Lab 2. VM NIC Mode, Port Forwarding

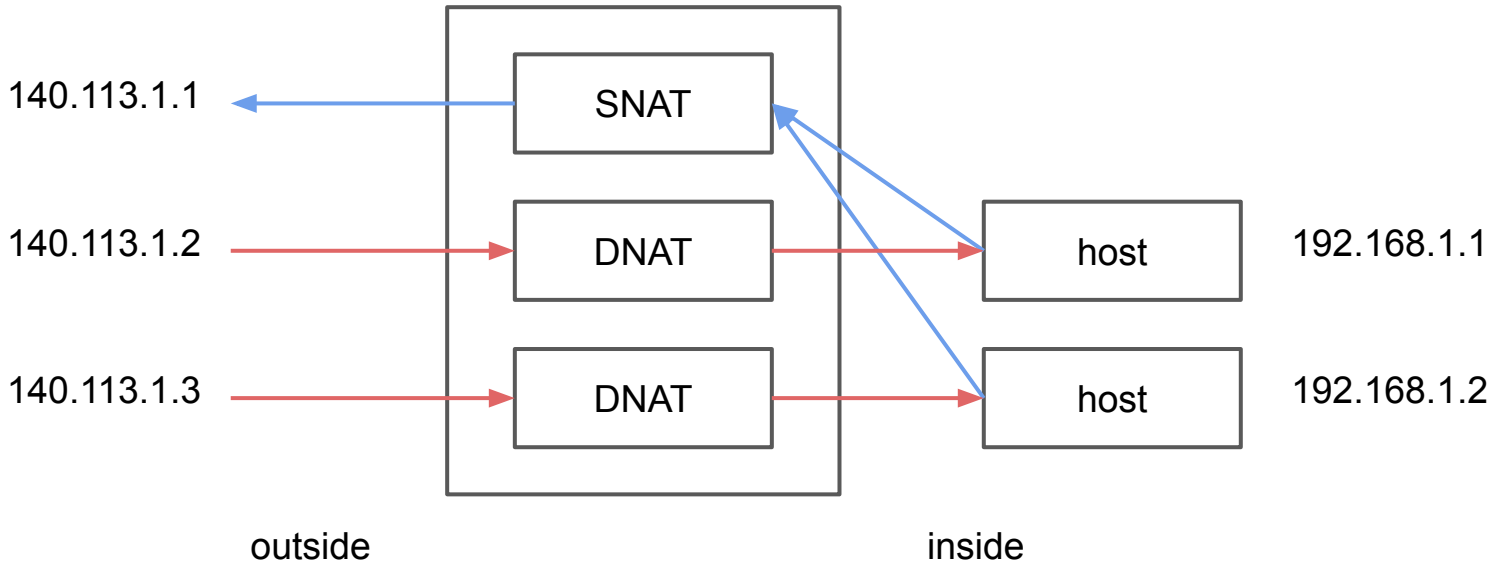
TA 施羿廷 (ytshih)
Credit to 紀政良 (clc)

Purpose

- Basic knowledge of VM (Virtual Box)
  - Virtaul Box hardware setting
  - Virtual Box network interface modes
- Port forwarding concept

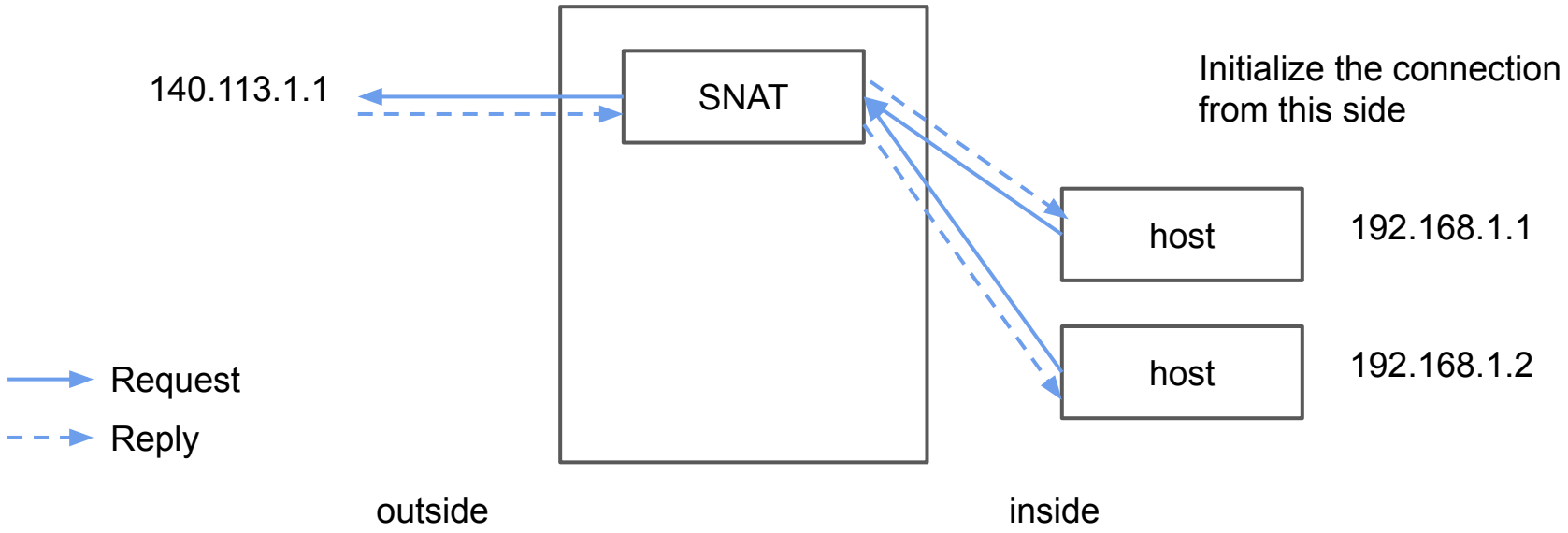
NAT Use Cases

- Perform SNAT when the inside packets pass through firewall or router
- Perform DNAT when the outside packets pass through firewall or router



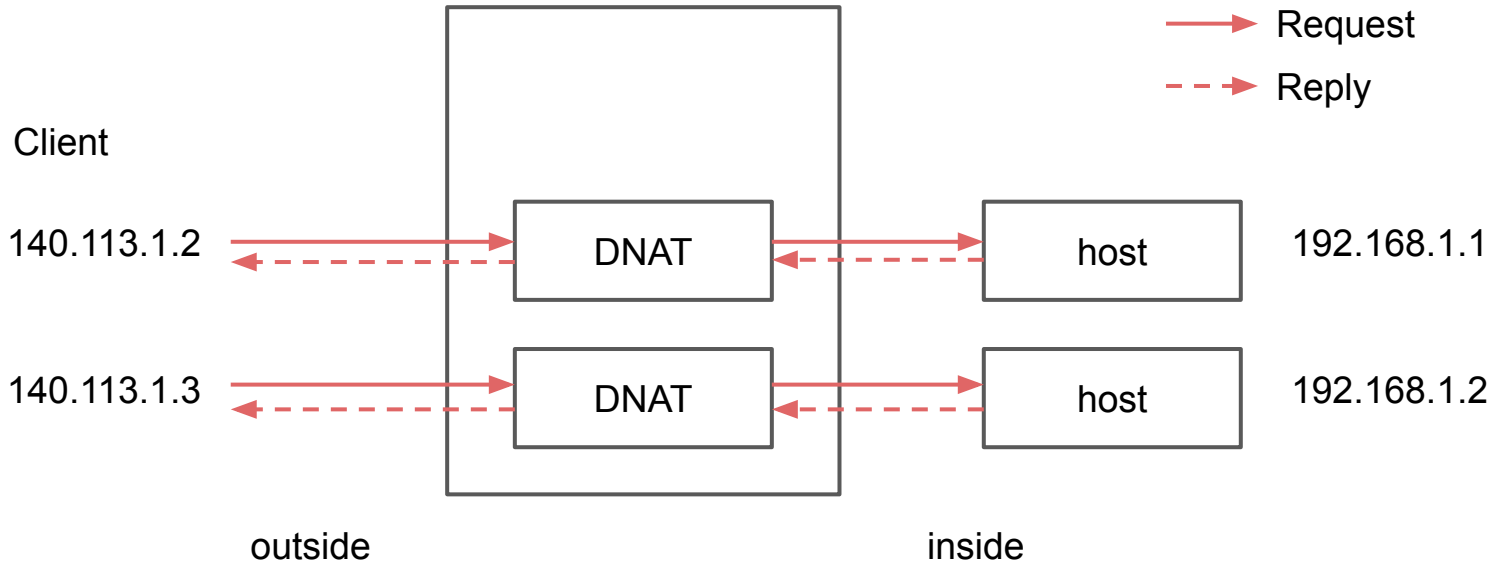
Source NAT

- Multiple private IPs treat the same public as a entrance

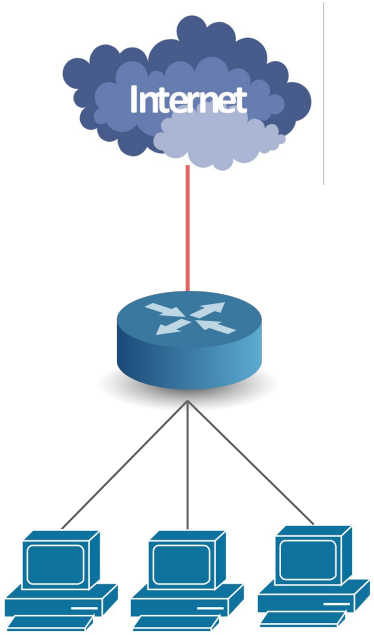


Destination NAT

- Outside network try to access inside network



NAT Common Topology



NAT Server

Hosts

Translate source address

Outgoing traffic

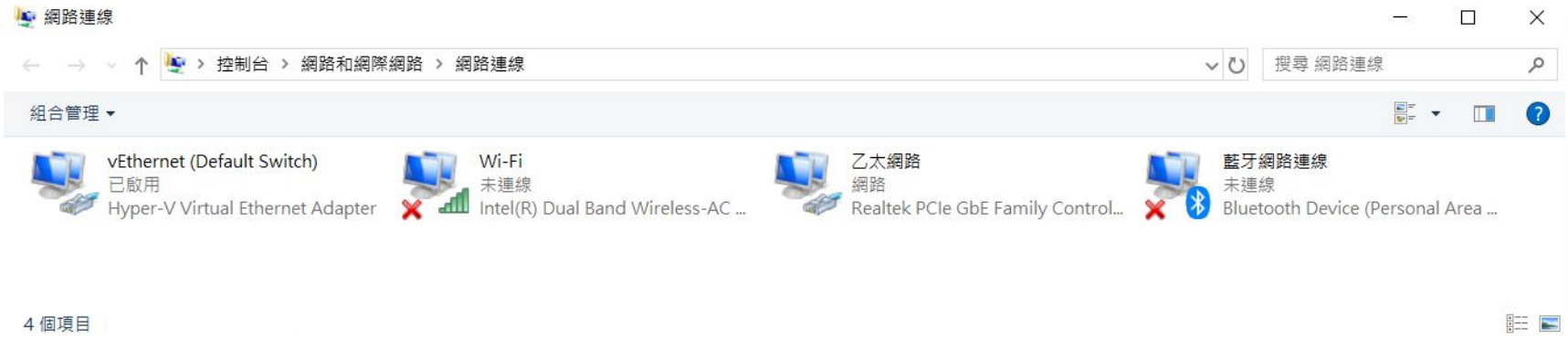
What is Virtual Machine (VM)?

- A software computer that, like a physical computer, runs an operating system and applications.
  - Has virtual devices that provide the same functionality as physical hardware
- Comprised of a set of specification and configuration files and is backed by the physical resources of a host
- Additional benefits
  - Portability
  - Manageability
  - Security

https://www.vmware.com/support/pubs/

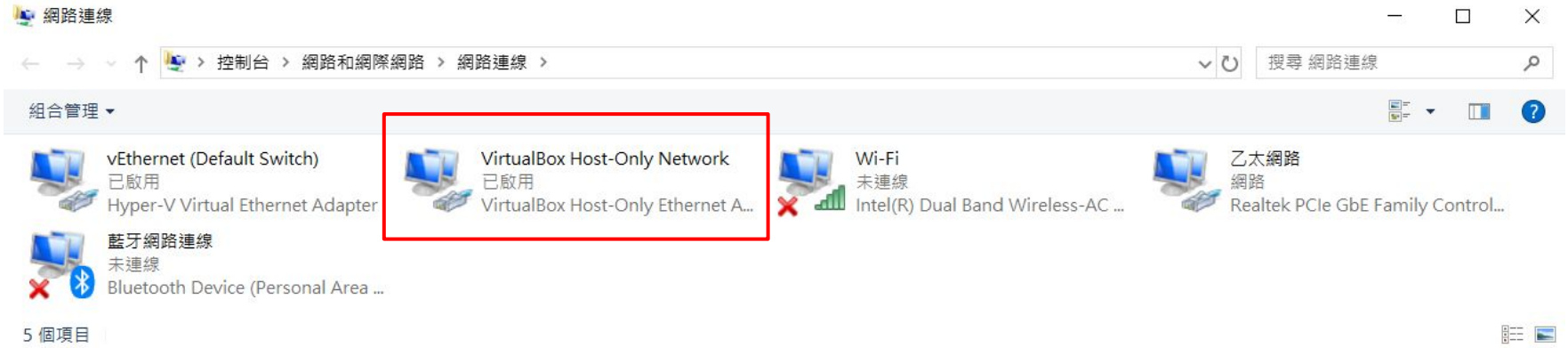
Virtual Networking

- Before install VirtualBox...



Virtual Networking

- After install VirtualBox…
  - What is the meaning of “Host-only”?



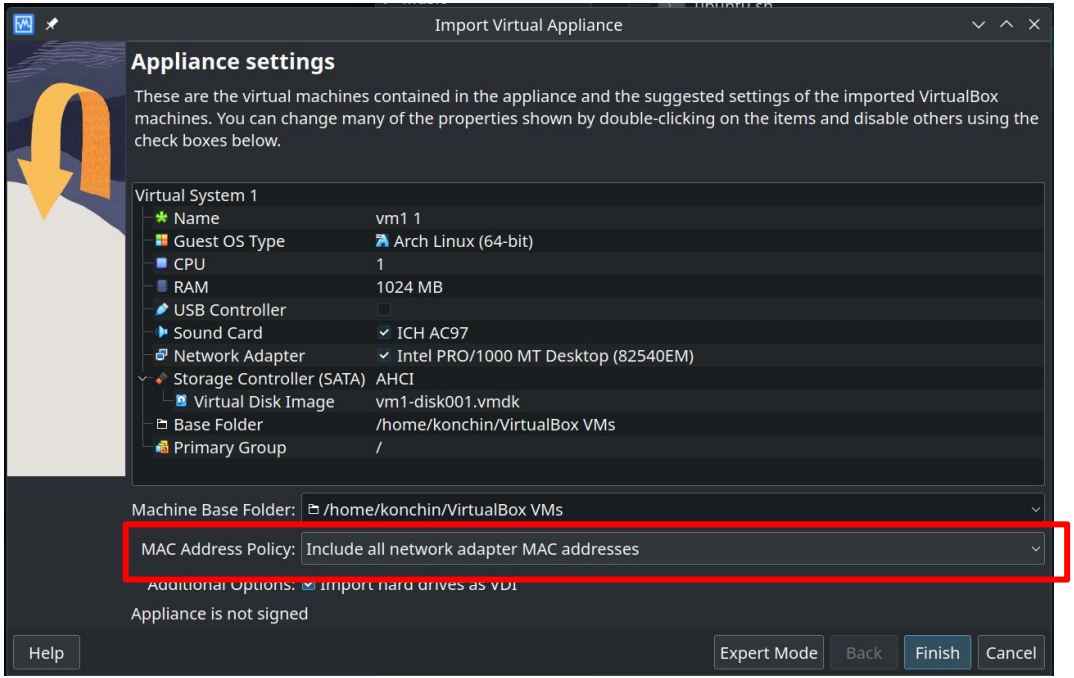
Virtual Networking

- After install VirtualBox…
  - What is this additional interface?

```
> ip --color address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: wlp0s20f3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
    link/ether 3c:21:9c:cd:eb:0f brd ff:ff:ff:ff:ff:ff
    inet 192.168.68.77/22 brd 192.168.71.255 scope global dynamic noprefixroute wlp0s20f3
       valid_lft 6671sec preferred_lft 6671sec
    inet6 fe80::6787:f334:86bf:6e5e/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: vboxnet0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 0a:00:27:00:00:00 brd ff:ff:ff:ff:ff:ff
```

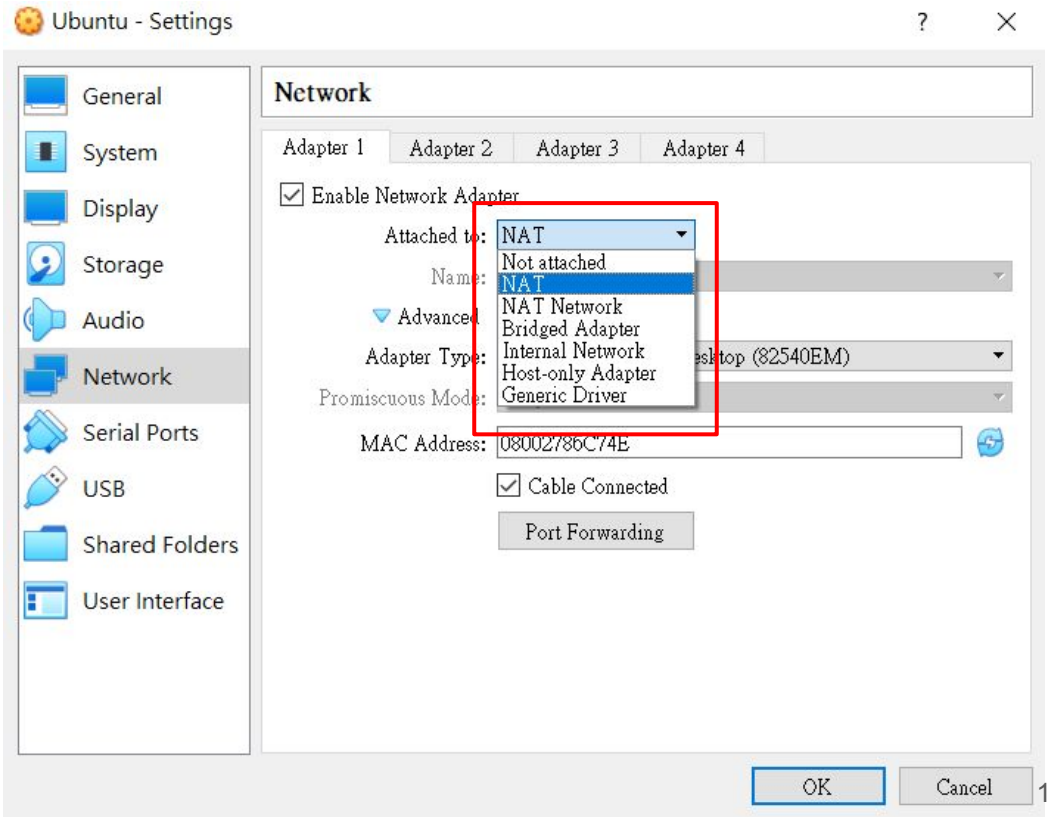
Virtualbox Import settings

- For **MAC Address Policy**, choose “**Include only NAT network adapter MAC addresses**”.



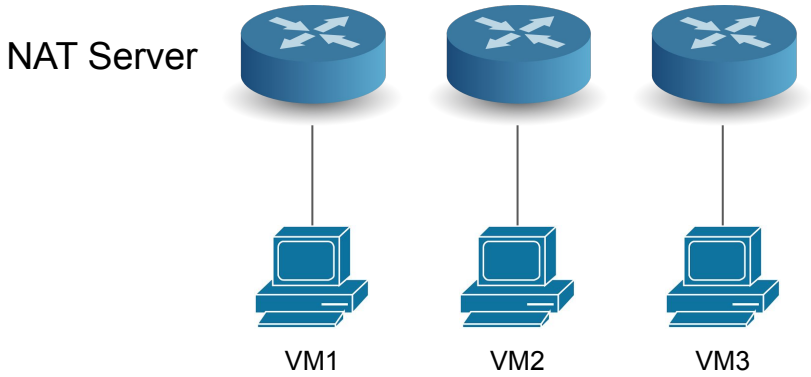
Virtual machine settings

- What is the meaning of these options?

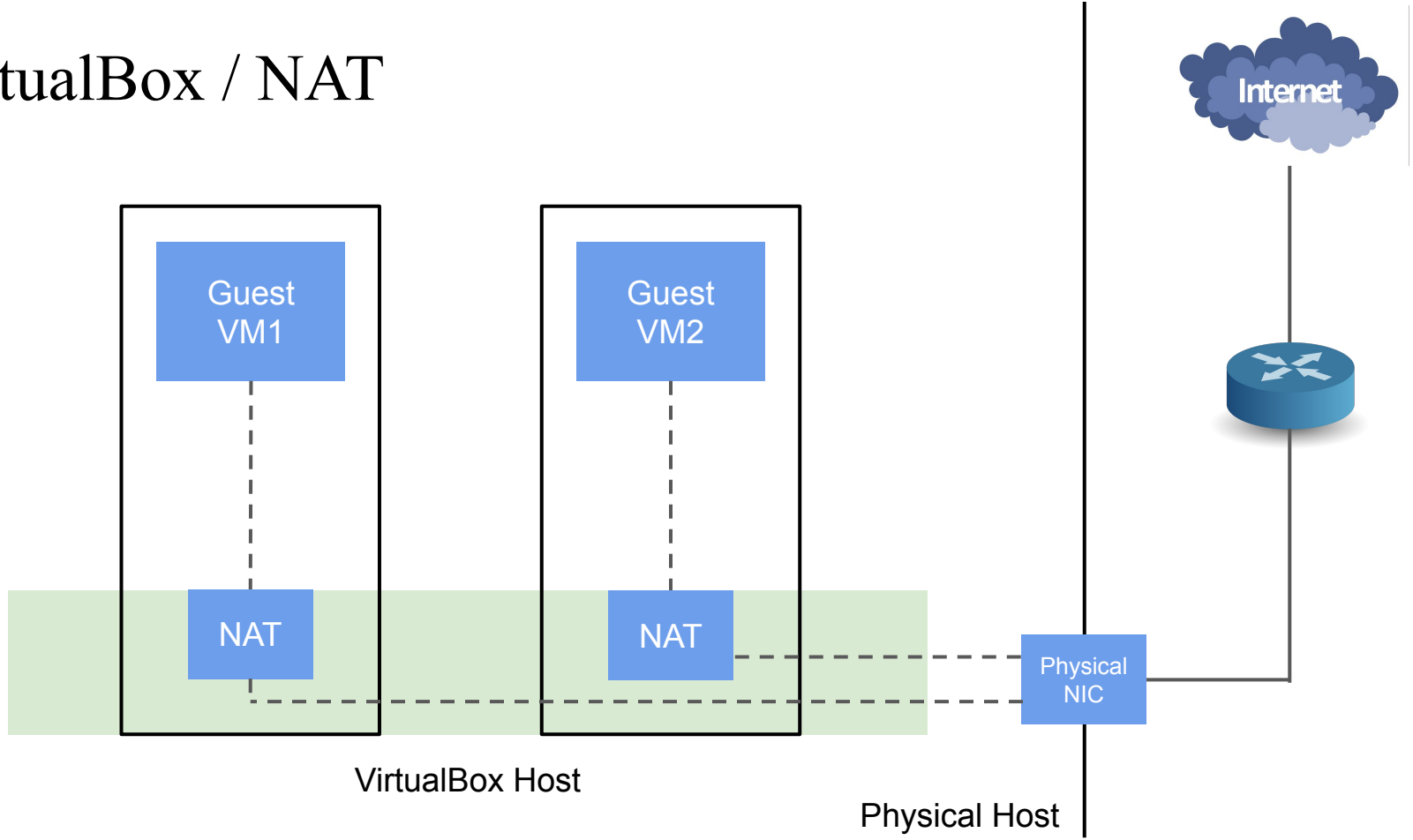


VirtualBox / NAT

- Default mode
- A virtual machine with NAT enabled acts much like a real computer that connects to the Internet through a router
- This router is placed between **each** virtual machine and the host
- This separation maximizes security since by default virtual machines **cannot talk to each other**



VirtualBox / NAT



VirtualBox / NAT

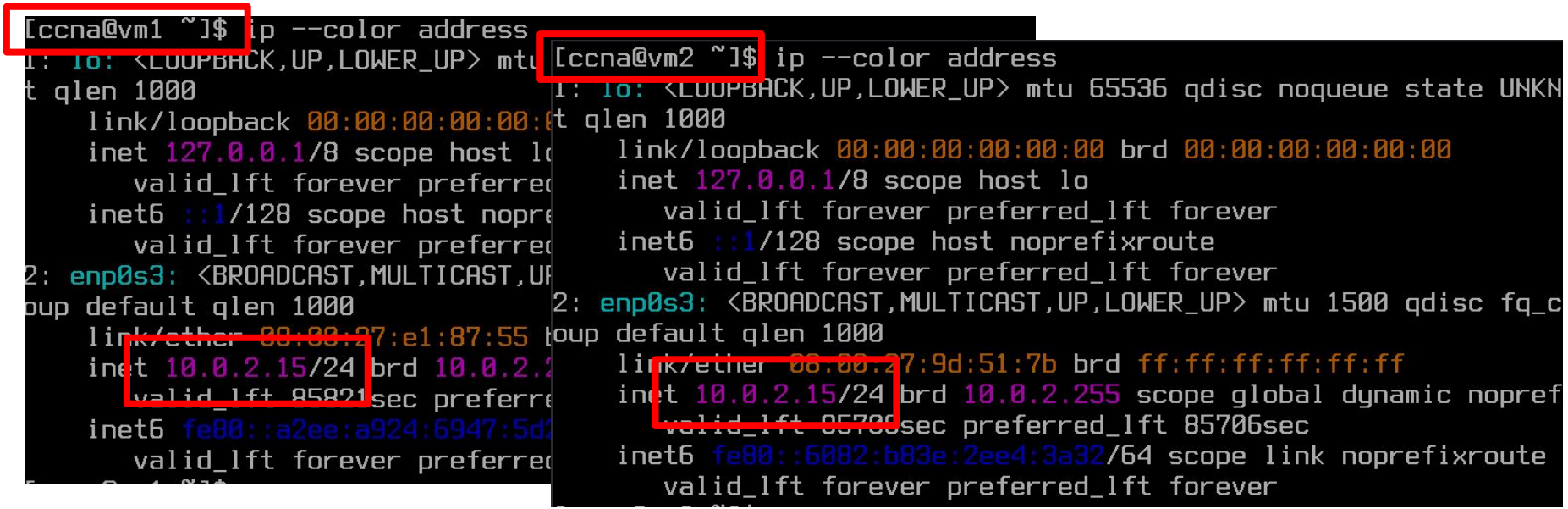
user/password: ccna/ccna

- Check the IP addresses of the vms.
  - So we can easily tell that they **can’t ping each other** directly.
- Ping 1.1.1.1
  - Every vm should be able to reach the Internet.
- Find the default gateway (*ip --color route*), which is **host** in this case.
- Ping the default gateway.
  - The host OS is Windows so it probably won’t reply.
  - However it should work if your host OS will reply **ICMP**.

VirtualBox / NAT

user/password: ccna/ccna

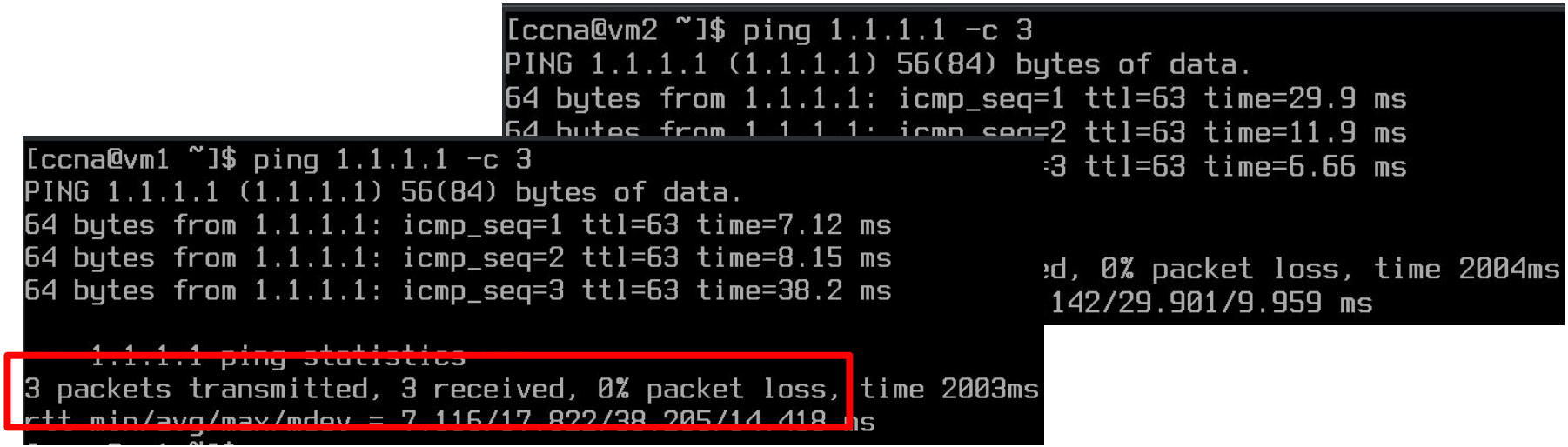
- Check the IP addresses of the vms.
  - They should have the same ip that NAT server gave to the vm.
  - So we can easily tell that they **can't ping each other** directly.



VirtualBox / NAT

user/password: ccna/ccna

- Ping 1.1.1.1
  - Every vm should be able to reach the Internet.



VirtualBox / NAT

user/password: ccna/ccna

- Find the default gateway (*ip --color route*), which is **host** in this case.

```
[ccna@vm1 ~]$ ip --color route
default via 10.0.2.2 dev enp0s3 proto dhcp src 10.0.2.15 metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15 metric 100
```

```
[ccna@vm2 ~]$ ip --color route
default via 10.0.2.2 dev enp0s3 proto dhcp src 10.0.2.15 metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15 metric 100
```

VirtualBox / NAT

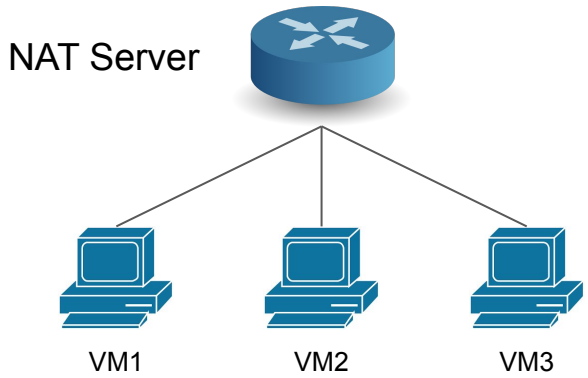
user/password: ccna/ccna

- Ping the default gateway.
  - The host OS is Windows so it probably won't reply.
  - However it should work if your host OS will reply **ICMP**.

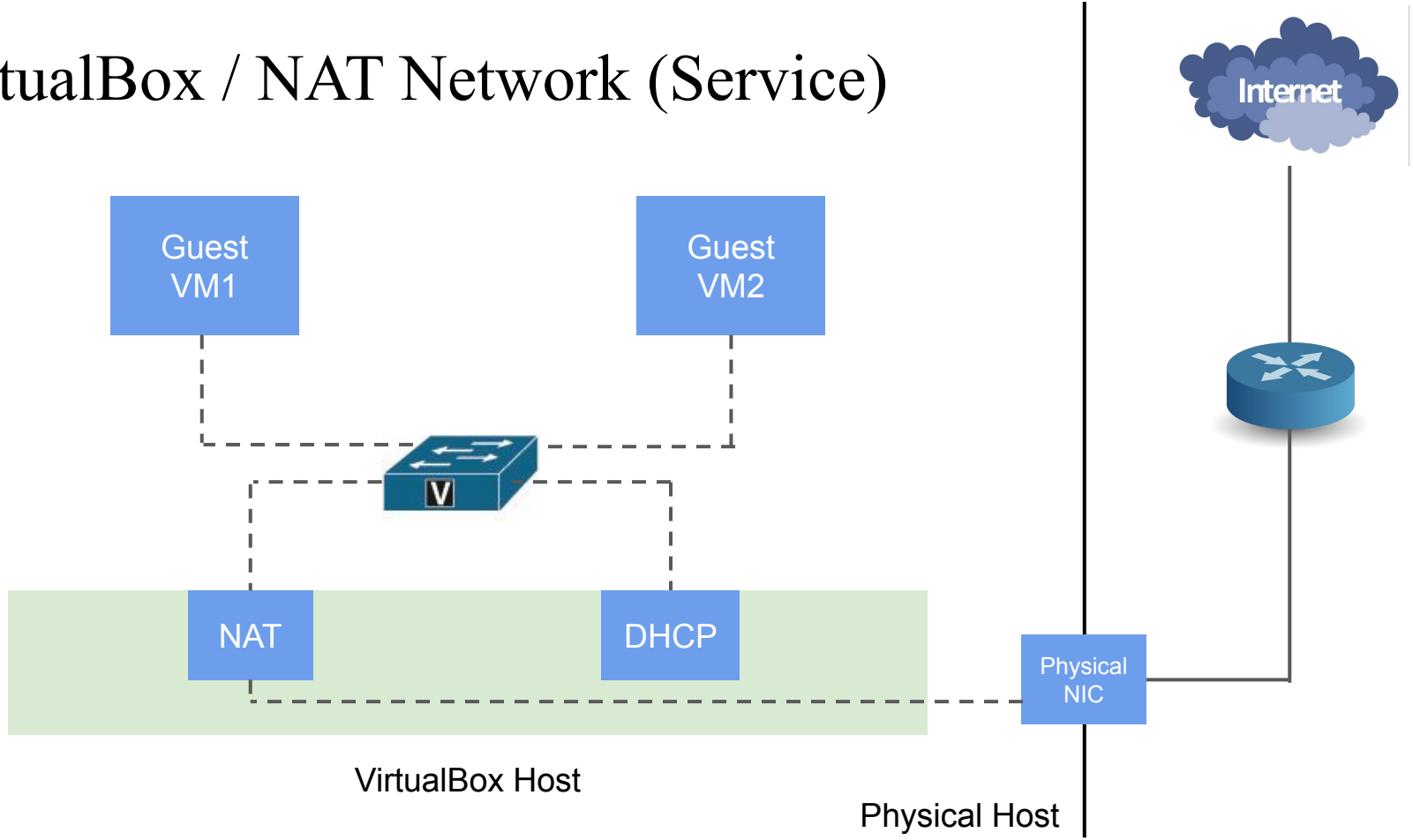
```
[ccna@vm1 ~]$ ip -c r
default via 10.0.2.2 dev enp0s3 proto dhcp src 10.0.2.15 metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15 metric 100
[ccna@vm1 ~]$ ping 10.0.2.2 -c 3
PING 10.0.2.2 (10.0.2.2) 56(84) bytes of data.
64 bytes from 10.0.2.2: icmp_seq=1 ttl=64 time=0.354 ms
64 bytes from 10.0.2.2: icmp_seq=2 ttl=64 time=0.374 ms
64 bytes from 10.0.2.2: icmp_seq=3 ttl=64 time=0.531 ms

--- 10.0.2.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2041ms
rtt min/avg/max/mdev = 0.354/0.419/0.531/0.079 ms
[ccna@vm1 ~]$ ssh 10.0.2.2 -l ytshih
ytshih@10.0.2.2's password:
Last login: Tue Jan 30 21:30:19 2024 from 127.0.0.1
[ytshih@arch-laptop ~]$
```

VirtualBox / NAT Network (Service)

- Works in a similar way to a home router
- Not like previous NAT mode, NAT Network letting systems inside **communicate with each other** and with systems outside using TCP and UDP over IPv4 and IPv6



VirtualBox / NAT Network (Service)

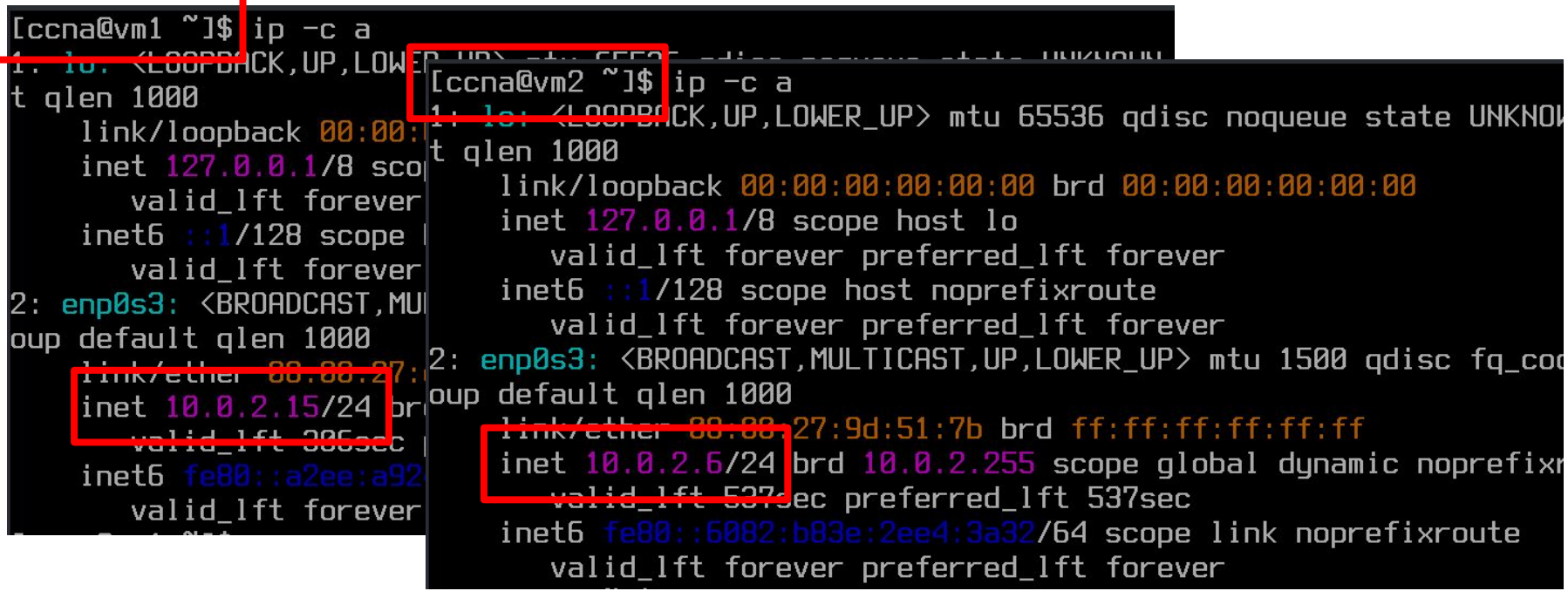


VirtualBox / NAT Network (Service)

- Check their IP addresses.
  - The IP addresses should be **different**, and in the **same subnet**.
- Ping 1.1.1.1
  - Every vm should be able to reach the Internet.
- Ping each other
- Find the default gateway (*ip --color route*), which is host in this case.
- Ping the default gateway.
  - The host OS is Windows so it probably won't reply.
  - However it should work if your host OS will reply ICMP.

VirtualBox / NAT Network (Service)

- Check their IP addresses.
  - The IP addresses should be **different**, and in the **same subnet**.



VirtualBox / NAT Network (Service)

- Ping 1.1.1.1
  - Every vm should be able to reach the Internet.

```
[ccna@vm2 ~]$ ping 1.1.1.1 -c 3
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data.
64 bytes from 1.1.1.1: icmp_seq=1 ttl=57 time=10.9 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=57 time=9.03 ms
64 bytes from 1.1.1.1: icmp_seq=3 ttl=57 time=176 ms

acket loss, time 2003ms
234/78.388 ms
```

```
[ccna@vm1 ~]$ ping 1.1.1.1 -c 3
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data.
64 bytes from 1.1.1.1: icmp_seq=1 ttl=57 time=7.33 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=57 time=6.73 ms
64 bytes from 1.1.1.1: icmp_seq=3 ttl=57 time=31.3 ms

--- 1.1.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2007ms
rtt min/avg/max/mdev = 6.728/15.110/31.273/11.431 ms
```

VirtualBox / NAT Network (Service)

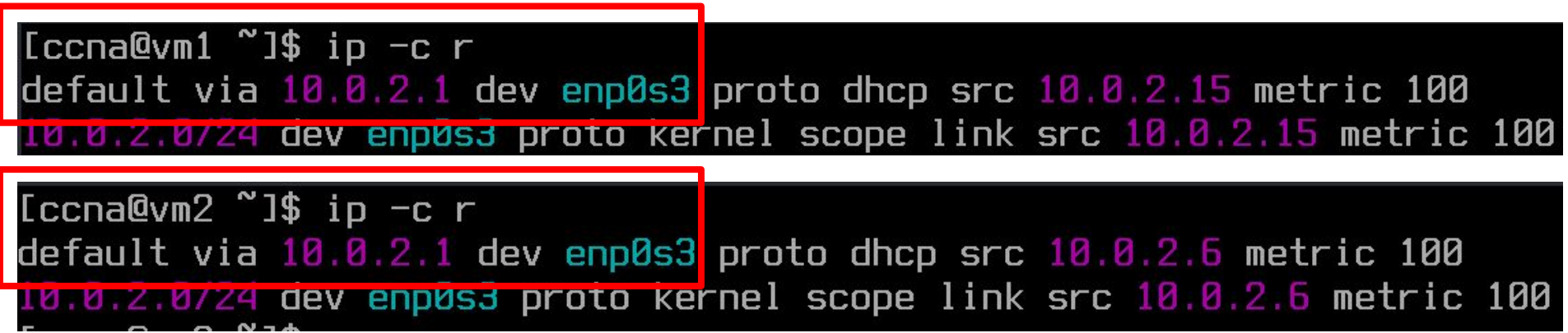
- Ping each other

```
[ccna@vm2 ~]$ ping 10.0.2.15 -c 3
PING 10.0.2.15 (10.0.2.15) 56(84) bytes of data.
64 bytes from 10.0.2.15: icmp_seq=1 ttl=64 time=0.429 ms
64 bytes from 10.0.2.15: icmp_seq=2 ttl=64 time=0.738 ms
64 bytes from 10.0.2.15: icmp_seq=3 ttl=64 time=0.640 ms
```

```
[ccna@vm1 ~]$ ping 10.0.2.6 -c 3
PING 10.0.2.6 (10.0.2.6) 56(84) bytes of data.
64 bytes from 10.0.2.6: icmp_seq=1 ttl=64 time=0.491 ms
64 bytes from 10.0.2.6: icmp_seq=2 ttl=64 time=1.20 ms
64 bytes from 10.0.2.6: icmp_seq=3 ttl=64 time=1.09 ms

--- 10.0.2.6 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2017ms
rtt min/avg/max/mdev = 0.491/0.927/1.203/0.312 ms
```

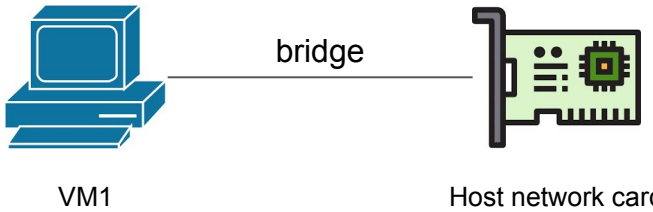
VirtualBox / NAT Network (Service)

- Find the default gateway (*ip --color route*), which is host in this case.

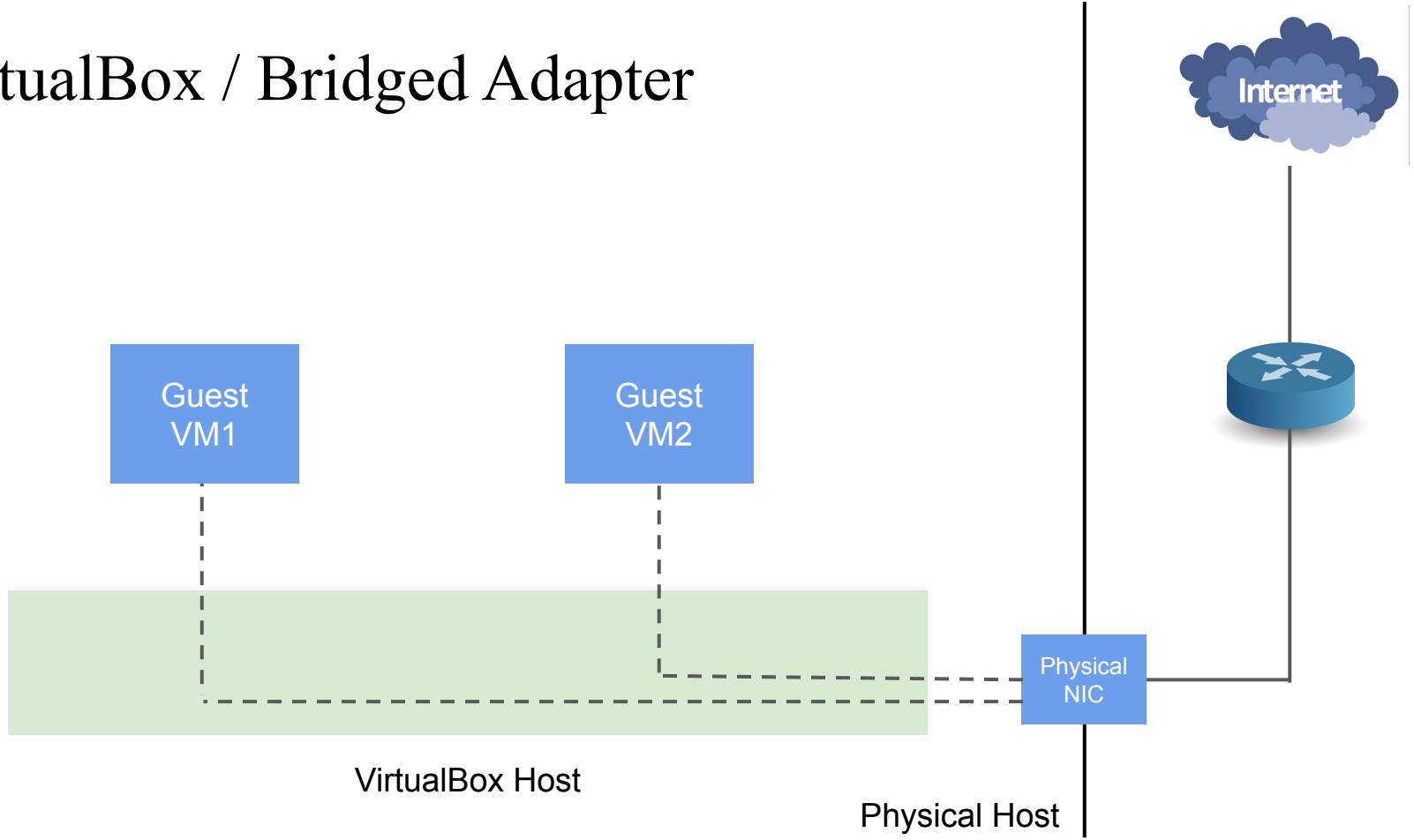


VirtualBox / Bridged Adapter

- Connect to one of your installed network cards and exchanges network packets **directly**
- Uses a device driver on your host system that filters data from your physical network adapter
  - called a *net filter* driver
- Notice that your VM might not get an IP if the gateway of your host does not support it. (e.g. your dorm)
  - You can use personal hotspot or bluetooth / usb tethering on your phone to do this exercise.

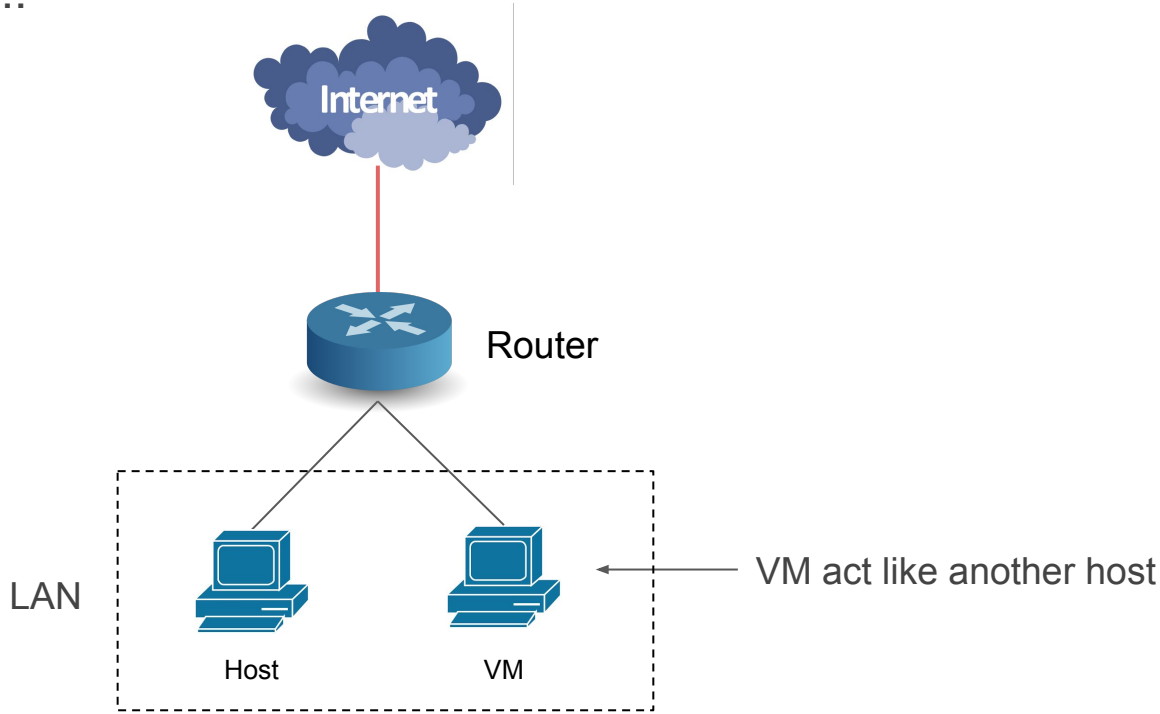


VirtualBox / Bridged Adapter



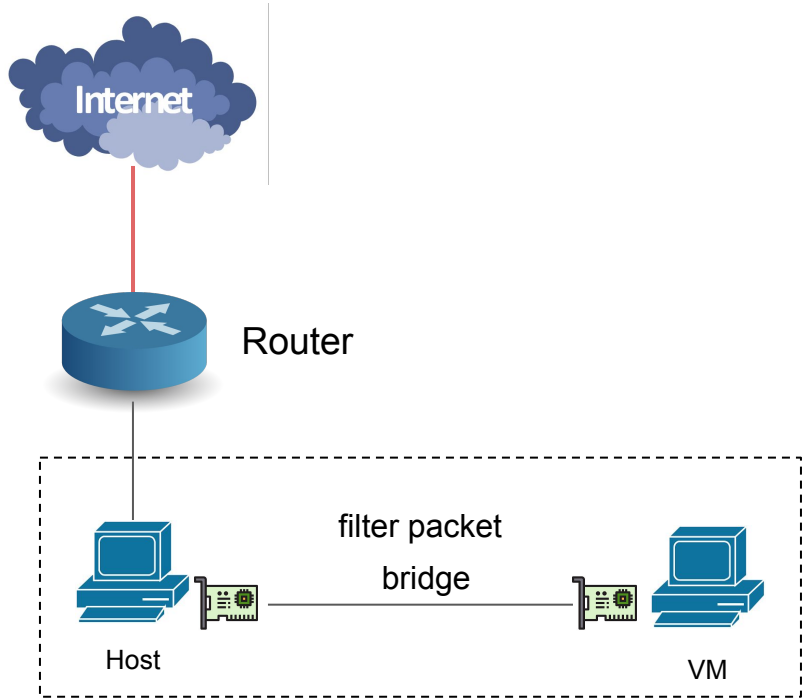
VirtualBox / Bridged Adapter Use Case

- What you try to do...



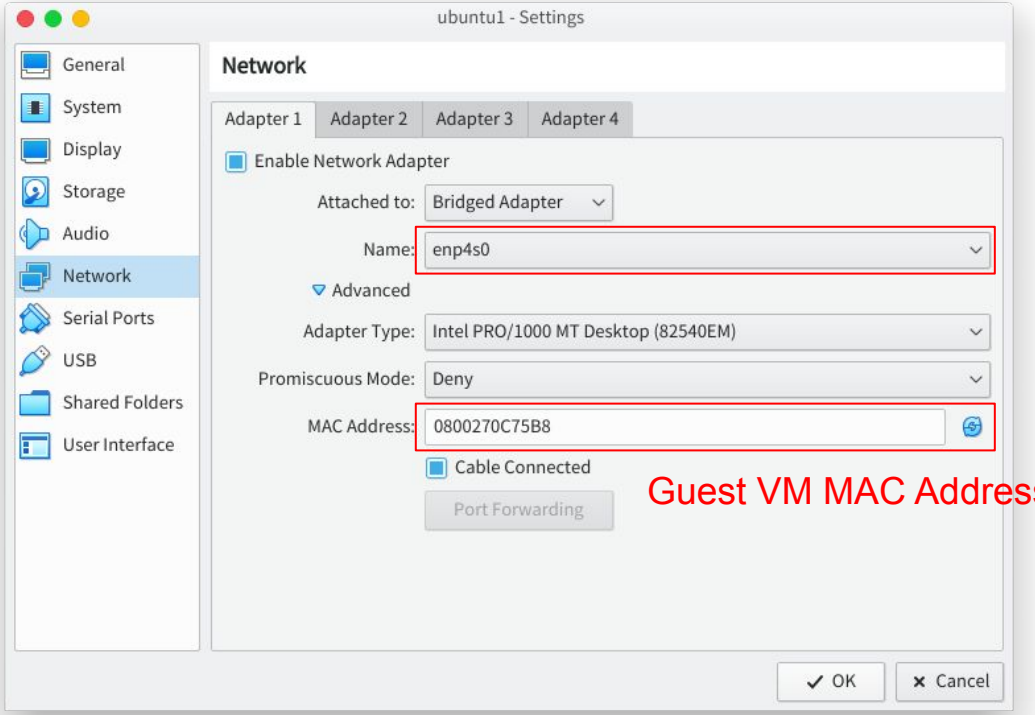
VirtualBox / Bridged Adapter Use Case

- What VirtualBox actually does...



In same machine

VirtualBox / Bridged Adapter Setting



VirtualBox / Bridged Adapter

- Check their IP addresses.
  - The IP addresses should be **different**, and in the same subnet **as host**.
- Ping 1.1.1.1
  - Every vm should be able to reach the Internet.
- Ping each other (vm1, vm2, host)
- Find the default gateway (*ip --color route*), which is **the same as host**.
- Ping the default gateway.
  - This should work unless your router won’t reply ICMP.

VirtualBox / Bridged Adapter

- Check their IP addresses.
  - The IP addresses should be **different**, and in the same subnet **as host**.

```
[ccna@vm1 ~]$ ip -c a show enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_coo
oup default qlen 1000
    link/ether 08:00:27:e1:87:55 brd ff:ff:ff:ff:ff:ff
    inet 192.168.68.74/22 brd 192.168.71.255 scope global dynamic
enp0s3
```

```
[ccna@vm2 ~]$ ip -c a show enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_co
oup default qlen 1000
    link/ether 08:00:27:9d:51:7b brd ff:ff:ff:ff:ff:ff
    inet 192.168.68.75/22 brd 192.168.71.255 scope global dynamic
enp0s3
```

```
[ytshih@arch-laptop ~]$ ip -c a show wlp0s20f3
2: wlp0s20f3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
ult qlen 1000
    link/ether 3c:21:9c:cd:eb:0f brd ff:ff:ff:ff:ff:ff
    inet 192.168.68.77/22 brd 192.168.71.255 scope global dyna
```

VirtualBox / Bridged Adapter

- Ping 1.1.1.1
  - Every vm should be able to reach the Internet.

```
[ccna@vm2 ~]$ ping 1.1.1.1 -c 3
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data.
64 bytes from 1.1.1.1: icmp_seq=1 ttl=58 time=93.7 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=58 time=18.3 ms
64 bytes from 1.1.1.1: icmp_seq=3 ttl=58 time=20.5 ms
```

```
[ccna@vm1 ~]$ ping 1.1.1.1 -c 3
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data.
64 bytes from 1.1.1.1: icmp_seq=1 ttl=58 time=16.3 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=58 time=11.5 ms
64 bytes from 1.1.1.1: icmp_seq=3 ttl=58 time=23.5 ms

--- 1.1.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 11.549/17.120/23.472/4.898 ms
```

VirtualBox / Bridged Adapter

- Ping each other (vm1, vm2, host)

```
[ccna@vm1 ~]$ ping 192.168.68.75 -c 3
PING 192.168.68.75 (192.168.68.75) 56(84) bytes of data.
64 bytes from 192.168.68.75: icmp_seq=1 ttl=64 time=0.703 ms
64 bytes from 192.168.68.75: icmp_seq=2 ttl=64 time=0.410 ms
64 bytes from 192.168.68.75: icmp_seq=3 ttl=64 time=1.00 ms

--- 192.168.68.75 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2022ms
rtt min/avg/max/mdev = 0.410/0.704/1.001/0.241 ms
[ccna@vm1 ~]$ ping 192.168.68.77 -c 3
PING 192.168.68.77 (192.168.68.77) 56(84) bytes of data.
64 bytes from 192.168.68.77: icmp_seq=1 ttl=64 time=0.812 ms
64 bytes from 192.168.68.77: icmp_seq=2 ttl=64 time=0.441 ms
64 bytes from 192.168.68.77: icmp_seq=3 ttl=64 time=0.210 ms

--- 192.168.68.77 ping statistics ---
3 packets transmitted, 3 received, 0% packet
rtt min/avg/max/mdev = 0.210/0.487/0.812/0.2
```

```
[ytshih@arch-laptop ~]$ ping 192.168.68.74 -c 3
PING 192.168.68.74 (192.168.68.74) 56(84) bytes of data.
64 bytes from 192.168.68.74: icmp_seq=1 ttl=64 time=0.467 ms
64 bytes from 192.168.68.74: icmp_seq=2 ttl=64 time=0.487 ms
64 bytes from 192.168.68.74: icmp_seq=3 ttl=64 time=0.326 ms

--- 192.168.68.74 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2026ms
rtt min/avg/max/mdev = 0.326/0.426/0.487/0.071 ms
```

VirtualBox / Bridged Adapter

- Find the default gateway (*ip --color route*), which is **the same as host**.

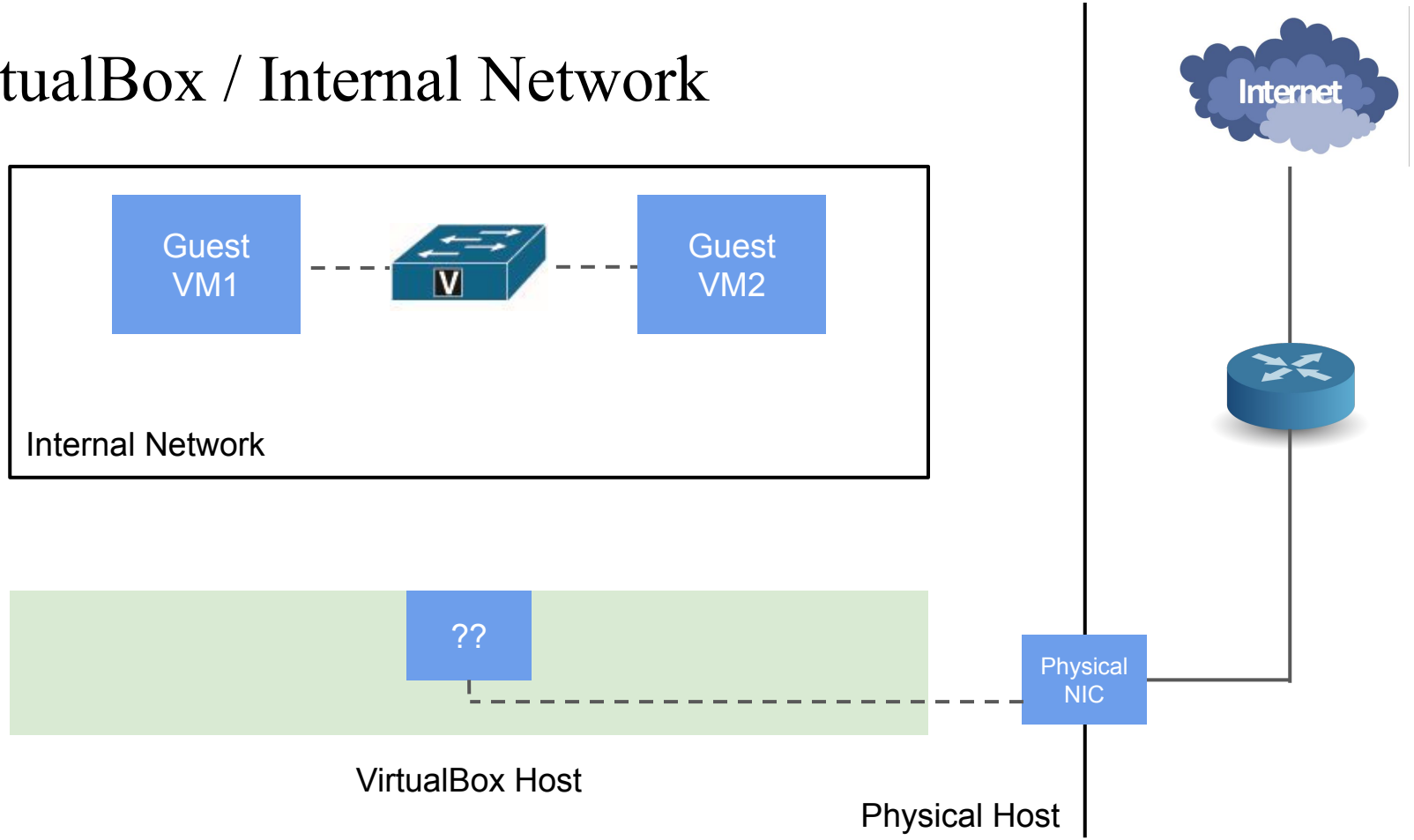
```
[ccna@vm1 ~]$ ip -c r
default via 192.168.68.1 dev enp0s3 proto dhcp src 192.168.68.74 metric 100
192.168.68.0/22 dev enp0s3 proto kernel scope link src 192.168.68.74 metric 100
```

```
[ytshih@arch-laptop ~]$ ip -c r
default via 192.168.68.1 dev wlp0s20f3 proto dhcp src 192.168.68.77 metric 600
192.168.68.0/22 dev wlp0s20f3 proto kernel scope link src 192.168.68.77 metric 600
```

VirtualBox / Internal Network

- Guest VM connected to an **isolated** virtual network
- VMs connected to same internal network can communicate with each other
- VMs cannot communicate with VirtualBox host
- VMs cannot communicate with any other hosts in external networks
- Internal network can be used for **modelling real networks**

VirtualBox / Internal Network



VirtualBox / Internal Network

- Check their IP addresses.
  - There are **no IP addresses** assigned to the vms (no DHCP).
- Set their IP addresses manually.
  - *sudo ip addr add dev enp0s3 10.0.0.1/24* on vm1, *10.0.0.2/24* on vm2
- Ping 1.1.1.1
  - Vms should **not** be able to access the internet.
- Ping each other.
  - Vms should be able to ping each other.
- Find the default gateway (*ip --color route*)
  - There is **no** default gateway unless you explicitly set it.

VirtualBox / Internal Network

- Check their IP addresses.
  - There are **no IP addresses** assigned to the vms (no DHCP).

```
[ccna@vm1 ~]$ ip -c a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defa
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
oup default qlen 1000
    link/ether 08:00:27:e1:87:55 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::a2ee:a924:6947:5d28/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

VirtualBox / Internal Network

- Set IP addresses manually.
  - *sudo ip addr add dev enp0s3 10.0.0.1/24* on vm1, *10.0.0.2/24* on vm2

```
[ccna@vm1 ~]$ sudo ip addr add dev enp0s3 10.0.0.1/24
[ccna@vm1 ~]$ ip -c a show enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
oup default qlen 1000
    link/ether 08:00:27:e1:87:55 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.1/24 scope global enp0s3
```

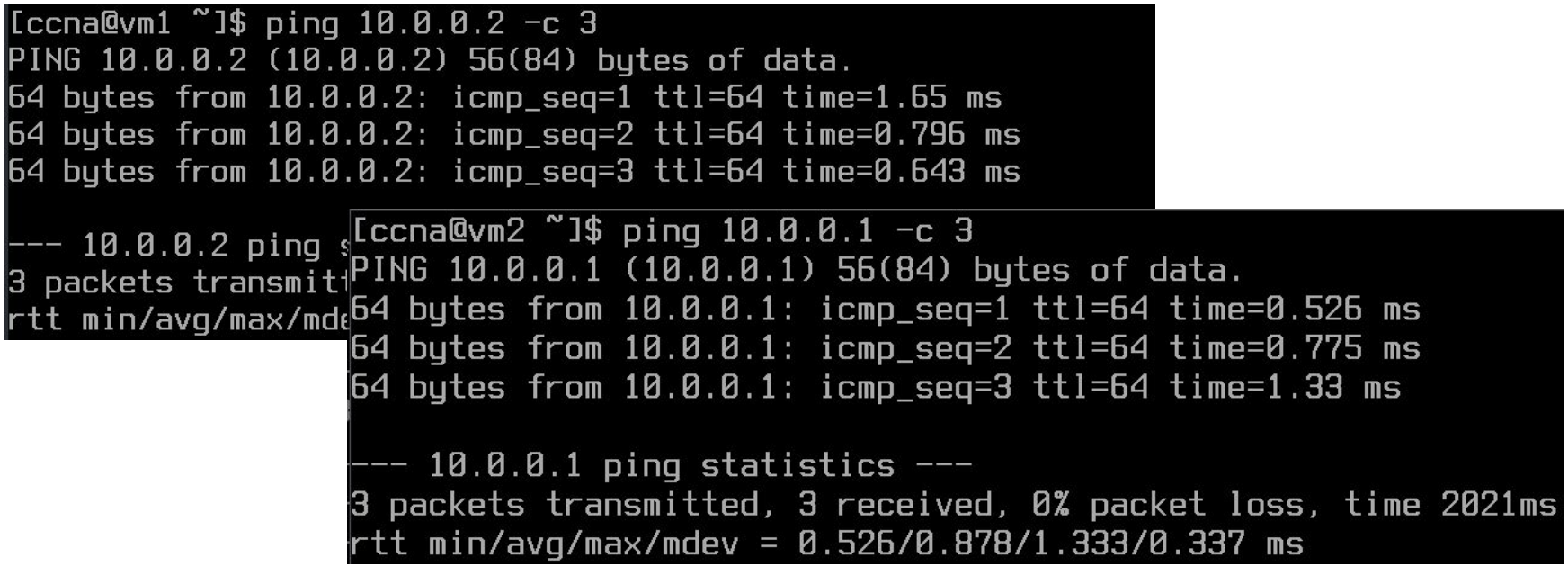
```
[ccna@vm2 ~]$ sudo ip addr add dev enp0s3 10.0.0.2/24
[ccna@vm2 ~]$ ip -c a show enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
oup default qlen 1000
    link/ether 08:00:27:9d:51:7b brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.2/24 scope global enp0s3
```

VirtualBox / Internal Network

- Ping 1.1.1.1
  - Vms should **not** be able to access the internet.

VirtualBox / Internal Network

- Ping each other.
  - Vms should be able to ping each other.



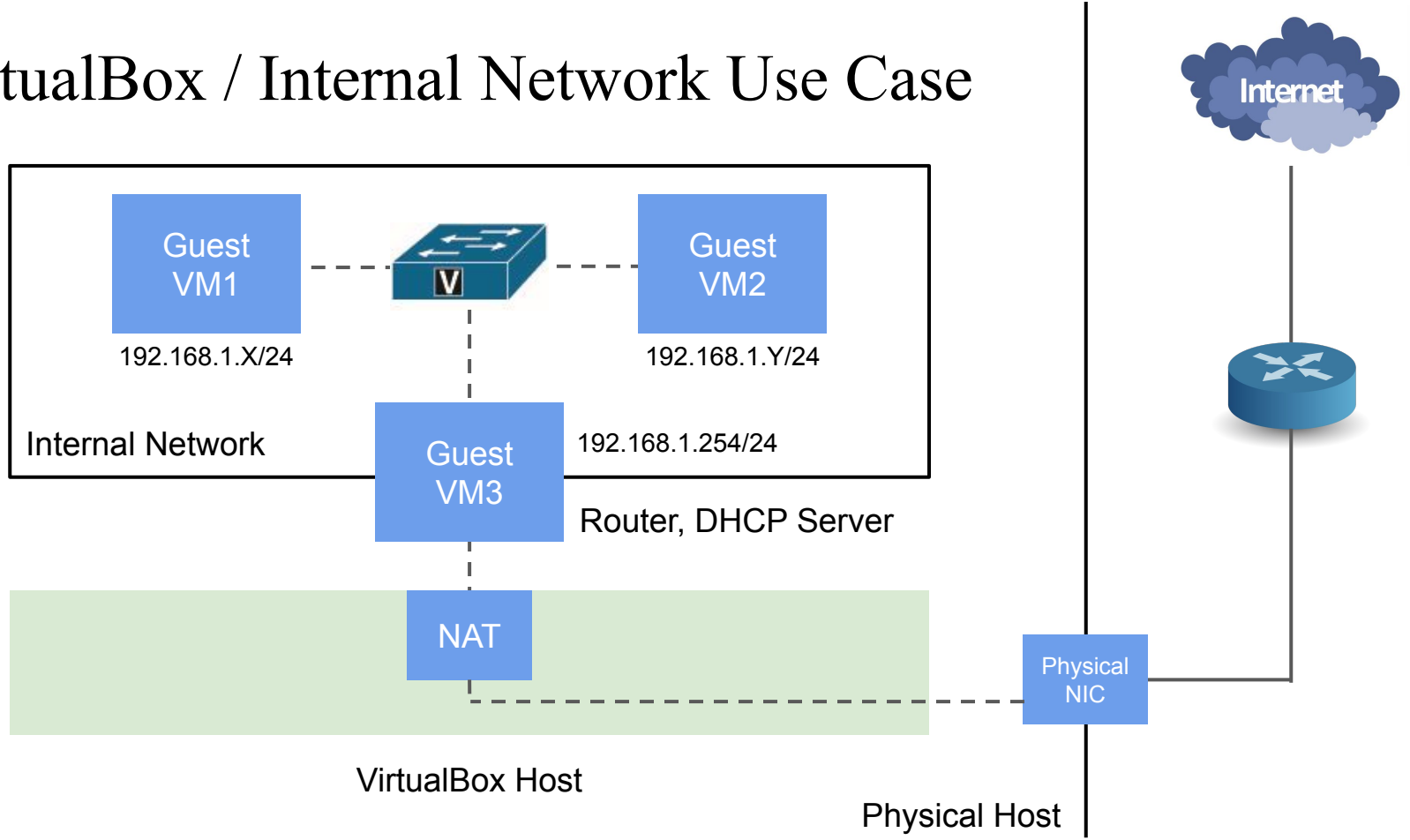
VirtualBox / Internal Network

- Find the default gateway (*ip --color route*)
  - There is **no** default gateway unless you explicitly set it.

```
[ccna@vm1 ~]$ ip -c r
10.0.0.0/24 dev enp0s3 proto kernel scope link src 10.0.0.1
```

```
[ccna@vm2 ~]$ ip -c r
10.0.0.0/24 dev enp0s3 proto kernel scope link src 10.0.0.2
```

VirtualBox / Internal Network Use Case



VirtualBox / Internal Network Use Case

- Make sure the MAC addresses on vm3 are correct.
  - Adapter 1: NAT / MACAddress: 08:00:00:00:00:01
  - Adapter 2: Internal Network / MACAddress: 08:00:00:00:00:02
- Check their IP addresses.
  - The IP addresses on vms should be **different**, and in the same subnet.
- Ping 1.1.1.1
  - Every vm should be able to reach the Internet.
- Ping each other.
- Find the default gateway of vm1 & vm2 (*ip --color route*), which is **vm3**.
- Ping the default gateway.
  - The default gateway of vm1 & vm2, aka vm3, will reply ICMP.

VirtualBox / Internal Network Use Case

user/password: ccna/ccna

- Check their IP addresses.
  - The IP addresses on vms should be **different**, and in the same subnet.

```
[ccna@vm1 ~]$ ip -c a show enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
oup default qlen 1000
    link/ether 08:00:27:e1:87:55 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.5/24 brd 192.168.1.255 scope global dynamic noprefixroute
```

```
[ccna@vm2 ~]$ ip -c a show enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
oup default qlen 1000
    link/ether 08:00:27:9d:51:7b brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.13/24 brd 192.168.1.255 scope global dynamic noprefixroute
```

VirtualBox / Internal Network Use Case

- Check their IP addresses.
  - The IP addresses on vms should be **different**, and in the same subnet.

user/password: ccna/ccna

```
[ccna@vm3 ~]$ ip -c a show enp0s9
2: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
oup default qlen 1000
    link/ether 08:00:27:eb:db:84 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.254/24 brd 192.168.1.255 scope global enp0s9
```

```
[ccna@vm3 ~]$ ip -c a show enp0s10
3: enp0s10: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
roup default qlen 1000
    link/ether 08:00:27:fd:a1:16 brd ff:ff:ff:ff:ff:ff
    inet 10.0.5.15/24 metric 1024 brd 10.0.5.255 scope global dynamic enp0s10
```

VirtualBox / Internal Network Use Case

- Ping 1.1.1.1
  - Every vm should be able to reach the Internet.

user/password: ccna/ccna

```
[ccna@vm1 ~]$ ping 1.1.1.1 -c 3
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data.
64 bytes from 1.1.1.1: icmp_seq=1 ttl=61 time=9.45 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=61 time=13.8 ms
64 bytes from
--- 1.1.1.1 pi
3 packets tran
rtt min/avg/ma
```

```
[ccna@vm2 ~]$ ping 1.1.1.1 -c 3
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data.
64 bytes from 1.1.1.1: icmp_seq=1 ttl=61 time=10.4 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=61 time=13.3 ms
64 bytes from 1
--- 1.1.1.1 pin
3 packets trans
rtt min/avg/max
```

```
[ccna@vm3 ~]$ ping 1.1.1.1 -c 3
PING 1.1.1.1 (1.1.1.1) 56(84) bytes of data.
64 bytes from 1.1.1.1: icmp_seq=1 ttl=63 time=20.7 ms
64 bytes from 1.1.1.1: icmp_seq=2 ttl=63 time=11.0 ms
64 bytes from 1.1.1.1: icmp_seq=3 ttl=63 time=11.6 ms

--- 1.1.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 11.043/14.444/20.698/4.427 ms
```

VirtualBox / Internal Network Use Case

- Ping each other.
  - vm1 ⇔ vm2
  - vm1 ⇔ vm3
  - vm2 ⇔ vm3

```
[ccna@vm1 ~]$ ping 192.168.1.13 -c 3
PING 192.168.1.13 (192.168.1.13) 56(84) bytes of data.
64 bytes from 192.168.1.13: icmp_seq=1 ttl=64 time=0.684 ms
64 bytes from 192.168.1.13: icmp_seq=2 ttl=64 time=0.442 ms
64 bytes from 192.168.1.13: icmp_seq=3 ttl=64 time=0.960 ms

--- 192.168.1.13 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2034ms
rtt min/avg/max/mdev = 0.442/0.695/0.960/0.211 ms
[ccna@vm1 ~]$ ping 192.168.1.254 -c 3
PING 192.168.1.254 (192.168.1.254) 56(84) bytes of data.
64 bytes from 192.168.1.254: icmp_seq=1 ttl=64 time=0.901 ms
64 bytes from 192.168.1.254: icmp_seq=2 ttl=64 time=1.37 ms
64 bytes from 192.168.1.254: icmp_seq=3 ttl=64 time=0.391 ms

--- 192.168.1.254 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.391/0.888/1.374/0.401 ms
```

VirtualBox / Internal Network Use Case

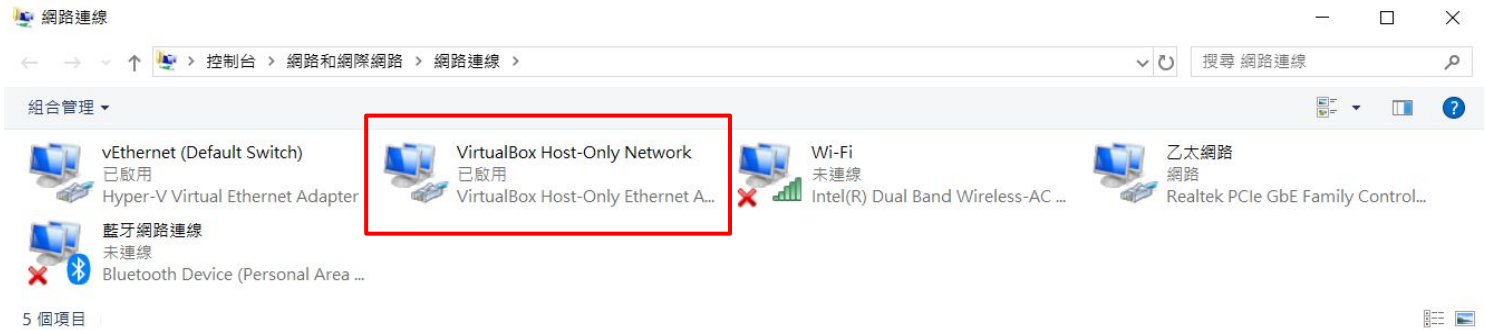
- Find the default gateway of vm1 & vm2 (*ip --color route*), which is **vm3**.

```
[ccna@vm1 ~]$ ip -c r
default via 192.168.1.254 dev enp0s3 proto dhcp src 192.168.1.5 metric 100
192.168.1.0/24 dev enp0s3 proto kernel scope link src 192.168.1.5 metric 100
```

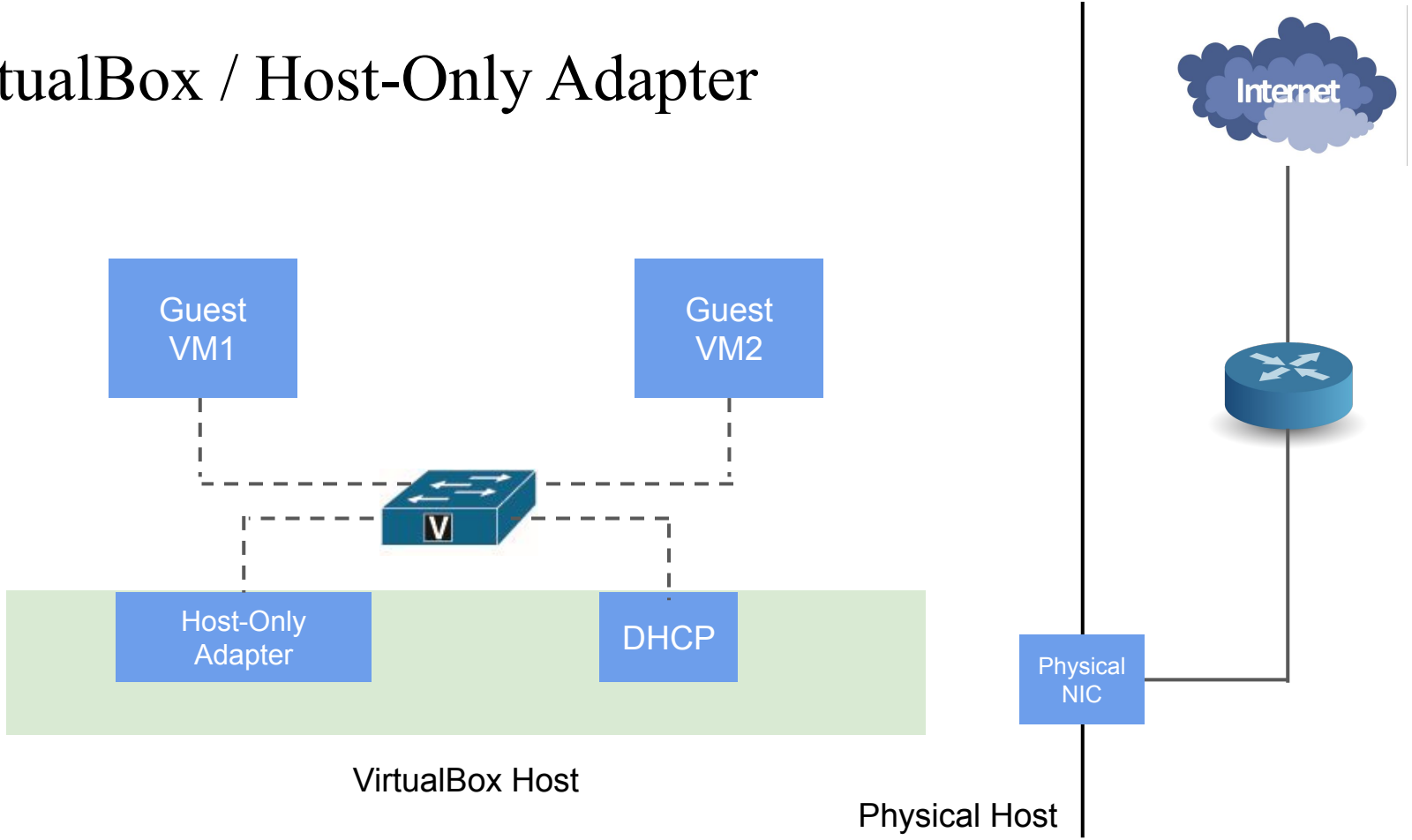
```
[ccna@vm2 ~]$ ip -c r
default via 192.168.1.254 dev enp0s3 proto dhcp src 192.168.1.13 metric 100
192.168.1.0/24 dev enp0s3 proto kernel scope link src 192.168.1.13 metric 100
```

VirtualBox / Host-Only Adapter

- Used for communicating between a host and guests
- A VM can communicate with other VMs connected to the host-only network, and **with the host machine**
- Instead, a virtual network interface, similar to a loopback interface, is created on the host, **providing connectivity among virtual machines and the host**



VirtualBox / Host-Only Adapter



VirtualBox / Host-Only Adapter

- Check their IP addresses.
  - The IP addresses on vms should be **different**, and in the same subnet **as the host-only NIC on host**.
- Ping 1.1.1.1
  - Every vm should **not** be able to reach the Internet.
- Ping each other and host.
- Find the default gateway
  - There is no default gateway on both vms.

VirtualBox / Host-Only Adapter

- Check their IP addresses.
  - The IP addresses on vms should be **different**, and in the same subnet **as the host-only NIC on host**. (You may have to manually set the vboxnet0 interface in linux host.)

```
[ytshih@arch-laptop ~]$ ip -c a show vboxnet0
3: vboxnet0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
ult qlen 1000
    link/ether 0a:00:27:00:00:00 brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.100/24 scope global vboxnet0
```

```
[ccna@vm1 ~]$ ip -c a show enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
oup default qlen 1000
    link/ether 08:00:27:e1:87:55 brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.101/24 brd 192.168.56.255 scope global dynamic noprefixro
 enp0s3
```

```
[ccna@vm2 ~]$ ip -c a show enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
oup default qlen 1000
    link/ether 08:00:27:9d:51:7b brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.102/24 brd 192.168.56.255 scope global dynamic noprefixrou
 enp0s3
```

VirtualBox / Host-Only Adapter

- Ping 1.1.1.1
  - Every vm should **not** be able to reach the Internet.

```
[ccna@vm1 ~]$ ping 1.1.1.1 -c 3
ping: connect: Network is unreachable
```

VirtualBox / Host-Only Adapter

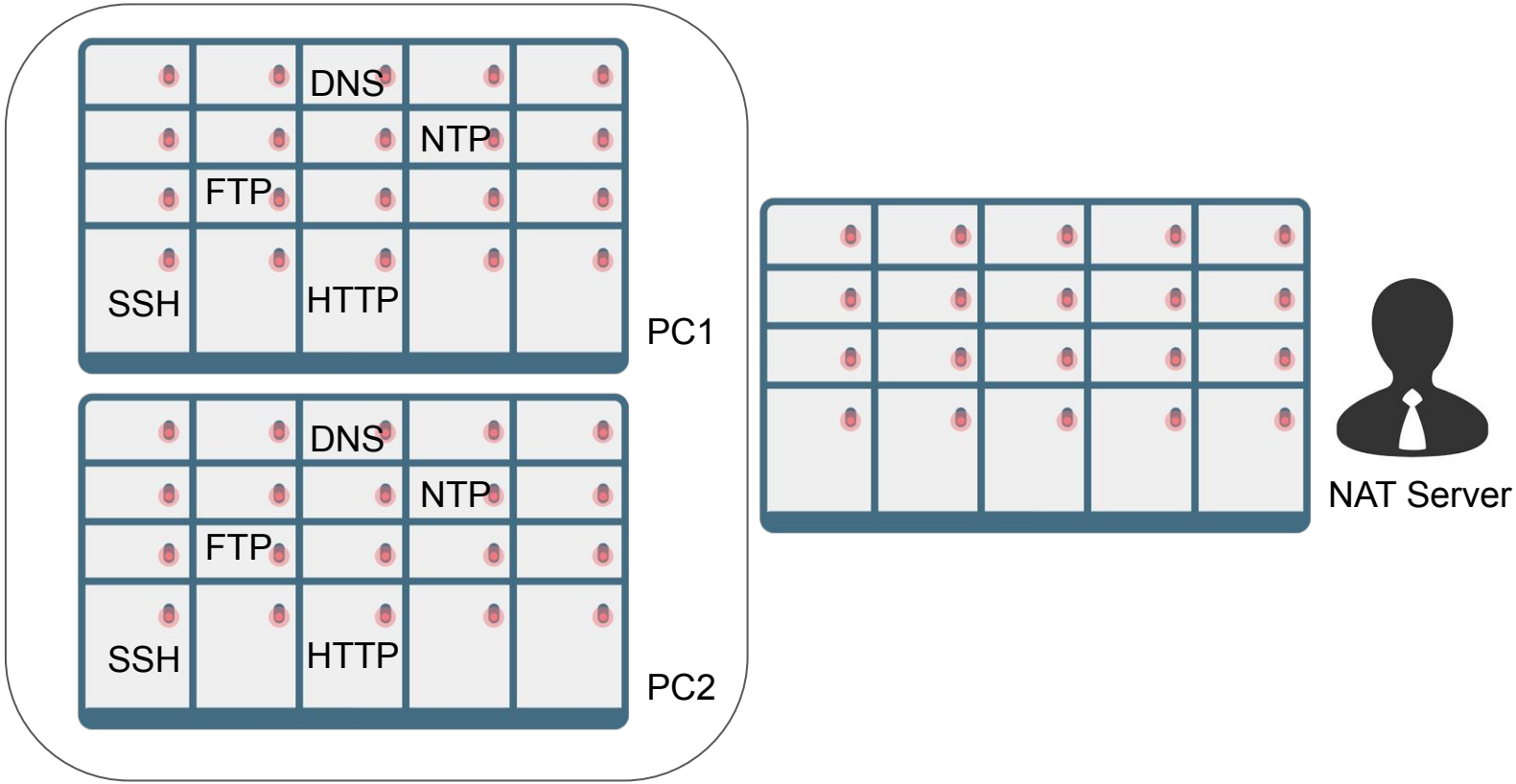
- Ping each other and host.

```
[ccna@vm1 ~]$ ping 192.168.56.100 -c 3
PING 192.168.56.100 (192.168.56.100) 56(84) bytes of data.
64 bytes from 192.168.56.100: icmp_seq=1 ttl=64 time=0.206 ms
64 bytes from 192.168.56.100: icmp_seq=2 ttl=64 time=0.222 ms
64 bytes from 192.168.56.100: icmp_seq=3 ttl=64 time=0.211 ms

--- 192.168.56.100 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2028ms
rtt min/avg/max/mdev = 0.206/0.213/0.222/0.006 ms
[ccna@vm1 ~]$ ping 192.168.56.102 -c 3
PING 192.168.56.102 (192.168.56.102) 56(84) bytes of data.
64 bytes from 192.168.56.102: icmp_seq=1 ttl=64 time=0.665 ms
64 bytes from 192.168.56.102: icmp_seq=2 ttl=64 time=0.545 ms
64 bytes from 192.168.56.102: icmp_seq=3 ttl=64 time=0.748 ms

--- 192.168.56.102 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.545/0.652/0.748/0.083 ms
```

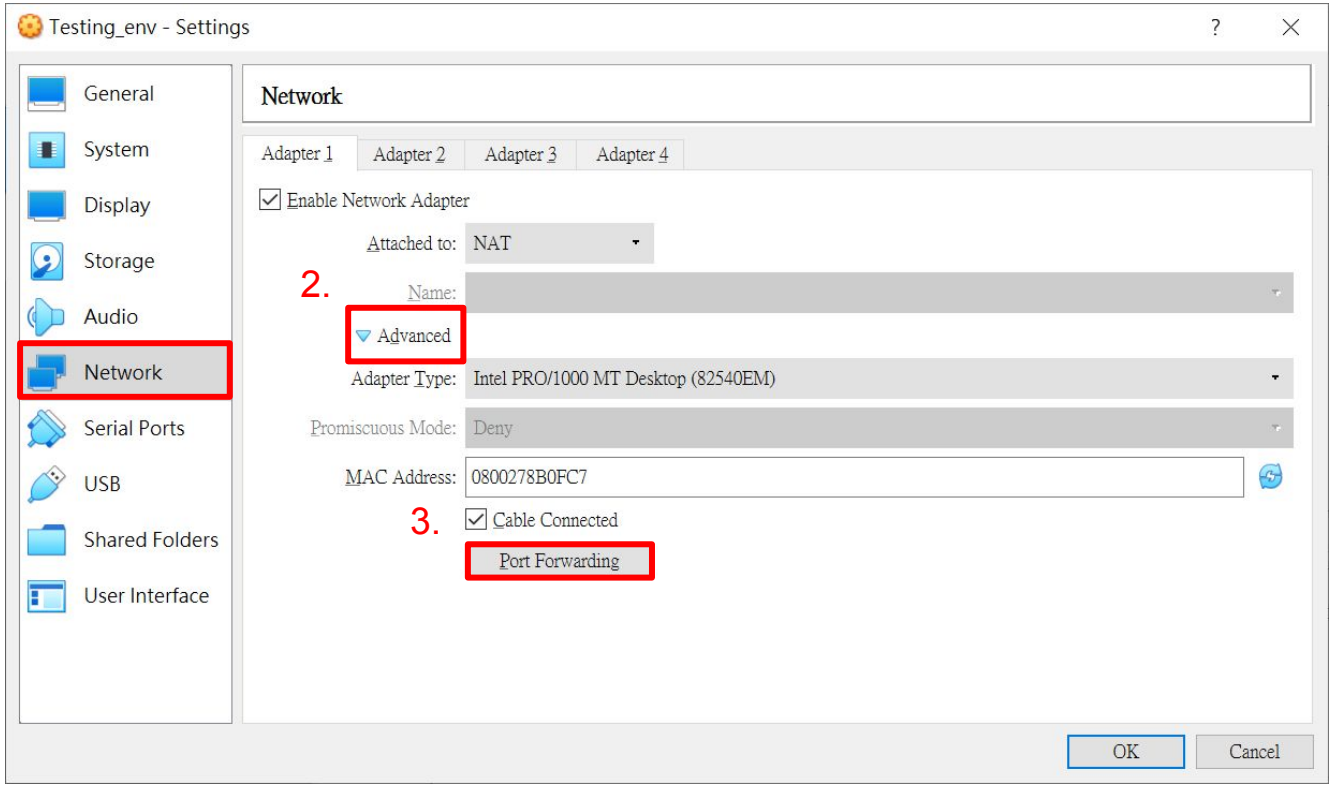
VM connection

Mode	**VM → Host**	**VM ← Host**	**VM1 ↔ VM2**	**VM → Net/LAN**	**VM ← Net/LAN**
Host-only	**+**	**+**	**+**	**–**	**–**
Internal	**–**	**–**	**+**	**–**	**–**
Bridged	**+**	**+**	**+**	**+**	**+**
NAT	**+**	Port forwarding	**–**	**+**	Port forwarding
NATservice	**+**	Port forwarding	**+**	**+**	Port forwarding

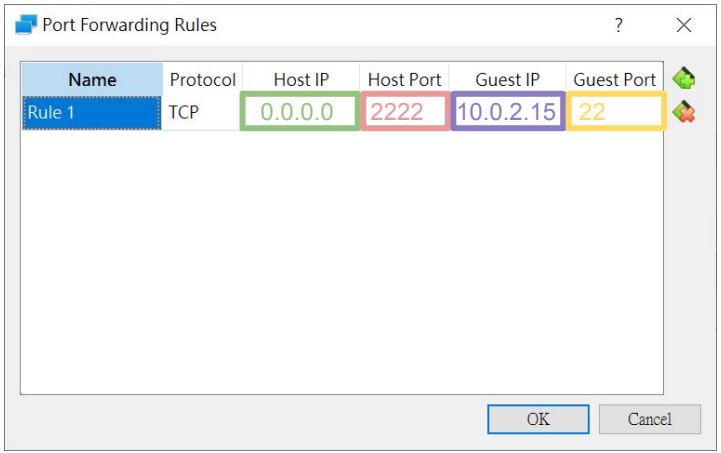
Port Forwarding

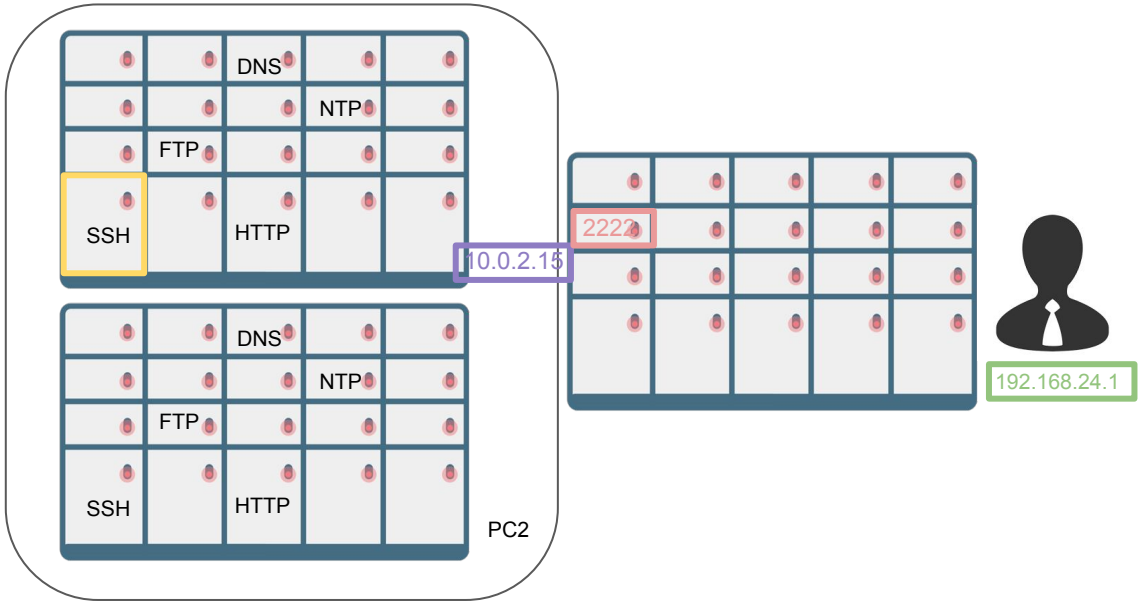


VB Port Forwarding Setting



VB Port Forwarding Setting





VB Port Forwarding Setting

- Try ssh from host.

```
[ytshih@arch-desktop ~]$ ssh localhost -p 2222 -l ccna
ccna@localhost's password:
Last login: Sat Feb 17 13:36:04 2024 from 10.0.2.2
[ccna@vm1 ~]$
```

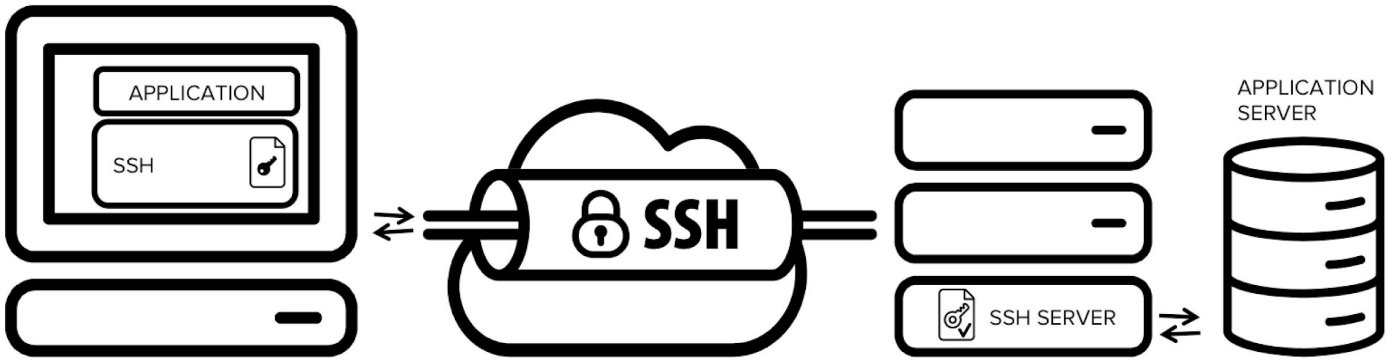
SSH Tunnel

- A method of transporting arbitrary networking data over an encrypted SSH connection.
- Provides a way to secure the data traffic of any given application using port forwarding, basically tunneling any TCP / IP port over SSH.

https://www.ssh.com/ssh/tunneling/

SSH Tunnel Procedure

1. Application contacts to a port that the SSH listens on.
2. SSH forwards the application over its encrypted tunnel to the other side.
3. SSH then connects to the actual application server.

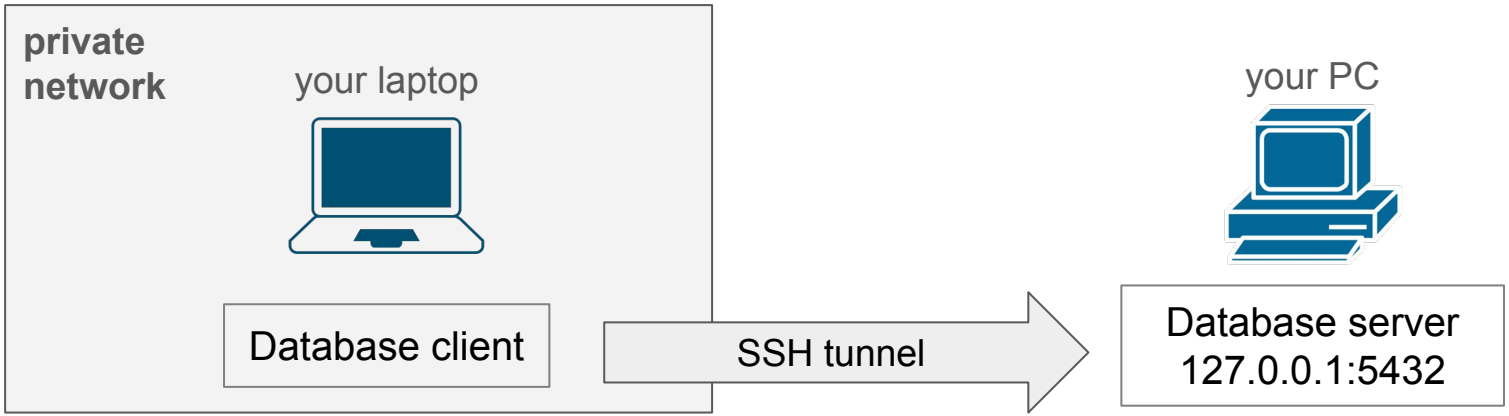


SSH Local Port Forwarding - Scenario

- You're doing your webapp homework with an laptop at Starbucks.
- Your webapp is running on your PC in your dorm with a public ip.
- You want to access your database from your laptop.
  - The database server will only accecpt connections from localhost.

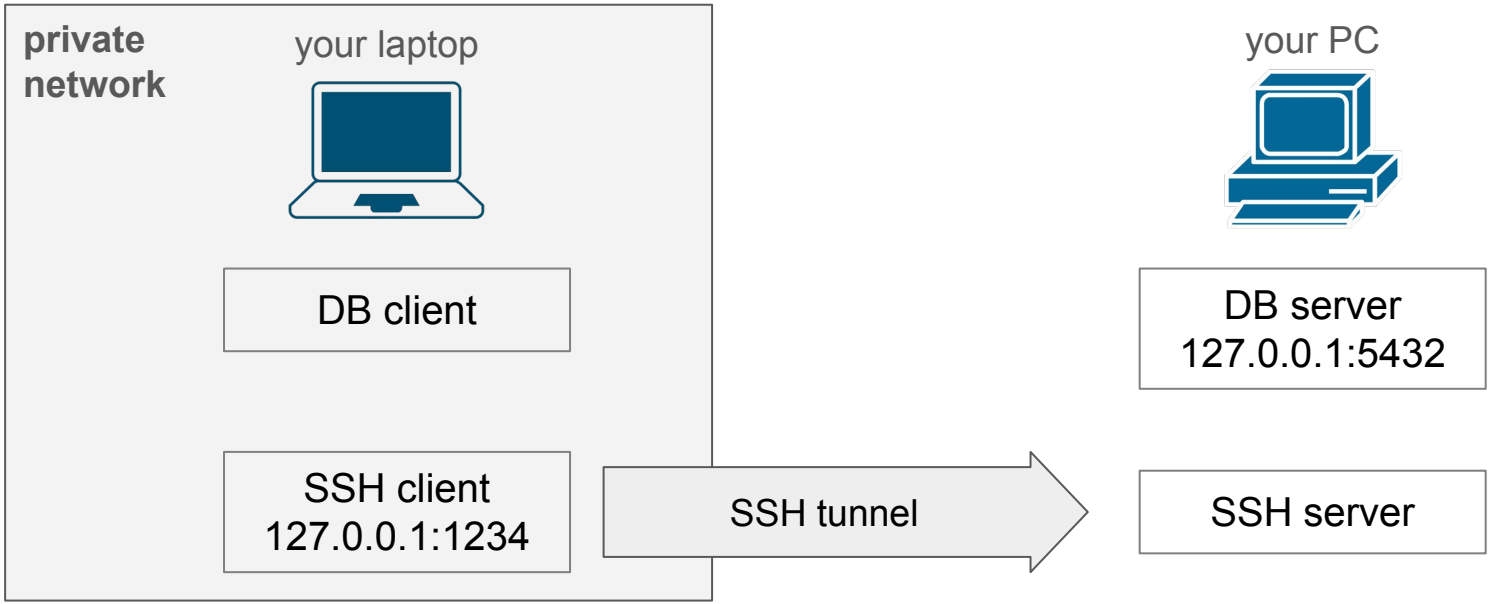
SSH Local Port Forwarding - Solution

- **`-L`** [*`bind_address`*`:`]*`port`*`:`*`host`*`:`*`hostport`*
- Specifies that the given port on the local (client) host is to be forwarded to the given host and port on the remote side.



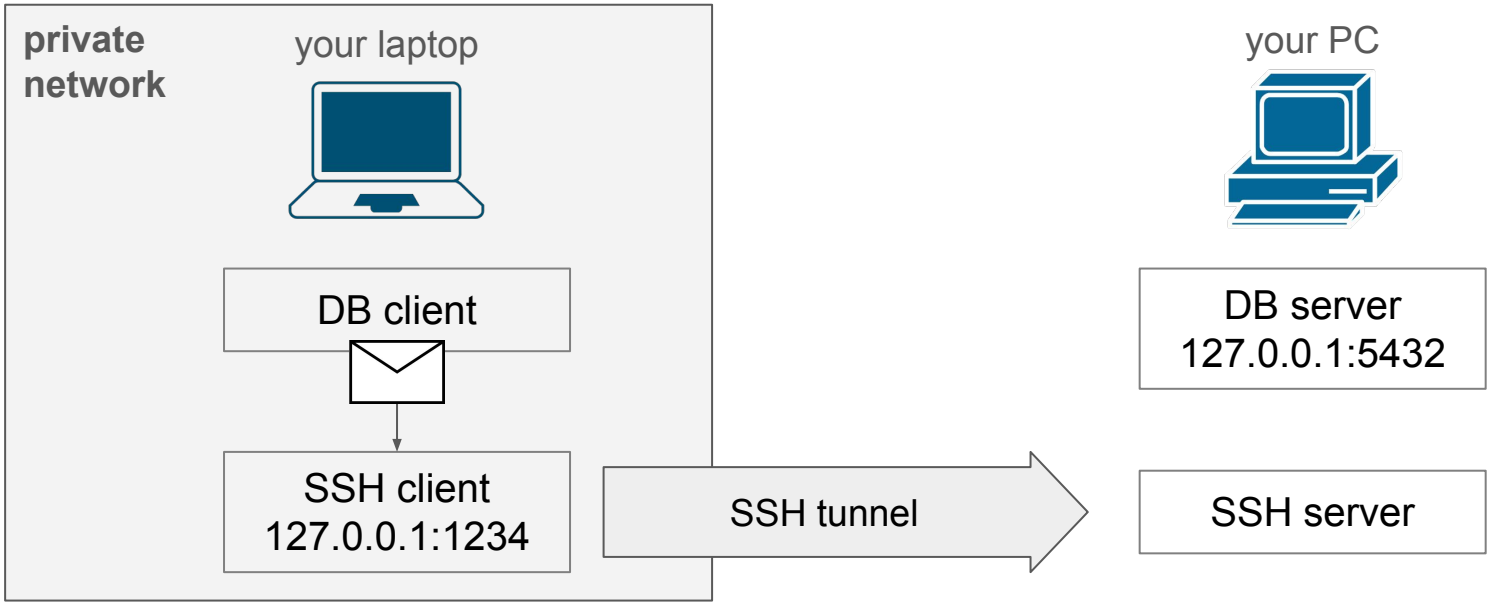
SSH Local Port Forwarding - Explanation

- Initialize an SSH connection from your laptop to your PC
  - Create an SSH tunnel



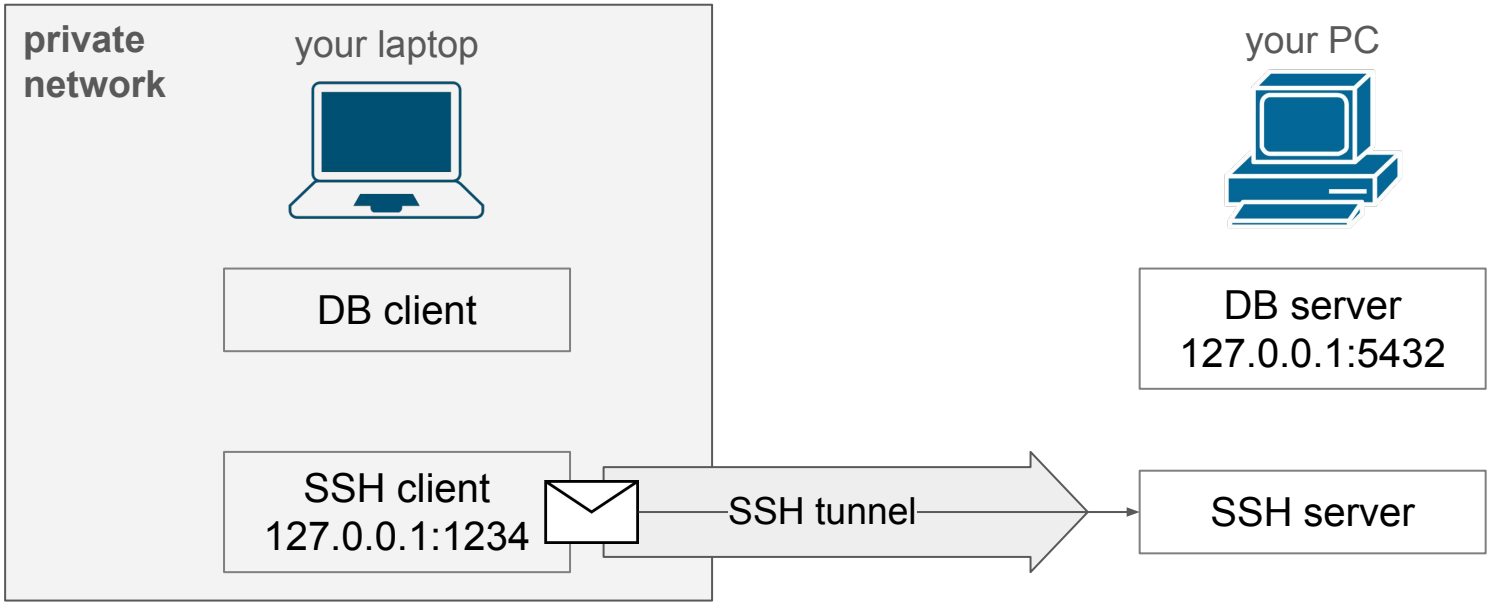
SSH Local Port Forwarding - Explanation

- Database client sends a packet to 127.0.0.1:1234



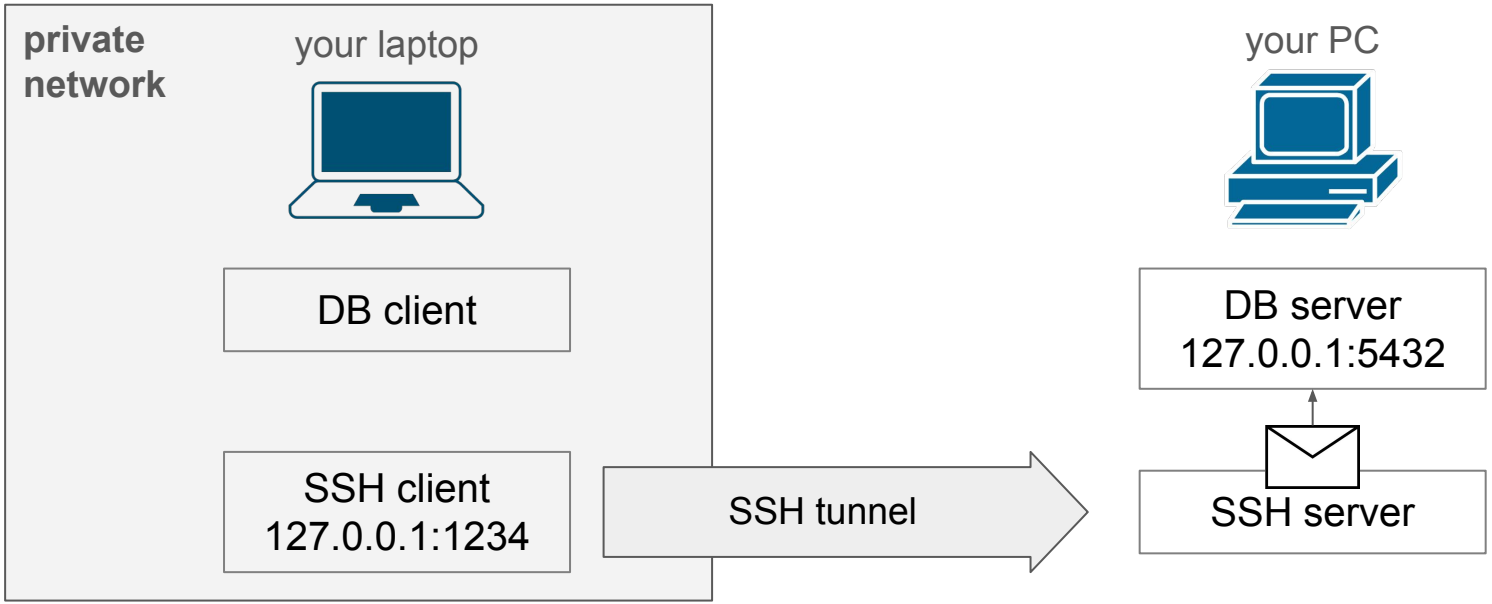
SSH Local Port Forwarding - Explanation

- SSH client forwards the packet to SSH server via SSH tunnel



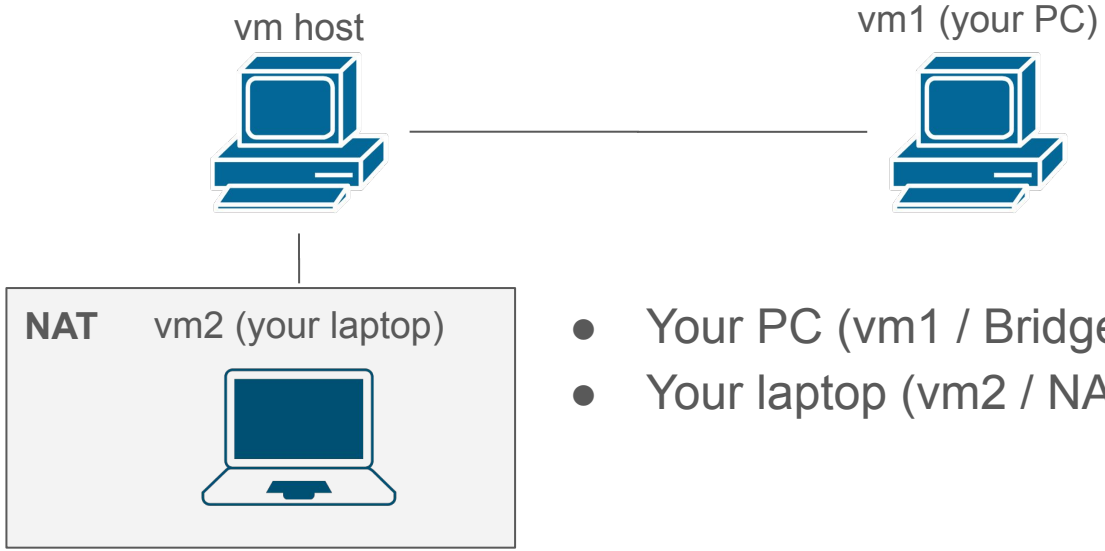
SSH Local Port Forwarding - Explanation

- SSH server then forward the packet to 127.0.0.1:5432



SSH Local Port Forwarding - Experiment Spec & Steps

1. Try to connect the database on vm1(your PC) from vm2(your laptop).
2. Use SSH local port forwarding to create tunnel.
3. Try to connect the database on vm1(your PC) again.



- Your PC (vm1 / Bridge *140.113.10.10*)
- Your laptop (vm2 / NAT *10.0.2.15*)

SSH Local Port Forwarding - Experiment Step 1

- Try to connect the database on vm1(your PC) from vm2(your laptop).
  - Use the ip addresses of your own machines.
  - Port 5432 is the default port for Postgresql database server.

```
[ccna@vm2 ~]$ psql --host 140.113.10.10 --port 5432 \
                   --username ccna --dbname ccna
psql: error: connection to server at “140.113.10.10”, port 5432 failed: No
route to host
    Is the server running on that host and accepting TCP/IP connections?
```

SSH Local Port Forwarding - Experiment Step 2

- Use SSH local port forwarding to create tunnel.
  - **`-N`** Do not execute a remote command.
  - **`-f`** Requests ssh to **go to background** just before command execution.

```
[ccna@vm2 ~]$ ssh 140.113.10.10 -L 1234:localhost:5432 -Nf
ccna@140.113.10.10's password:
[ccna@vm2 ~]$
```

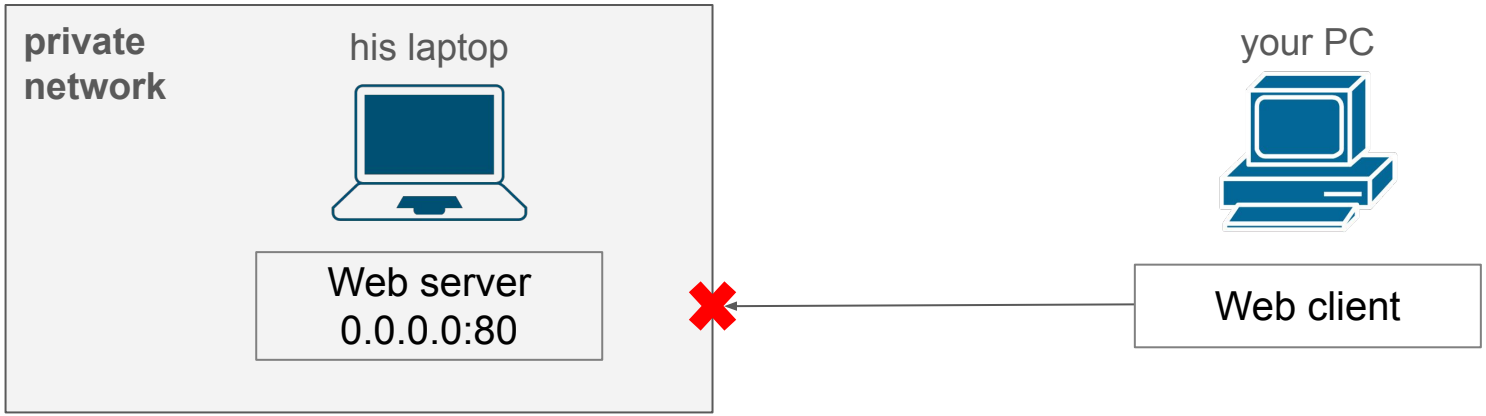
SSH Local Port Forwarding - Experiment Step 3

- Try to connect the database on vm1(your PC) again.
  - Use **127.0.0.1 (localhost) and 1234 port** this time.
  - Type **exit** to quit.

```
[ccna@vm2 ~]$ psql --host localhost --port 1234 \
                   --username ccna --dbname ccna
psql (16.1)
Type “help” for help.

ccna=#
```

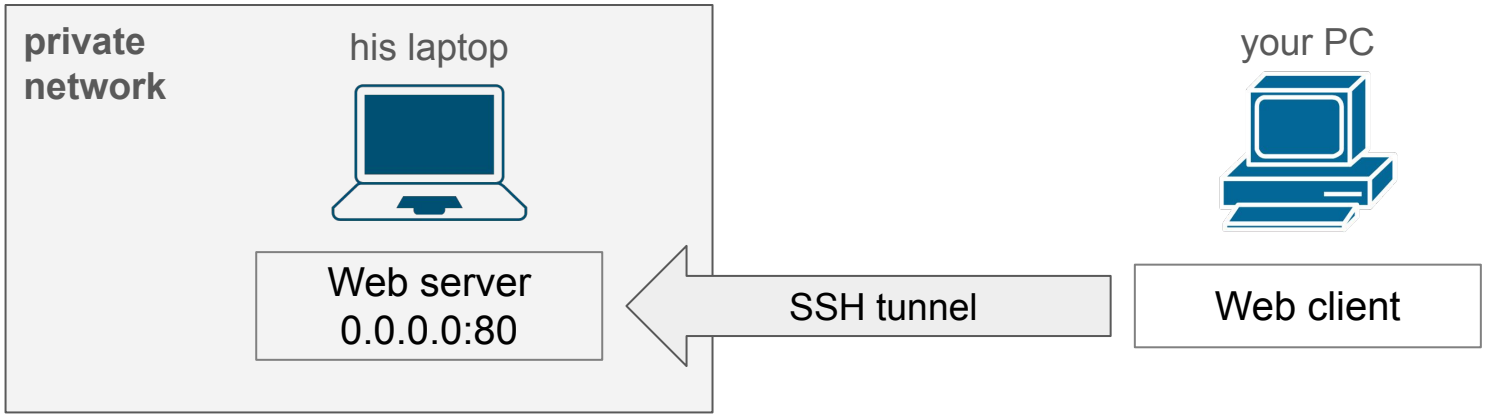
SSH Remote Port Forwarding - Scenario

- Your webapp classmate want to show you his webapp on his laptop behind a private network, which is not publicliy accessible.
- On the other hand, your PC in your dorm is publicliy accessible.
- How can you access his laptop from your PC?



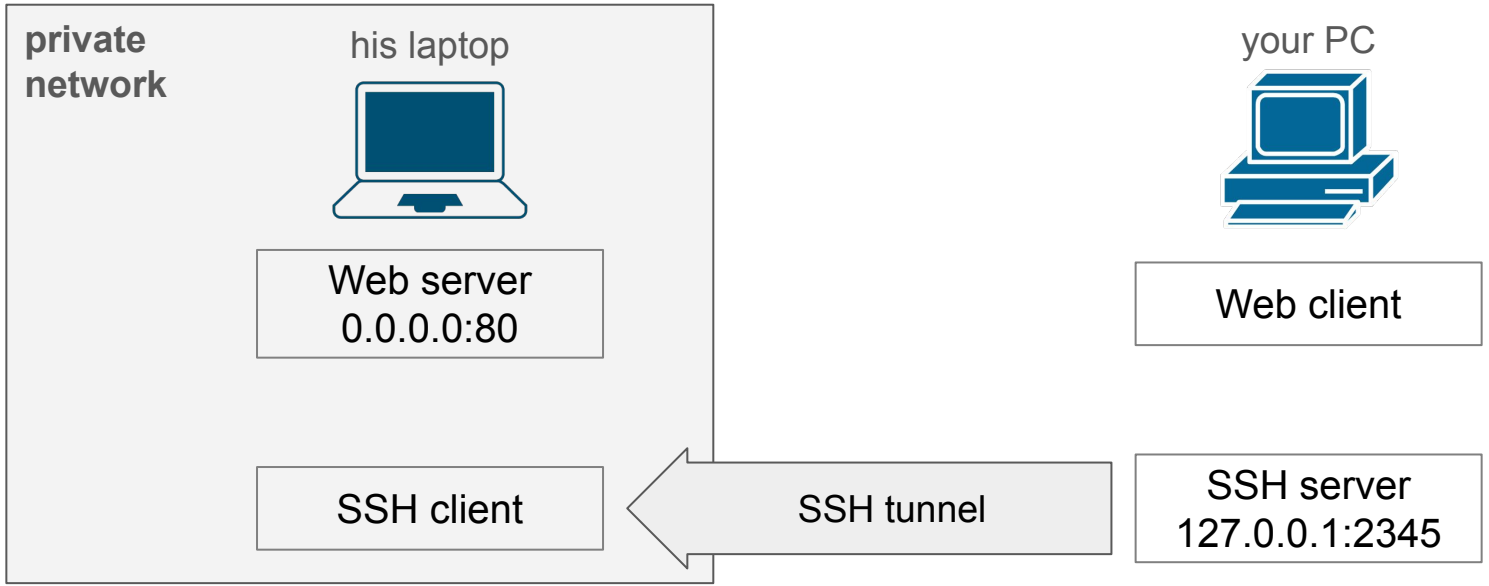
SSH Remote Port Forwarding - Solution

- **`-R`** [*`bind_address`*`:`]*`port`*`:`*`host`*`:`*`hostport`*
- Specifies that the given port on the remote (server) host is to be forwarded to the given host and port on the local side.



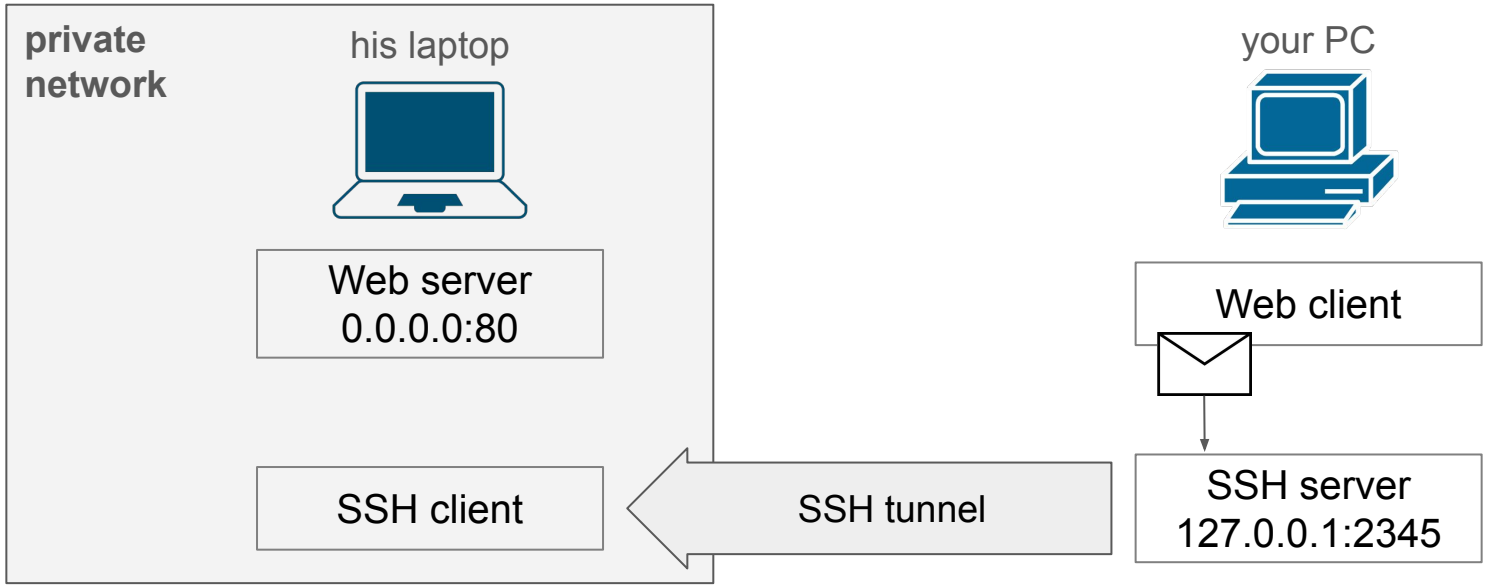
SSH Remote Port Forwarding - Explanation

- Initialize an SSH connection from his laptop to your PC
  - Create an SSH tunnel



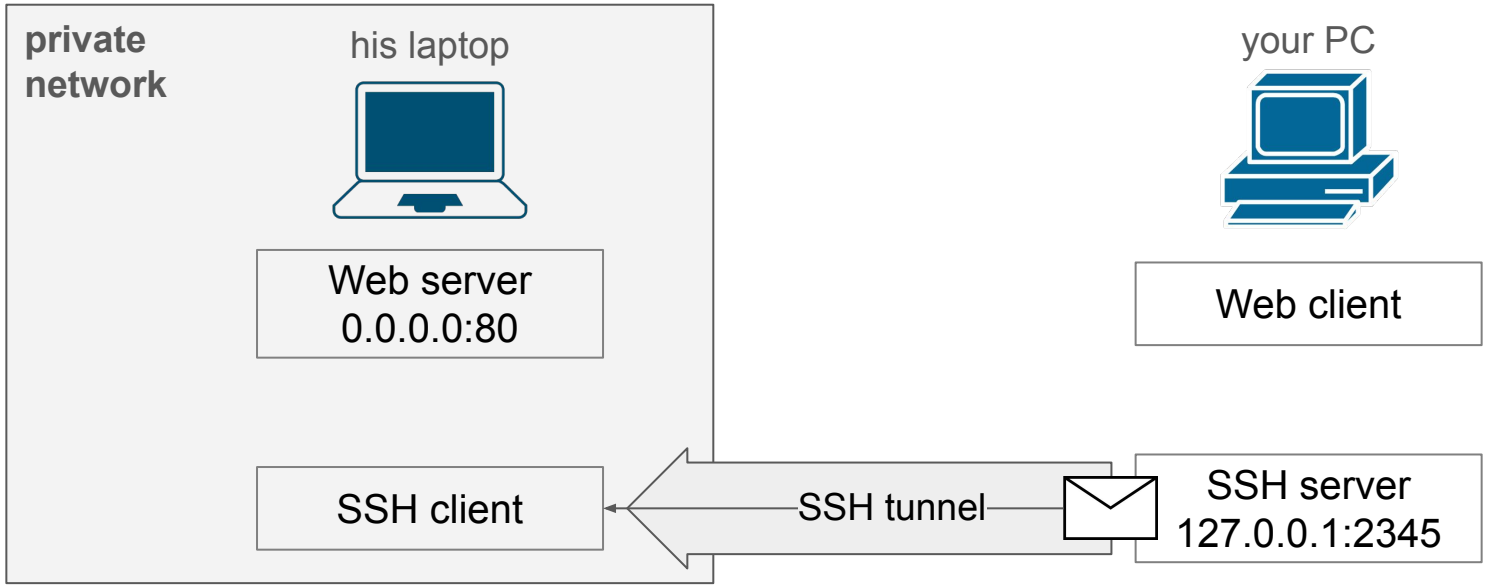
SSH Remote Port Forwarding - Explanation

- Web client sends a packet to 127.0.0.1:2345



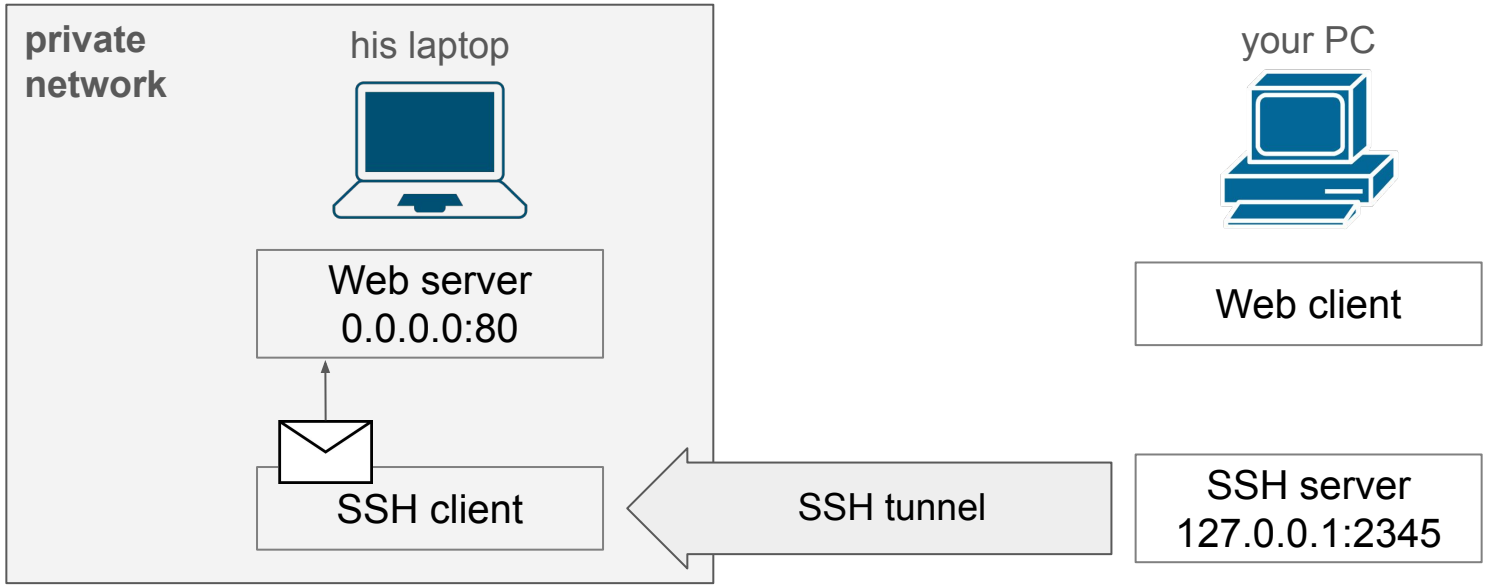
SSH Remote Port Forwarding - Explanation

- SSH server forwards the packet to SSH client via the SSH tunnel



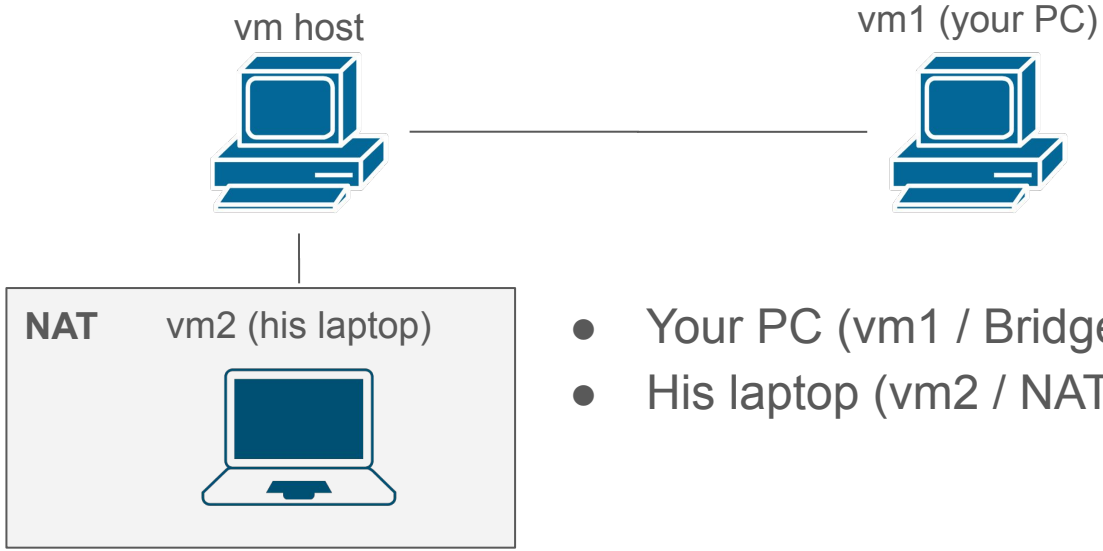
SSH Remote Port Forwarding - Explanation

- SSH client forwards the packet to the web server on his laptop.



SSH Remote Port Forwarding - Experiment Spec & Steps

1. Try to access the web server on vm2(his laptop) from vm1(your PC).
2. Use SSH remote port forwarding to create tunnel.
3. Try to access the web server again.



- Your PC (vm1 / Bridge *140.113.10.10*)
- His laptop (vm2 / NAT *10.0.2.15*)

SSH Remote Port Forwarding - Experiment Step 1

- Try to access the web server on vm2(his laptop) from vm1(your PC).
  - Use the ip addresses of your own machines.
- You can stop the previous ssh connections by running `killall ssh`.

```
[ccna@vm1 ~]$ curl 10.0.2.15
curl: (28) Connection timed out after 10002 milliseconds
[ccna@vm1 ~]$ curl localhost:2345
curl: (7) Failed to connect to localhost port 2345 after 0 ms: Couldn’t
connect to server
```

SSH Remote Port Forwarding - Experiment Step 2

- Use SSH remote port forwarding to create tunnel.
  - **-R** [*bind_address*:]*port*:*host*:*hostport*

```
[ccna@vm2 ~]$ ssh 140.113.10.10 -R 2345:localhost:80 -Nf
ccna@140.113.10.10's password:
[ccna@vm2 ~]$
```

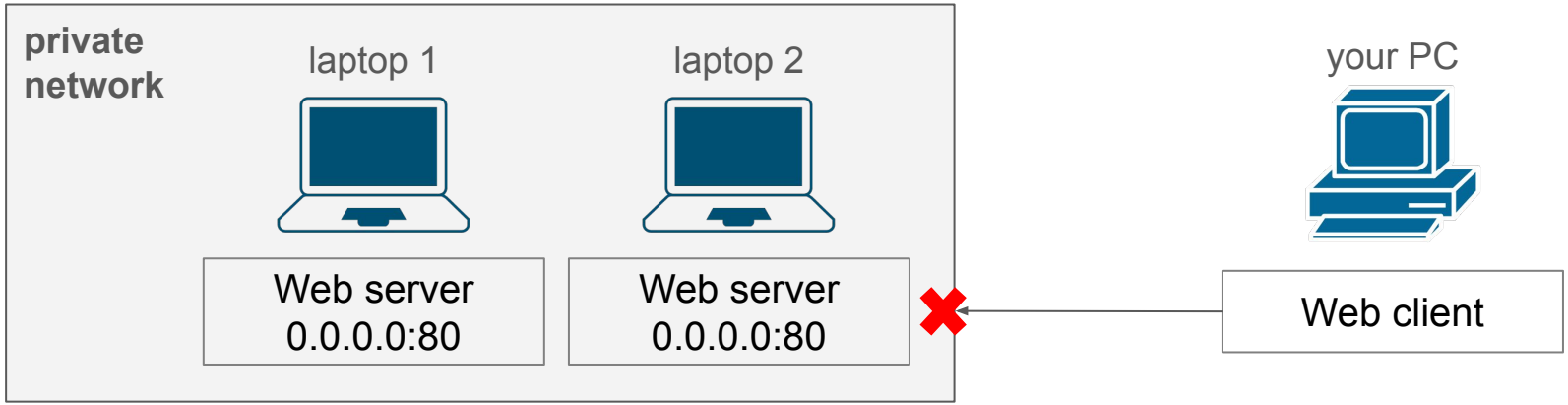
SSH Remote Port Forwarding - Experiment Step 3

- Try to access the web server again.

```
[ccna@vm2 ~]$ curl localhost:2345
hello, world from vm2
[ccna@vm2 ~]$
```

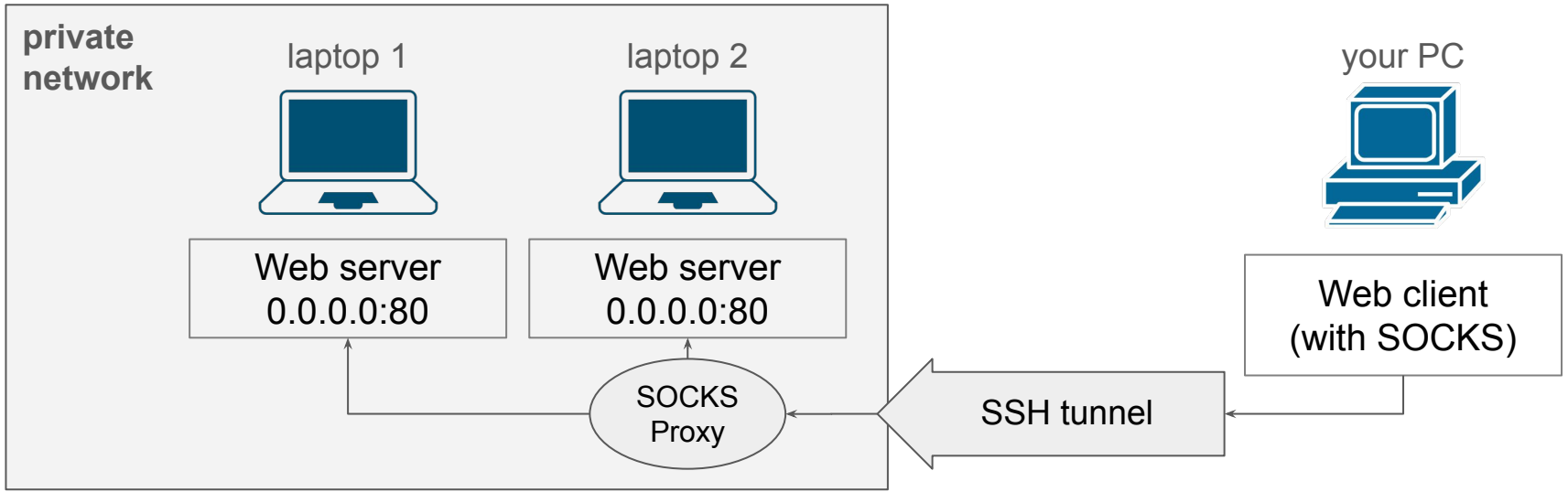
SSH Dynamic Port Forwarding - Scenario

- Another classmate who's also your classmate's roommate has joined.
- They're using their own laptops to host the webapp.
- Their laptops are behind the same NAT network.
- However, you're too lazy to do two remote port forwarding.
  - How can you meet the requirements with using two port forwarding?



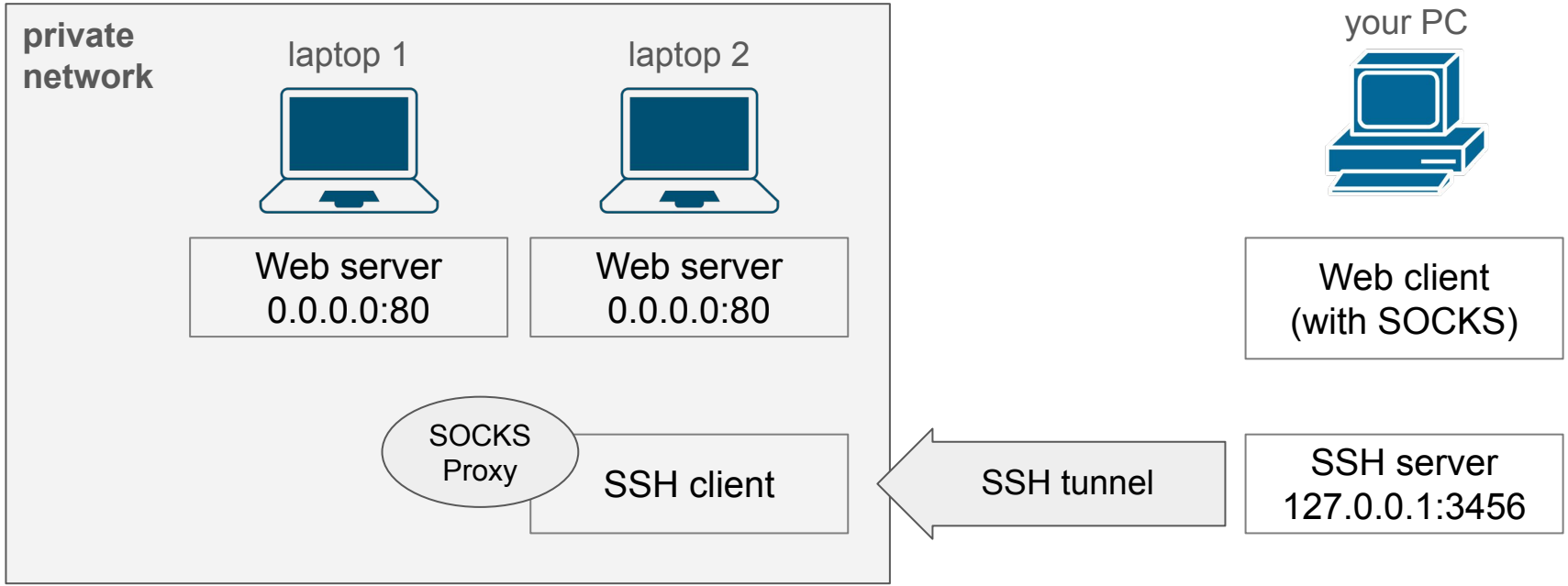
SSH Dynamic Port Forwarding - Solution

- **`-R`** *`[bind_address:]port`*
- ssh will act as a SOCKS 4/5 proxy and forward connections to the destinations requested by the remote SOCKS client.
  - There is also **`-D`** which listen on local port. See **man ssh** for more details.



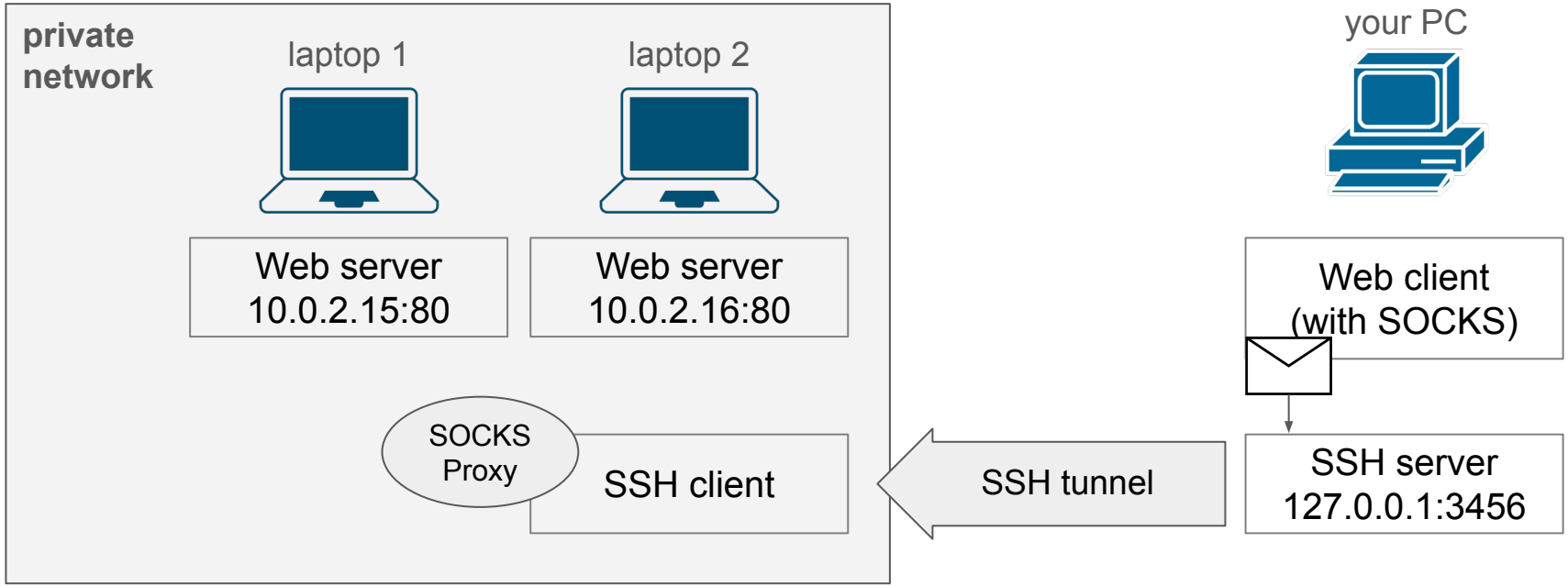
SSH Dynamic Port Forwarding - Explanation

- Initialize an SSH connection from your laptop to your PC
  - Create an SSH tunnel and SOCKS Proxy server



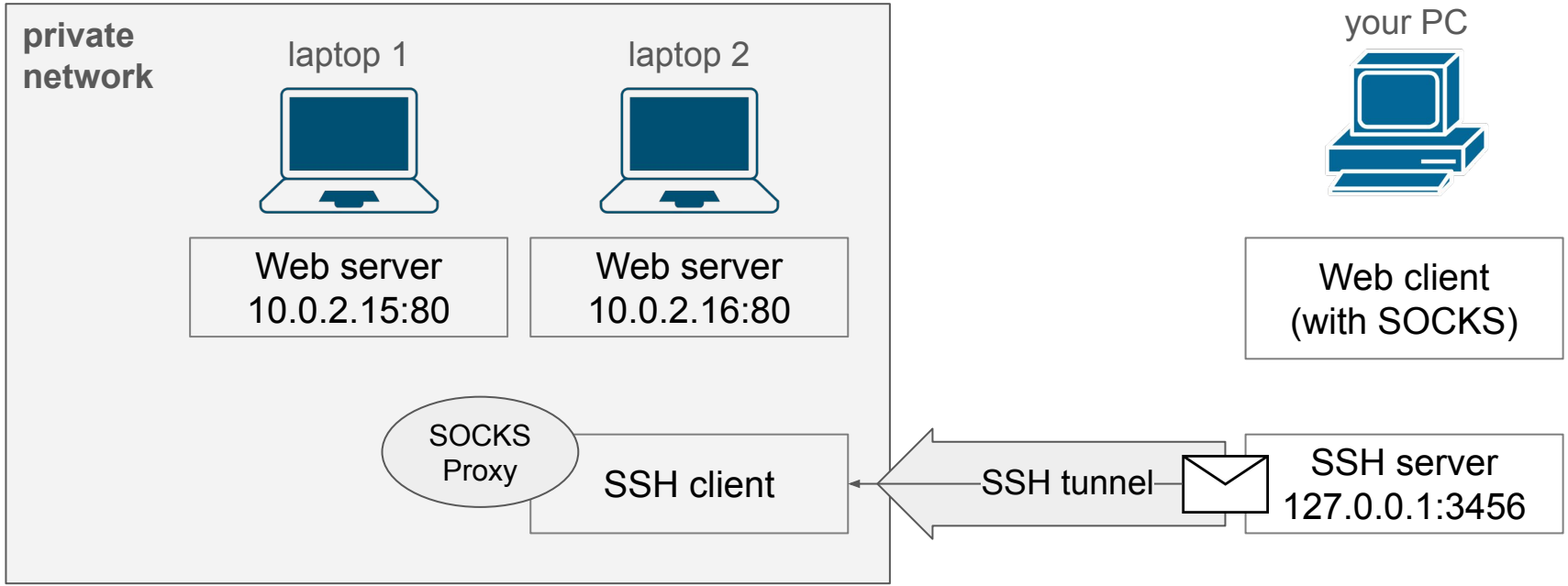
SSH Dynamic Port Forwarding - Explanation

- Web client sends a packet to 10.0.2.15:80 via SOCKS proxy on 127.0.0.1:3456



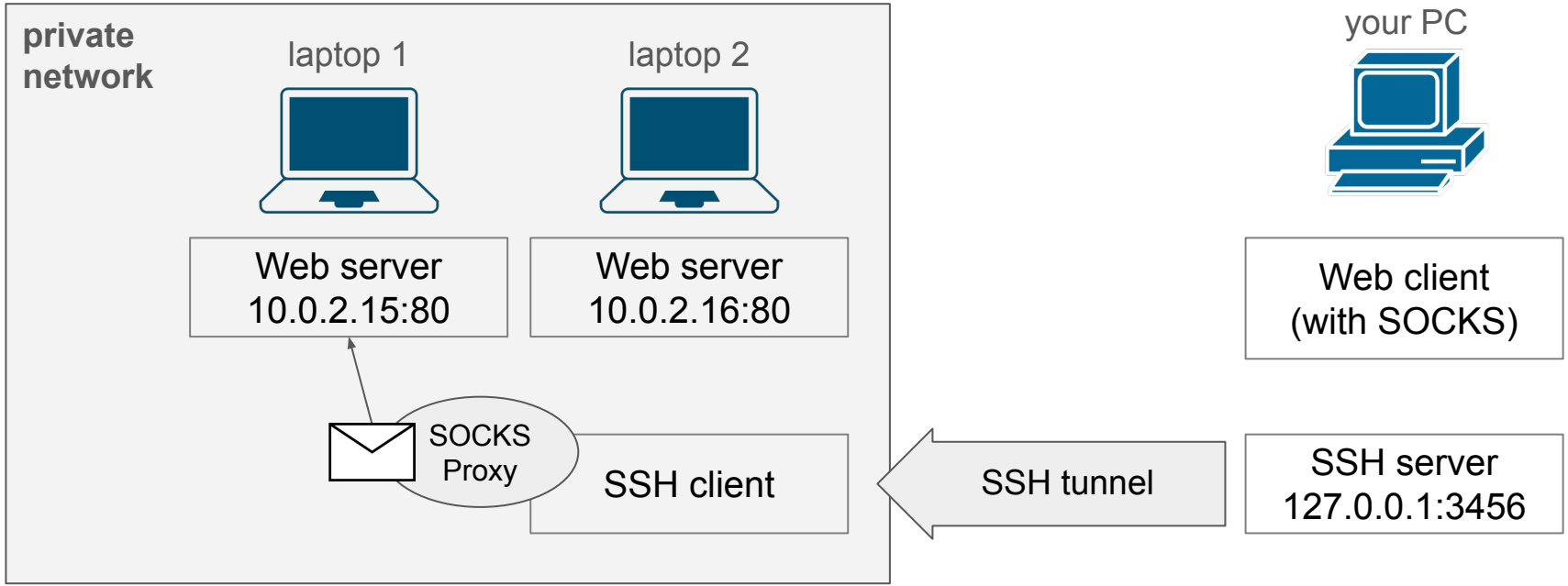
SSH Dynamic Port Forwarding - Explanation

- SSH server forward the packet to SSH client via SSH tunnel.



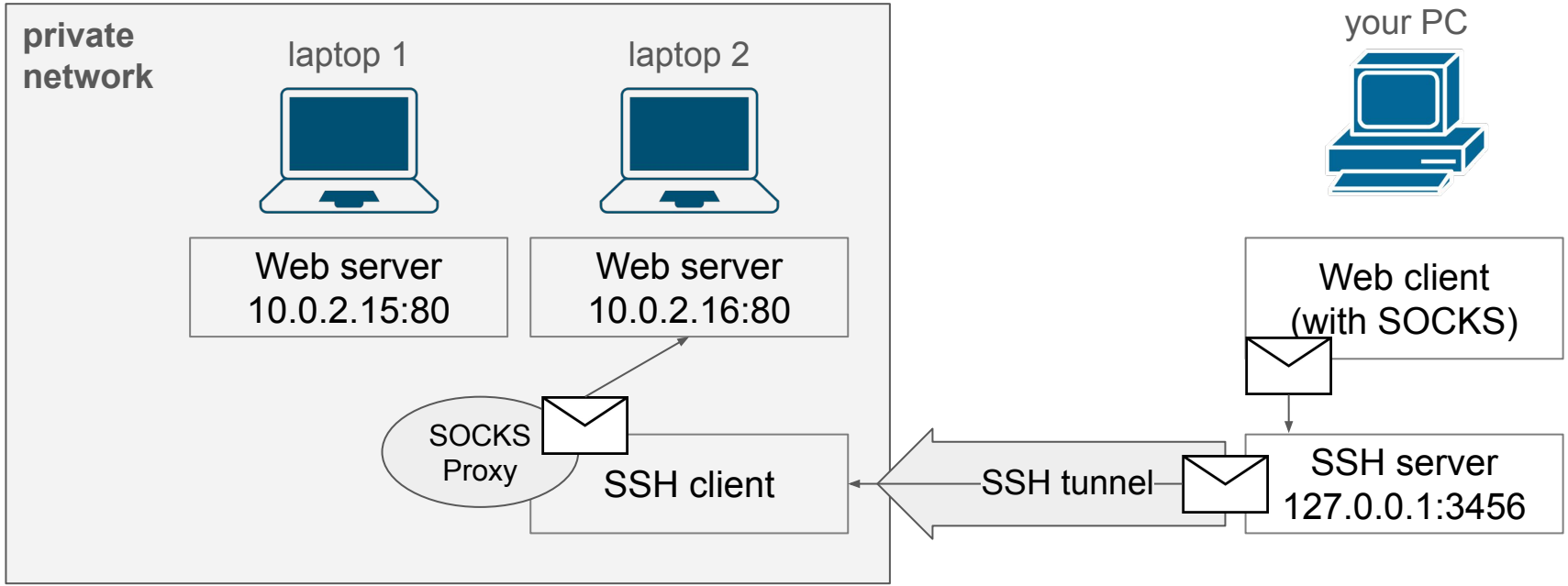
SSH Dynamic Port Forwarding - Explanation

- SSH client (which act like a SOCKS Proxy) then forwards the packet to its original destination 10.0.2.15:80.



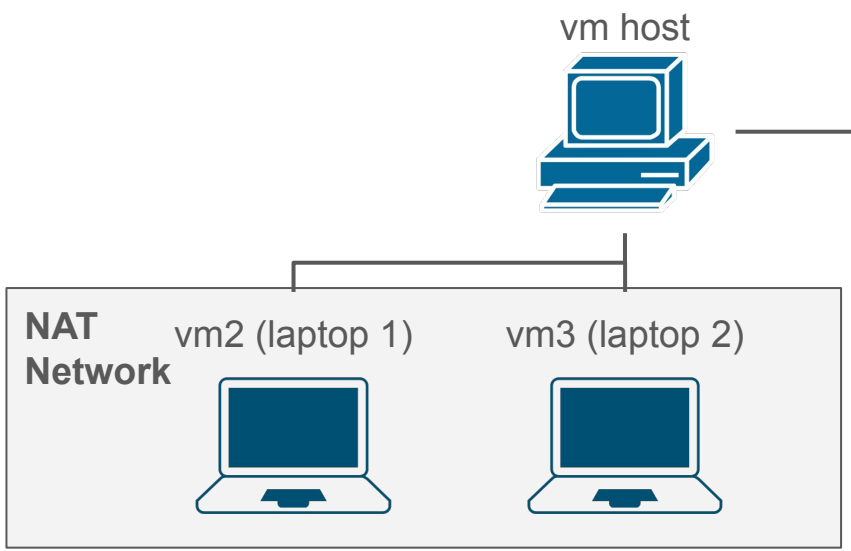
SSH Dynamic Port Forwarding - Explanation

- Web client can also send packets to 10.0.2.16:80 using the same procedure.



SSH Dynamic Port Forwarding - Experiment Spec & Steps

1. Try to access the web servers on vm2 & vm3 (laptop 1 & 2) from vm1(your PC).
2. Use SSH remote port forwarding to create tunnel.
3. Try to access the web servers again.



- Your PC (vm1 / Bridge *140.113.10.10*)
- Laptop 1 (vm2 / NAT Network *10.0.2.15*)
- Laptop 2 (vm2 / NAT Network *10.0.2.16*)

SSH Dynamic Port Forwarding - Experiment Step 1

- Try to access the web servers on vm2 & vm3 (laptop 1 & 2) from vm1(your PC).
  - Use the ip addresses of your own machines.
- You can stop the previous ssh connections by running `killall ssh`.

```
[ccna@vm1 ~]$ curl 10.0.2.15 --max-time 10
curl: (28) Connection timed out after 10002 milliseconds
[ccna@vm1 ~]$ curl 10.0.2.16 --max-time 10
curl: (28) Connection timed out after 10003 milliseconds
[ccna@vm1 ~]$
```

SSH Dynamic Port Forwarding - Experiment Step 2

- Use SSH remote port forwarding to create tunnel.
  - **-R** [*bind_address*:]*port* to initialize dynamic remote port forwarding

```
[ccna@vm2 ~]$ ssh 140.113.10.10 -R 3456 -Nf
ccna@140.113.10.10's password:
[ccna@vm2 ~]$
```

SSH Dynamic Port Forwarding - Experiment Step 3

- Try to access the web servers again.
  - **`--socks5`** *`host[:port]`* to use socks5 proxy while sending requests

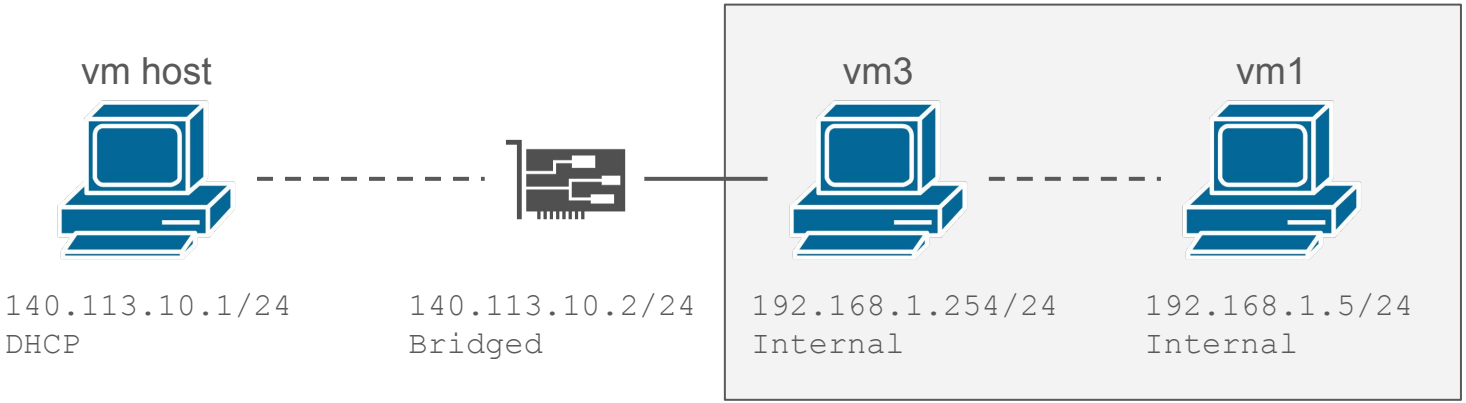
```
[ccna@vm1 ~]$ curl localhost
hello, world from vm1
[ccna@vm1 ~]$ curl 10.0.2.15 --socks5 localhost:3456
hello, world from vm2
[ccna@vm1 ~]$ curl 10.0.2.16 --socks5 localhost:3456
hello, world from vm3
```

SSH Port Forwarding - Comparison

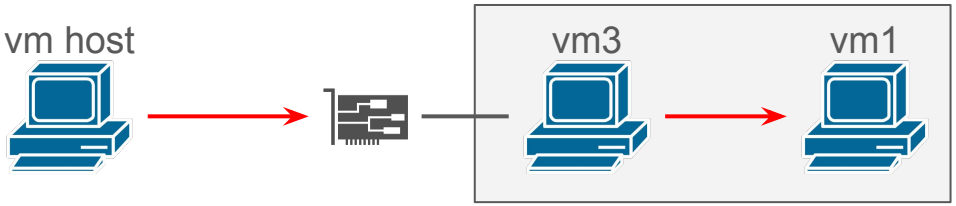
Type	Command	Listen port	Number of ports being forwarded
Local	`-L`	On local side	1 to 1
Remote	`-R`	On remote side	1 to 1
Dynamic	`-D / -R`	Either on local or remote side	Many to many

SSH Proxy Jump - Scenario & Spec

- You have to access *vm1*, but *vm1* is in a private network.
- However, there's another machine *vm3*, which have two NIC.
  - One is in the private network.
  - The other is in a network that *host* can access.



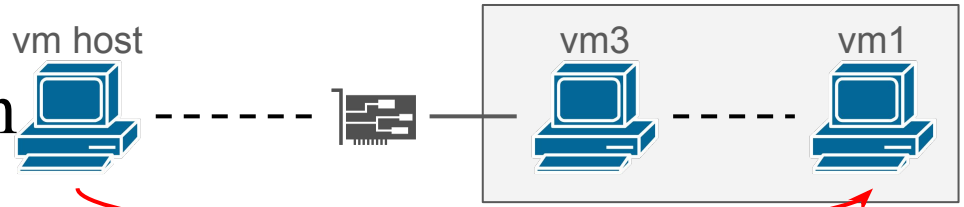
SSH Proxy Jump - Issue



- ssh to *vm3* and then ssh to *vm1* might be a good approach.
- This can be annoying if you have to constantly access *vm1*.

```
[ytshih@desktop ~]$ ssh 140.113.10.2 -l ccna
ccna@192.168.1.2's password:
Last login: Sat Feb 17 14:05:00 2024 from 140.113.10.1
[ccna@vm3 ~]$ ssh 192.168.1.5
ccna@10.0.2.16's password:
Last login: Sat Feb 17 14:05:23 2024 from 192.168.1.254
[ccna@vm1 ~]$
```

SSH Proxy Jump - Solution



- **`-J`** *`[user@]destination[:port]`*
- Setting this option will cause ssh to connect to the target host by first making a ssh connection to the specified ProxyJump host and then establishing a TCP forwarding to the ultimate target from there.

```
[ytshih@desktop ~]$ ssh 192.168.1.5 -l ccna -J ccna@140.113.10.2
ccna@192.168.1.2's password:
ccna@10.0.2.16's password:
Last login: Sat Feb 17 14:12:05 2024 from 192.168.1.254
[ccna@vm1 ~]$
```

That’s all

- Practice the experiments in the slides.
- Project 1 will release on E3 soon.
- Feel free to ask any question.