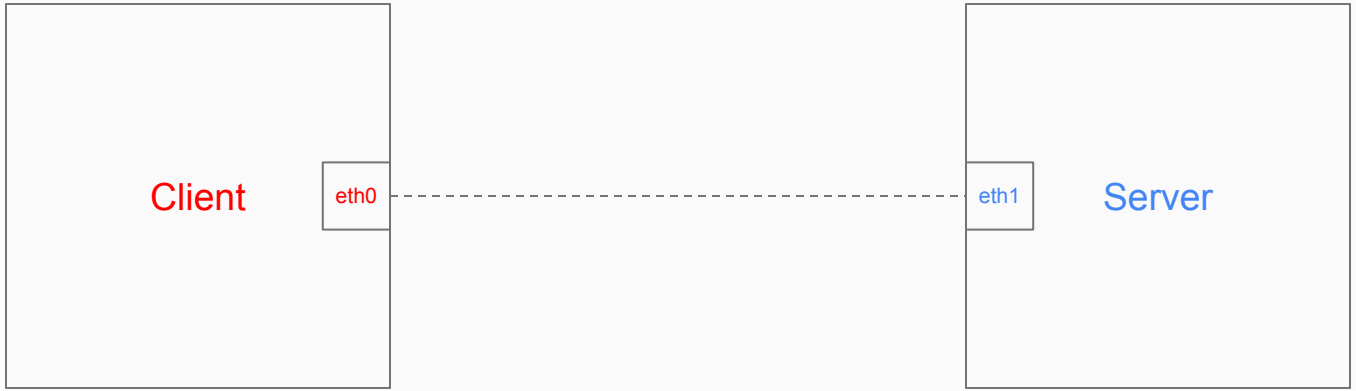

```
# ip netns add client
# ip netns add server
# ip link add eth0 type veth peer name eth1
# ip link set eth0 netns client
# ip link set eth1 netns server
# ip netns exec client ip link set lo up
# ip netns exec server ip link set lo up
# ip netns exec client ip link set eth0 up
```



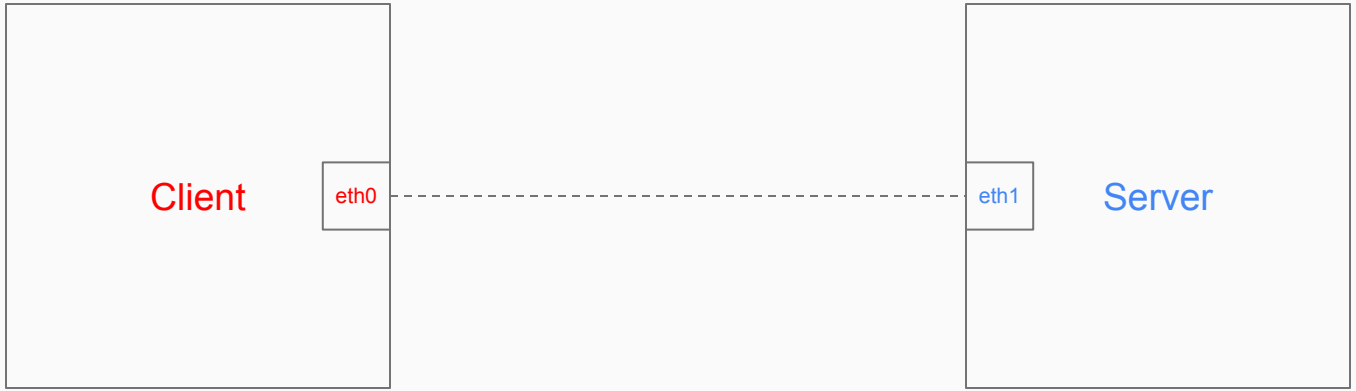

```
# ip netns add server
# ip link add eth0 type veth peer name eth1
# ip link set eth0 netns client
# ip link set eth1 netns server
# ip netns exec client ip link set lo up
# ip netns exec server ip link set lo up
# ip netns exec client ip link set eth0 up
#
```



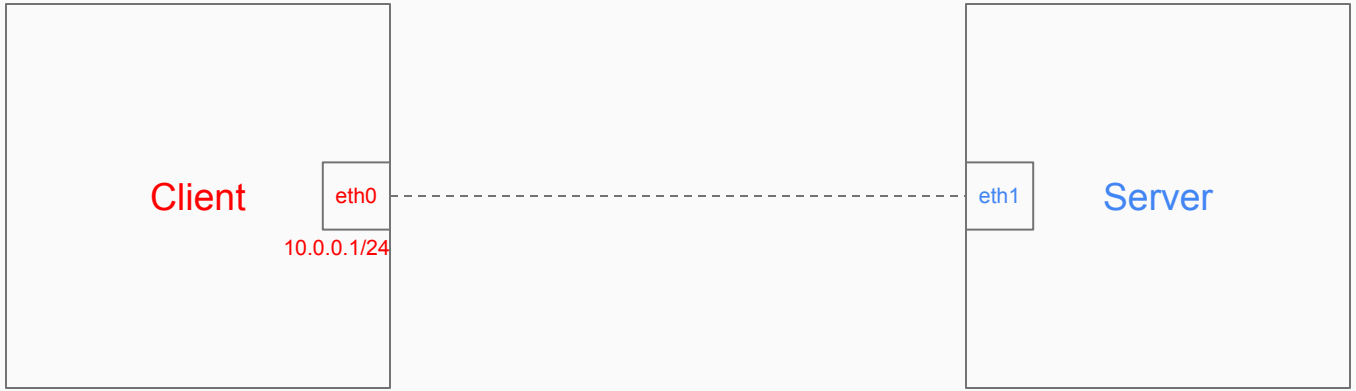

```
# ip netns add server
# ip link add eth0 type veth peer name eth1
# ip link set eth0 netns client
# ip link set eth1 netns server
# ip netns exec client ip link set lo up
# ip netns exec server ip link set lo up
# ip netns exec client ip link set eth0 up
# ip netns exec server ip link set eth1 up
```



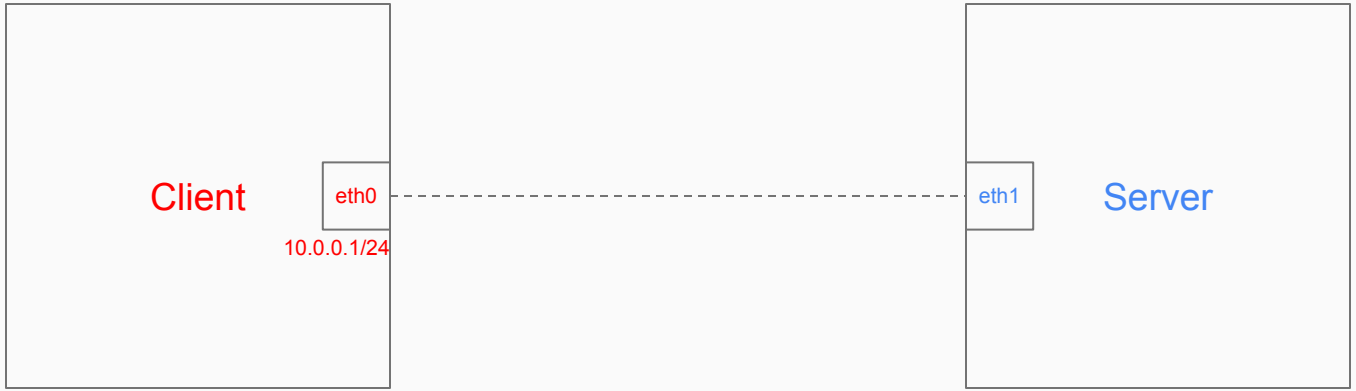

```
# ip link add eth0 type veth peer name eth1
# ip link set eth0 netns client
# ip link set eth1 netns server
# ip netns exec client ip link set lo up
# ip netns exec server ip link set lo up
# ip netns exec client ip link set eth0 up
# ip netns exec server ip link set eth1 up
#
```



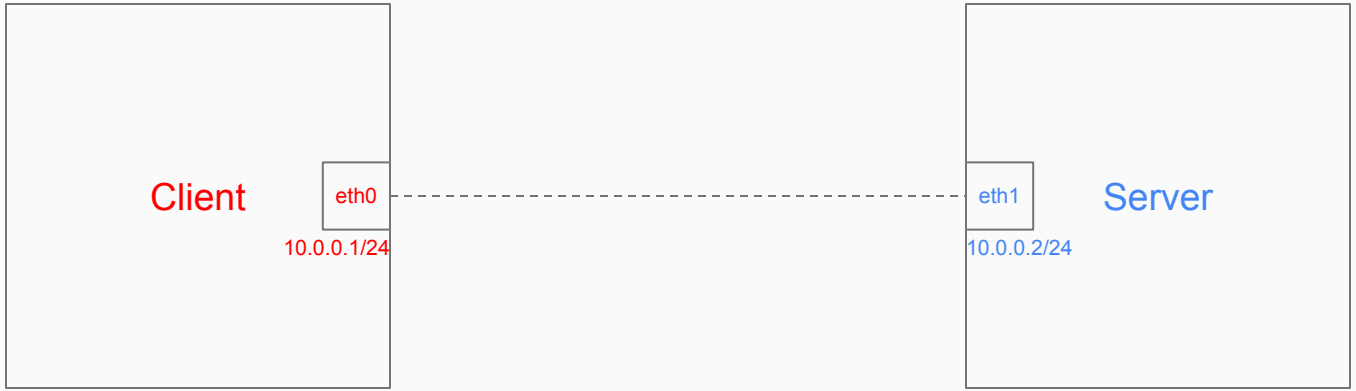

```
# ip link add eth0 type veth peer name eth1
# ip link set eth0 netns client
# ip link set eth1 netns server
# ip netns exec client ip link set lo up
# ip netns exec server ip link set lo up
# ip netns exec client ip link set eth0 up
# ip netns exec server ip link set eth1 up
# ip netns exec client ip address add 10.0.0.1/24 dev eth0
```



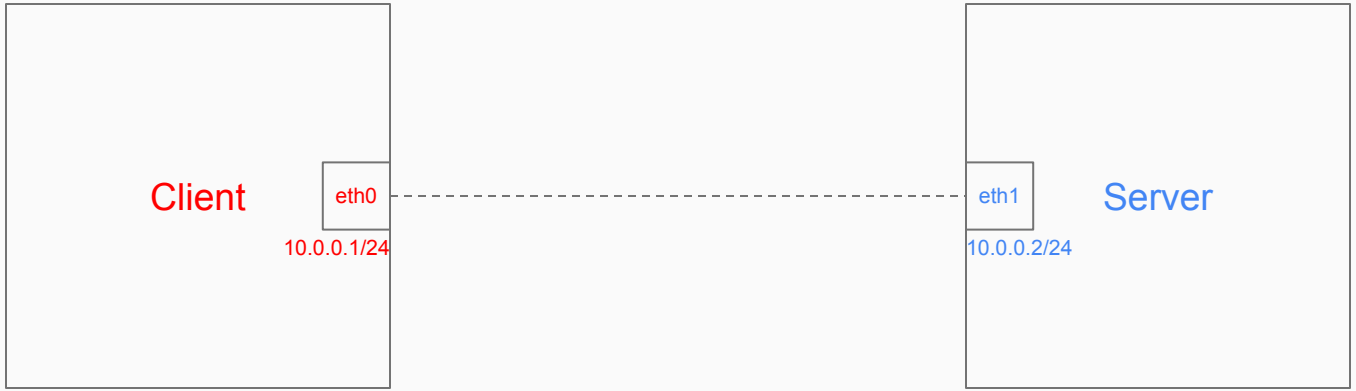

```
# ip link set eth0 netns client
# ip link set eth1 netns server
# ip netns exec client ip link set lo up
# ip netns exec server ip link set lo up
# ip netns exec client ip link set eth0 up
# ip netns exec server ip link set eth1 up
# ip netns exec client ip address add 10.0.0.1/24 dev eth0
#
```




```
# ip link set eth0 netns client
# ip link set eth1 netns server
# ip netns exec client ip link set lo up
# ip netns exec server ip link set lo up
# ip netns exec client ip link set eth0 up
# ip netns exec server ip link set eth1 up
# ip netns exec client ip address add 10.0.0.1/24 dev eth0
# ip netns exec server ip address add 10.0.0.2/24 dev eth1
```

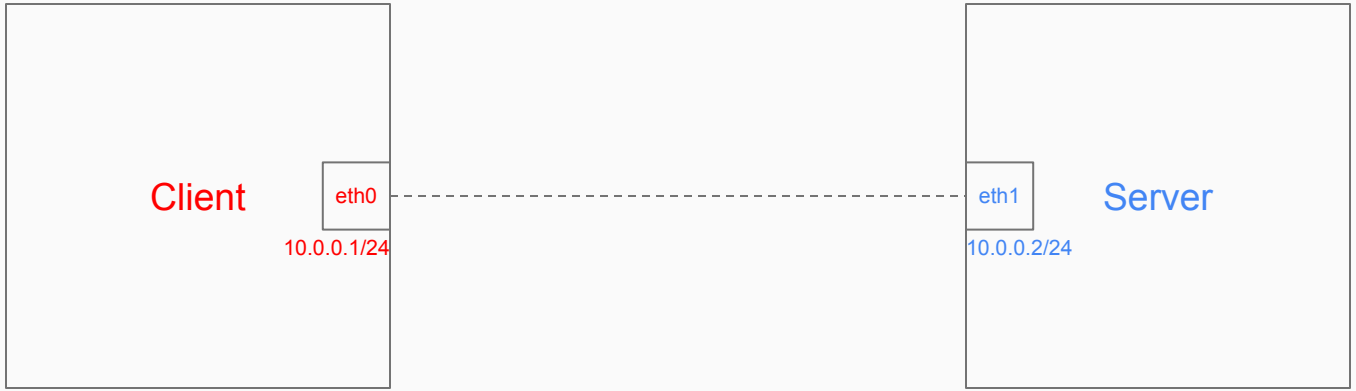



```
# ip link set eth1 netns server
# ip netns exec client ip link set lo up
# ip netns exec server ip link set lo up
# ip netns exec client ip link set eth0 up
# ip netns exec server ip link set eth1 up
# ip netns exec client ip address add 10.0.0.1/24 dev eth0
# ip netns exec server ip address add 10.0.0.2/24 dev eth1
#
```


Client
eth0
10.0.0.1/24
eth1
Server
10.0.0.2/24

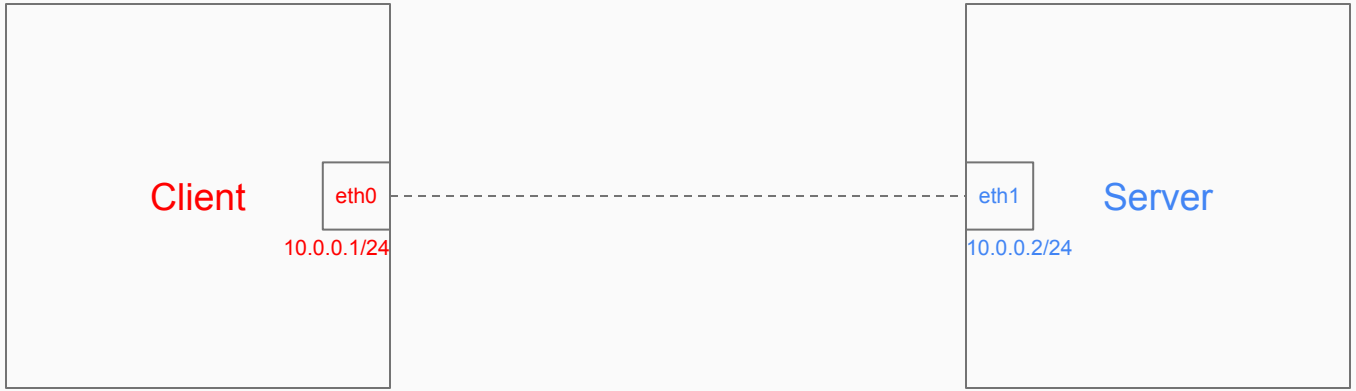
```
#
```

```
#
```



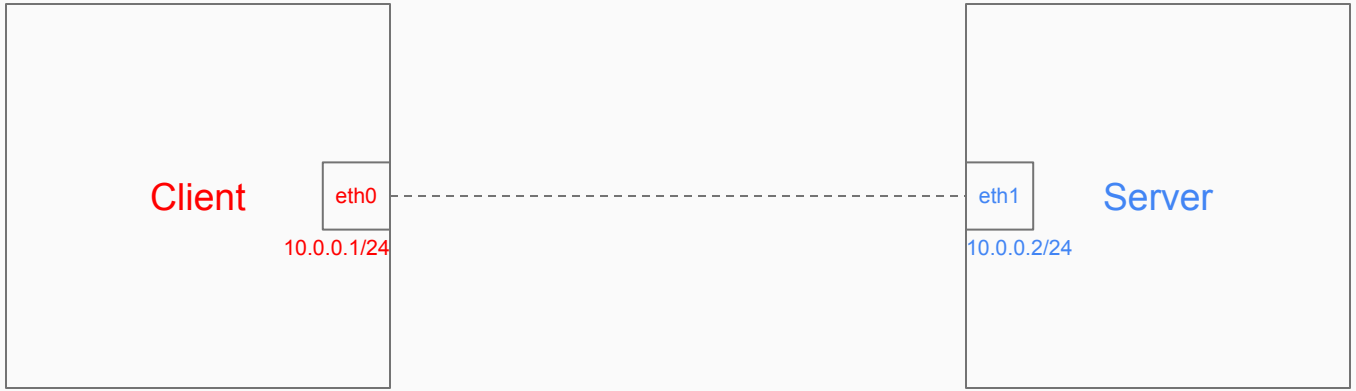

```
# ip netns exec client bash
```

```
# ip netns exec server bash
```



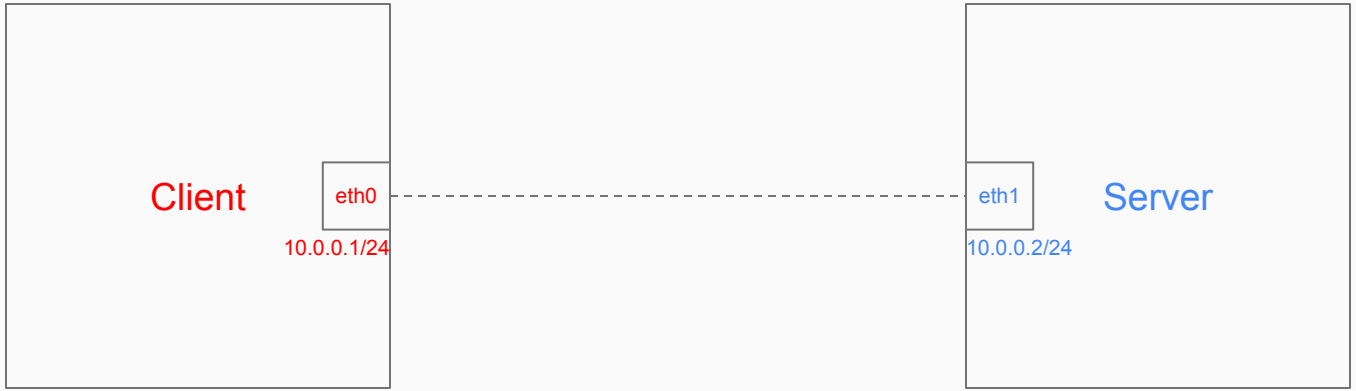

```
# ip netns exec client bash
#
```

```
# ip netns exec server bash
#
```



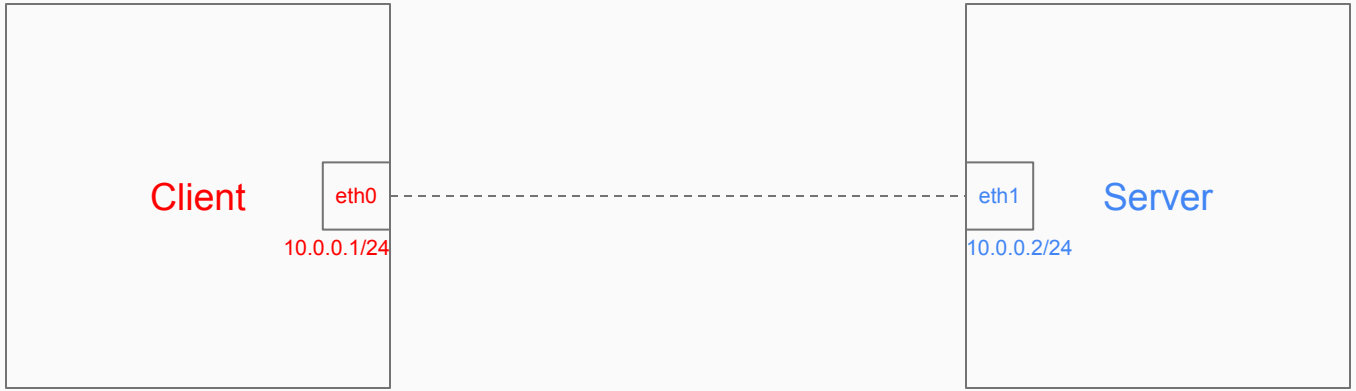

```
# ip netns exec client bash
# ip address
```

```
# ip netns exec server bash
# ip address
```



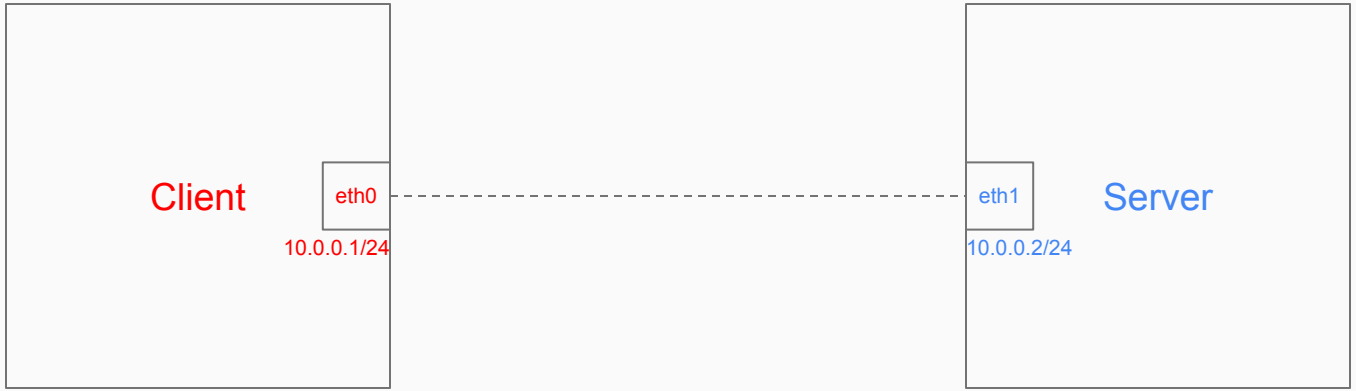

```
# ip netns exec client bash
# ip address
<output>
#
```

```
# ip netns exec server bash
# ip address
<output>
#
```



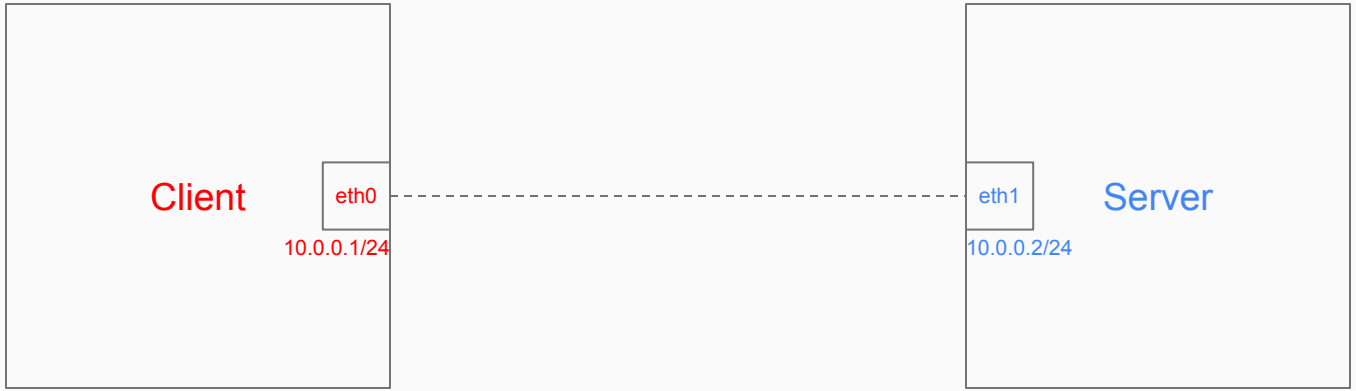

```
# ip netns exec client bash
# ip address
<output>
# ping 10.0.0.2
```

```
# ip netns exec server bash
# ip address
<output>
#
```



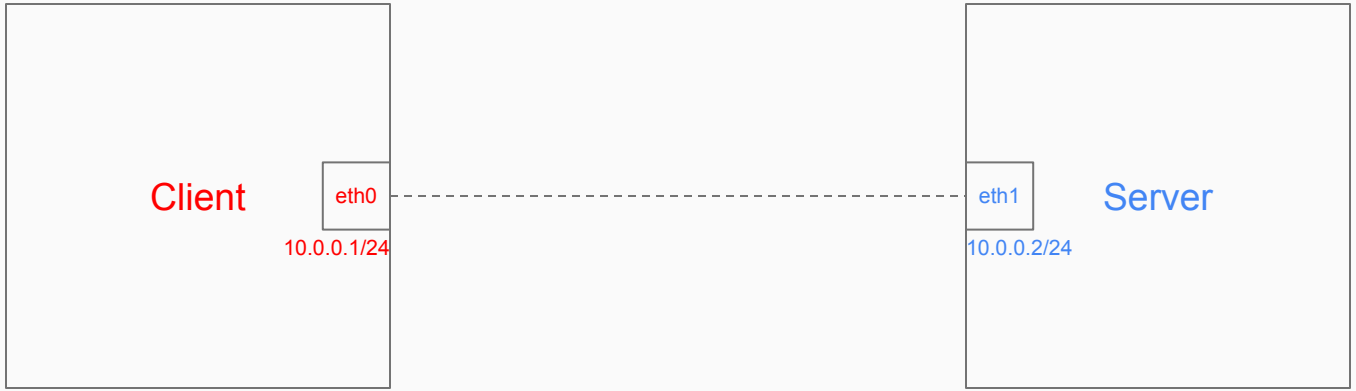

```
# ip netns exec client bash
# ip address
<output>
# ping 10.0.0.2
<output>
```

```
# ip netns exec server bash
# ip address
<output>
#
```



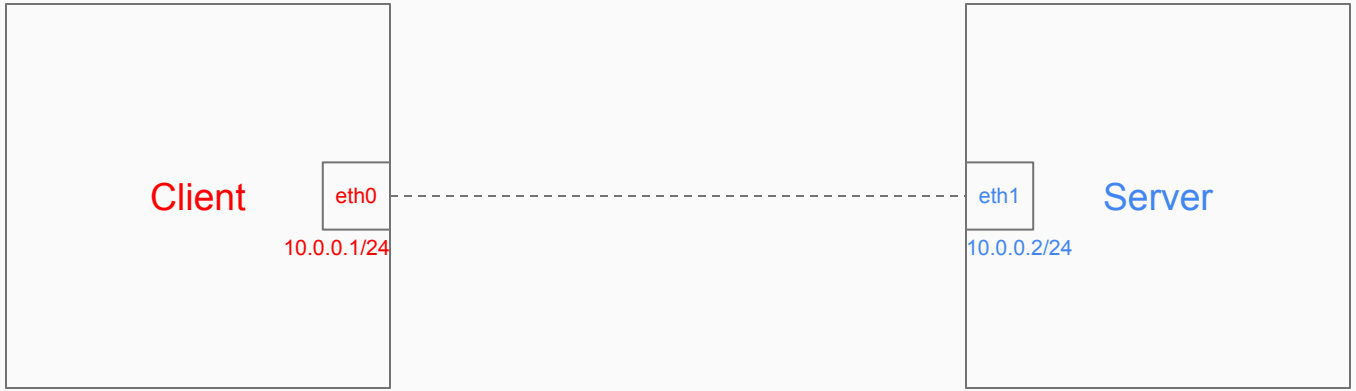

```
# ip netns exec client bash
# ip address
<output>
# ping 10.0.0.2
<output>^C
#
```

```
# ip netns exec server bash
# ip address
<output>
#
```



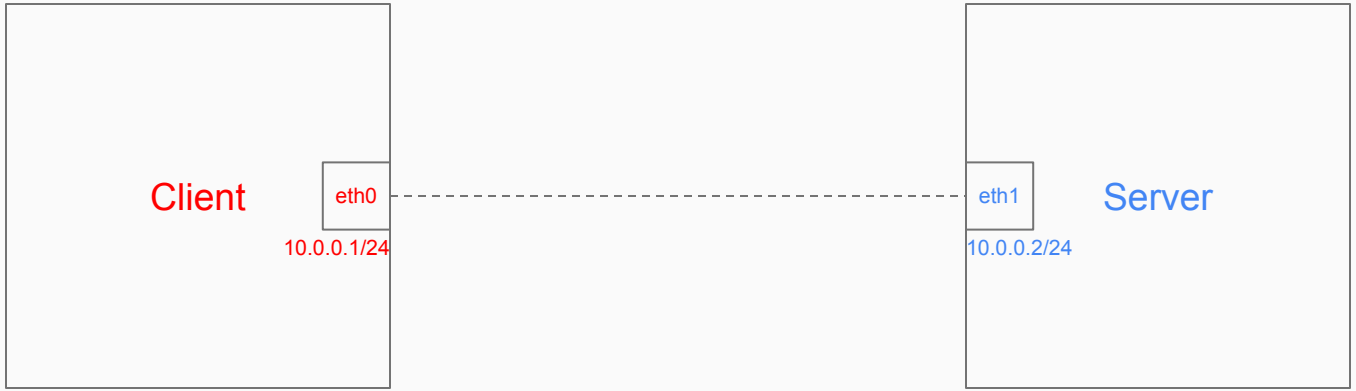

```
# ip netns exec client bash
# ip address
<output>
# ping 10.0.0.2
<output>^C
#
```

```
# ip netns exec server bash
# ip address
<output>
# ping 10.0.0.1
```



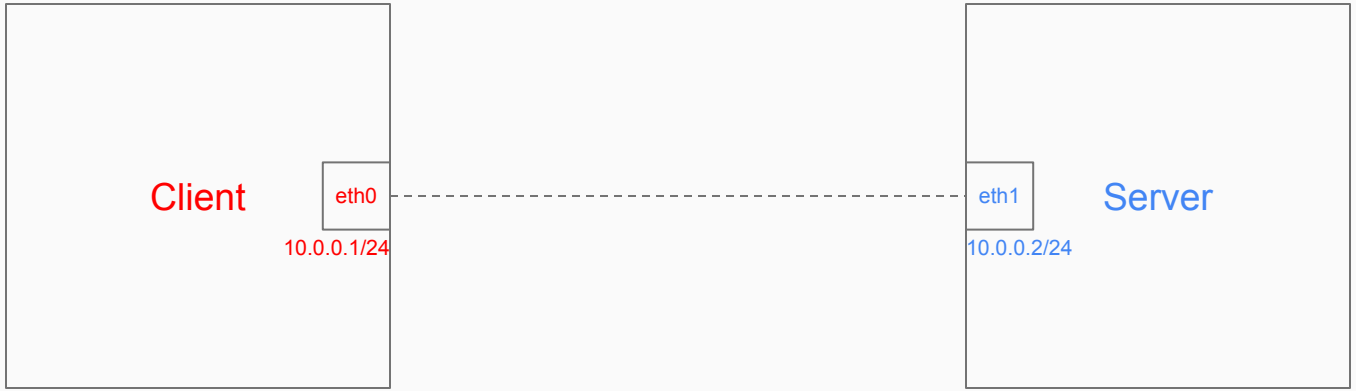

```
# ip netns exec client bash
# ip address
<output>
# ping 10.0.0.2
<output>^C
#
```

```
# ip netns exec server bash
# ip address
<output>
# ping 10.0.0.1
<output>
```



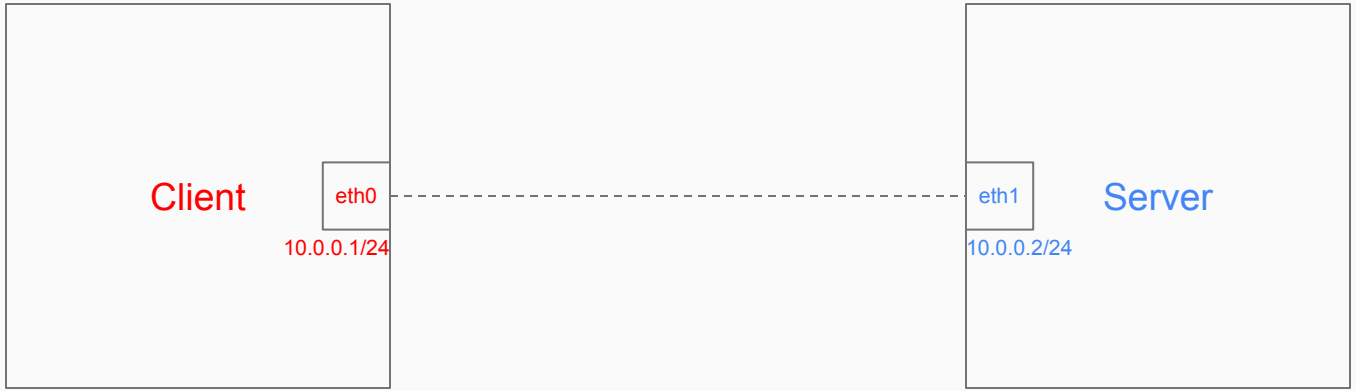

```
# ip netns exec client bash
# ip address
<output>
# ping 10.0.0.2
<output>^C
#
```

```
# ip netns exec server bash
# ip address
<output>
# ping 10.0.0.1
<output>^C
#
```



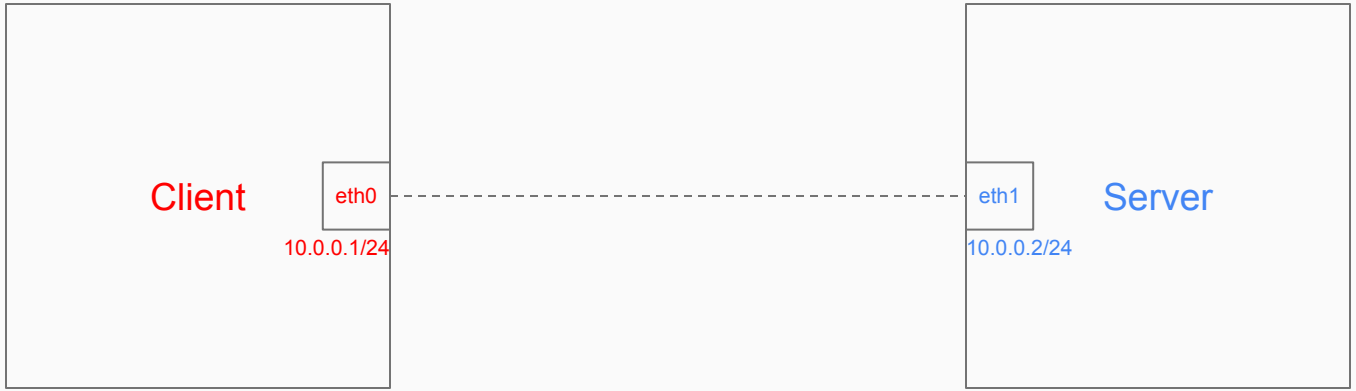

```
# ip netns exec client bash
# ip address
<output>
# ping 10.0.0.2
<output>^C
#
```

```
# ip netns exec server bash
# ip address
<output>
# ping 10.0.0.1
<output>^C
# iperf -s
```



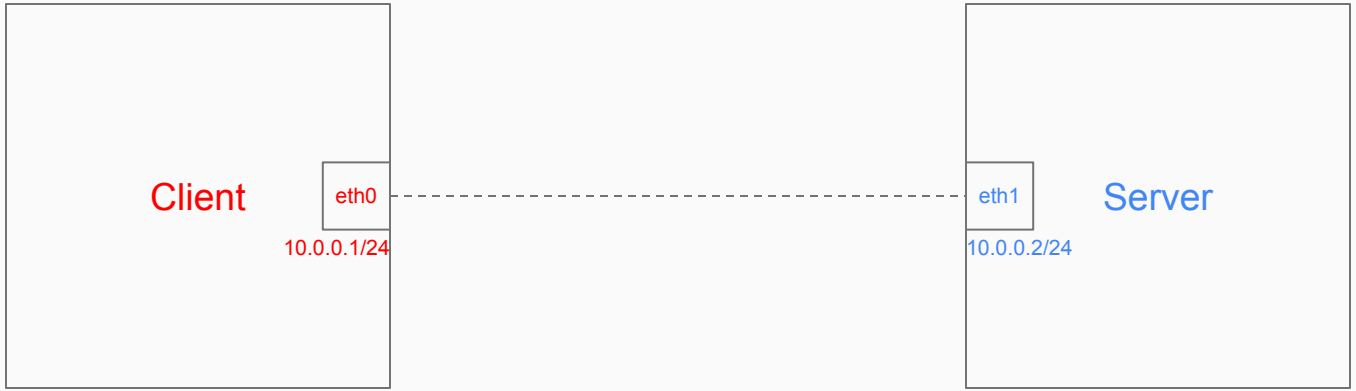

```
# ip netns exec client bash
# ip address
<output>
# ping 10.0.0.2
<output>^C
#
```

```
# ip netns exec server bash
# ip address
<output>
# ping 10.0.0.1
<output>^C
# iperf -s
<output>
```



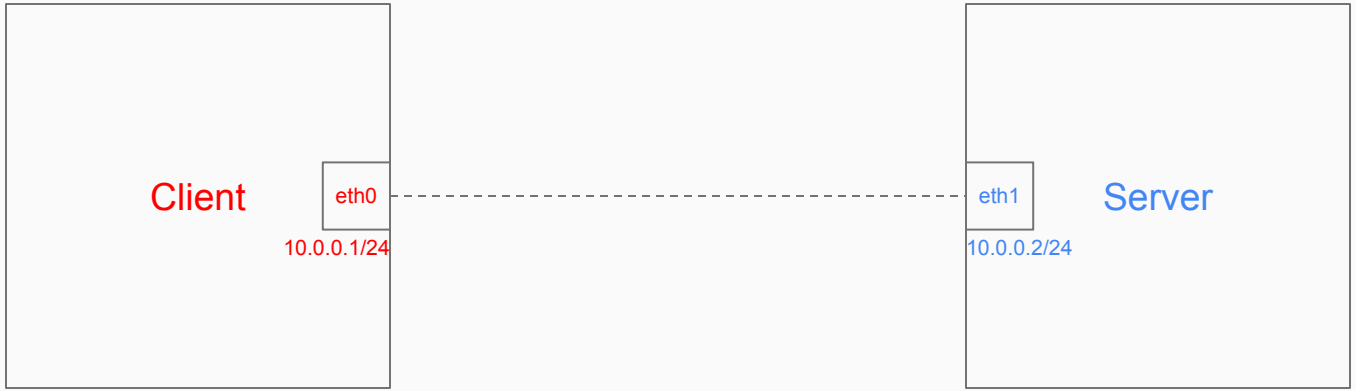

```
# ip netns exec client bash
# ip address
<output>
# ping 10.0.0.2
<output>^C
# iperf -c 10.0.0.2
```

```
# ip netns exec server bash
# ip address
<output>
# ping 10.0.0.1
<output>^C
# iperf -s
<output>
```



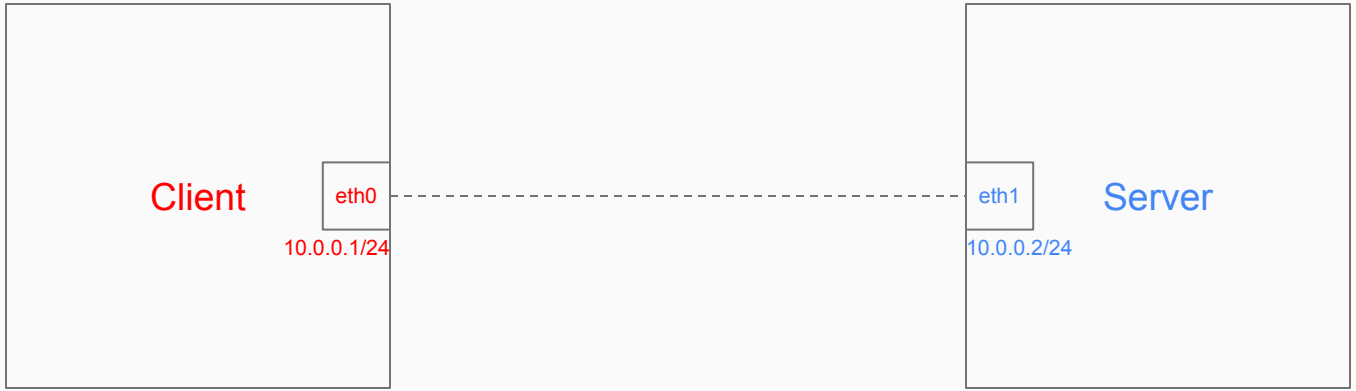

```
# ip netns exec client bash
# ip address
<output>
# ping 10.0.0.2
<output>^C
# iperf -c 10.0.0.2
<output>
```

```
# ip netns exec server bash
# ip address
<output>
# ping 10.0.0.1
<output>^C
# iperf -s
<output>
```



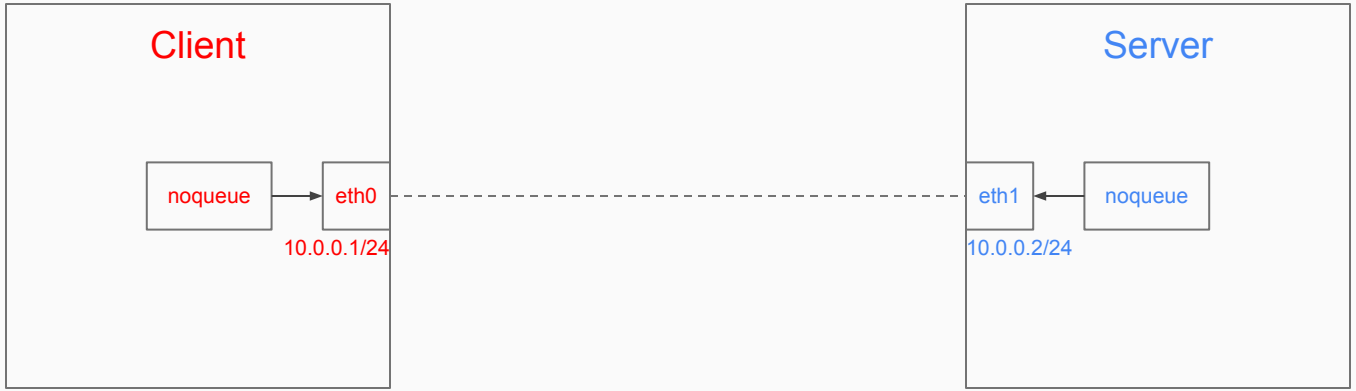

```
# ip netns exec client bash
# ip address
<output>
# ping 10.0.0.2
<output>^C
# iperf -c 10.0.0.2
<output>^C
```

```
# ip netns exec server bash
# ip address
<output>
# ping 10.0.0.1
<output>^C
# iperf -s
<output>
```



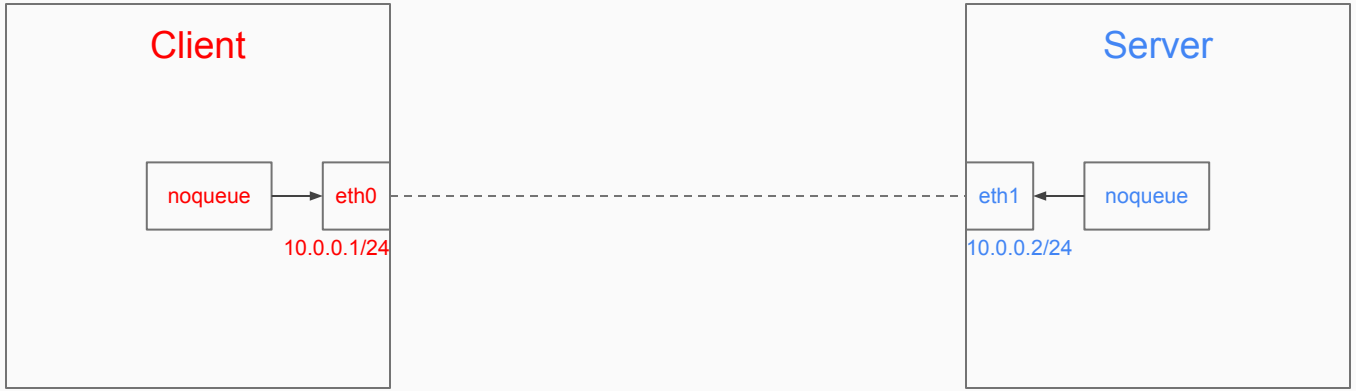

```
# ip netns exec client bash
# ip address
<output>
# ping 10.0.0.2
<output>^C
# iperf -c 10.0.0.2
<output>^C
# iperf -c 10.0.0.2 -u
```

```
# ip netns exec server bash
# ip address
<output>
# ping 10.0.0.1
<output>^C
# iperf -s
<output>
```


Client
noqueue
eth0
10.0.0.1/24
Server
eth1
noqueue
10.0.0.2/24

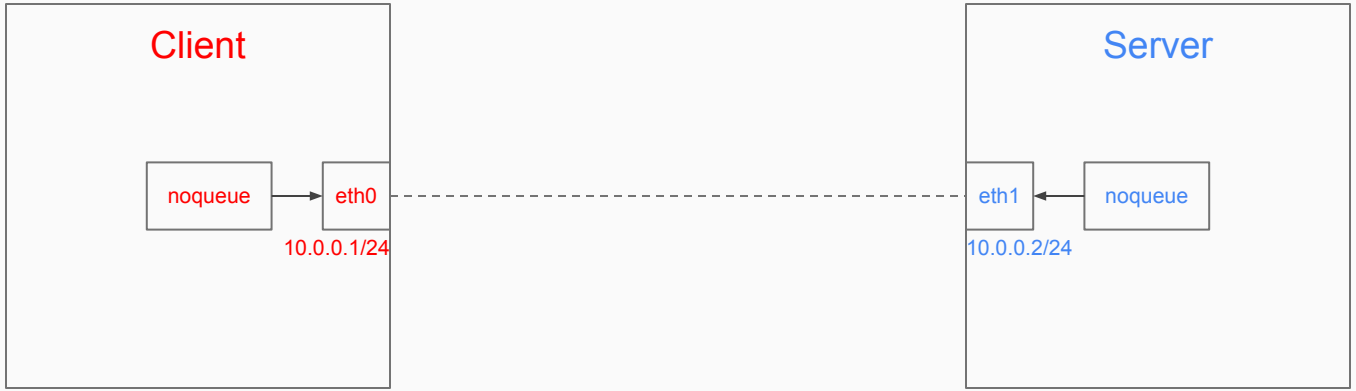
```
#
```

```
#
```



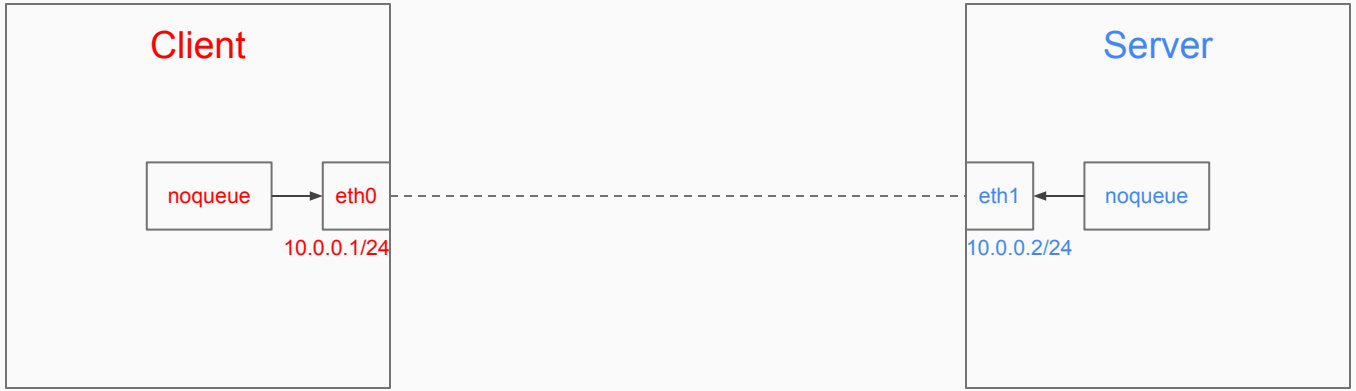

```
# tc qdisc show dev eth0
```

```
# tc qdisc show dev eth1
```



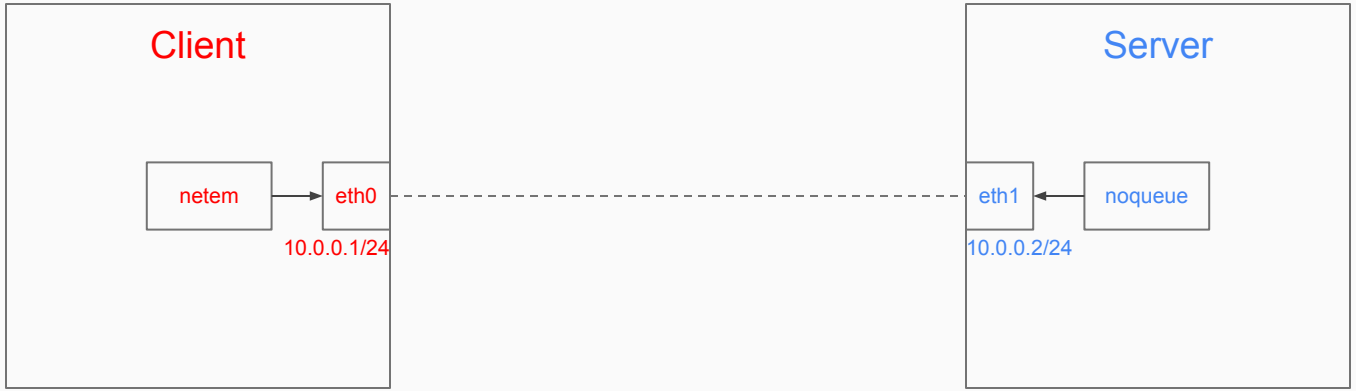

```
# tc qdisc show dev eth0
<output>
#
```

```
# tc qdisc show dev eth1
<output>
#
```



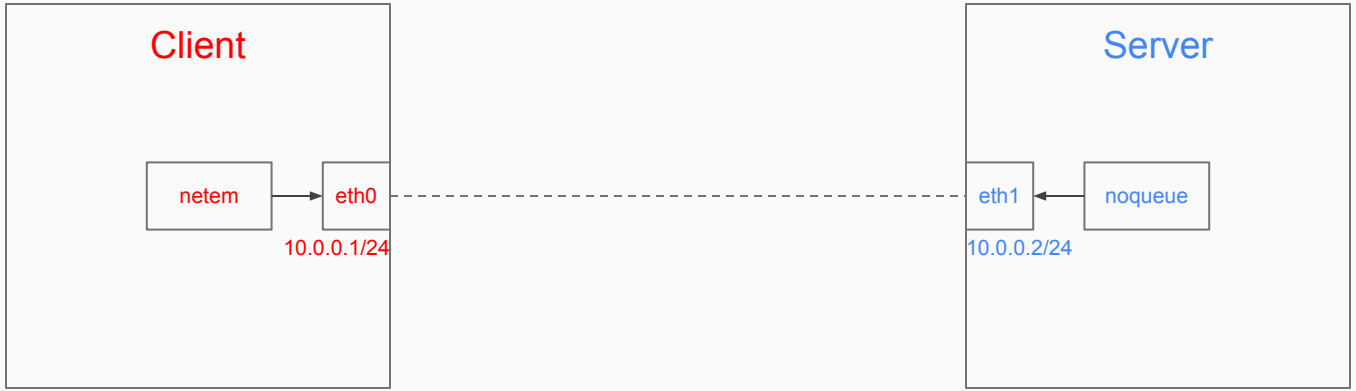

```
# tc qdisc show dev eth0
<output>
# tc qdisc add dev eth0 root \
  netem delay 5ms rate 10mbit
```

```
# tc qdisc show dev eth1
<output>
#
```



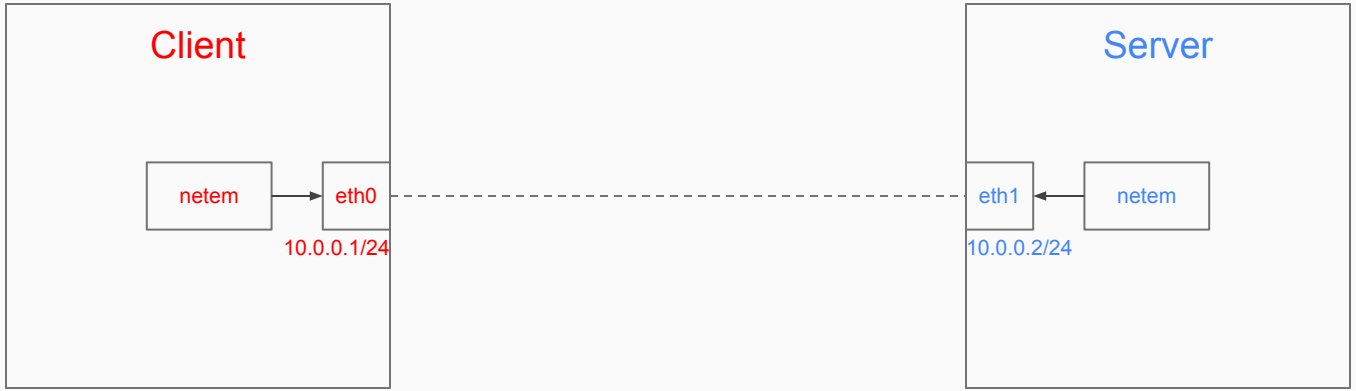

```
# tc qdisc show dev eth0
<output>
# tc qdisc add dev eth0 root \
  netem delay 5ms rate 10mbit
#
```

```
# tc qdisc show dev eth1
<output>
#
```




```
# tc qdisc show dev eth0
<output>
# tc qdisc add dev eth0 root \
  netem delay 5ms rate 10mbit
#
```

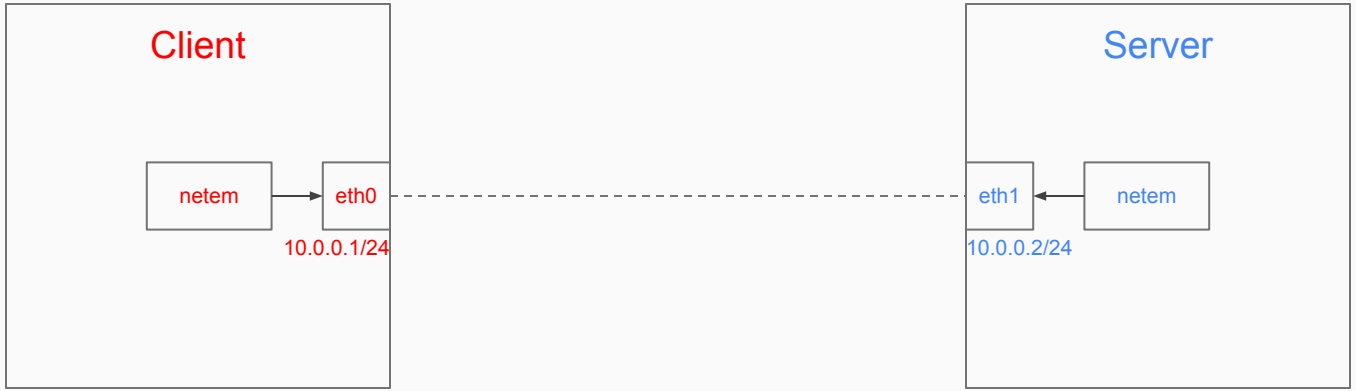
```
# tc qdisc show dev eth1
<output>
# tc qdisc add dev eth1 root \
  netem delay 10ms rate 50mbit
```




```
# tc qdisc show dev eth0
<output>
# tc qdisc add dev eth0 root \
  netem delay 5ms rate 10mbit
#
```

```
# tc qdisc show dev eth1
<output>
# tc qdisc add dev eth1 root \
  netem delay 10ms rate 50mbit
#
```

Example with CoAP Client and Server using network namespaces




```
# ./coap-client -m get
coap://[10.0.0.2]/.well-known/co
re
```

```
# ./coap-server
```

Topology setup with NeST:
Create two nodes connected via a direct link

What is NeST?

- A Python package
- Uses network namespaces to simplify network experimentation
- Provides intuitive APIs to
  - Build a virtual network
  - Run experiments on the virtual network
  - Collect statistics
  - Plot results

Shanthanu S. Rai, Narayan G., Dhanasekhar M., Leslie Monis, and Mohit P. Tahiliani. "NeST: Network Stack Tester." In Proceedings of the Applied Networking Research Workshop, pp. 32-37. 2020.

Why NeST?

- Simplifies the process to reproduce network experiments
- Less physical resources, less error prone and less prerequisites
- Multiple instances of the same network topology can co-exist, and different experiments can be run in parallel on every instance
- Open source tool released under GPLv2 License

Installing NeST

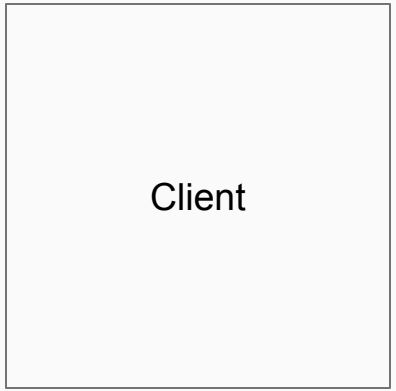
```
$ sudo apt install python3-pip netperf

$ python3 -m pip install -U pip

$ python3 -m pip install nitk-nest
```

Website: https://nitk-nest.github.io/
Repository: https://gitlab.com/nitk-nest/nest
NeST users list: https://groups.google.com/g/nest-users
NeST documentation: https://nitk-nest.github.io/docs/index.html

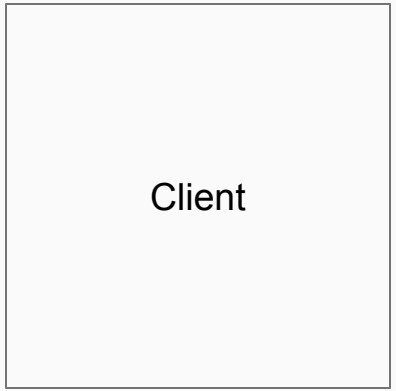
Note: Do not write these commands in a terminal. Kindly open a file editor (e.g., gedit) and type these commands in it. Remember, you have to save it as a python file (with .py extension)

```
from nest.experiment import *
from nest.topology import *
```

```
from nest.experiment import *
from nest.topology import *

client = Node('client')
```



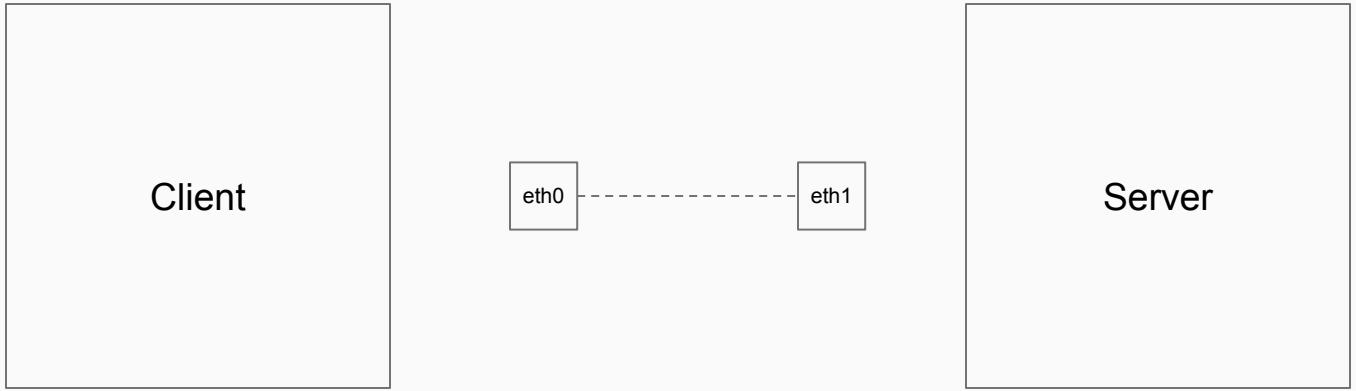

```
from nest.experiment import *
from nest.topology import *

client = Node('client')
```



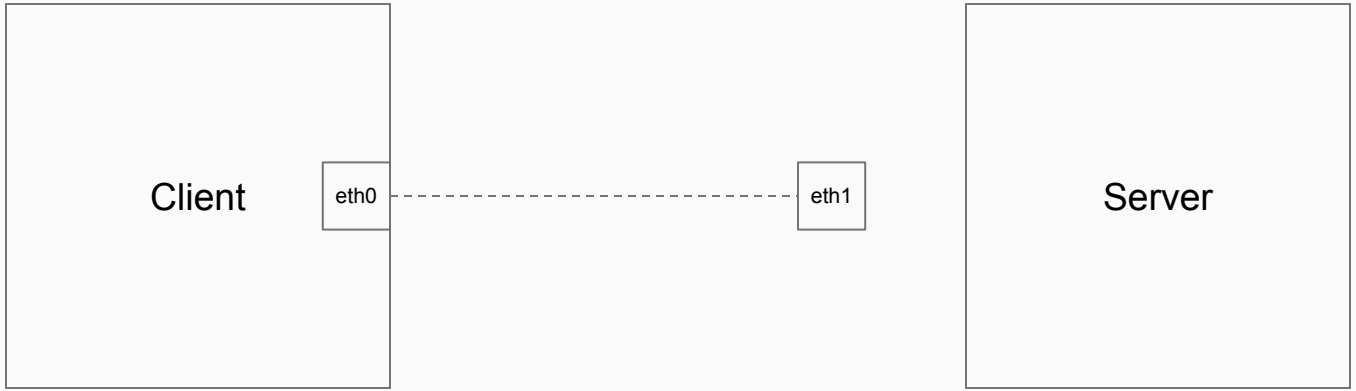

```
from nest.experiment import *
from nest.topology import *

client = Node('client')
server = Node('server')
```


```
from nest.experiment import *
from nest.topology import *

client = Node('client')
server = Node('server')
```

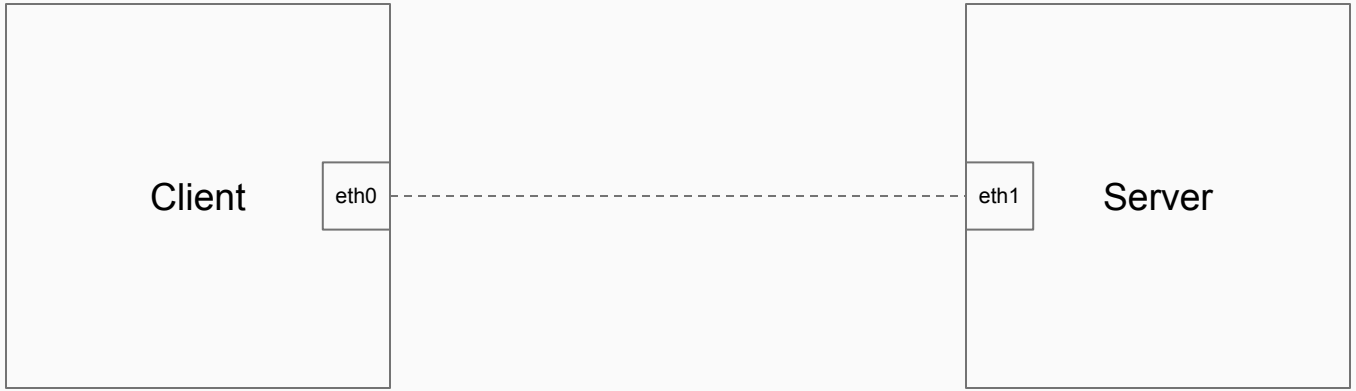

```
from nest.topology import *

client = Node('client')
server = Node('server')

(eth0, eth1) = connect(client, server)
```



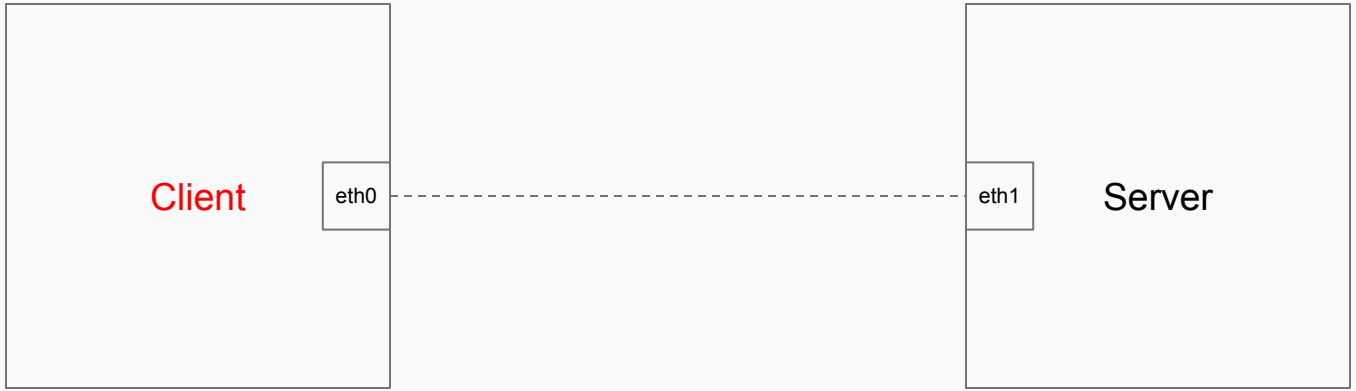

```
from nest.topology import *

client = Node('client')
server = Node('server')

(eth0, eth1) = connect(client, server)
```



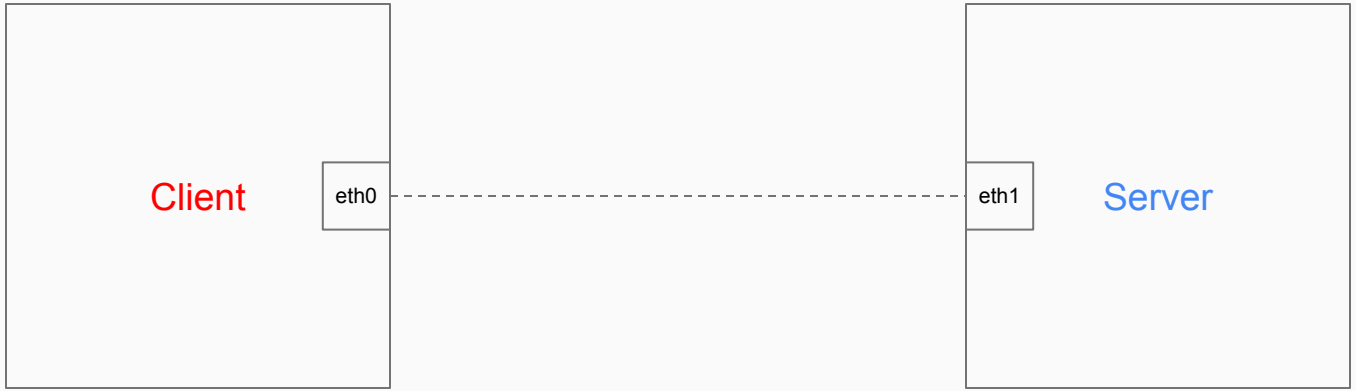

```
from nest.topology import *

client = Node('client')
server = Node('server')

(eth0, eth1) = connect(client, server)
```



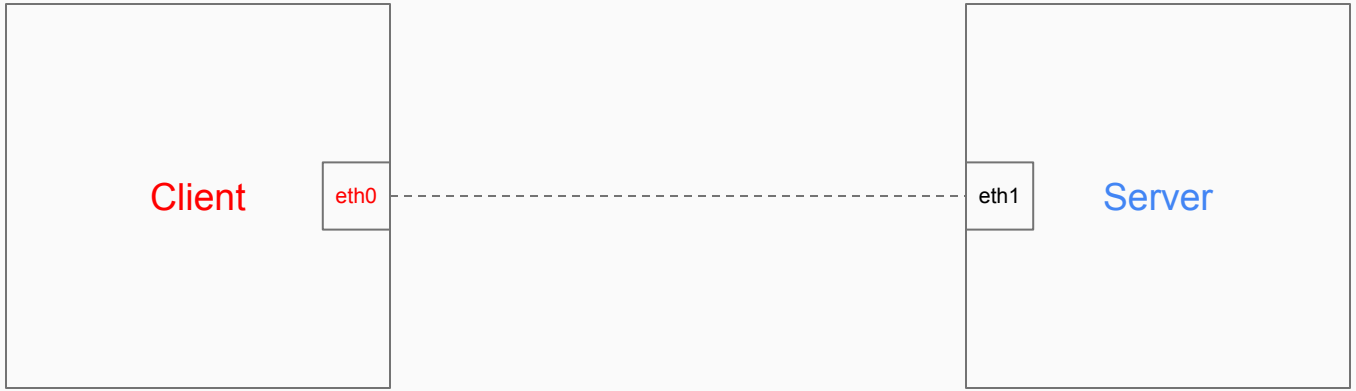

```
from nest.topology import *

client = Node('client')
server = Node('server')

(eth0, eth1) = connect(client, server)
```



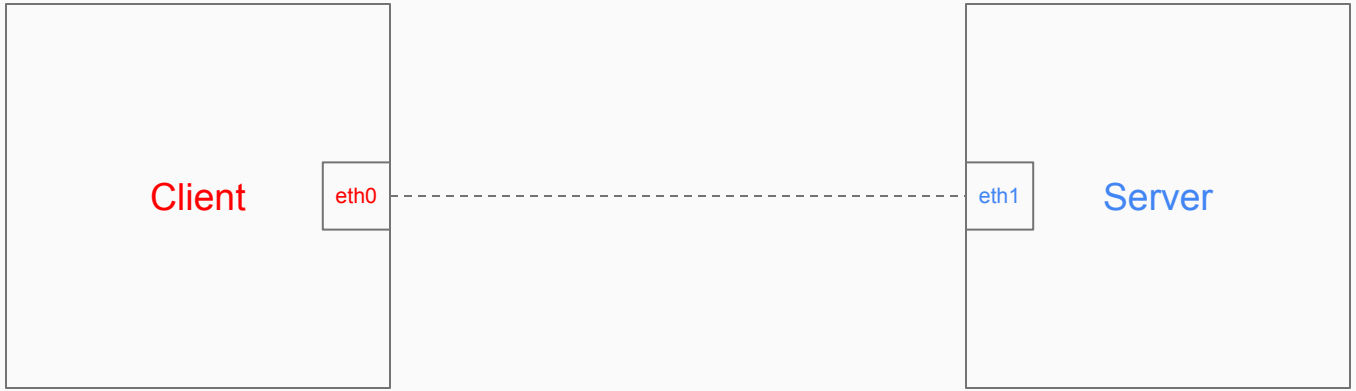

```
from nest.topology import *

client = Node('client')
server = Node('server')

(eth0, eth1) = connect(client, server)
```



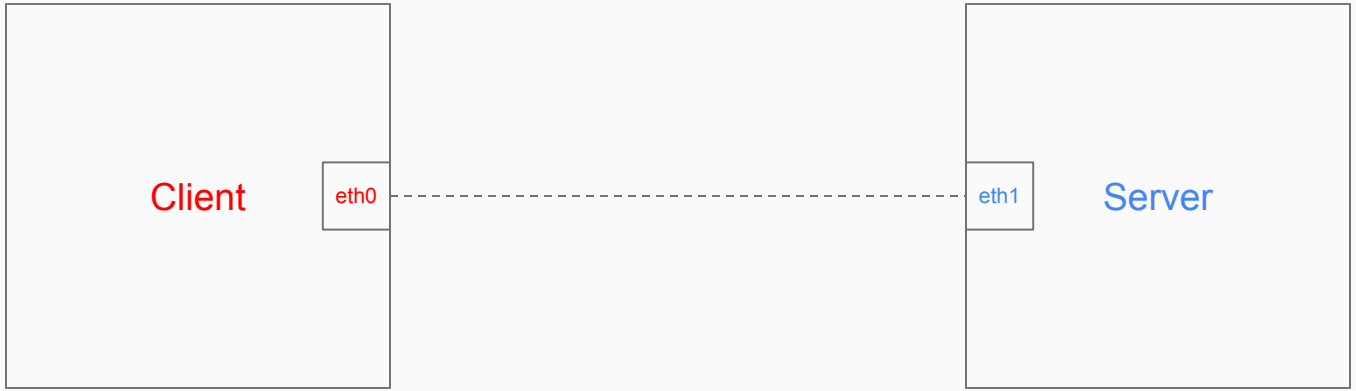

```
from nest.topology import *

client = Node('client')
server = Node('server')

(eth0, eth1) = connect(client, server)
```



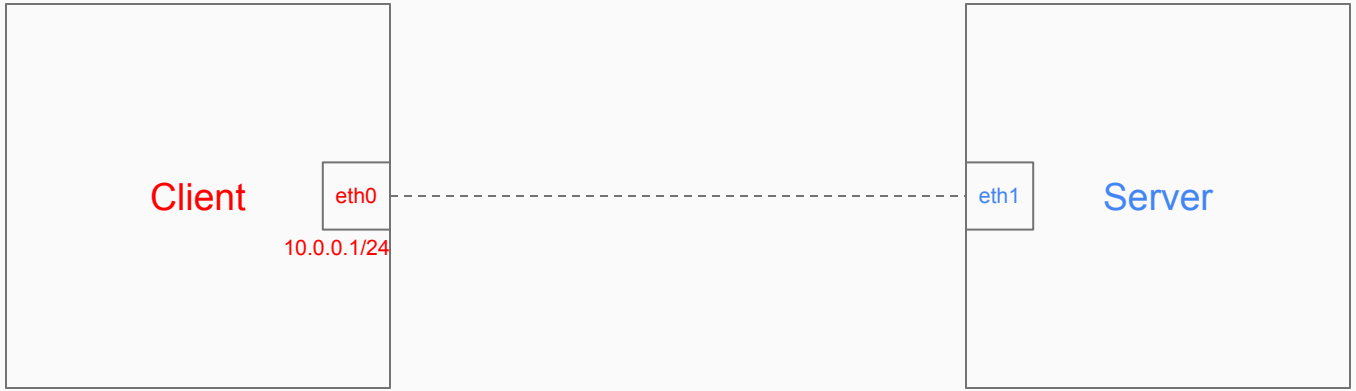

```
from nest.topology import *

client = Node('client')
server = Node('server')

(eth0, eth1) = connect(client, server)
```



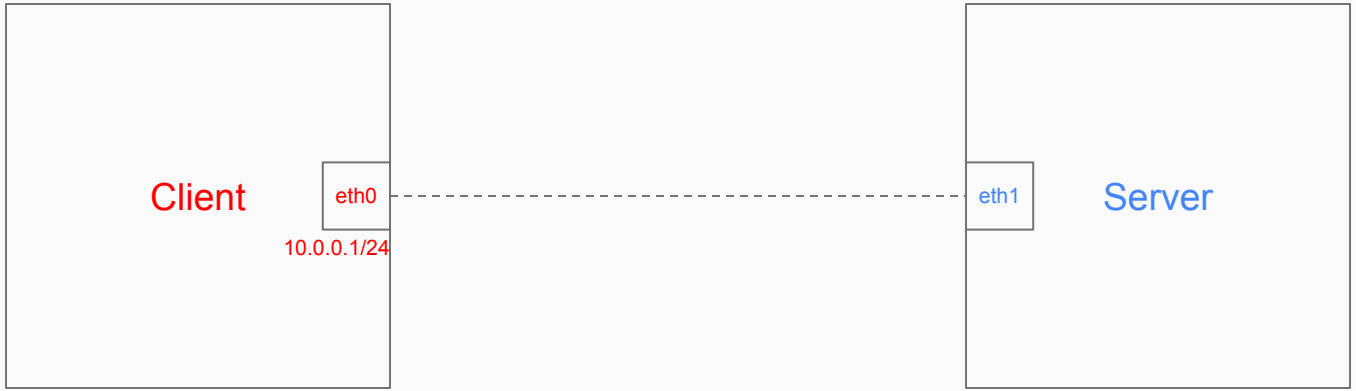

```
from nest.topology import *

client = Node('client')
server = Node('server')

(eth0, eth1) = connect(client, server)
```



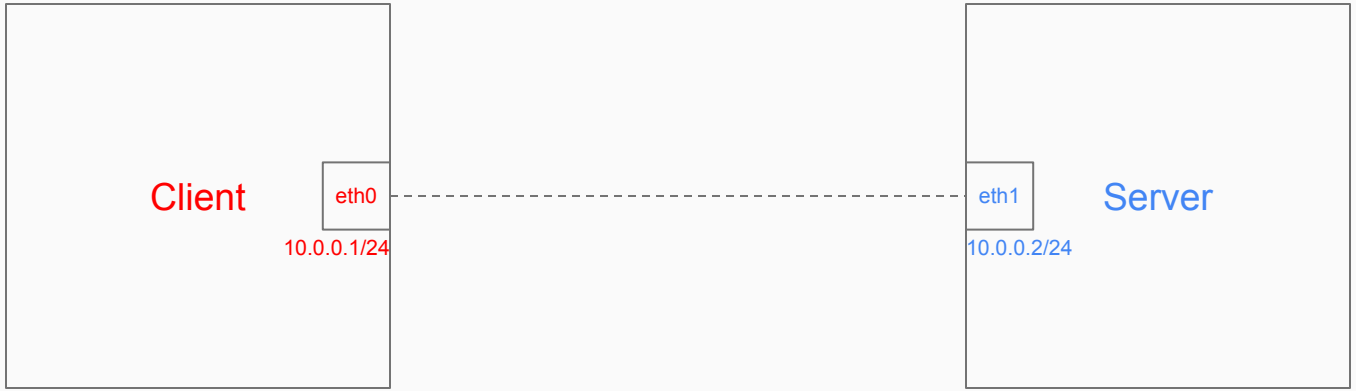

```
client = Node('client')
server = Node('server')

(eth0, eth1) = connect(client, server)

eth0.set_address('10.0.0.1/24')
```



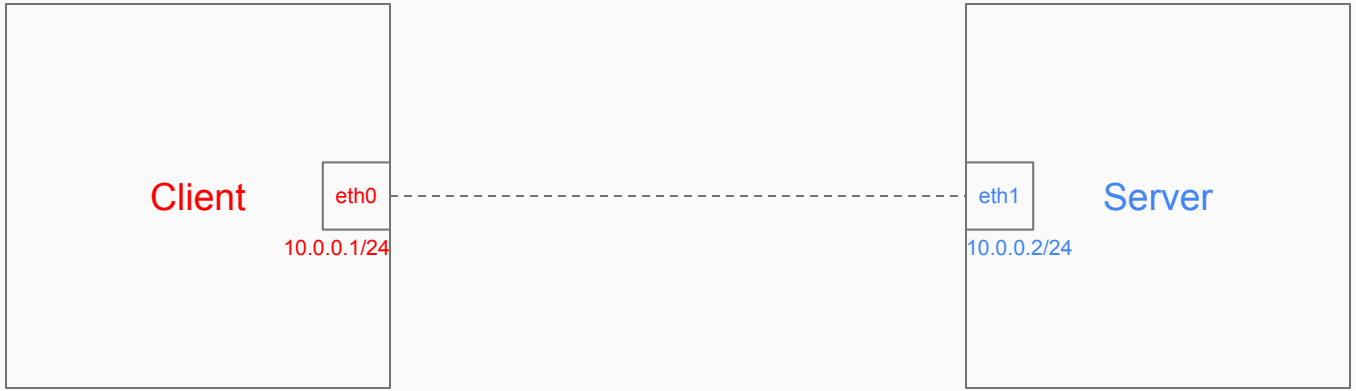

```
client = Node('client')
server = Node('server')

(eth0, eth1) = connect(client, server)

eth0.set_address('10.0.0.1/24')
```



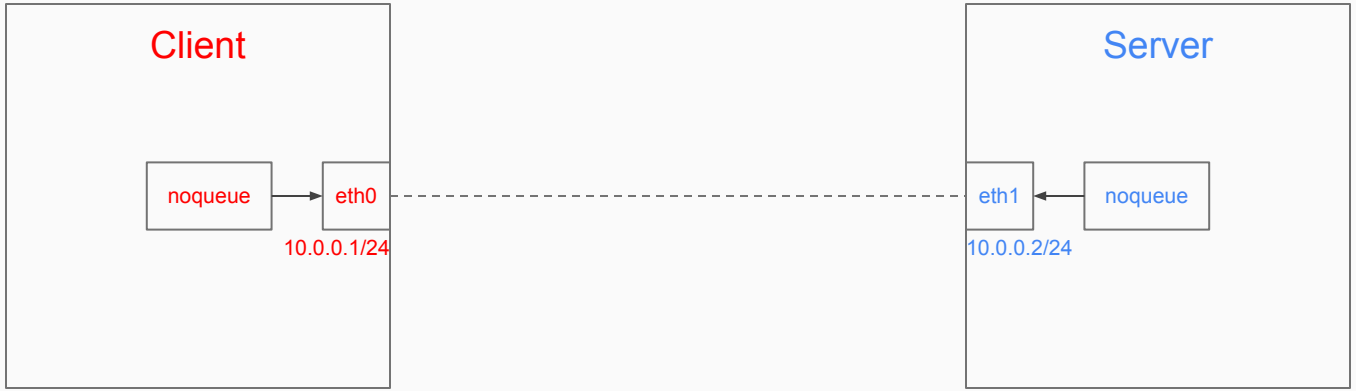

```
server = Node('server')

(eth0, eth1) = connect(client, server)

eth0.set_address('10.0.0.1/24')
eth1.set_address('10.0.0.2/24')
```



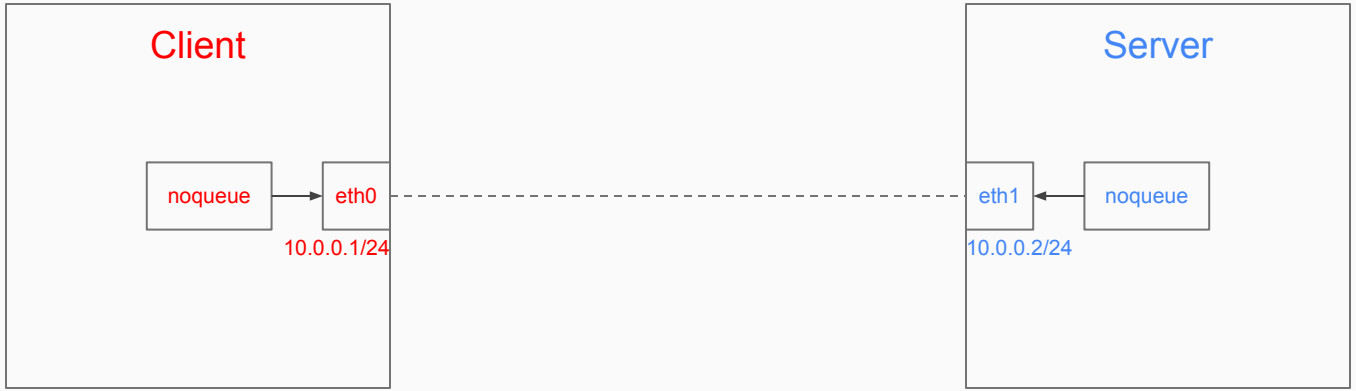

```
server = Node('server')

(eth0, eth1) = connect(client, server)

eth0.set_address('10.0.0.1/24')
eth1.set_address('10.0.0.2/24')
```



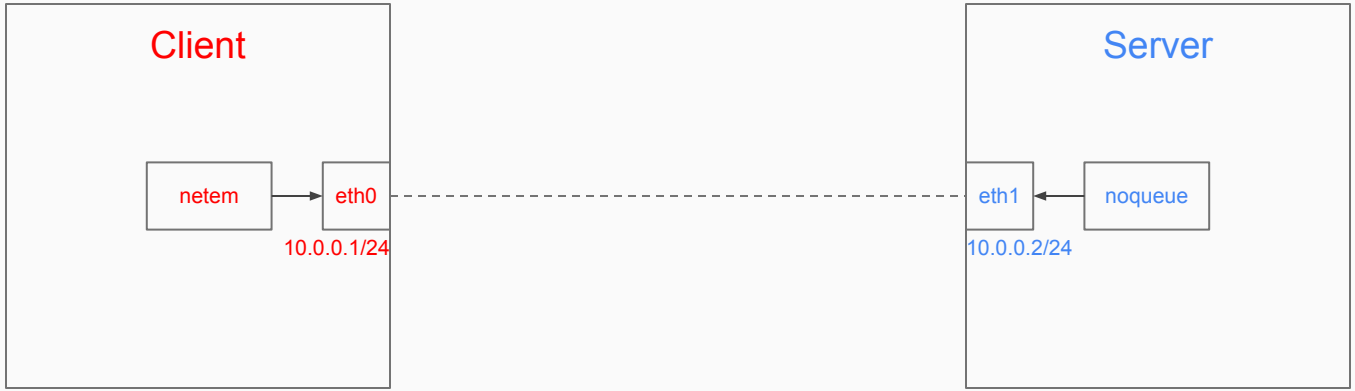

```
(eth0, eth1) = connect(client, server)

eth0.set_address('10.0.0.1/24')
eth1.set_address('10.0.0.2/24')

client.ping(eth1.address)      # client.ping('10.0.0.2') also works
```



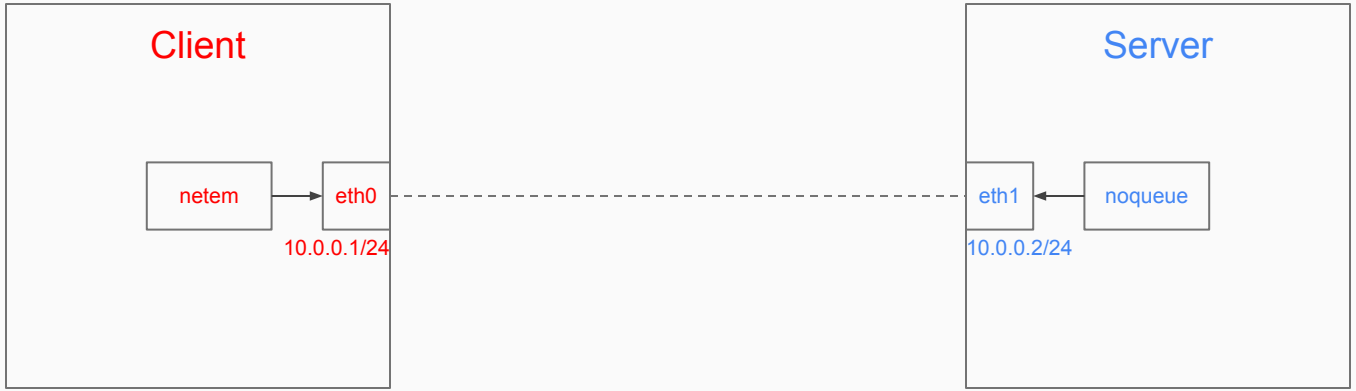

```
(eth0, eth1) = connect(client, server)

eth0.set_address('10.0.0.1/24')
eth1.set_address('10.0.0.2/24')

client.ping(eth1.address)      # client.ping('10.0.0.2') also works
```



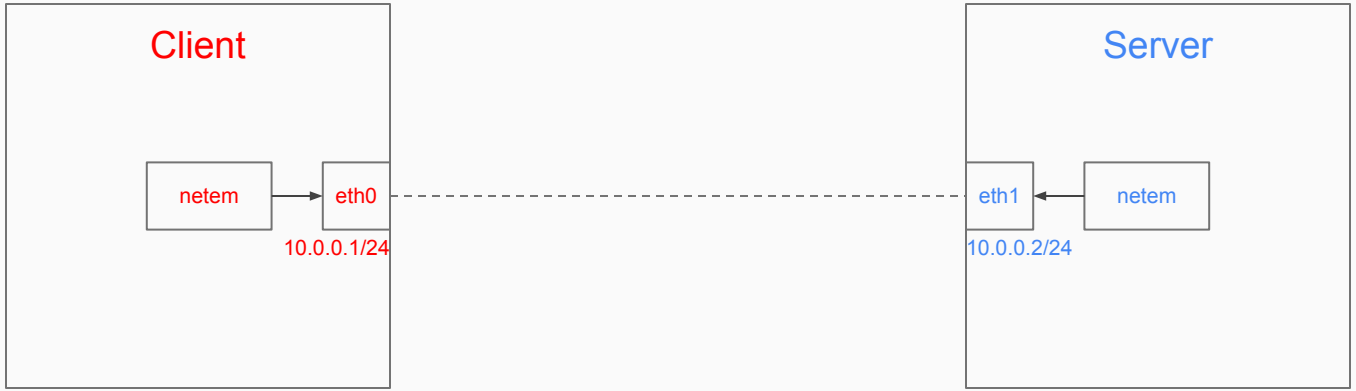

```
eth0.set_address('10.0.0.1/24')
eth1.set_address('10.0.0.2/24')

client.ping(eth1.address)      # client.ping('10.0.0.2') also works

eth0.set_attributes('5mbit', '5ms')
```




```
eth0.set_address('10.0.0.1/24')
eth1.set_address('10.0.0.2/24')

client.ping(eth1.address)      # client.ping('10.0.0.2') also works

eth0.set_attributes('5mbit', '5ms')
```




```
eth1.set_address('10.0.0.2/24')

client.ping(eth1.address)      # client.ping('10.0.0.2') also works

eth0.set_attributes('5mbit', '5ms')
eth1.set_attributes('10mbit', '100ms')
```




```
eth1.set_address('10.0.0.2/24')

client.ping(eth1.address)      # client.ping('10.0.0.2') also works

eth0.set_attributes('5mbit', '5ms')
eth1.set_attributes('10mbit', '100ms')
```

Thank you!

Contact:

Mohit P. Tahiliani
tahiliani@nitk.edu.in