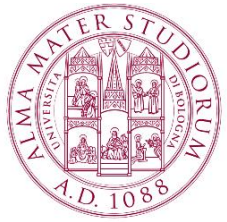


ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Practical experiments about IP networking and routing

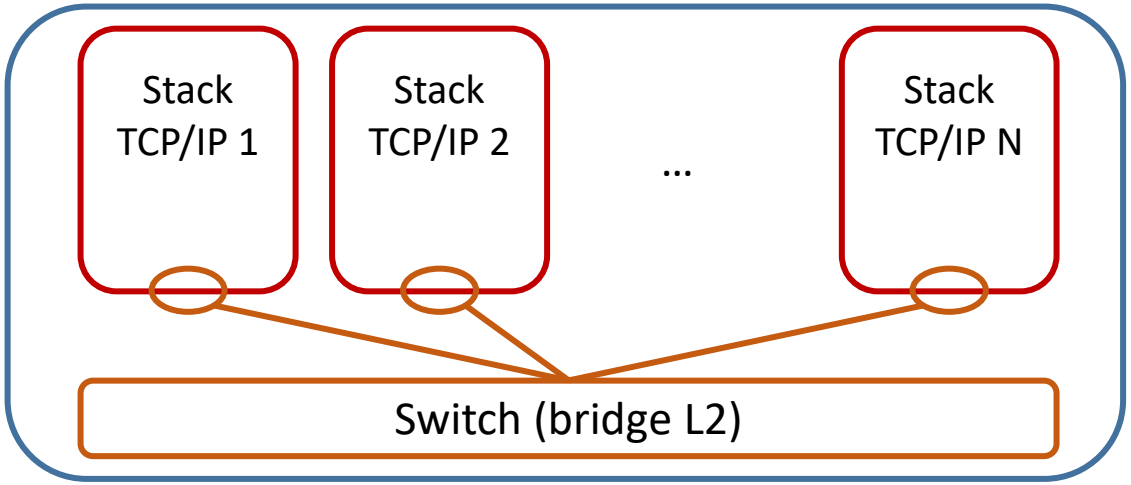
Franco CALLEGATI – Walter CERRONI

Chiara CONTOLI – Chiara GRASSELLI

AA 2021-2022

Experiments with networking

- Practical experiments in networking with physical infrastructure are complex to organize and expensive
- In the last decade:
  - Virtualization technologies in ICT gives new opportunities
- Question?
  - Can we run a whole network «inside» a single physical machine, experimenting all protocols and features «as if» in a real environment

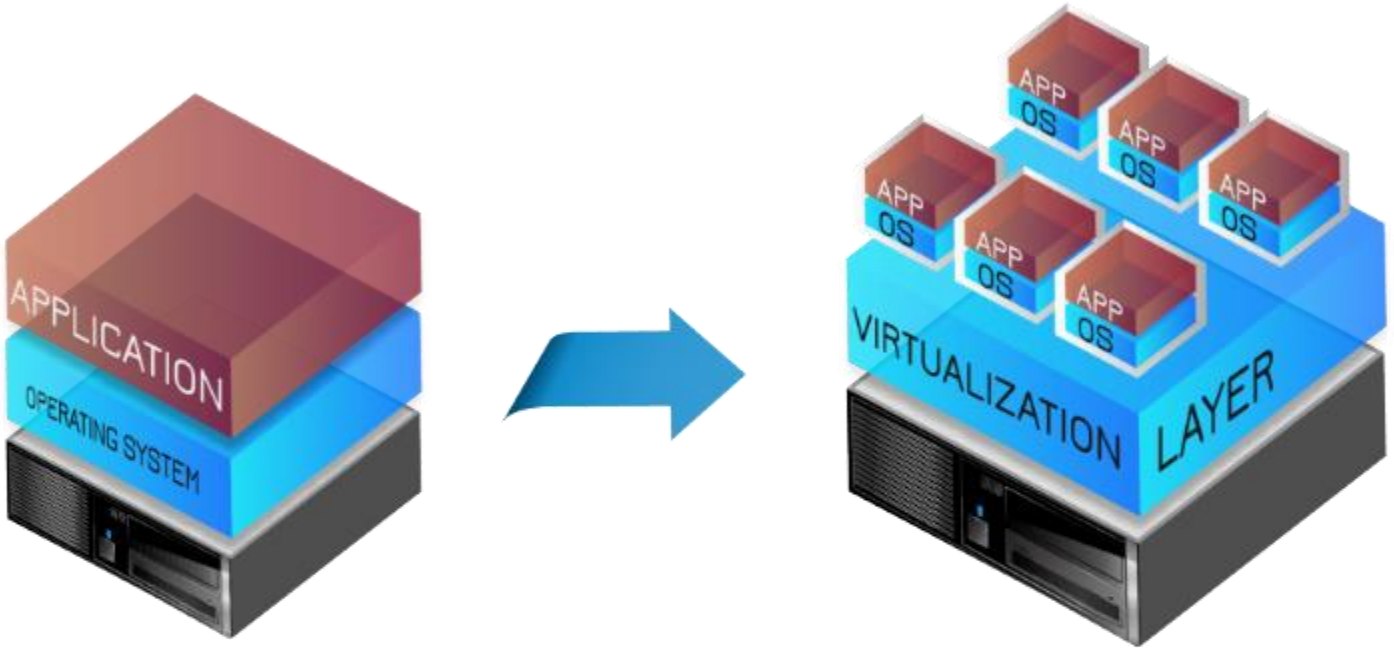


Single physical machine

Virtualization

- **Virtualization** in ICT
  - creation and use of *virtual* (i.e., abstract, software-based) versions of computing, storage and network elements
  - evolution of a concept invented in the ’60s to logically separate physical resources available on a mainframe for different applications
- Advantages
  - physical resource sharing → optimal utilization
  - decoupling software from hardware
  - scalability, flexibility, and mobility
- Issues
  - virtual resource isolation
  - security and privacy between different users

Computing element virtualization



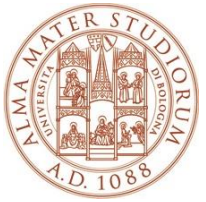
Traditional Architecture

Virtual Architecture

Source: http://exelos.com/solutions/virtualization/

Example of a virtualized system

Some dates about virtualization



- Late ’60s
  - First ideas from IBM
- 1987
  - Insignia Solutions demonstrated SoftPC, running DOS systems under Unix workstations
- 1998
  - VMWare was founded and soon started to commercialize VirtualPC
- 2007
  - Virtualbox is released

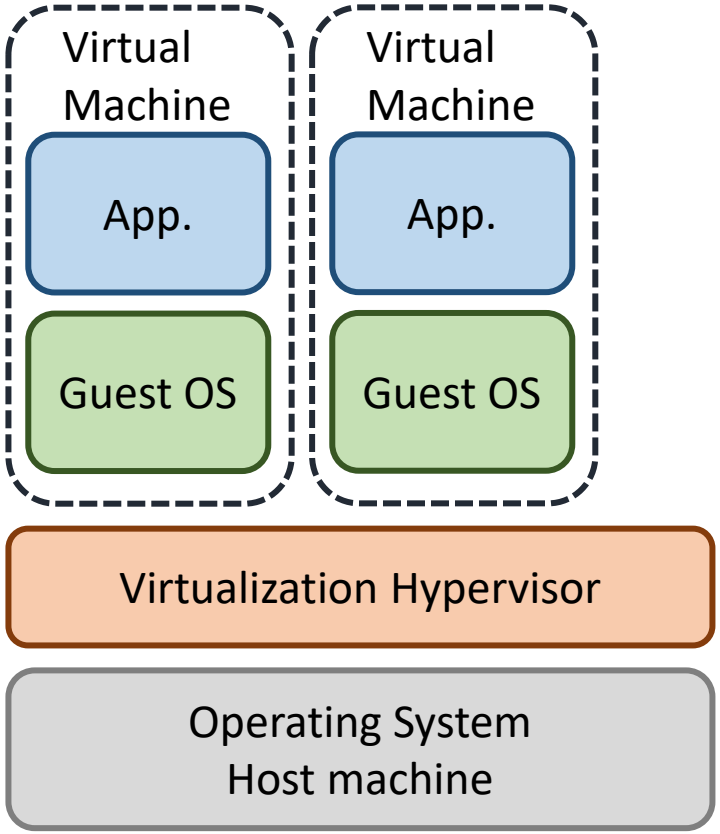
Virtualization categories

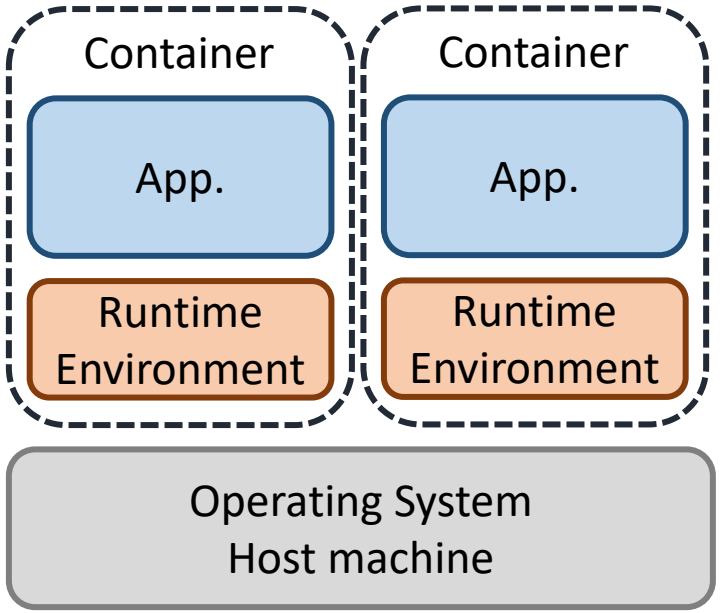
- Full virtualization
  - emulation of each HW device by the *hypervisor*
  - transparent installation of the virtual machine (VM)
  - e.g., Virtualbox, VMware, QEMU
- Paravirtualization
  - some HW functions are not emulated in SW, but shared among VMs through the hypervisor
  - VMs must be “aware” of being virtualized (driver or modified kernel)
  - e.g., Xen, KVM+Virtio
- Operating system virtualization
  - isolated “slices” of the operating system seen by applications as independent computing/storage/networking environments
  - limited VM heterogeneity (e.g., Linux on Linux)
  - e.g., Linux Containers (LXC), Docker

Linux containers

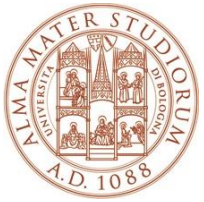
- Virtualization
  - Share the hardware
  - Execute different OSs on the same hardware
- Container
  - Share the hardware
  - Share the kernel of the OS
  - Execute application processes in isolation with the related environment (libraries, config files etc.)
- A VM may have a OS completely different from the host OS (linux on Windows, Windows on Linux etc.)
- A container must be compatible with the host OS

Virtualization VS Containers



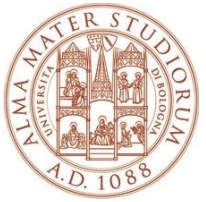


Some dates about containers



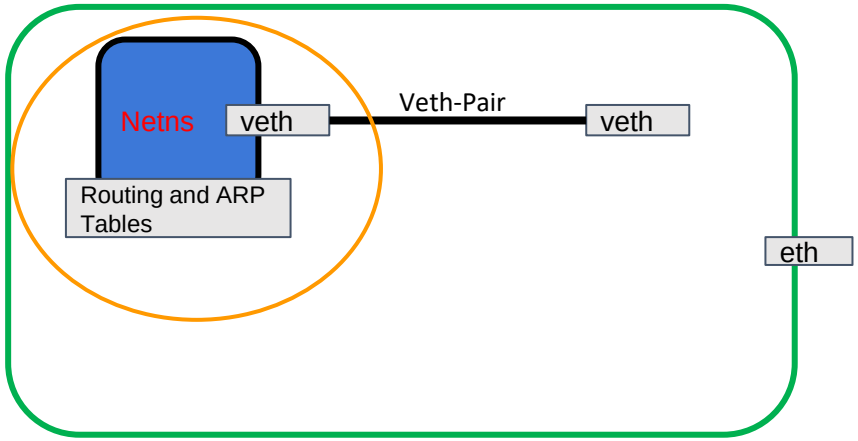
- 2000
  - First ideas in FreeBSD
- 2001
  - First project on process isolation in Linux (Linux Vserver)
- 2006-2008
  - cgroups and namespaces added to Linux
- 2014
  - Linux containers (LXC)
- 2014
  - Docker

How to build LANs in the virtual space



- Networking with Linux namespaces
  - Run multiple and separate network protocol stacks on the same machine
    - Different IP addresses
    - Different routing tables
    - Different forwarding strategies
  - The IP network is built with a switch interconnecting protocol stacks by means of their interfaces
  - Need for a software switch or bridge

A Linux namespace looks like…



Physical Host (global namespace)

Isolated Network namespace (e.g.: Netns is the name of the namespace)

The Veth Pair (Virtual Ethernet Pair) is the network cable abstraction (Ethernet tunnel). This element is *always* built as a couple (two endpoints).

Allow the namespace to communicate with (physical) host namespace or with another name space.

Communications between different namespaces can be implemented also via virtual switches, e.g.:

- Linux bridge
- OpenVirtualSwitch (OVS)

Virtual switches belongs to the global namespace

The key commands (1)

- `ip netns`
  - Network namespace management
    - `ip netns add NAMESPACE`
      - Create a namespace
    - `ip netns del NAMESPACE`
      - Delete a namespace
    - `ip netns list`
      - List the existing namespaces
    - `ip netns exec NAME COMMAND`
      - Execute a specific COMMAND in namespace NAME
    - `ip netns id`
      - Gives the ID of the current namespace

The key commands (2)

- `ip link`
  - Manage and configure network devices
    - `ip link add NAME type TYPE`
      - NAME is the symbolinc name of the device
      - TYPE is the type of network device
        - Veth = virtual ethernet interface
        - Bridge = Ethernet bridge (switch)
    - `ip link add NAME1 type veth peer name NAME2`
      - Create a ethernet link between two logical interfaces named NAME1 and NAME2s
    - `ip link set NAME`
      - Configure properties of NAME

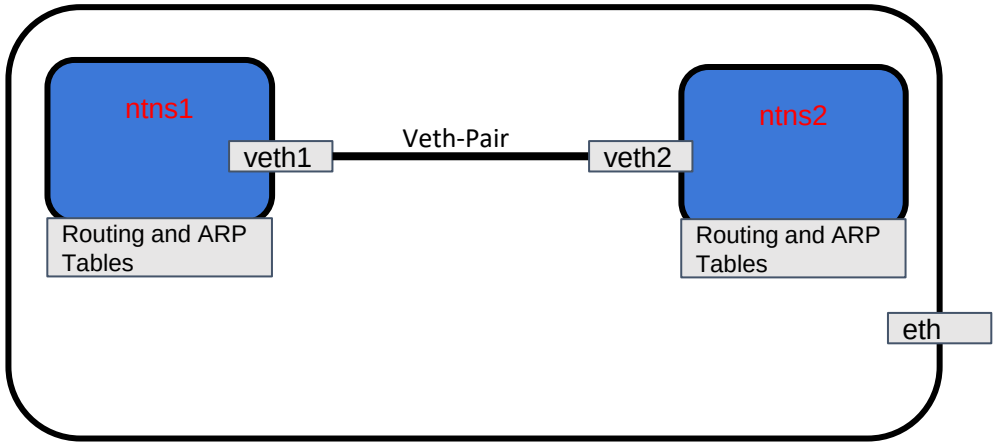
The key commands (3)

- `bridge`
  - Manage bridge addressing
    - `bridge link`
      - Set properties of interfaces to links
    - `bridge fdb`
      - Manage bridge forwarding database

Example 1

Create two namespaces and interconnect them through a Veth-Pair.
Consider the following:

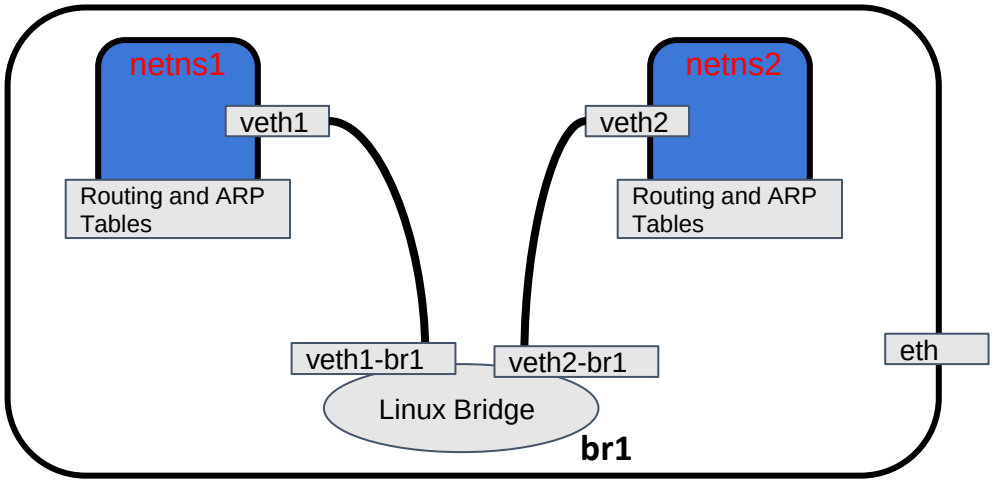
- names on topology: must be used to create components
  - **Note**: names must be unique inside each namespace!
- it is a point-to-point connection: 172.0.0.0/30



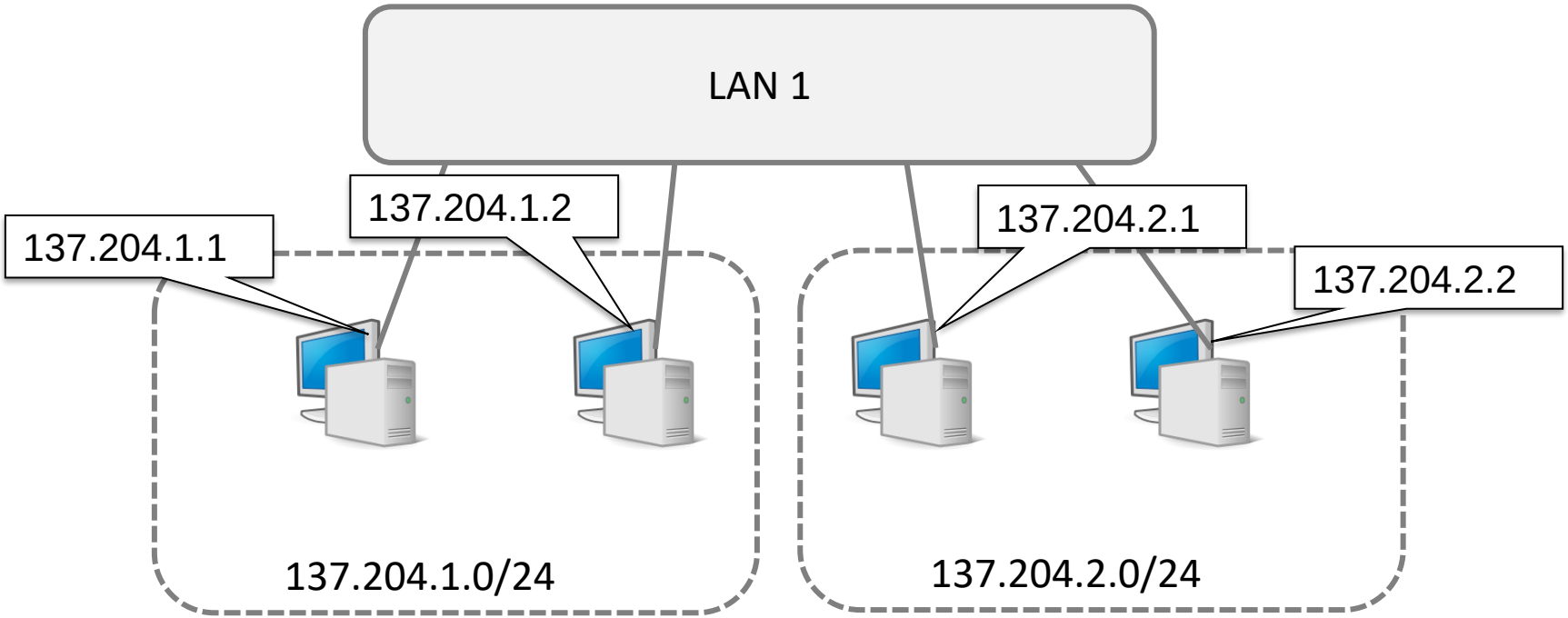
Which elements are part of the network? Which steps it is necessary to carry out to to build such scenario?
It is always important to check current configuration and those performed later.

Example 2

- Two namespaces interconnected through a Linux bridge:



Topology 1


LAN 1
137.204.1.1
137.204.1.2
137.204.2.1
137.204.2.2
137.204.1.0/24
137.204.2.0/24

Check connectivity

- Host 137.204.1.1/24 can reach host 137.204.1.2/24
  - Same LAN same IP network

```
H1_1> Ping –c 5 137.204.1.2
PING 137.204.1.2 (137.204.1.2) 56(84) bytes of data.
64 bytes from 137.204.1.2: icmp_seq=1 ttl=64 time=0.103 ms
64 bytes from 137.204.1.2: icmp_seq=2 ttl=64 time=0.054 ms
64 bytes from 137.204.1.2: icmp_seq=3 ttl=64 time=0.055 ms
64 bytes from 137.204.1.2: icmp_seq=4 ttl=64 time=0.056 ms
64 bytes from 137.204.1.2: icmp_seq=5 ttl=64 time=0.055 ms

--- 137.204.1.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 104ms
rtt min/avg/max/mdev = 0.054/0.064/0.103/0.021 ms
```

Check connectivity

- Host 137.204.1.1/24 can’t reach host 137.204.2.2/24
  - Same LAN but different IP networks

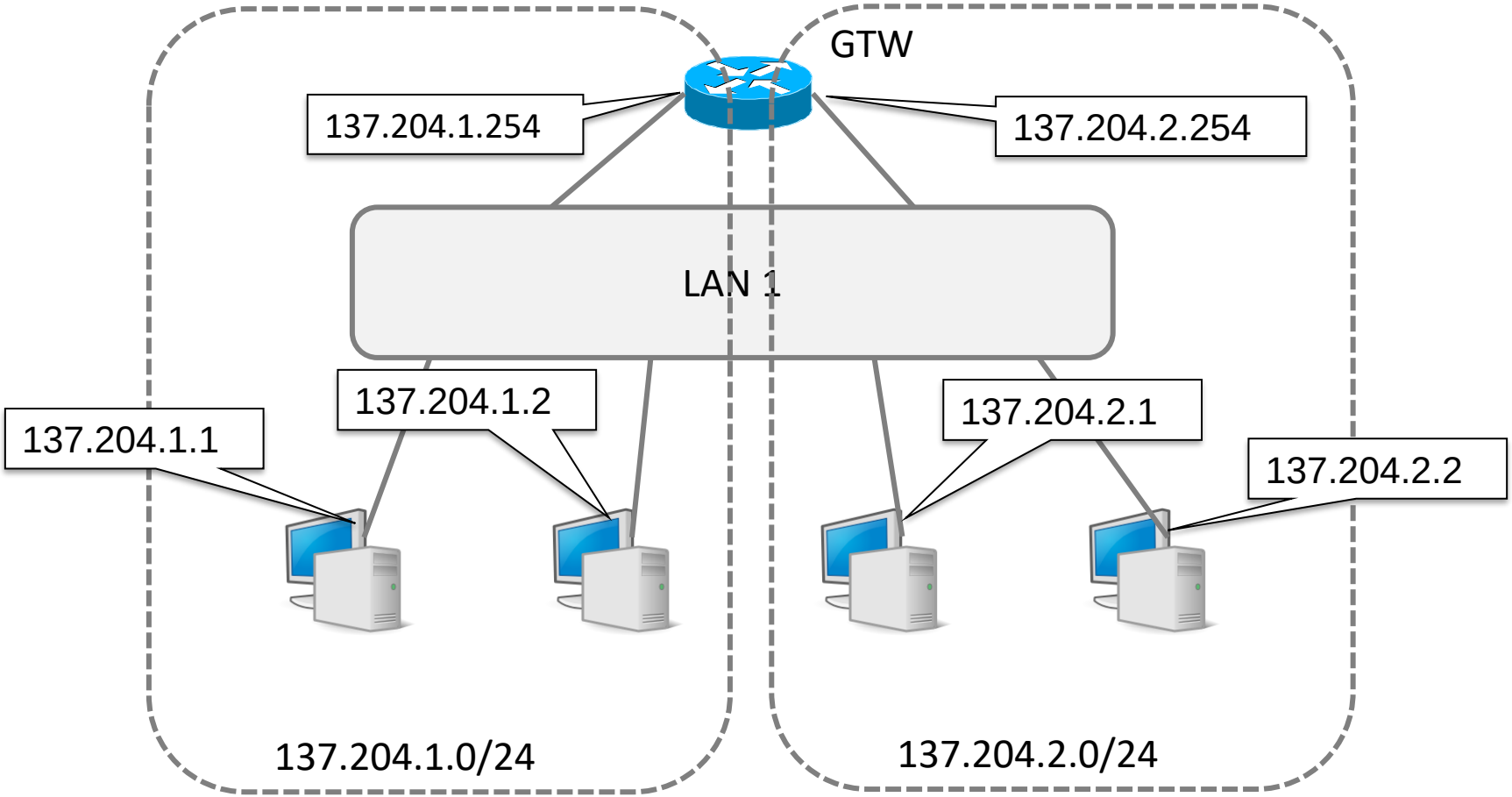
```
H1_1> Ping –c 5 137.204.2.2
connect: Network is unreachable

H1_1> route -n
Kernel IP routing table
Destination  Gateway   Genmask       Flags Metric Ref Use Iface
137.204.1.0  0.0.0.0   255.255.255.0 U     0      0   0   veth0
```

Key message!

- Connecting different hosts to the same LAN does not guarantee correct communications
- Hosts must belong to the same IP network to talk on the LAN

- The **LAN** is the physical infrastructure with physical addresses linked to hardware equipment
- The **IP network** is the logical entity that guarantees correct communications

Topology 2


GTW
137.204.1.254
137.204.2.254
LAN 1
137.204.1.2
137.204.1.1
137.204.2.1
137.204.2.2
137.204.1.0/24
137.204.2.0/24

Add Gateway

- Bridge IP networks
- Must be connected to both IP networks
- Hosts must know the gateway exists

```
H1_1> route -n
Kernel IP routing table
Destination  Gateway   Genmask     Flags Metric Ref Use Iface
0.0.0.0   137.204.1.254  0.0.0.0   UG   0      0   0   veth0
137.204.1.0  0.0.0.0   255.255.255.0 U  0      0   0   veth0
```

Check connectivity

- H1_1 can reach the gateway
- For some reasons the packet do not travel

```
H1_1> ping -c 5 137.204.2.2
PING 137.204.2.2 (137.204.2.2) 56(84) bytes of data.

--- 137.204.2.2 ping statistics ---
5 packets transmitted, 0 received, 100% packet loss, time 108ms
```

A gateway must be a gateway

- Usually hosts operating system do not forward packets from one interface to another
- Hosts recieve or send packets … that’s it but … we can force forwarding in most cases

```
H1_1> ip netns exec GTW sysctl -w net.ipv4.ip_forward=1
```

Finally

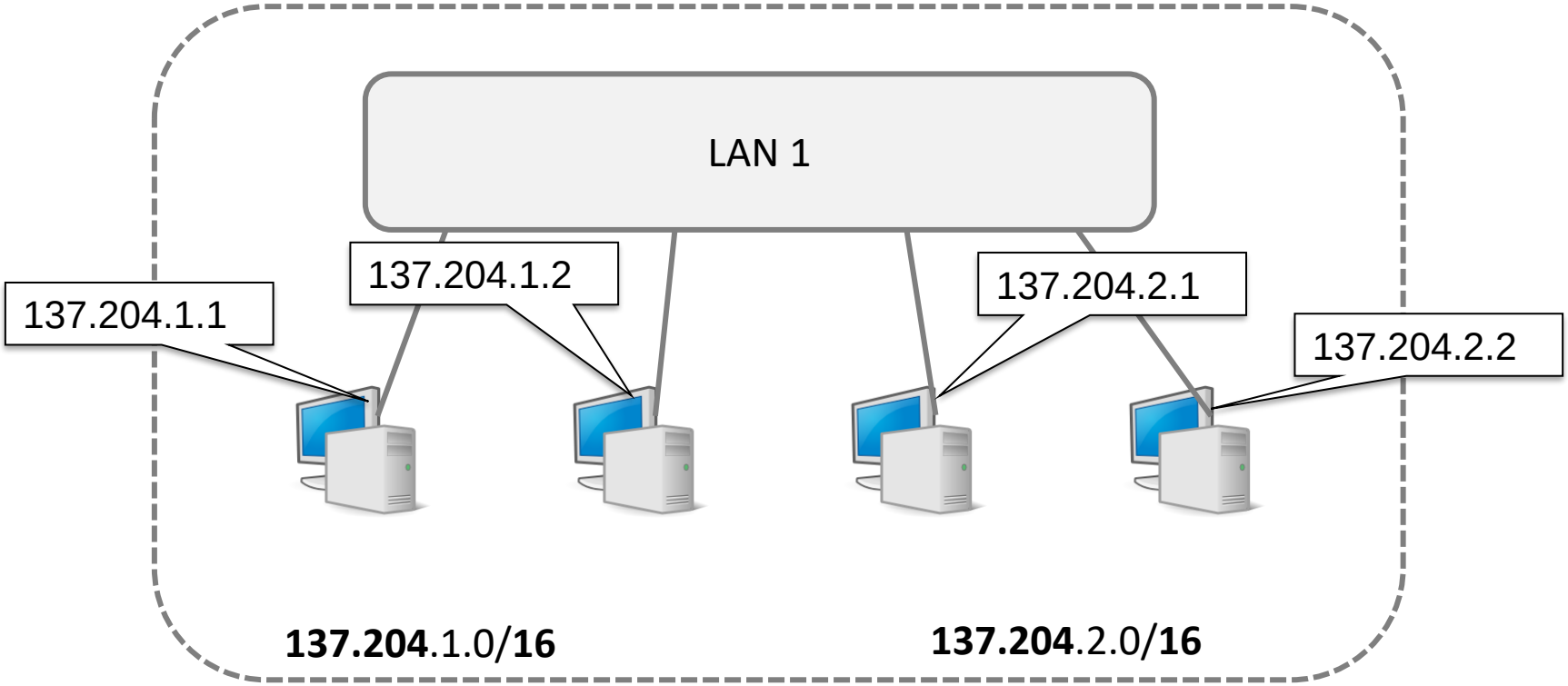
```
H1_1> ip netns exec h1_1 ping 137.204.2.1
PING 137.204.2.1 (137.204.2.1) 56(84) bytes of data.
64 bytes from 137.204.2.1: icmp_seq=1 ttl=63 time=0.114 ms
64 bytes from 137.204.2.1: icmp_seq=2 ttl=63 time=0.050 ms
64 bytes from 137.204.2.1: icmp_seq=3 ttl=63 time=0.073 ms
64 bytes from 137.204.2.1: icmp_seq=4 ttl=63 time=0.074 ms
64 bytes from 137.204.2.1: icmp_seq=5 ttl=63 time=0.034 ms
64 bytes from 137.204.2.1: icmp_seq=6 ttl=63 time=0.032 ms

--- 137.204.2.1 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 125ms
rtt min/avg/max/mdev = 0.032/0.062/0.114/0.030 ms
```

Avoid tricks

- In Topology 1 there is physical connectivity between hosts of different IP networks
- This can be exploited to cross IP network boundaries

```
H1_1> ip netns exec h1_1 ifconfig
veth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
inet 137.204.1.1  netmask 255.255.255.0  broadcast 0.0.0.0
ether 6a:63:16:e7:49:ce  txqueuelen 1000  (Ethernet)
```

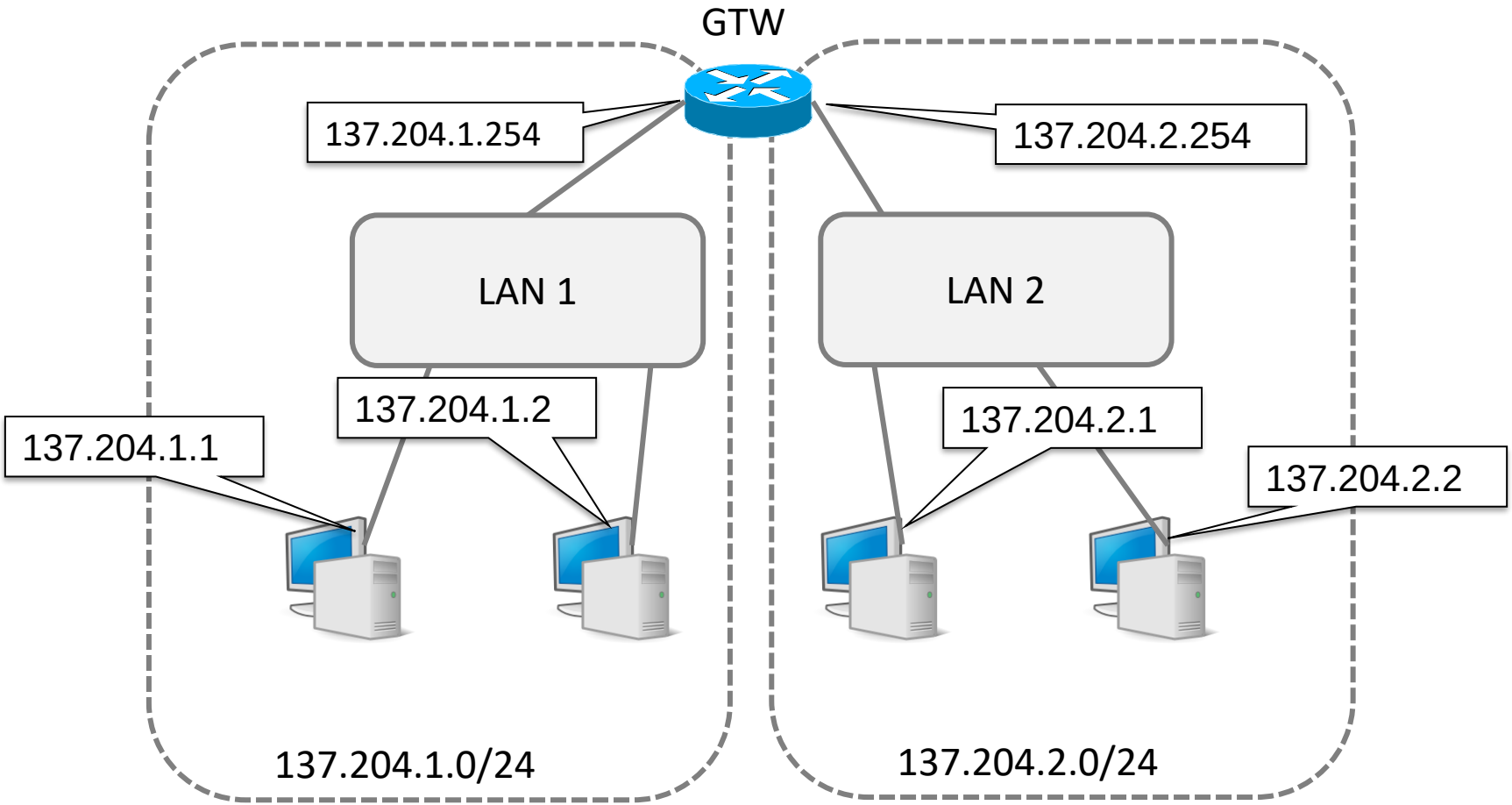
Topology 2


LAN 1
137.204.1.1
137.204.1.2
137.204.2.1
137.204.2.2
137.204.1.0/16
137.204.2.0/16

Here is how!

```
H1_1> ip netns exec h1_1 ifconfig veth0 netmask 255.255.0.0
H1_1> root@kali:~/Documents/BBS# ip netns exec h1_1 ifconfig
veth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
inet 137.204.1.1  netmask 255.255.0.0  broadcast 0.0.0.0
ether 6a:63:16:e7:49:ce  txqueuelen 1000  (Ethernet)

H2_1> ip netns exec h2_1 ifconfig veth0 netmask 255.255.0.0

H1_1> ip netns exec h1_1 ping 137.204.2.1
PING 137.204.2.1 (137.204.2.1) 56(84) bytes of data.
64 bytes from 137.204.2.1: icmp_seq=1 ttl=64 time=0.048 ms
64 bytes from 137.204.2.1: icmp_seq=2 ttl=64 time=0.044 ms
64 bytes from 137.204.2.1: icmp_seq=3 ttl=64 time=0.026 ms
64 bytes from 137.204.2.1: icmp_seq=4 ttl=64 time=0.037 ms
64 bytes from 137.204.2.1: icmp_seq=5 ttl=64 time=0.092 ms
64 bytes from 137.204.2.1: icmp_seq=6 ttl=64 time=0.022 ms
--- 137.204.2.1 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 107ms
rtt min/avg/max/mdev = 0.022/0.044/0.092/0.024 ms
```

Topology 2


GTW
137.204.1.254
137.204.2.254
LAN 1
LAN 2
137.204.1.2
137.204.2.1
137.204.1.1
137.204.2.2
137.204.1.0/24
137.204.2.0/24

Logical and Physical separation

- In Topology two there is no way to trick hosts
- They can communicate if and only if there is a gateways between the LANs
- If we try as before it is not going to work …