به نام خدا

پروژه درس شبکه

مولف:پرنیان ملک زاده
دستیار آموزشی: آرمان سعیدی
استاد حیدرپور

۱.
این دستور namespace هایی که نام گذاری شده اند را نشان می دهد. اما namespace های پیشفرض در دایرکتوری ما ذخیره نشده اند همانطور که اسمی هم ندارند.

۲.

```
parnian@parnian:~$ sudo ip netns add test
parnian@parnian:~$ ip netns list
test
```

۳.

```
parnian@parnian:~$ sudo ip netns exec test ifconfig -a
lo: flags=8<LOOPBACK>  mtu 65536
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

parnian@parnian:~$ ifconfig -a
ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.16.238.129  netmask 255.255.255.0  broadcast 172.16.23
        inet6 fe80::20c:29ff:fea0:5be0  prefixlen 64  scopeid 0x20<link
        ether 00:0c:29:a0:5b:e0  txqueuelen 1000  (Ethernet)
        RX packets 21386  bytes 30872169 (30.8 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2671  bytes 204883 (204.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 40  memory 0x3fe00000-3fe20000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 239  bytes 20189 (20.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 239  bytes 20189 (20.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

If config -a تمام اینترفیس های در دسترس رو نشون میده

Sudo ip netns exec test ifconfig -aاینترفیس هایی که در نیم اسپیسی که ساختیم و اسمشو تست گذاشتیم نشون میده

۴.

```
root@parnian:/home/parnian# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
    link/ether 00:0c:29:a0:5b:e0 brd ff:ff:ff:ff:ff:ff
3: s-eth1@H1-eth0: <BROADCAST,MULTICAST,M-DOWN> mtu 1500 qdisc noop state DOWN mode DEFAULT group default qlen 1000
    link/ether e2:07:43:43:3b:2f brd ff:ff:ff:ff:ff:ff
4: H1-eth0@s-eth1: <BROADCAST,MULTICAST,M-DOWN> mtu 1500 qdisc noop state DOWN mode DEFAULT group default qlen 1000
    link/ether ae:01:89:12:5b:06 brd ff:ff:ff:ff:ff:ff
5: s-eth2@H2-eth0: <BROADCAST,MULTICAST,M-DOWN> mtu 1500 qdisc noop state DOWN mode DEFAULT group default qlen 1000
    link/ether 52:b1:ba:02:51:42 brd ff:ff:ff:ff:ff:ff
6: H2-eth0@s-eth2: <BROADCAST,MULTICAST,M-DOWN> mtu 1500 qdisc noop state DOWN mode DEFAULT group default qlen 1000
    link/ether 02:9f:71:03:db:d1 brd ff:ff:ff:ff:ff:ff
```

از آنجا که ما هنوز نیم اسپیسی برای اینترفیس هایی که ساختیم در نظر نگرفته ایم ، در نیم اسپیس دیفالت یا پیشفرض ذخیره می شود(root namespace)

```
root@parnian:/home/parnian# ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default qlen 1000
    link/ether 00:0c:29:a0:5b:e0 brd ff:ff:ff:ff:ff:ff
3: s-eth1@if4: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group default qlen 1000
    link/ether e2:07:43:43:3b:2f brd ff:ff:ff:ff:ff:ff link-netns H1
5: s-eth2@if6: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group default qlen 1000
    link/ether 52:b1:ba:02:51:42 brd ff:ff:ff:ff:ff:ff link-netns H2
```

۵.

زمانیکه دو لینک h1-eth0 , h2-eht0 را به نت ان اس های h1, h2 میبریم لینک ۴و ۶ حذف میشوند و این تفاوت این تصویر با تصویر سوال قبل است.

```
root@parnian:/home/parnian# ip netns exec H1 ip link show
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
4: H1-eth0@if3: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group default qlen 1000
    link/ether ae:01:89:12:5b:06 brd ff:ff:ff:ff:ff:ff link-netnsid 0
root@parnian:/home/parnian# ip netns exec H2 link show
link: missing operand after ‘show’
Try 'link --help' for more information.
root@parnian:/home/parnian# ip netns exec H2 ip link show
1: lo: <LOOPBACK> mtu 65536 qdisc noop state DOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
6: H2-eth0@if5: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group default qlen 1000
    link/ether 02:9f:71:03:db:d1 brd ff:ff:ff:ff:ff:ff link-netnsid 0
```

اگر ما دستور قبل را برای نیم اسپیس های h1,h2 اجرا کنیم ، لینک های ۴ و ۶ را داخل آنها میبینیم

۶.

```
root@parnian:/home/parnian# ovs-vsctl show
e487bf42-77e1-4bf2-9245-de71dbe0565f
    Bridge s
        Port s
            Interface s
                type: internal
        Port s-eth2
            Interface s-eth2
        Port s-eth1
            Interface s-eth1
    ovs_version: "2.13.8"
```

همانطور که در تصویر قابل مشاهده است، ۳ اینترفیس برای سوییچ s وجود دارد که یکی از آنها internal interface است.

```
The internal interface and port in each bridge is both an implementation
requirem ent and exists for historical reasons relating to the implementation of
Linux bridging module.

The purpose is to hold the IP for the bridge itself (just like some physical
bridges do). This is also useful in cases where a bridge has a physical
interface that would normally have its own IP. Since assigning a port to an IP
wouldn't happen in a physical bridge, assigning an IP to the physical interface
would be incorrect, as packets would stop at the port and not be passed across
the bridge.

A physical Ethernet device that is part of an Open vSwitch bridge should not
have an IP address. You can restore functionality by moving the IP address to an
Open  thevSwitch "internal" device, such as the network device named after the
bridge itself.
```

.۷

```
root@parnian:/home/parnian# ip netns exec H1 ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.625 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.069 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.201 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.170 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.205 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.163 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=0.168 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=0.179 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=0.193 ms

64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=0.198 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=0.149 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=0.162 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=0.102 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=0.185 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=0.191 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=0.347 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=0.172 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=0.225 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=0.169 ms
64 bytes from 10.0.0.2: icmp_seq=20 ttl=64 time=0.154 ms
64 bytes from 10.0.0.2: icmp_seq=21 ttl=64 time=0.178 ms
^C
--- 10.0.0.2 ping statistics ---
21 packets transmitted, 21 received, 0% packet loss, time 20468ms
rtt min/avg/max/mdev = 0.069/0.200/0.625/0.107 ms
```

The switch **uses the MAC address to identify which device's outgoing packets are being sent, and where to deliver incoming packets**. The MAC address identifies the physical device and doesn't change, while the network layer (Layer 3) IP address, can be assigned dynamically to a device and change over time.

۸.

```
mininet> nodes
available nodes are:
h1 h2 s1
mininet> links
h1-eth0<->s1-eth1 (OK OK)
h2-eth0<->s1-eth2 (OK OK)
```

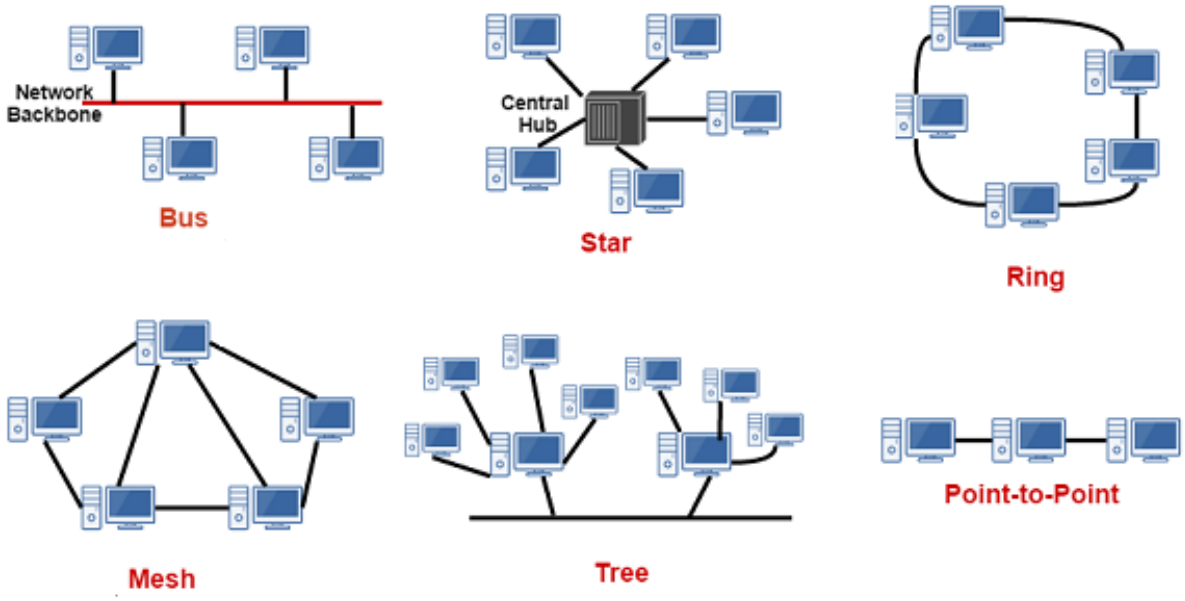
۹.

```
parnian@parnian:~$ sudo ovs-vsctl show
e487bf42-77e1-4bf2-9245-de71dbe0565f
    Bridge s
        Port s
            Interface s
                type: internal
        Port s-eth2
            Interface s-eth2
        Port s-eth1
            Interface s-eth1
    Bridge s1
        Controller "ptcp:6654"
        fail_mode: standalone
        Port s1
            Interface s1
                type: internal
        Port s1-eth1
            Interface s1-eth1
        Port s1-eth2
            Interface s1-eth2
    ovs_version: "2.13.8"
```

The diffrence is controller:
By default, mininet boots its default controller and connects to the controller at localhost:6633. A Controller is **a Node that is running (or has execed?) an OpenFlow controller**. An OpenFlow controller uses the OpenFlow protocol to connect and configure network devices to determine the best path for application traffic.

Topologies in mininet:

In mininet we have various topologies like minimal, sin- gle, reversed, linear, tree topology etc. We have run various topologies over Mininet to see their startup, stop and completion time in seconds. First we run minimal topology which simply creates two hosts and one switch.

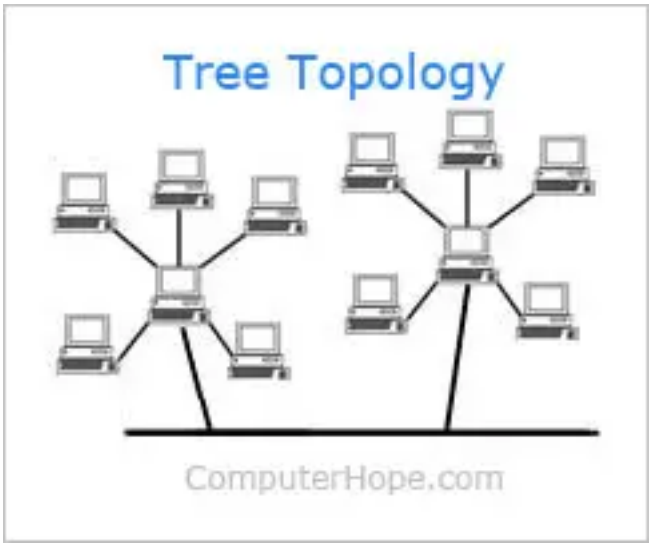


A **tree topology** is a special type of structure where many connected elements are arranged like the branches of a tree. For example, tree topologies are frequently used to organize the computers in a corporate network, or the information in a database.

In a tree topology, there can be only one connection between any two connected nodes. Because any two nodes can have only one mutual connection, tree topologies create a natural parent and child hierarchy.

Tree topology in computer networking

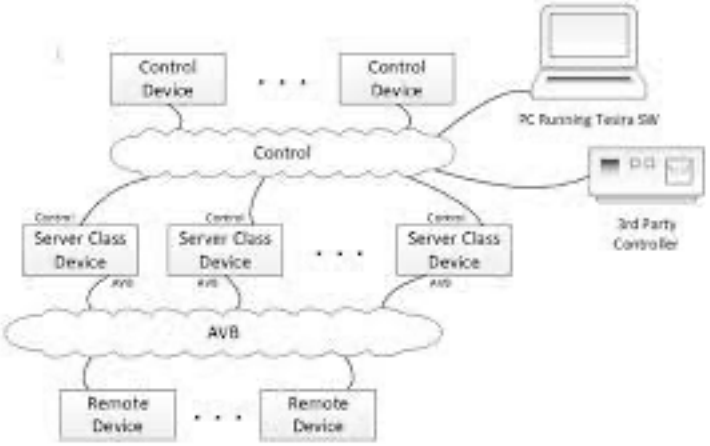
In computer networks, a tree topology is also known as a **star bus topology**. It incorporates elements of both a bus topology and a star topology. Below is an example network diagram of a tree topology, where the central nodes of two star networks are connected to one another.



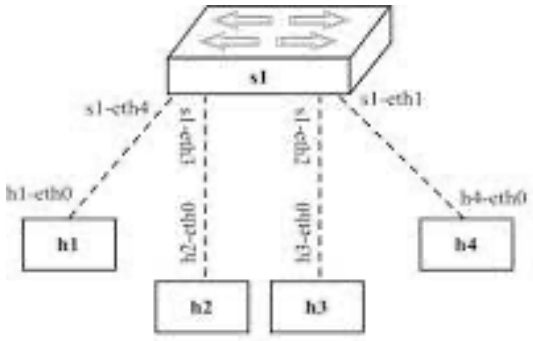
In the picture, if the main cable (trunk) between the two star topology networks failed, those networks would be unable to communicate with each other. However, computers on the same star topology would still be able to communicate.

SINGLE TOPOLOGY:

It is a simple topology with one openflow switch and k hosts. It also creates a link between switch and k hosts.



REVERSED TOPOLOGY:
It is similrder to single topology but connection order is reversed.




```
mininet> net
h1 h1-eth0:s1-eth4
h2 h2-eth0:s1-eth3
h3 h3-eth0:s1-eth2
h4 h4-eth0:s1-eth1
s1 lo:  s1-eth1:h4-eth0 s1-eth2:h3-eth0 s1-eth3:h2-eth0 s1-eth4:h1-eth0
c0
```

According to picture host 1 is connected on port 4, and host 4 is connected on port 1. So the port numbers have been reversed.

.١١

```
mininet> sh ovs-ofctl dump-flows s1
 cookie=0x0, duration=802.732s, table=0, n_packets=24, n_bytes=1832, priority=0 actions=NORMAL
```