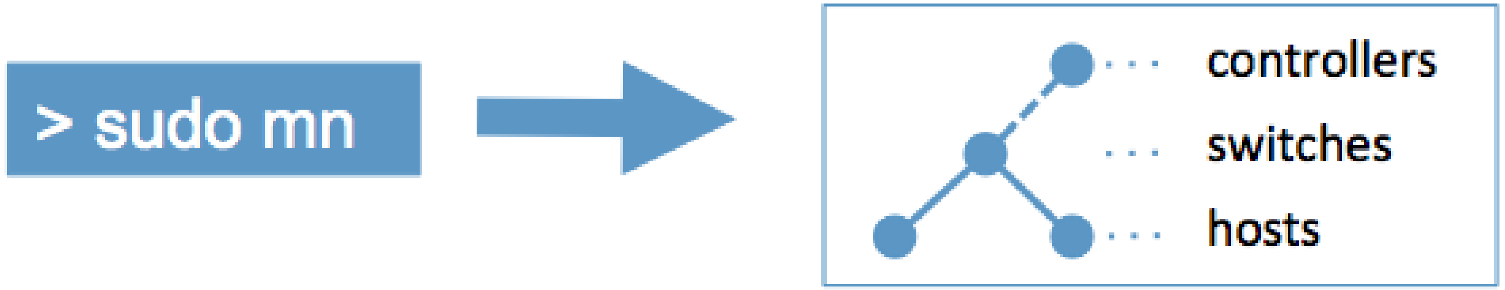
Mininet



```
> sudo mn
```

What is Mininet?

- A **virtual network environment** that can run on a single PC
- Runs real kernel, switch, and application code on a single machine
  - Command-line, UI, Python interfaces
- Many **OpenFlow features** are built-in
  - Useful: developing, deploying, and sharing

SDN

OpenFlow is a communications protocol that gives access to the forwarding plane of a network switch or router over the network.

ONF defines OpenFlow as the first standard communications interface defined between the control and forwarding layers of an SDN architecture. OpenFlow allows direct access to and manipulation of the forwarding plane of network devices such as switches and routers, both physical and virtual (hypervisor-based).

Why Use Mininet?

- Fast
- Possible to create custom topologies
- Can run real programs (anything that can run on Linux can run on a Mininet host)
- Programmable OpenFlow switches
- Easy to use
- Open source

Alternatives

- **Real system:** Pain to configure

- **Networked VMs:** Scalability

virtual machines such as VMware and virtual Box

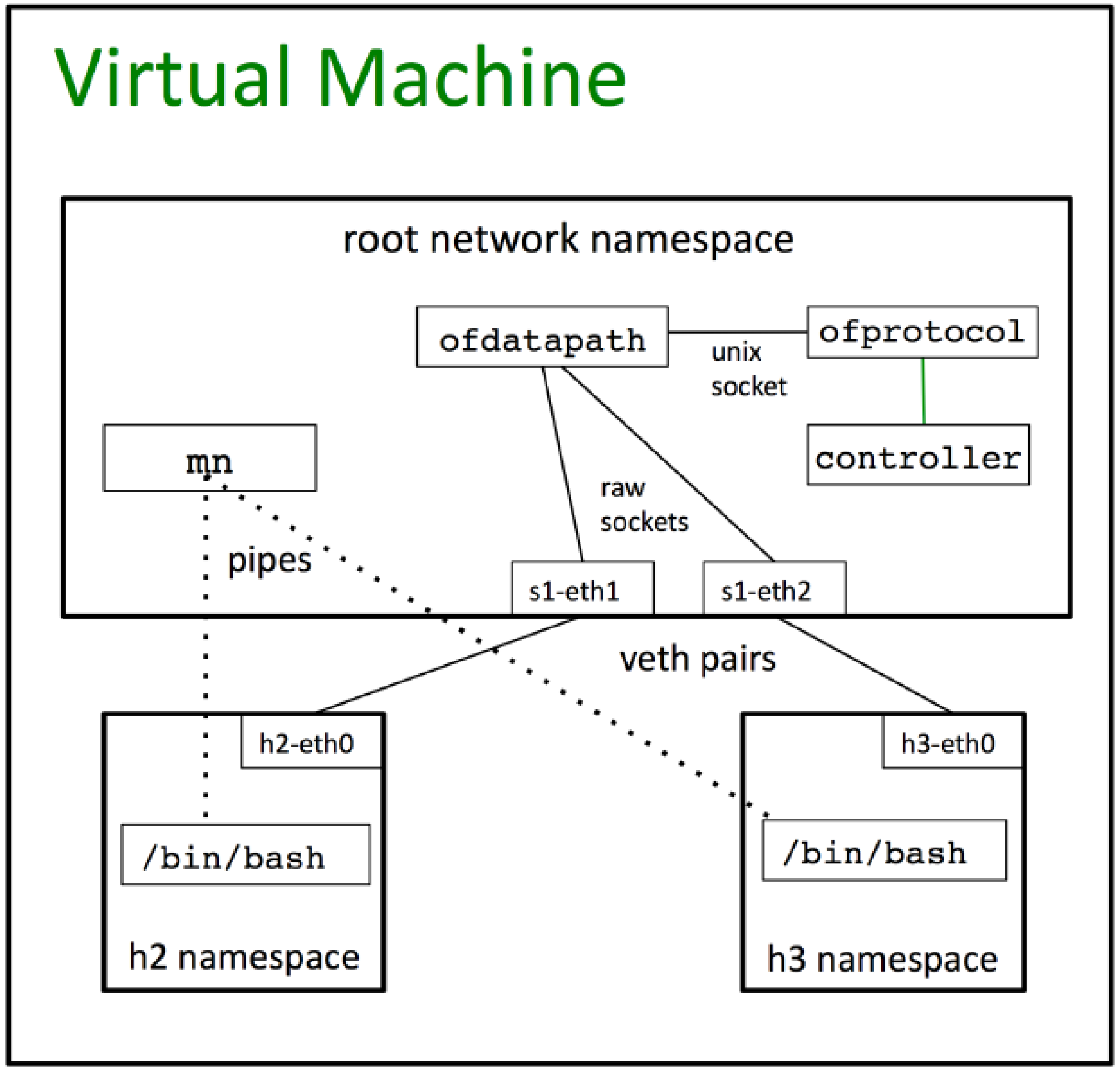
- **Simulator:** No path to hardware deployment

Mininet is Emulator(simulate hardware too!)

nutshell: Nutshell is an easy-to-use, drag-and-drop app builder that runs in a web browser. It's the only platform that allows non-technical people to build sophisticated business apps from scratch, without writing a single line of code.

مخلص کلام - پوست فندق و بادام و غیره

The Mininet VM in a Nutshell



- Launch mininet process
- Per host
  - Bash process
  - Network namespace
- Create veth pairs and assign to namespaces
- Create OpenFlow switch to connect hosts
- Create OpenFlow controller

The -bash process that is present after booting Linux is the default shell, known as Bash (short for "Bourne Again SHell"). The shell is a command line interpreter that allows users to interact with the operating system by running commands.

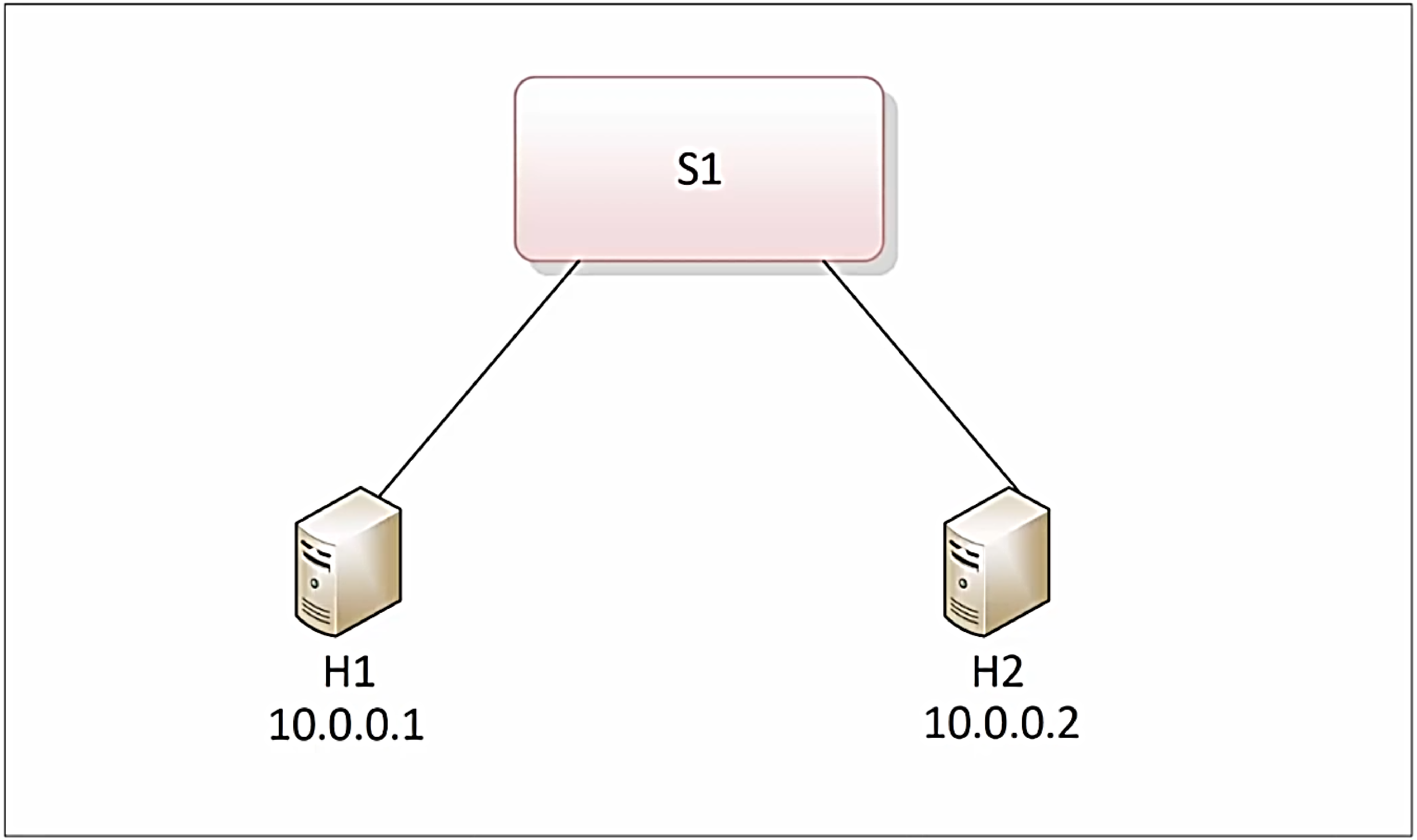
https://github.com/mininet/mininet/wiki/pdf/mininet-hotnets2010-final.pdf

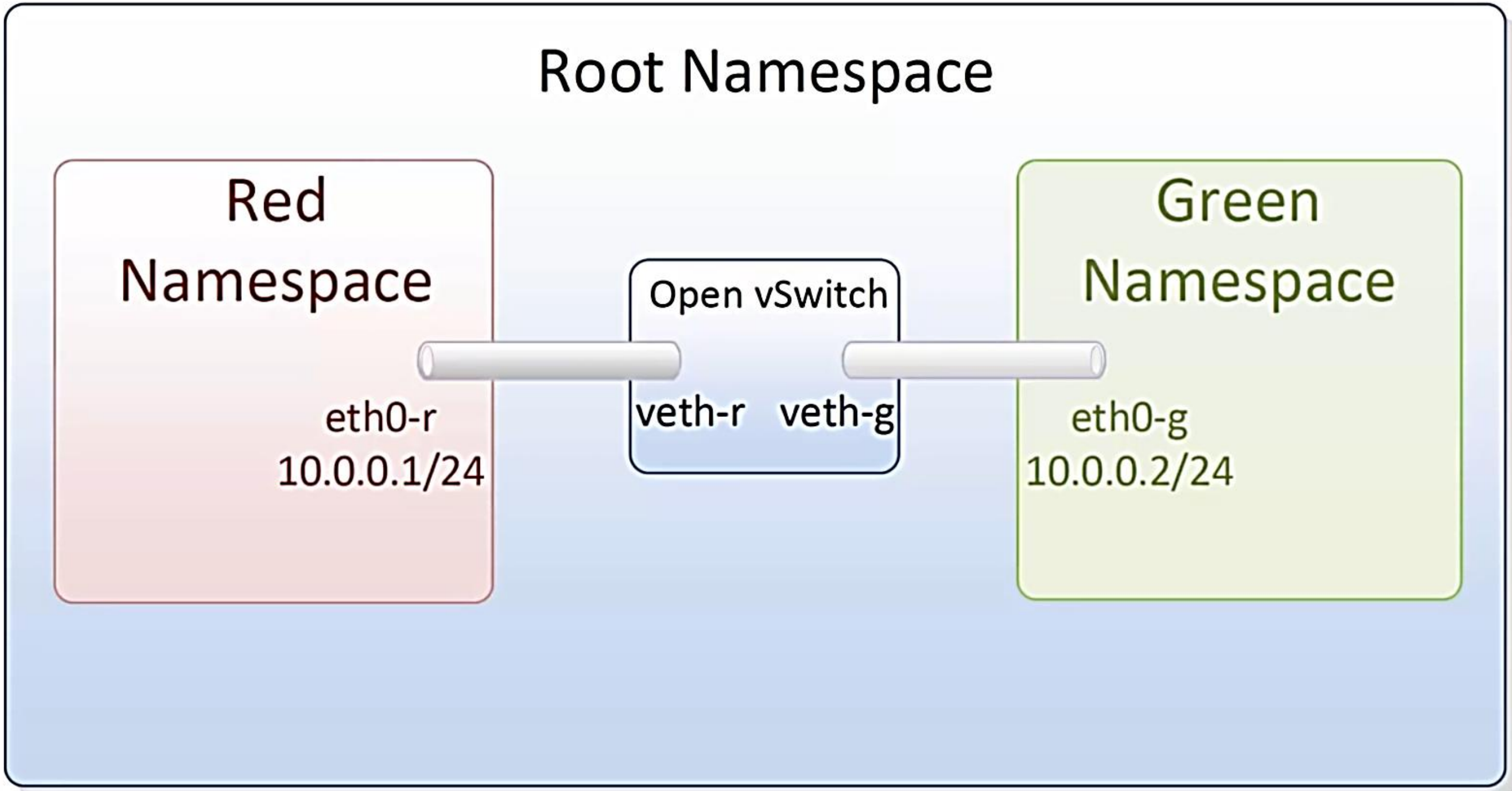
The hard, protective covering of a nut is called a nutshell. If you hear someone say "in a nutshell," they are most likely summing something up in a few words. It's more common to refer to a nutshell as simply a shell.

What are Linux Network Namespaces?

- Multiple isolated networking environments running on a single physical host or VM
- Each network namespace has its own interfaces, routing tables and forwarding tables
- Processes can be dedicated to one network namespace
- Used in OpenStack, Mininet, Docker, more...

Example

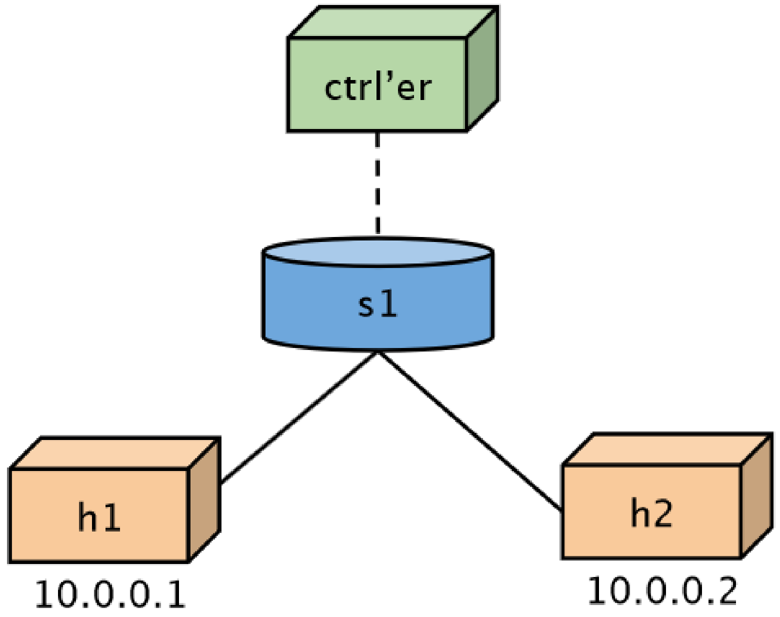



Root Namespace
Red Namespace
eth0-r
10.0.0.1/24
Open vSwitch
veth-r
veth-g
Green Namespace
eth0-g
10.0.0.2/24

Demo: basic network setup in Linux

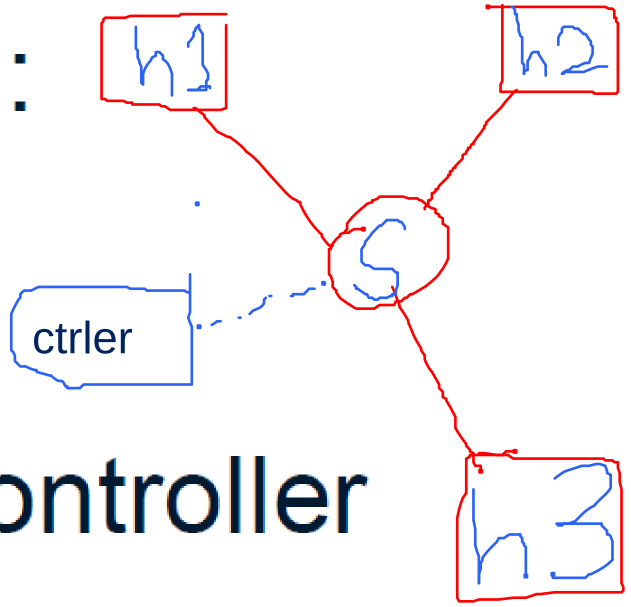
```
sudo bash
# Create host namespaces
ip netns add h1
ip netns add h2
# Create switch
ovs-vsctl add-br s1
# Create links
ip link add h1-eth0 type veth peer name s1-eth1
ip link add h2-eth0 type veth peer name s1-eth2
ip link show
# Move host ports into namespaces
ip link set h1-eth0 netns h1
ip link set h2-eth0 netns h2
ip netns exec h1 ip link show
ip netns exec h2 ip link show
# Connect switch ports to OVS
ovs-vsctl add-port s1 s1-eth1
ovs-vsctl add-port s1 s1-eth2
ovs-vsctl show
# Set up OpenFlow controller
ovs-vsctl set-controller s1 tcp:127.0.0.1
ovs-controller ptcp: &
ovs-vsctl show
```

```
# Configure network
ip netns exec h1 ifconfig h1-eth0 10.1
ip netns exec h1 ifconfig lo up
ip netns exec h2 ifconfig h2-eth0 10.2
ip netns exec h1 ifconfig lo up
ifconfig s1-eth1 up                        h2
ifconfig s1-eth2 up
# Test network
ip netns exec h1 ping -c1 10.2
```



Testing a Simple Mininet Setup

- Try setting up a simple topology with three hosts connected to a single switch:
  - sudo mn --test pingall --topo single,3



- This setup uses a default switch controller and switch
  - Mininet also allows you to use custom remote controllers (and custom switches)

Basic Mininet Command Line

- **--topo** – defines a topology via command line upon mininet start-up.
- **--switch** – defines the switch to be used. By default the OVSK software switch is used.
  openflow virtual switch
- **--controller** – defines the controller to be used. If unspecified default controller is used with a default hub behavior.

Trying Out Different Mininet Topologies

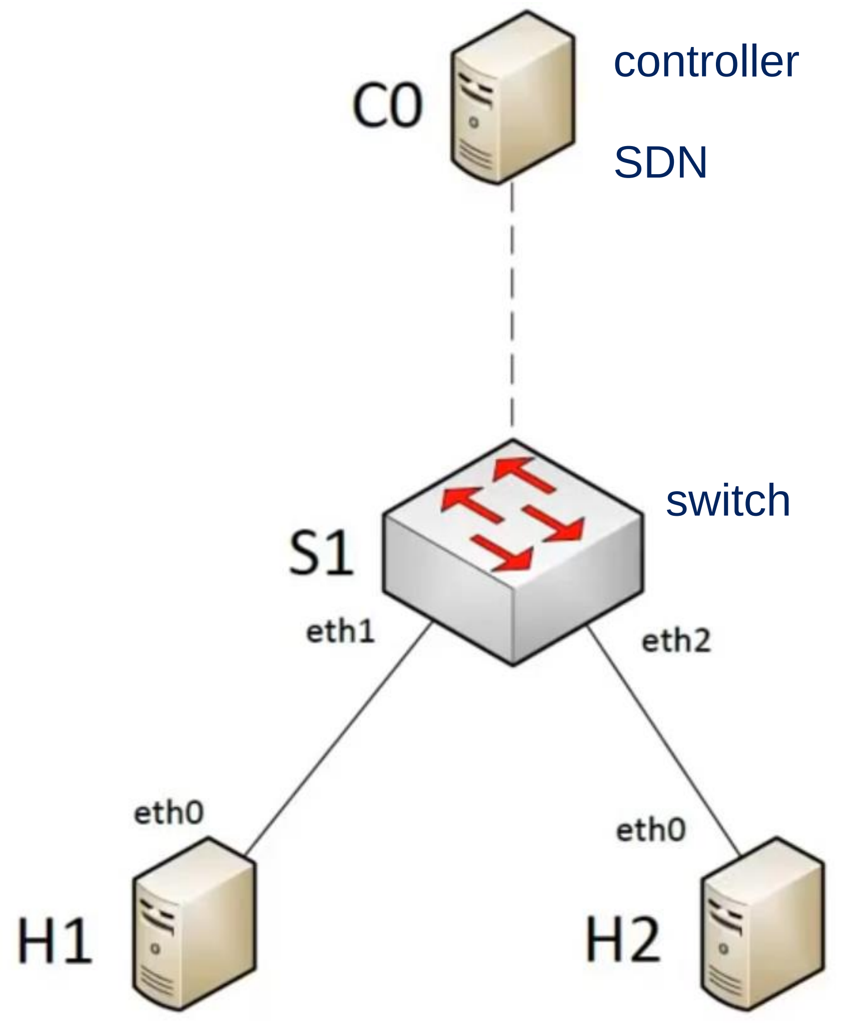
- Minimal network with two hosts, one (1) switch
  - sudo mn –topo minimal
- Example with 4 hosts and 4 switches
  - sudo mn --topo linear,4
- Example with 4 hosts all connected to one switch.
  - sudo mn --topo single,4
- Tree topology with defined depth and fan-out.
  - sudo mn --topo tree,depth=2,fanout=2

هواکش - گنجایش خروجی

branch factor

leaves are hosts
root and intermediate nodes are switches

- Minimal network with two hosts, one (1) switch
  - sudo mn –topo minimal

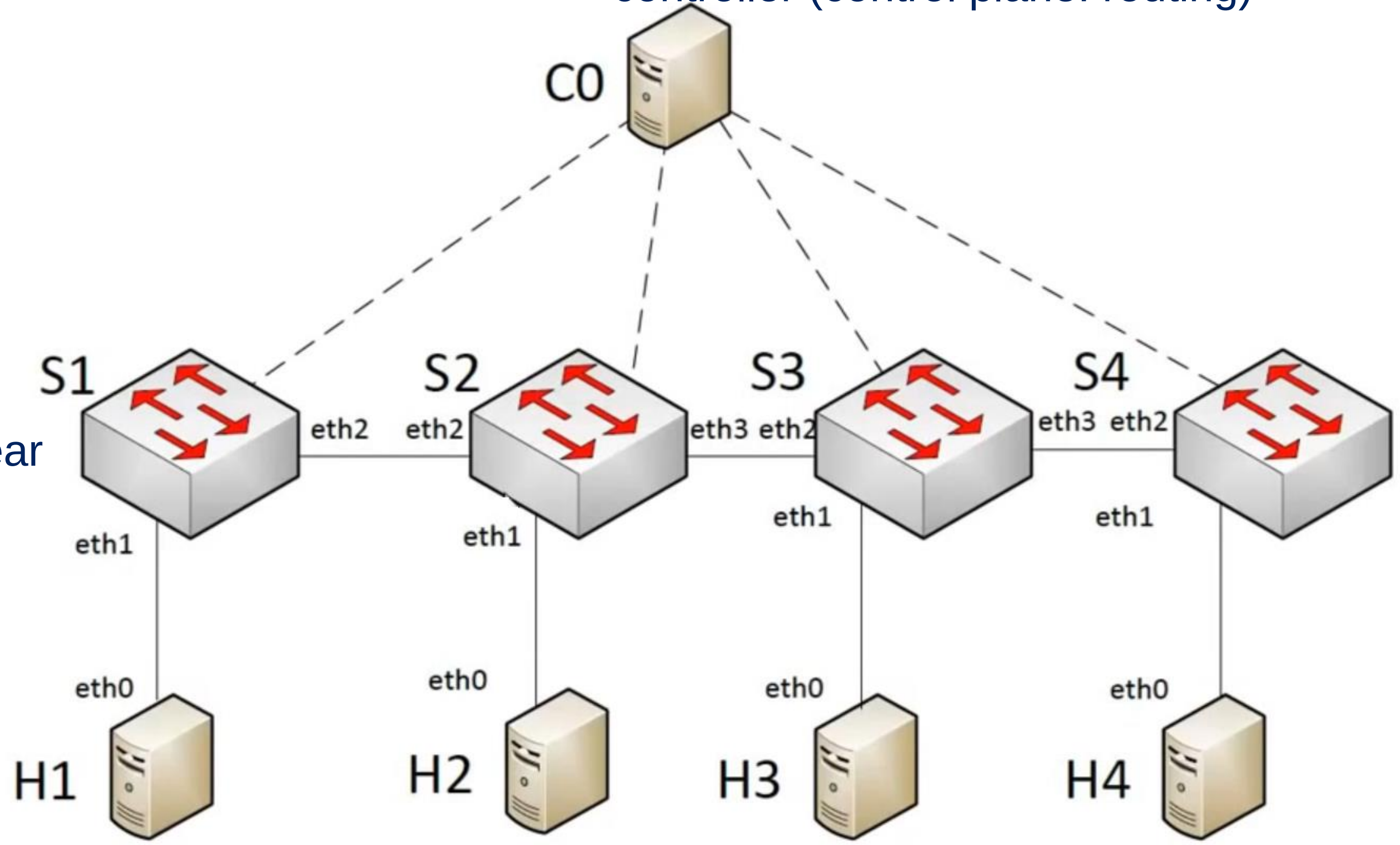


Example with 4 hosts and 4 switches

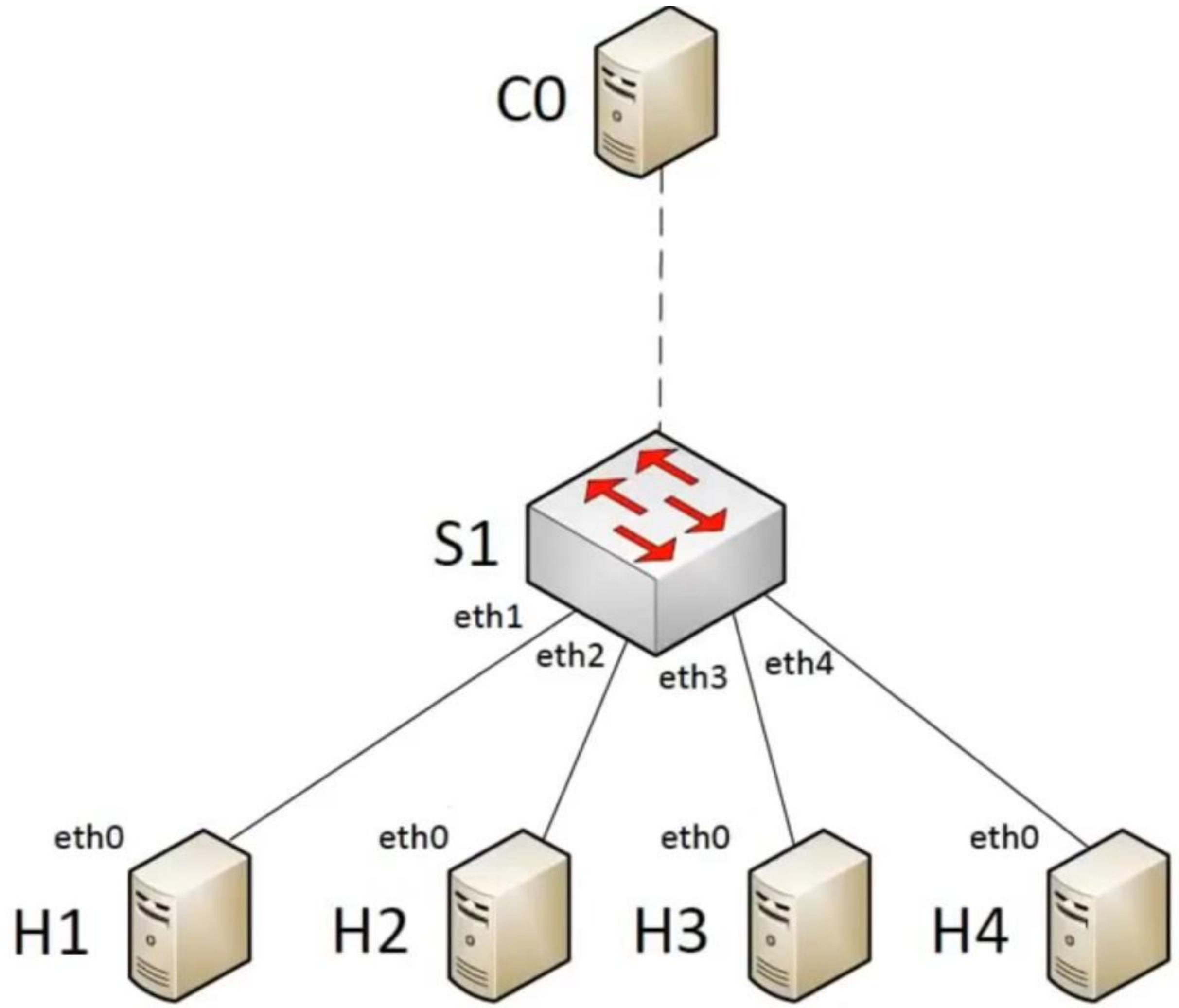
- sudo mn --topo linear,4

controller (control plane: routing)

switches(data plane: forwarding) are connected linear



- Example with 4 hosts all connected to one switch.
  - sudo mn --topo single,4



Tree topology with defined depth and fan-out.

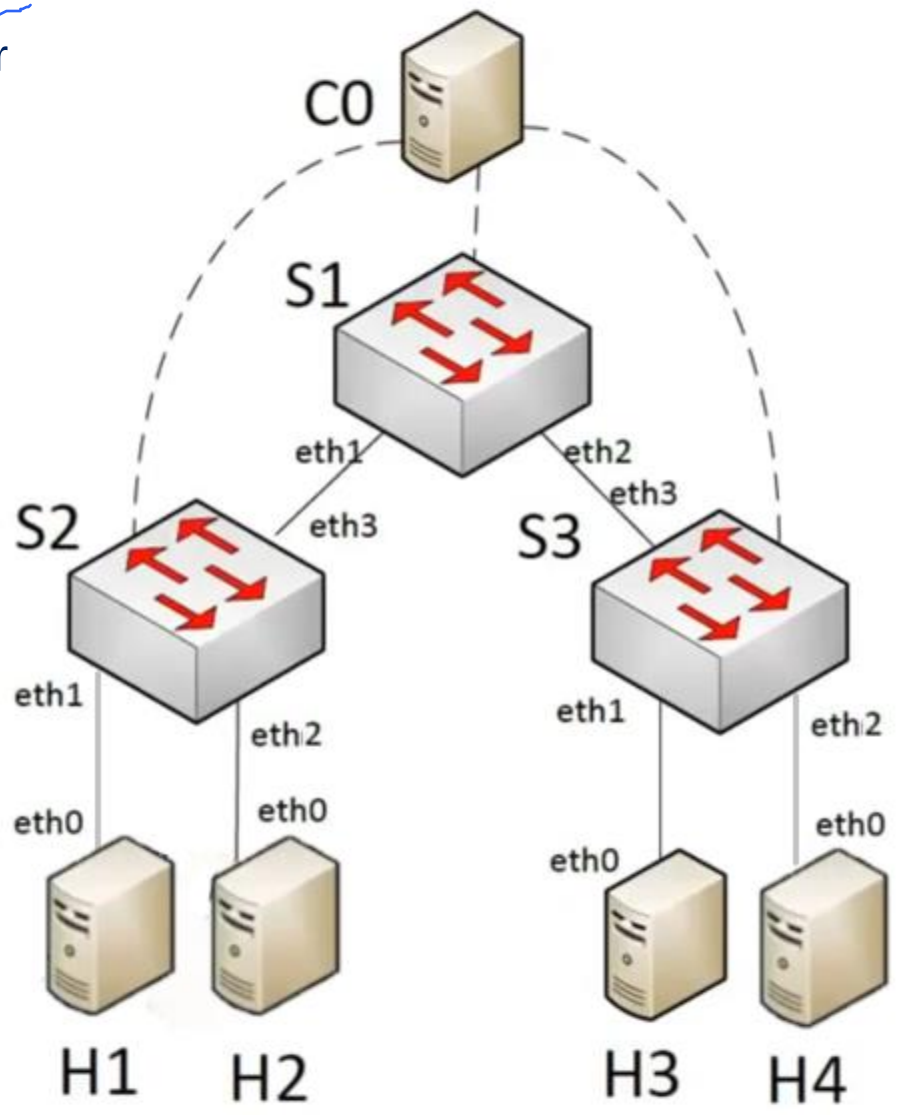
- `sudo mn --topo tree,depth=2,fanout=2`

branch factor

normal network:
routers, host, link-layer switches, other middle boxes
routers --> forwarding, routing

SDN(Software Defined Network):
controllers, switches, host, link-layer switches, other middle boxes
switches --> forwarding
controllers --> routing

**
forwarding: data plane
routing: control plane



leaves are hosts
root and intermediate nodes are switches

How mn Works: mn executes Python

یک اسکریپت راه اندازی است که پایتون را اجرا می کند

- “mn” is a launch script that executes Python
- Consider: “—topo linear,4”

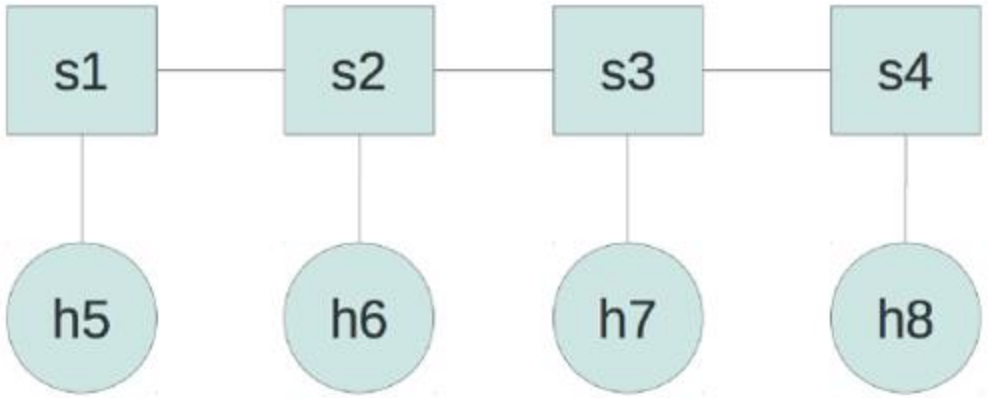
```
from mininet.net import Mininet

from mininet.topo import LinearTopo

Linear = LinearTopo(k=4)

net = Mininet(topo=Linear)

net.start()
net.pingAll()
net.stop()
```



Writing Your Own Mininet Topologies

- Example: Two hosts, one switch
- **mininet.cli.CLI(net)** before net.stop() will escape to interactive CLI before script terminates
- **addLink** allows you to specify: Bandwidth (bw) in Mbps, Delay (delay), Maximum Queue Size (max_queue_size), Loss (loss) in percentage

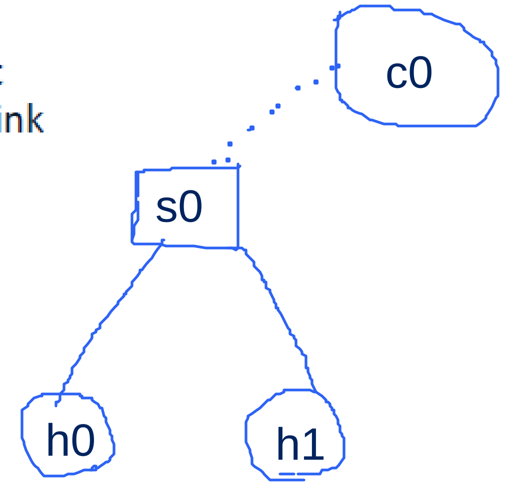
```
from mininet.net import Mininet
from mininet.util import createLink
net = Mininet()

# Creating nodes in the network.
c0 = net.addController()
h0 = net.addHost('h0')
s0 = net.addSwitch('s0')
h1 = net.addHost('h1')

# Creating links between nodes in network (2-ways)
net.addLink(h0, s0)
net.addLink(h1, s0)

# Configuration of IP addresses in interfaces
h0.setIP('192.168.1.1', 24)
h1.setIP('192.168.1.2', 24)

net.start()
net.pingAll()
net.stop()
```



https://github.com/mininet/mininet/wiki/Introduction-to-Mininet

More Complicated Topology Generation

```
#!/usr/bin/python
```

روبرداری کردن

```
from mininet.topo import Topo
from mininet.net import Mininet
from mininet.util import dumpNodeConnections
from mininet.log import setLogLevel
```

inheritance from Topo class

```
class SingleSwitchTopo(Topo):
   "Single switch connected to n hosts."
   def __init__(self, n=2, **opts):
```

تعداد متغیر آرگومان

```
      # Initialize topology and default options
      Topo.__init__(self, **opts)
      switch = self.addSwitch('s1')

      # Python's range(N) generates 0..N-1
      for h in range(n):
         host = self.addHost('h%s' % (h + 1))
         self.addLink(host, switch)
```

```
def simpleTest():
   "Create and test a simple network"
   topo = SingleSwitchTopo(n=4)
   net = Mininet(topo)
   net.start()
   print "Dumping host connections"
   dumpNodeConnections(net.hosts)
   print "Testing network connectivity"
   net.pingAll()
   net.stop()

if __name__ == '__main__':
   # Tell mininet to print useful information
   setLogLevel('info')
   simpleTest()
```

Mininet Command Line Interface Usage

❖ Mininet Command Line Interface Usage

- *Interact with hosts and switches*
  - **Start a minimal topology**

```
$ sudo mn
```

  - **Start a minimal topology using a remote controller**

```
$ sudo mn --controller=remote,ip=[IP_ADDDR],port=[listening port]
```

  - **Start a custom topology**

```
$ sudo mn --custom [topo_script_path] --topo=[topo_name]
```

  - **Display nodes**

```
mininet> nodes
```

  - **Display links**

```
mininet> net
```

  - **Dump information about all nodes** اطلاعات مربوط به همه گره ها را تخلیه کنید

```
mininet> dump
```

Mininet Command Line Interface Usage

❖ Mininet Command Line Interface Usage

- Interact with hosts and switches
  - Check the IP address of a certain node

```
mininet> h1 ifconfig -a
```

  - Print the process list from a host process

```
mininet> h1 ps -a
```

- Test connectivity between hosts
  - Verify the connectivity by pinging from host1 to host2

```
mininet> h1 ping -c 1 h2
```

  - Verify the connectivity between all hosts

```
mininet> pingall
```

MiniNet commands

sudo mn

percentage

- (...) --link tc,bw=100,delay=1ms,loss=0,max_queue_size=1000,...

- ping (10 echo requests)
  - h1 ping h2 -c 10

count

- iperf
  - To perform a TCP bandwidth test between hosts
  - iperf h1 h2
- exit
  - Release resources

* external terminal:

```
xterm h1
```

opening wireshark on xterm: wireshark &