Network Namespace 개요

이름 공간

독립된 공간

가상화기술, Docker등 Container 기술

Virtual Ethernet(veth), pairs

Pipe

# Linux Namespace 정의

http://man7.org/linux/man-pages/man7/namespaces.7.html

A namespace wraps a global system resource in an abstraction that makes it appear to the processes within the namespace that they have their own isolated instance of the global resource. Changes to the global resource are visible to other processes that are members of the namespace, but are invisible to other processes. One use of namespaces is to implement containers.

Linux provides the following namespaces:

Namespace	Constant	Isolates
IPC	CLONE_NEWIPC	System V IPC, POSIX message queues
Network	CLONE_NEWNET	Network devices, stacks, ports, etc.
Mount	CLONE_NEWNS	Mount points
PID	CLONE_NEWPID	Process IDs
User	CLONE_NEWUSER	User and group IDs
UTS	CLONE_NEWUTS	Hostname and NIS domain name

# Network Namespace 정의

Network namespaces (CLONE_NEWNET)

Network namespaces provide isolation of the system resources associated with networking: network devices, IPv4 and IPv6 protocol stacks, IP routing tables, firewalls, the /proc/net directory, the /sys/class/net directory, port numbers (sockets), and so on.

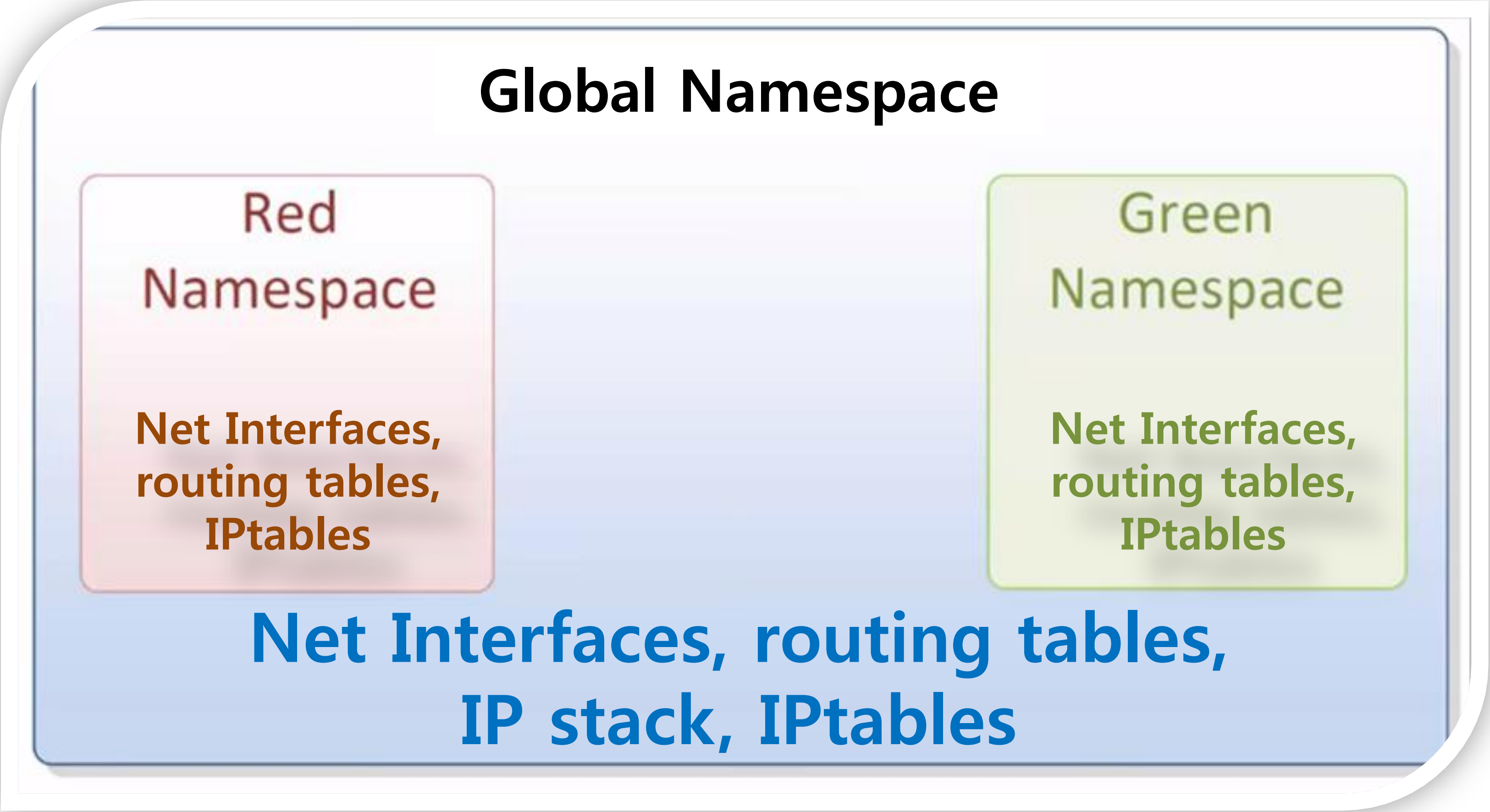
A physical network device can live in exactly one network namespace.

A virtual network device ("veth") pair provides a pipe-like abstraction that can be used to create tunnels between network namespaces, and can be used to create a bridge to a physical network device in another namespace.

When a network namespace is freed (i.e., when the last process in the namespace terminates), its physical network devices are moved back to the initial network namespace (not to the parent of the process).

Use of network namespaces requires a kernel that is configured with the CONFIG_NET_NS option.

# Network Namespace 정의



Global Namespace

Red Namespace

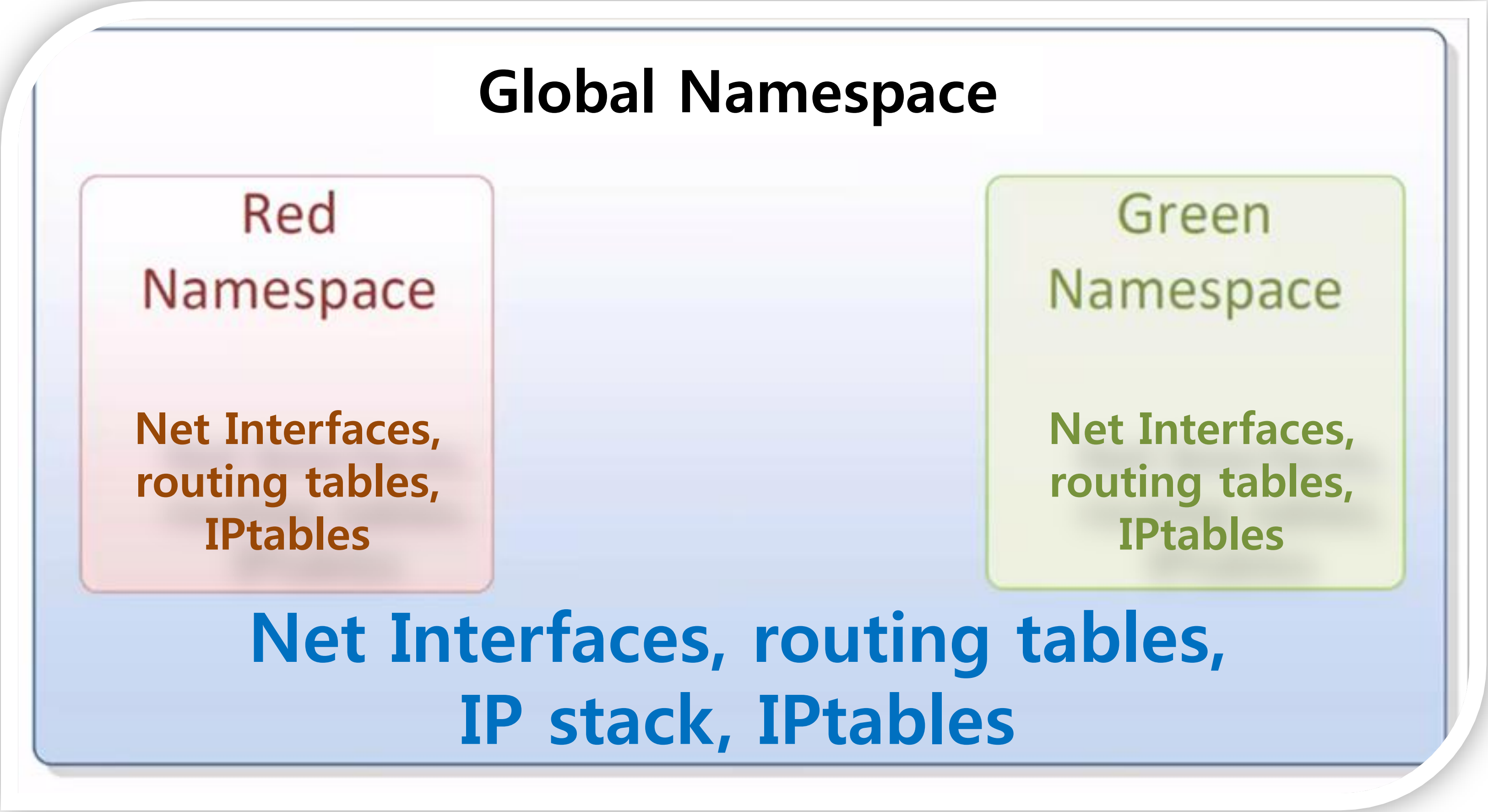
Net Interfaces, routing tables, IPtables

Green Namespace

Net Interfaces, routing tables, IPtables

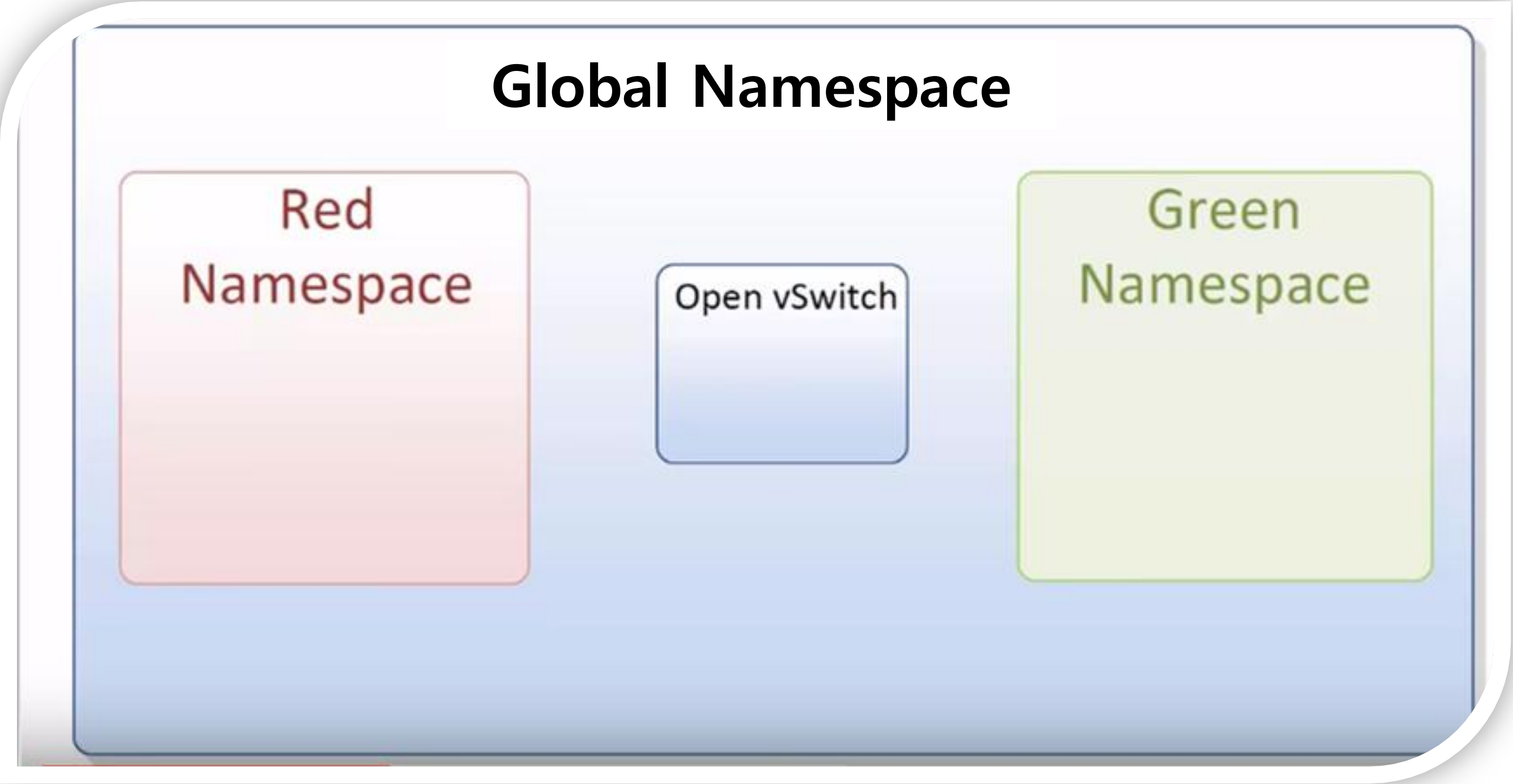
Net Interfaces, routing tables, IP stack, IPtables

```
# ip link
# ip address (ip a)
# ip route
# ip netns add red    => red namespace 생성
# ip netns add green  => green namespace 생성
# ip netns
# ls –l /var/run/netns
```

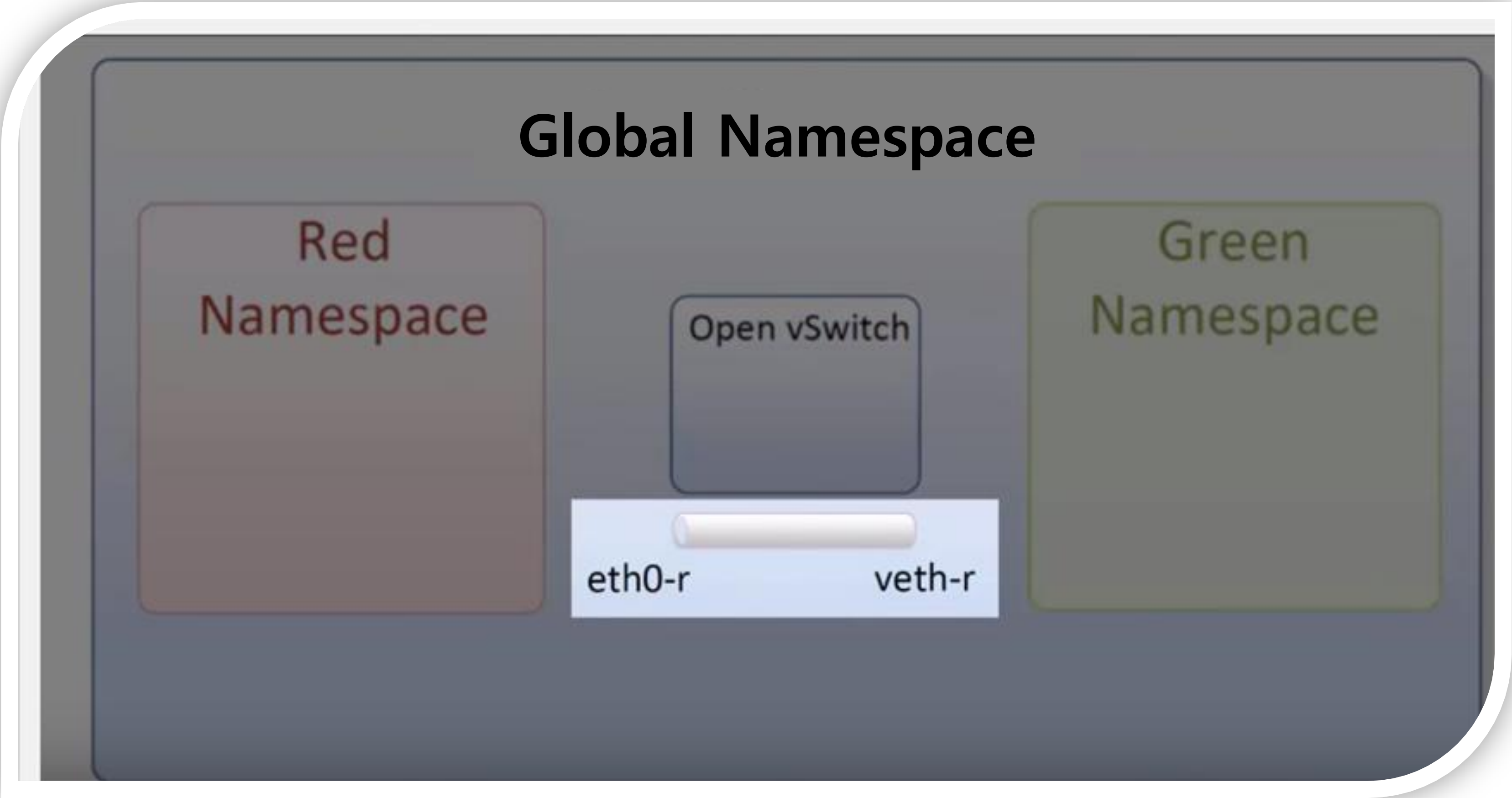


Global Namespace
Red Namespace
Net Interfaces, routing tables, IPtables
Green Namespace
Net Interfaces, routing tables, IPtables
Net Interfaces, routing tables, IP stack, IPtables

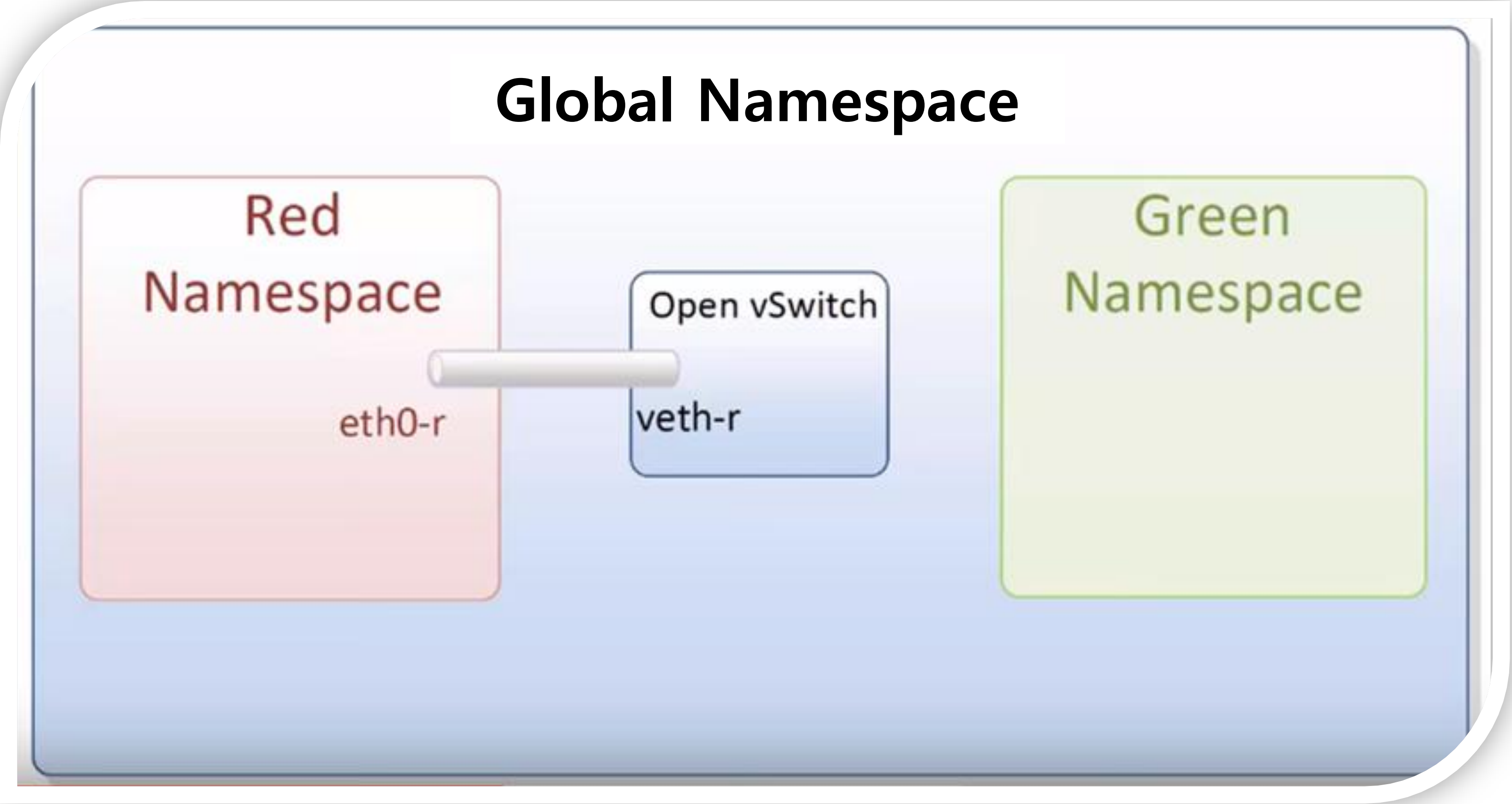
```
# ip netns exec red ip link
# ip netns exec green ip link

# ovs-vsctl add-br OVS1   => Open vSwitch 생성
# ovs-vsctl show
# ip link
```



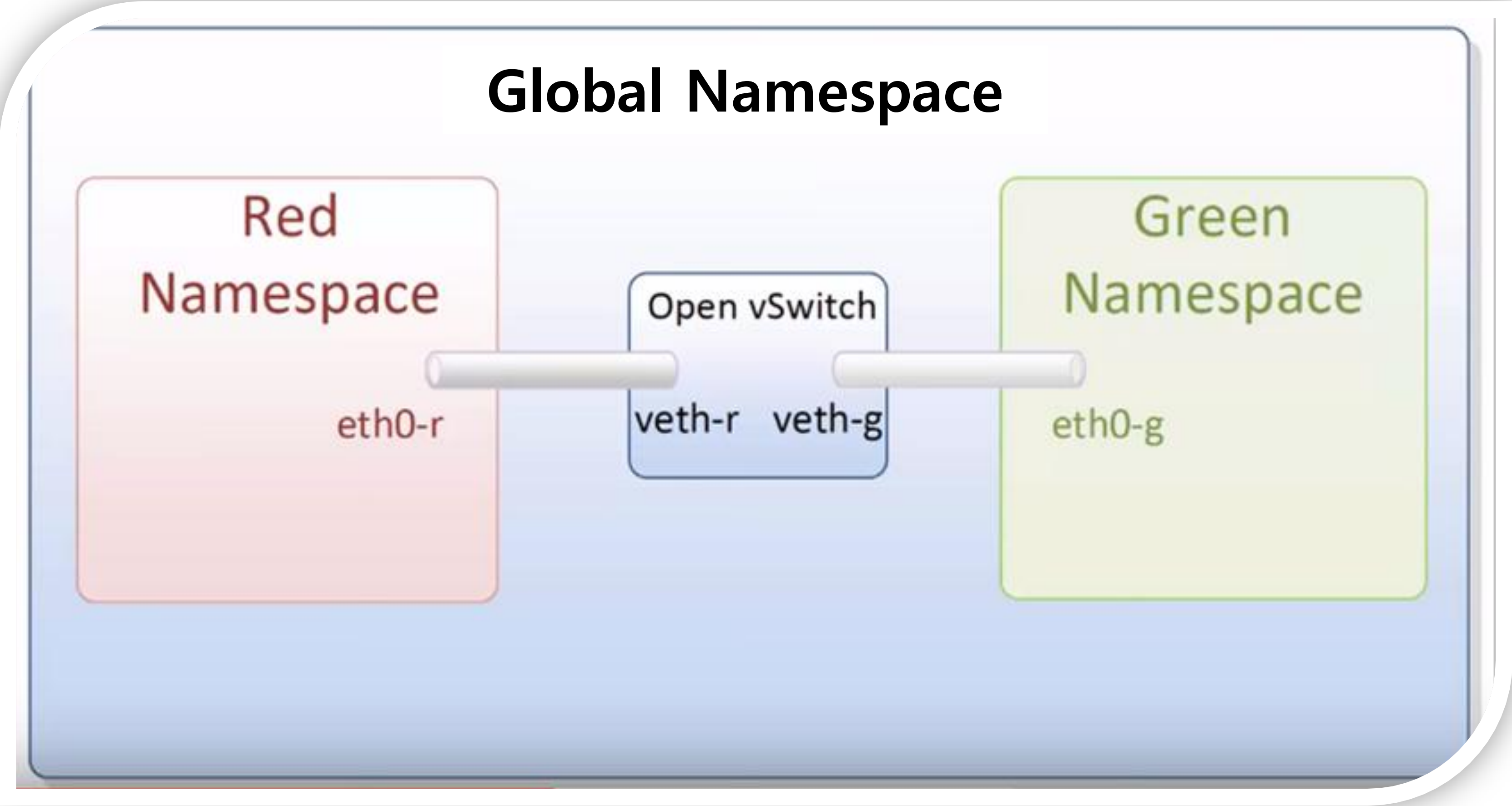
```
# ip link add eth0-r type veth peer name veth-r   => veth생성 (Red)
# ip link
# ip link set eth0-r netns red
# ip link
# ip netns exec red ip link
```



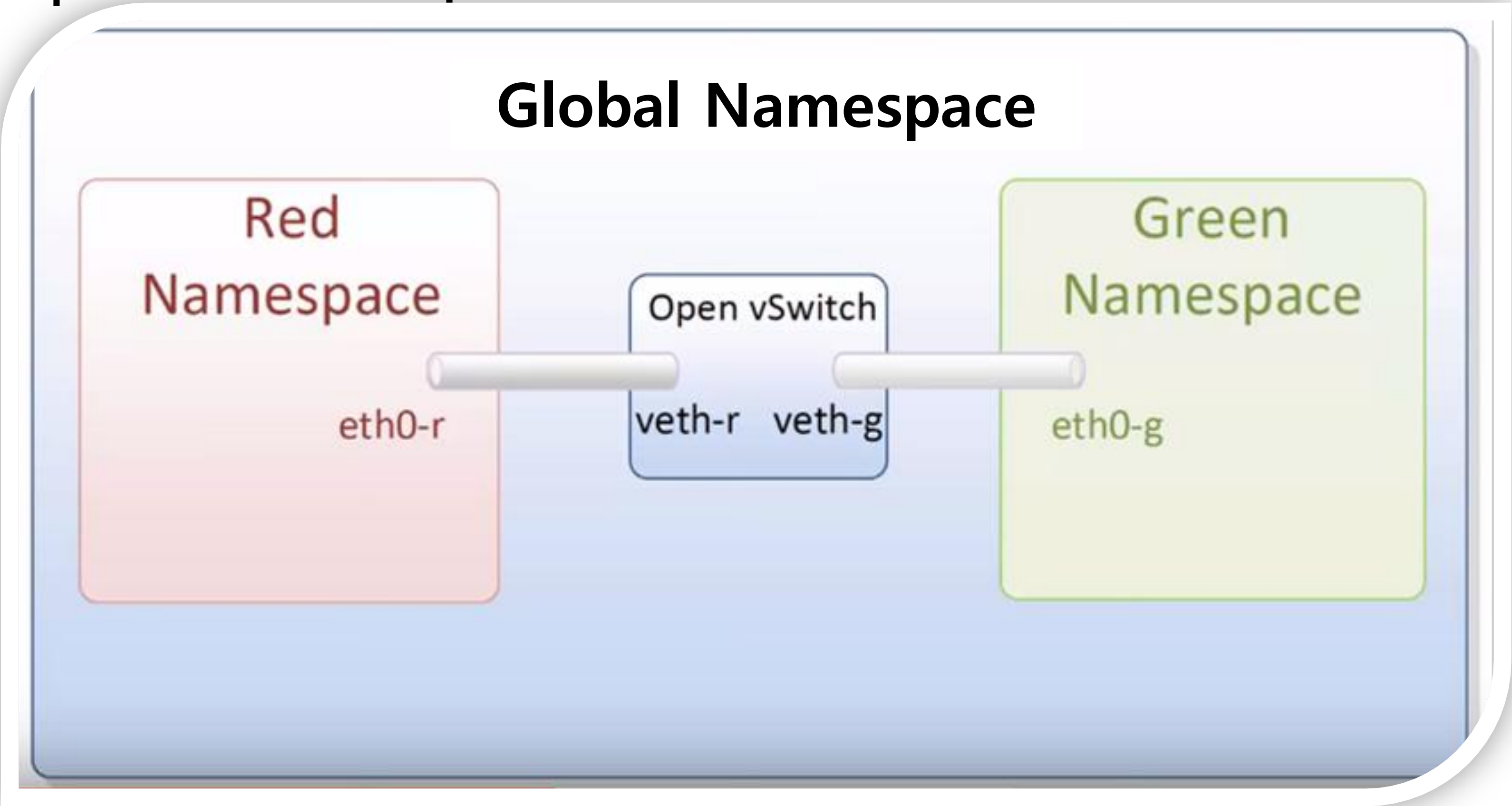
```
# ovs-vsctl add-port OVS1 veth-r   => OVS내 veth-r 추가
# ovs-vsctl show
```



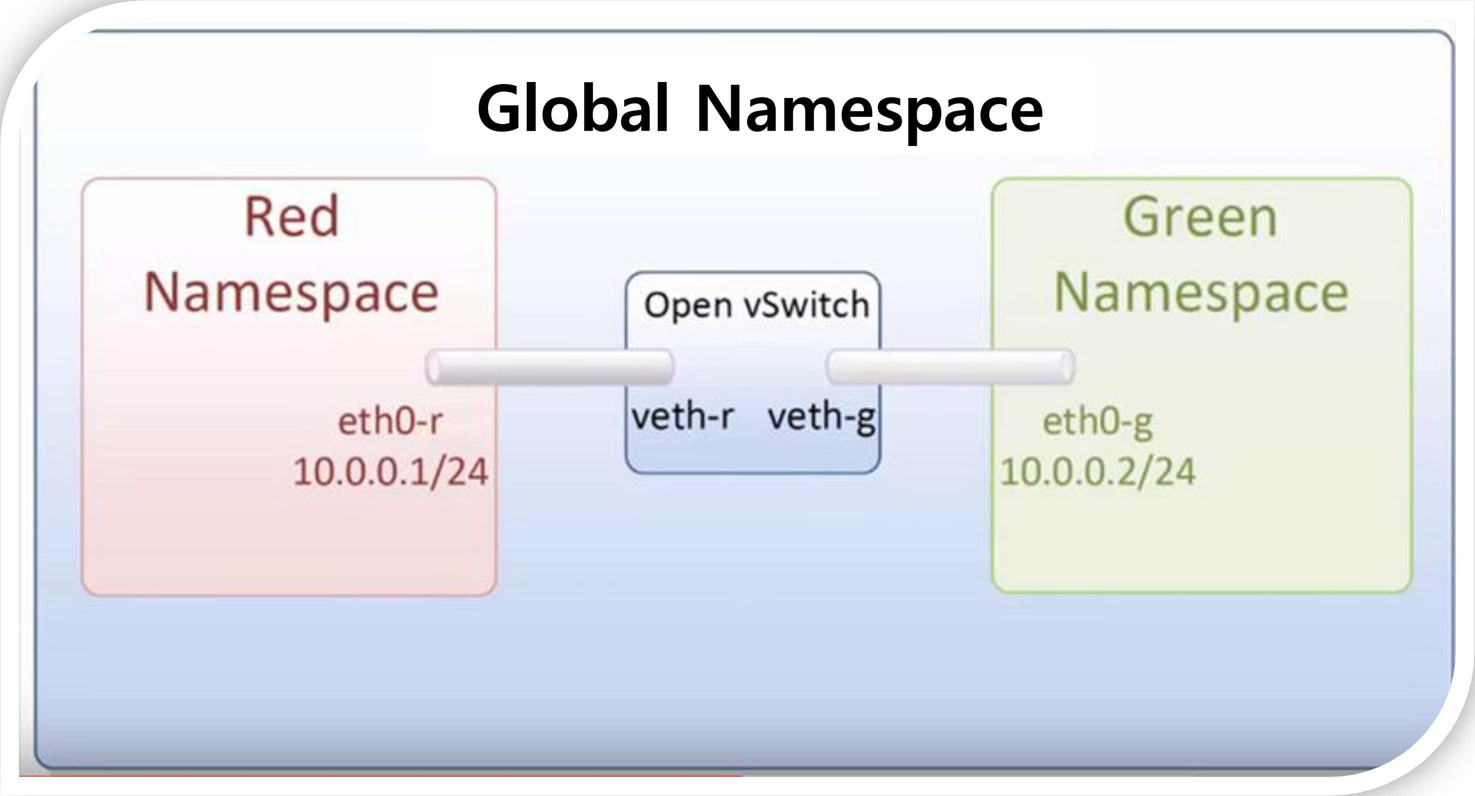
```
# ip link add eth0-g type veth peer name veth-g => veth생성 (Green)
# ip link
# ip link set eth0-g netns green
# ip netns exec green ip link
# ovs-vsctl add-port OVS1 veth-g  => OVS내 veth-g 추가
# ovs-vsctl show
```



```
# ip link
# ip link set veth-r up
# ip link set veth-g up

# ip netns exec red ip link set dev lo up
# ip netns exec red ip link set eth0-r up
# ip netns exec red ip address add 10.0.0.1/24 dev eth0-r
# ip netns exec red ip a
# ip netns exec red ip route
```



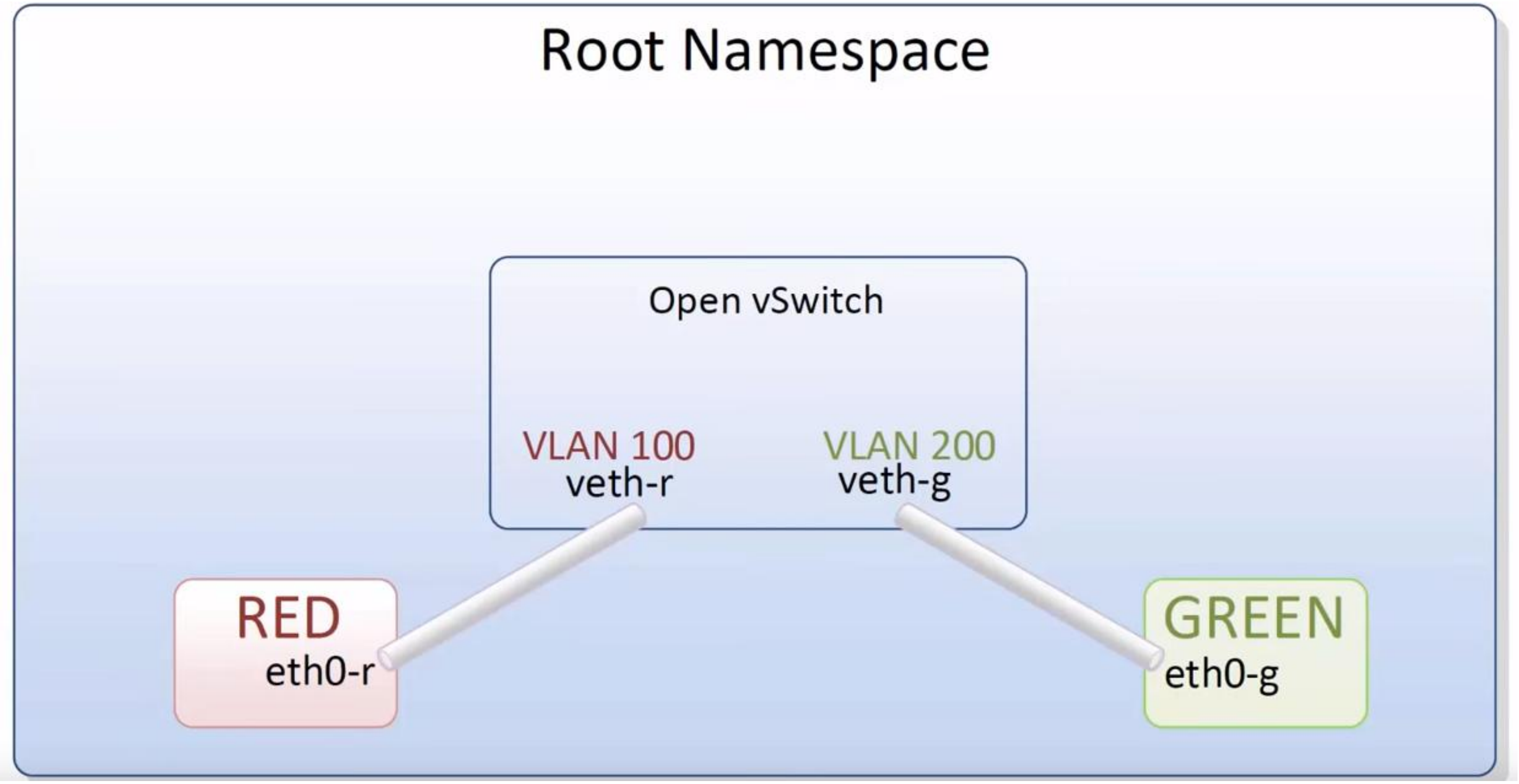
```
# ip link set dev veth-g up

# ip netns exec green bash
# ip link set dev lo up
# ip link set eth0-g up
# ip address add 10.0.0.2/24 dev eth0-g
# ip a
# ip route
# ping 10.0.0.1; ping 10.0.0.2
```



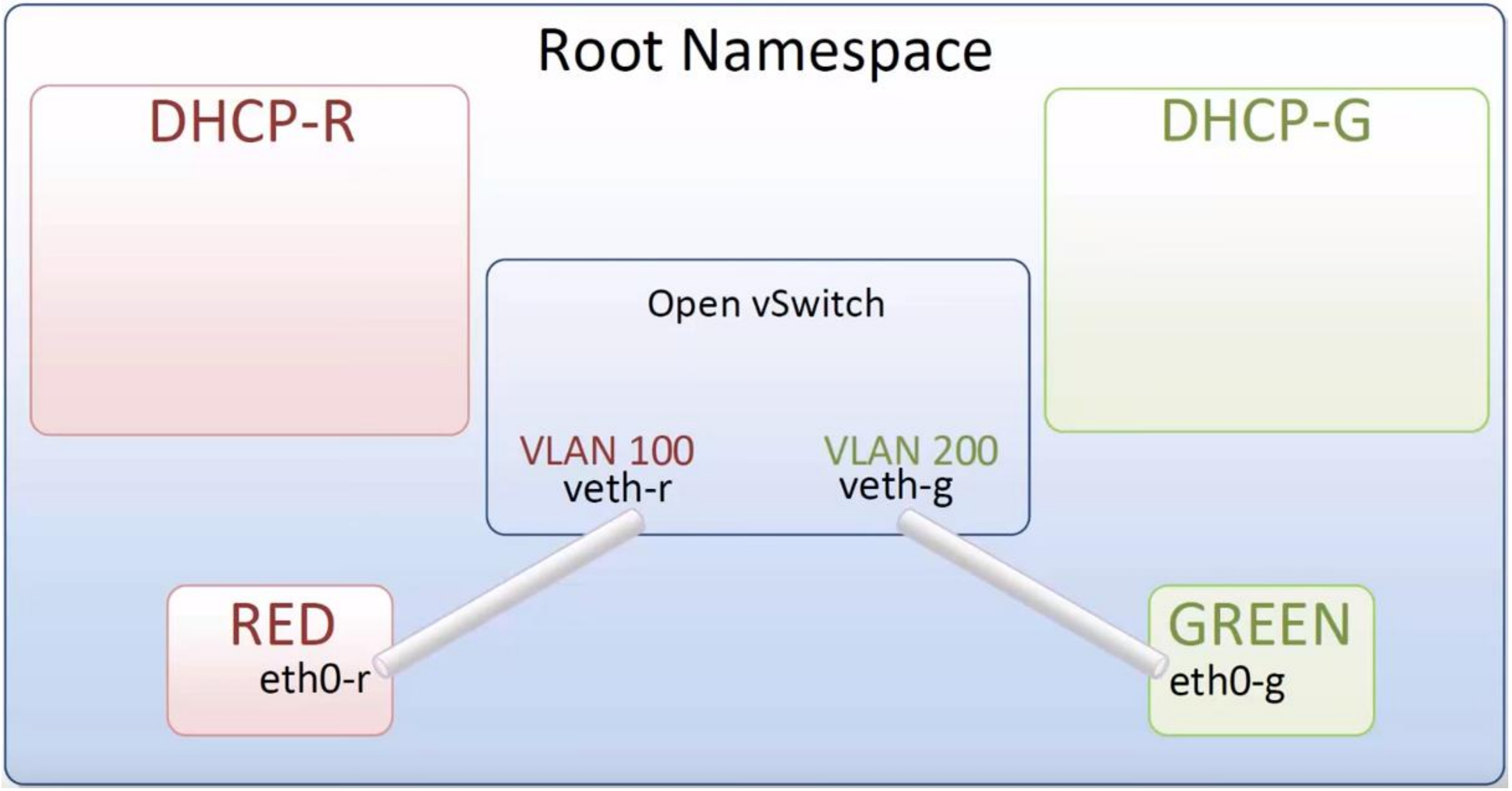
```
# openstack network namespace naming rule
# ip netns
qdhcp-[f130c5a0-1879-4e83-91c0-0c9950eb2182, 네트워크 UUID]
  : DHCP 프로토콜을 통해 인스턴스에게 IP주소를 할당
snat-[a62685ce-7c40-4836-96f1-0455e332986d, Source NAT UUID]
  : 여러 VM이 공용 IP 하나를 공유하여 사용
qrouter-[ec565b6f-b0a1-4b41-bfc5-1393b18bc1d9, 라우터 UUID]
  : 현재 연결된 서브넷 상의 인스턴스 트래픽 라우팅
qlbaas-[a24f3b84-30d3-4251-8149-092a8696038c, 로드밸런서 UUID]
  : 인스턴스에 대한 트래픽을 로드밸런싱하는 Haproxy와 같은 서비스 제공
```

```
# ip netns help
Usage: ip netns list
       ip netns add NAME
       ip netns delete NAME
       ip netns identify PID
       ip netns pids NAME
       ip netns exec NAME cmd ...
       ip netns monitor
```

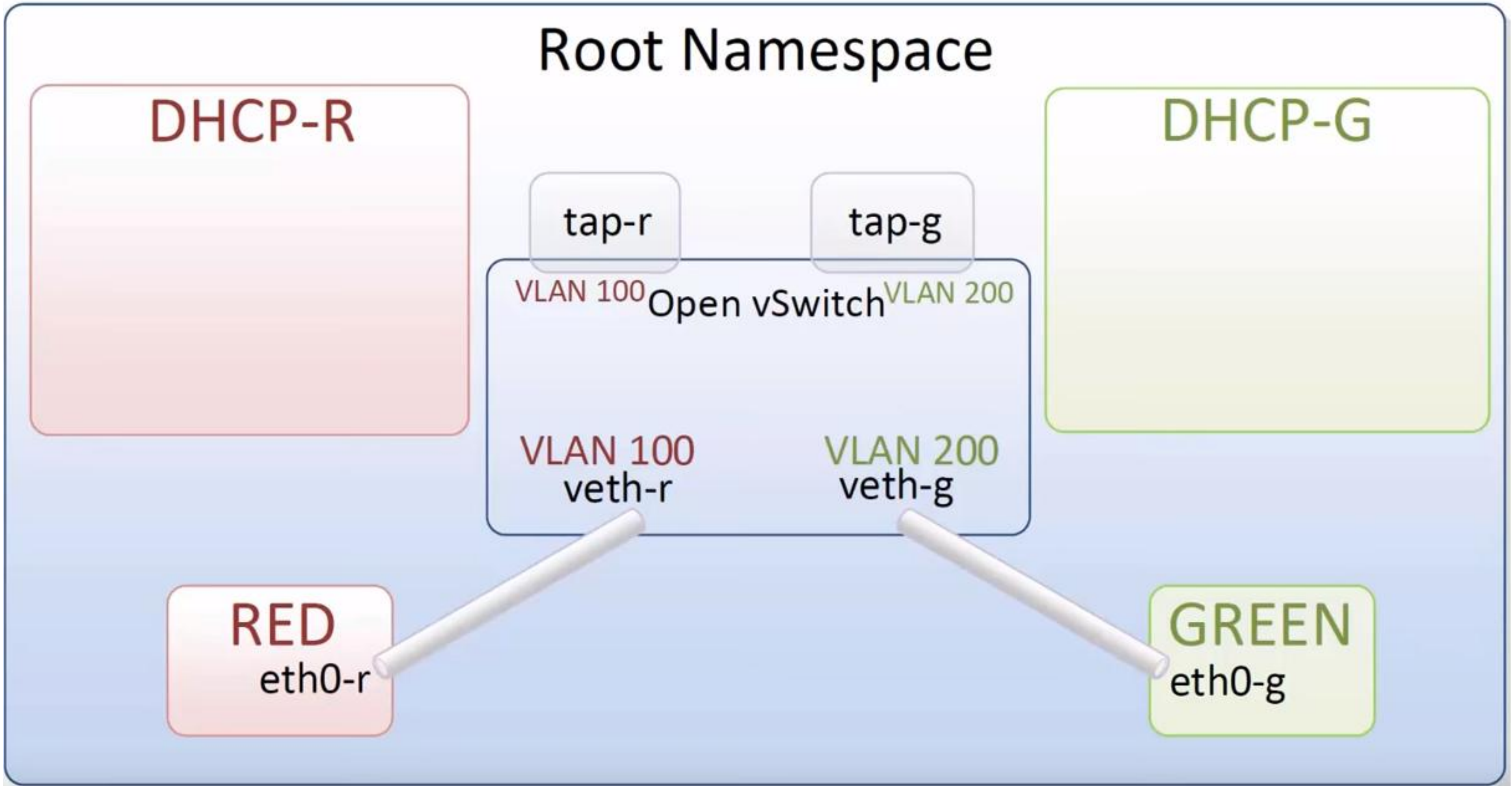
```
# ovs-vsctl set port veth-r tag=100
# ovs-vsctl set port veth-g tag=200
# ovs-vsctl show
# ip netns exec red ip a ; ip netns exec green ip a
# ip netns exec red ip address del 10.0.0.1/24 dev eth0-r
# ip netns exec green ip address del 10.0.0.2/24 dev eth0-g
# ip netns exec red ip a ; ip netns exec green ip a
```



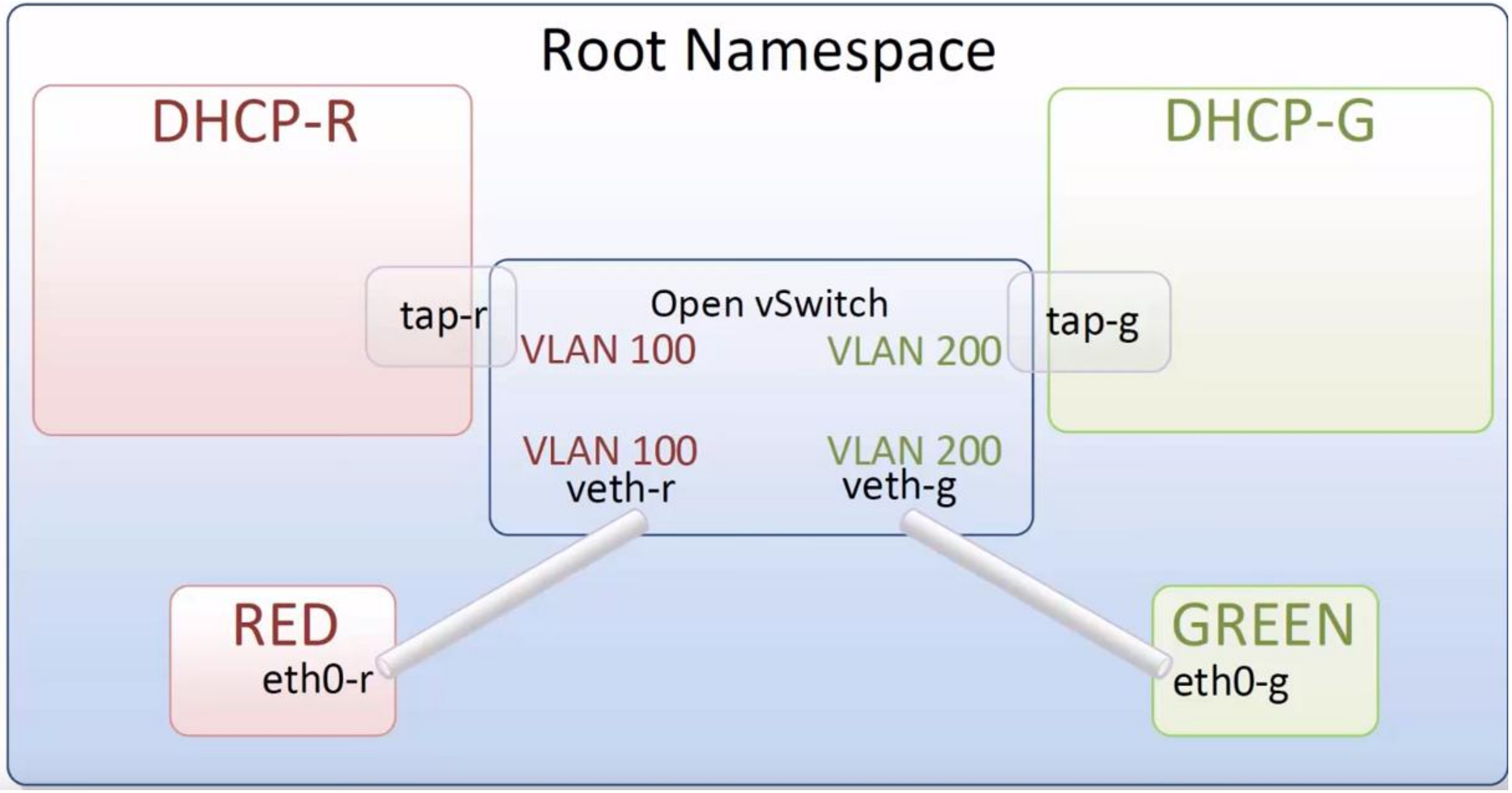
```
# ip netns add dhcp-r  ; ip netns add dhcp-g
# ovs-vsctl show
```



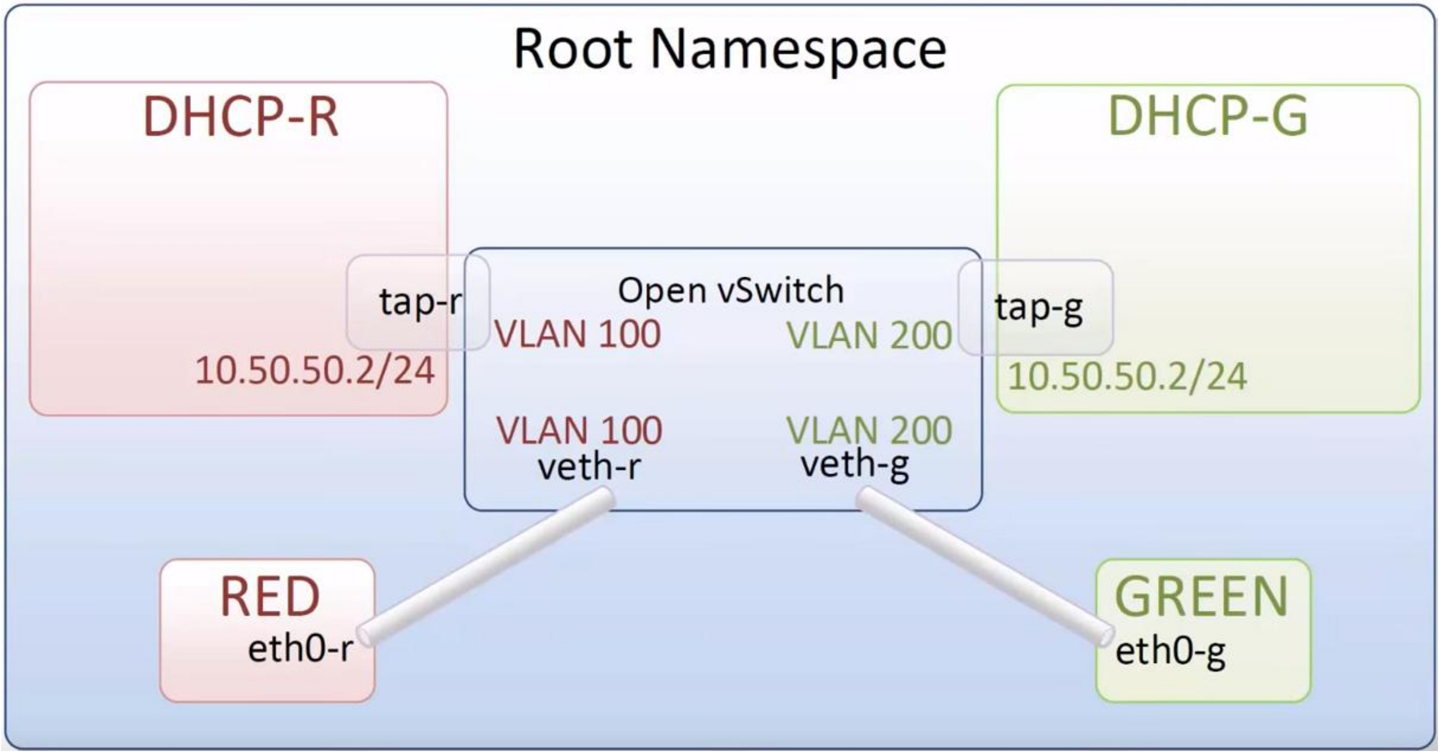
```
# ovs-vsctl add-port OVS1 tap-r
# ovs-vsctl set interface tap-r type=internal
# ovs-vsctl set port tap-r tag=100
# ovs-vsctl add-port OVS1 tap-g
# ovs-vsctl set interface tap-g type=internal
# ovs-vsctl set port tap-g tag=200
```



```
# ip netns exec dhcp-r ip link ; ip netns exec dhcp-g ip link
# ip link set tap-r netns dhcp-r
# ip link set tap-g netns dhcp-g
# ip link
# ovs-vsctl show
```



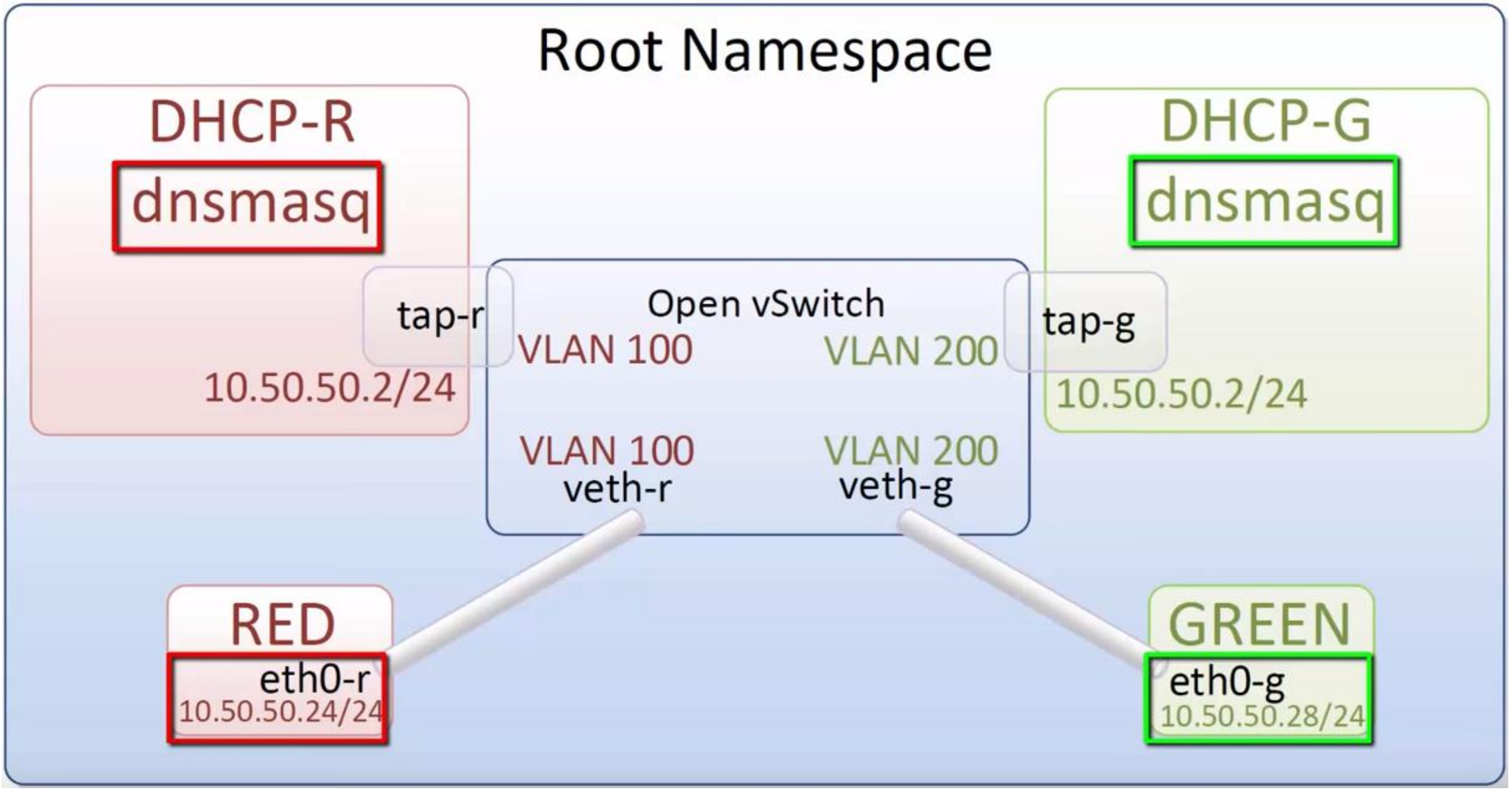
```
# ip netns exec dhcp-r bash
# ip link set dev lo up
# ip link set dev tap-r up
# ip address add 10.50.50.2/24 dev tap-r

# ip netns exec dhcp-g bash
# ip link set dev lo up
# ip link set dev tap-g up
# ip address add 10.50.50.2/24 dev tap-g
```



```
# ip netns exec dhcp-r dnsmasq --interface=tap-r --dhcp-
range=10.50.50.10,10.50.50.100,255.255.255.0
# ip netns exec dhcp-g dnsmasq --interface=tap-g --dhcp-
range=10.50.50.10,10.50.50.100,255.255.255.0
```

```
# ps -eaf |grep dns
libvirt+  1590     1  0 10:45 ?        00:00:00 /usr/sbin/dnsmasq --conf-file=/var/lib/libvirt/dnsmasq/default.conf
nobody    2536   833  0 10:46 ?        00:00:00 /usr/sbin/dnsmasq --no-resolv --keep-in-foreground --no-hosts --bind-
interfaces --pid-file=/run/sendsigs.omit.d/network-manager.dnsmasq.pid --listen-address=127.0.1.1 --conf-
file=/var/run/NetworkManager/dnsmasq.conf --cache-size=0 --proxy-dnssec --enable-
dbus=org.freedesktop.NetworkManager.dnsmasq --conf-dir=/etc/NetworkManager/dnsmasq.d
nobody    4295     1  0 18:01 ?        00:00:00 dnsmasq --interface=tap-r --dhcp-
range=10.50.50.10,10.50.50.100,255.255.255.0
nobody    4368     1  0 18:01 ?        00:00:00 dnsmasq --interface=tap-g --dhcp-
range=10.50.50.10,10.50.50.100,255.255.255.0
root      4730  7732  0 18:04 pts/25   00:00:00 grep --color=auto dns
root@stack-VirtualBox:~#
# ip netns identify 4295
dhcp-r
# ip netns identify 4368
dhcp-g
# ip netns exec red dhclient eth0-r
# ip netns exec red ip a
# ip netns exec green dhclient eth0-g
# ip netns exec green ip a
```

#


Root Namespace
DHCP-R
dnsmasq
tap-r
10.50.50.2/24
DHCP-G
dnsmasq
tap-g
10.50.50.2/24
Open vSwitch
VLAN 100
VLAN 200
VLAN 100
veth-r
VLAN 200
veth-g
RED
eth0-r
10.50.50.24/24
GREEN
eth0-g
10.50.50.28/24

참고

1. https://www.youtube.com/watch?v=_WgUwUf1d34
2. http://man7.org/linux/man-pages/man8/ip-netns.8.html
3. http://man7.org/linux/man-pages/man7/namespaces.7.html
4. http://man7.org/linux/man-pages/man2/clone.2.html
5. http://man7.org/linux/man-pages/man2/setns.2.html
6. http://man7.org/linux/man-pages/man2/unshare.2.html