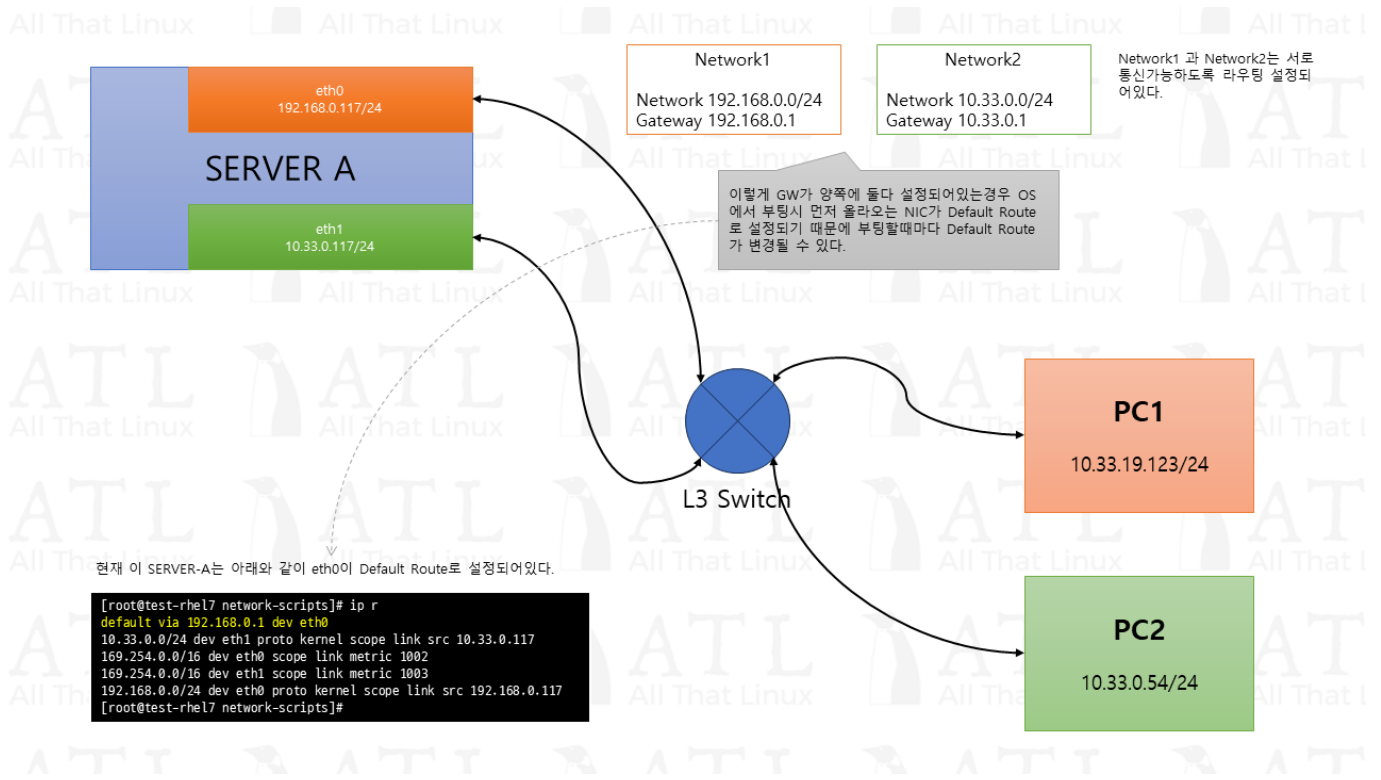


- Multiple Gateway** 3
- 3
- PING** 3
- NIC** 4
- 6
- NetworkManager** 8
- rp_filter** 9
- 10
- 10

Multiple Gateway

— 2024/04/19 04:09

NIC, Default Gateway

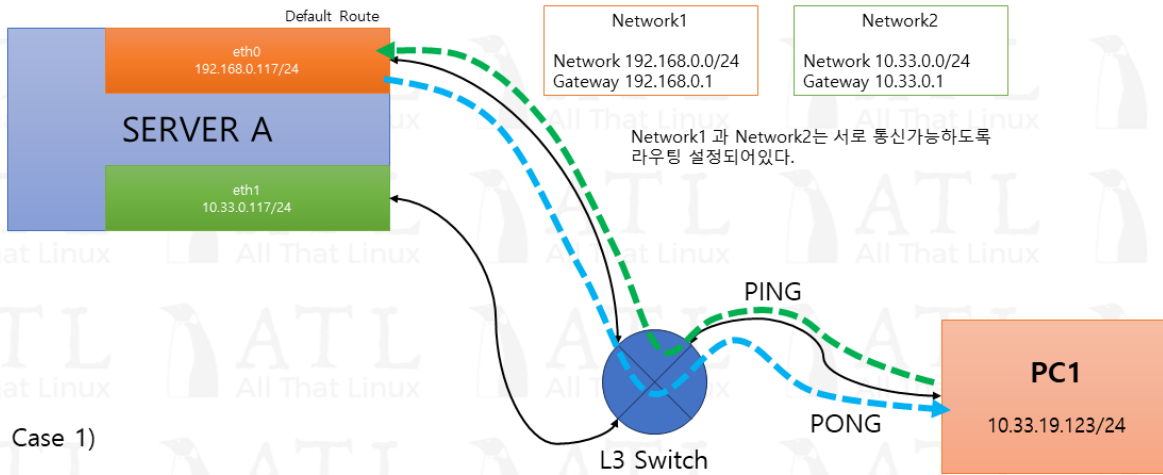


가

SERVER-A

```
[root@test-rhel7 network-scripts]# ip r
default via 192.168.0.1 dev eth0
10.33.0.0/24 dev eth1 proto kernel scope link src 10.33.0.117
169.254.0.0/16 dev eth0 scope link metric 1002
169.254.0.0/16 dev eth1 scope link metric 1003
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.117
[root@test-rhel7 network-scripts]#
```

PING



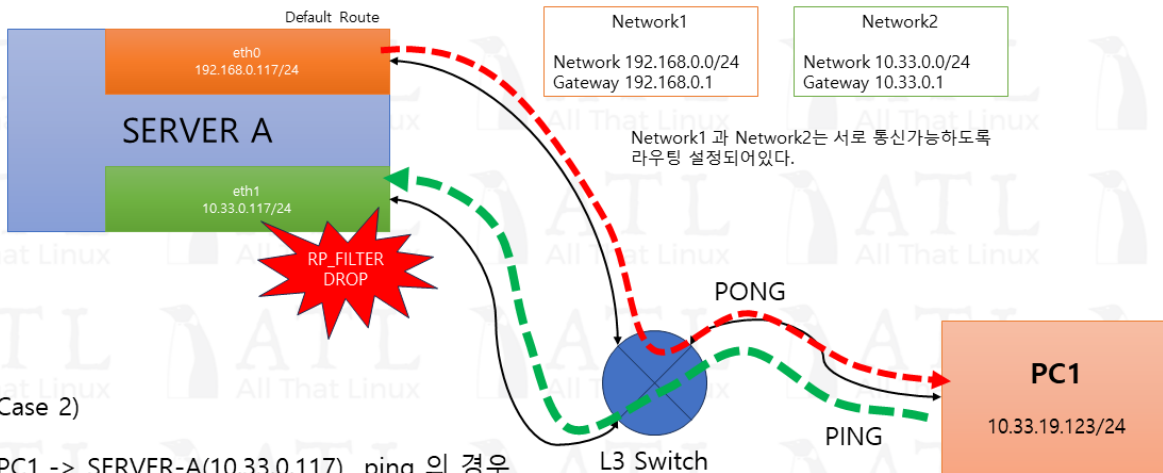
Case 1)

PC1 -> SERVER A(192.168.0.117) ping 의 경우

Default Route가 eth0 이므로 위 그림과 같이 eth0을 통해 정상적으로 ping이 동작한다.

PC1 SERVER-A default route eth0 NIC 192.168.0.117

NIC



Case 2)

PC1 -> SERVER-A(10.33.0.117) ping 의 경우

SERVER-A 에서 응답을 보내기 위한 대상 PC1의 주소가 10.33.19.123 이기 때문에 default route인 eth0 NIC로 응답을 해야만한다.

애초에 그 이전에 RP_FILTER에 의해 SRC ADDRESS가 적절한 인터페이스로 들어온것인지 체크되기 때문에 이런경우 eth1 에서 패킷이 DROP되어 버린다. 따라서 PONG 응답 패킷자체가 생성되지 않는다.

SERVER-A NIC eth1 10.33.0.117

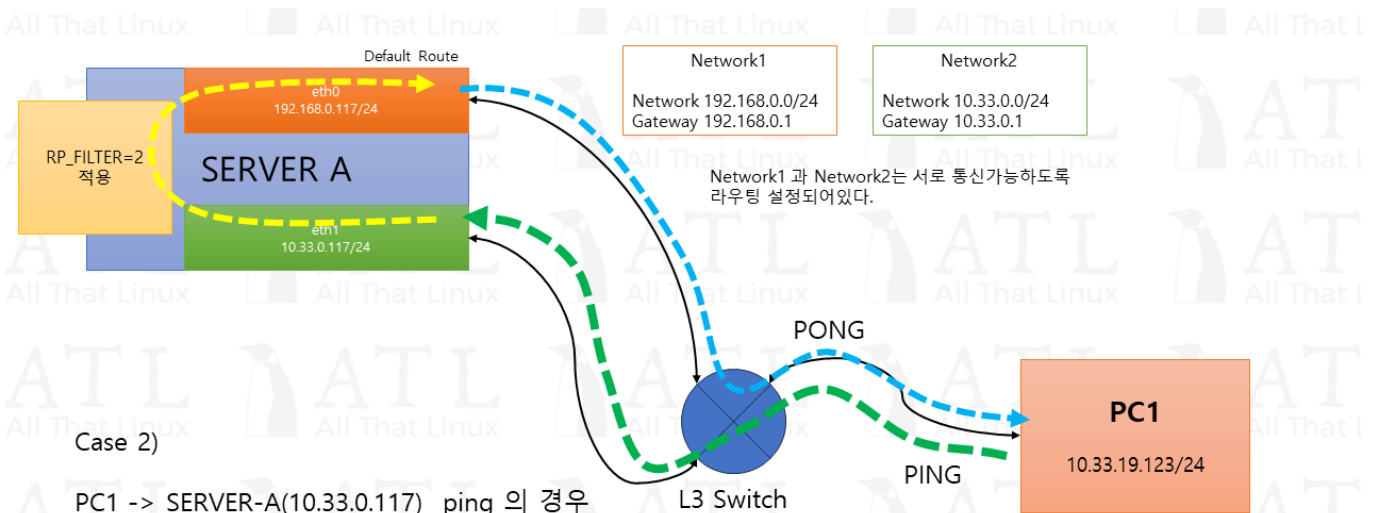
tcpdump

```

tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on lo:, link-type EN10MB (Ethernet), capture size 262144 bytes
[Interface:eth1:] 00:16:15.843703 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 50, seq 54, length 64
[Interface:eth1:] 00:16:16.867902 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 50, seq 55, length 64
[Interface:eth1:] 00:16:17.891712 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 50, seq 56, length 64
[Interface:eth1:] 00:16:18.915671 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 50, seq 57, length 64
[Interface:eth1:] 00:16:19.939879 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 50, seq 58, length 64
[Interface:eth1:] 00:16:20.963866 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 50, seq 59, length 64
[Interface:eth1:] 00:16:21.987843 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 50, seq 60, length 64

```

PING eth1 가 SERVER-A OS (RHEL 7)
 rp_filter가 RFC3704 Strict mode (=1)
 rp_filter
 rp_filter



Case 2)

PC1 -> SERVER-A(10.33.0.117) ping 의 경우

SERVER-A의 sysctl.conf 에 아래설정을 추가하면

```
net.ipv4.conf.default.rp_filter = 2
```

```
net.ipv4.conf.all.rp_filter = 2
```

엄격한 RP_FILTER규칙을 느슨하게 설정하여 응답을 적절한 인터페이스로 할 수 있게 된다.

rp_filter loose mode (=2)
 가

rp_filter /etc/sysctl.conf 가 .

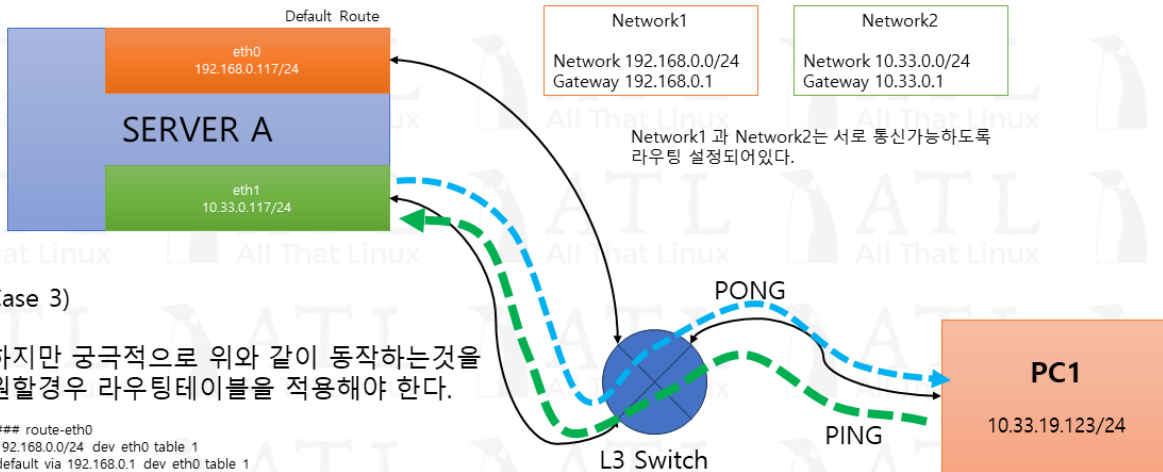
```
net.ipv4.conf.default.rp_filter = 2
net.ipv4.conf.all.rp_filter = 2
```

PING SERVER-A tcpdump

```
[Interface:eth1:] 00:35:12.483899 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 50, seq 1164, length 64
[Interface:eth0:] 00:35:13.507619 IP 10.33.0.117 > 10.33.19.123: ICMP
echo reply, id 50, seq 1165, length 64
[Interface:eth1:] 00:35:13.507587 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 50, seq 1165, length 64
[Interface:eth0:] 00:35:14.531632 IP 10.33.0.117 > 10.33.19.123: ICMP
echo reply, id 50, seq 1166, length 64
[Interface:eth1:] 00:35:14.531606 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 50, seq 1166, length 64
[Interface:eth0:] 00:35:15.555904 IP 10.33.0.117 > 10.33.19.123: ICMP
echo reply, id 50, seq 1167, length 64
[Interface:eth1:] 00:35:15.555858 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 50, seq 1167, length 64
```

PING request eth1 eth0 .

가 .



Case 3)

하지만 궁극적으로 위와 같이 동작하는것을 원할경우 라우팅테이블을 적용해야 한다.

```

### route-eth0
192.168.0.0/24 dev eth0 table 1
default via 192.168.0.1 dev eth0 table 1

### route-eth1
10.33.0.0/24 dev eth1 table 2
default via 10.33.0.1 dev eth1 table 2

### rule-eth0
iif eth0 priority 1 table 1
from 192.168.0.117 priority 1 table 1

### rule-eth1
iif eth1 priority 1 table 1
from 10.33.0.117 priority 1 table 2

```

eth1 route PC1 → SERVER-A eth0 가 eth1(10.33.0.117) 가 PC1 IP가 10.33.19.123 SERVER-A default

```

eth0 1 192.168.0.1
./etc/sysconfig/network-script/route-eth0

```

```

### /etc/sysconfig/network-script/route-eth0
192.168.0.0/24 dev eth0 table 1
default via 192.168.0.1 dev eth0 table 1

```

```

가 eth1 2 10.33.0.1
./etc/sysconfig/network-script/route-eth1

```

```

### /etc/sysconfig/network-script/route-eth1
10.33.0.0/24 dev eth1 table 2
default via 10.33.0.1 dev eth1 table 2

```

```

eth0 iif(income interface)가 eth0 1 rule
eth0 IP(192.168.0.117) 1 rule
./etc/sysconfig/network-script/rule-eth0

```

```

### /etc/sysconfig/network-script/rule-eth0
iif eth0 priority 100 table 1

```

```
from 192.168.0.117 priority 100 table 1
```

```
가 eth1 iif(income interface)가 eth1 2
rule . eth1 IP(10.33.0.117) 2 rule
```

```
/etc/sysconfig/network-script/rule-eth1
```

```
### /etc/sysconfig/network-script/rule-eth1
iif eth1 priority 100 table 1
from 10.33.0.117 priority 100 table 2
```

```
priority 가 .
```

PING

```
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1:, link-type EN10MB (Ethernet), capture size 262144 bytes
[Interface:eth1:] 00:44:19.811711 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 51, seq 22, length 64
[Interface:eth1:] 00:44:19.811739 IP 10.33.0.117 > 10.33.19.123: ICMP
echo reply, id 51, seq 22, length 64
[Interface:eth1:] 00:44:20.835779 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 51, seq 23, length 64
[Interface:eth1:] 00:44:20.835822 IP 10.33.0.117 > 10.33.19.123: ICMP
echo reply, id 51, seq 23, length 64
[Interface:eth1:] 00:44:21.859569 IP 10.33.19.123 > 10.33.0.117: ICMP
echo request, id 51, seq 24, length 64
[Interface:eth1:] 00:44:21.859601 IP 10.33.0.117 > 10.33.19.123: ICMP
echo reply, id 51, seq 24, length 64
```

```
eth1 .
```

NetworkManager

```
ifconfig NetworkManager
ifcfg-eth0 route-eth0, rule-eth0 .
```

```
NetworkManager dispatcher
script .
```

```
/etc/NetworkManager/dispatcher.d/10-multiple-gw
```



```
#!/bin/bash

INTERFACE=$1
ACTION=$2

if [ "$INTERFACE" == "eth0" ]; then
    if [ "$ACTION" == "up" ]; then
        ip route add 192.168.0.0/24 dev eth0 table 1
        ip route add default via 192.168.0.1 dev eth0 table 1
        ip rule add iif eth0 priority 100 table 1
        ip rule add from 192.168.0.117 priority 100 table 1
    fi
fi

if [ "$INTERFACE" == "eth1" ]; then
    if [ "$ACTION" == "up" ]; then
        ip route add 10.33.0.0/24 dev eth1 table 2
        ip route add default via 10.33.0.1 dev eth1 table 2
        ip rule add iif eth1 priority 100 table 2
        ip rule add from 10.33.0.117 priority 100 table 2
    fi
fi
```

(+x)

```
chmod +x /etc//etc/NetworkManager/dispatcher.d/10-multiple-gw
```

rp_filter

Reverse Path Filtering(rp_filter)

IP forwarding

RP Filtering

IP

가

<https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt>

rp_filter

rp_filter - INTEGER

0 - No source validation.

1 - Strict mode as defined in RFC3704 Strict Reverse Path

Each incoming packet is tested against the FIB and if the interface is not the best reverse path the packet check will fail.

By default failed packets are discarded.

2 - Loose mode as defined in RFC3704 Loose Reverse Path

Each incoming packet's source address is also tested against the FIB and if the source address is not reachable via any interface

the packet check will fail.

Current recommended practice in RFC3704 is to enable strict mode to prevent IP spoofing from DDos attacks. If using asymmetric routing or other complicated routing, then loose mode is recommended.

The max value from conf/{all,interface}/rp_filter is used when doing source validation on the {interface}.

Default value is 0. Note that some distributions enable it in startup scripts.

- 0
- RHEL 6 <http://tools.ietf.org/html/rfc3704>
1
- Strict mode() _____ IP 가
_____ 가 IP
_____ 가
RHEL 7+ IPReversePathFilter 가 가 . 가
- **IP**
IP
- RHEL 5 0() 1() 가 가 . 1()

```
[root@localhost ~]# sysctl -w "net.ipv4.conf.default.rp_filter=2"
[root@localhost ~]# sysctl -w "net.ipv4.conf.all.rp_filter=2"
```

/etc/sysctl.conf 가 .

- <https://access.redhat.com/solutions/53031>
- <https://access.redhat.com/solutions/30564>
- <https://access.redhat.com/solutions/19596>
- <https://access.redhat.com/solutions/288823>
- <http://capsuleer.tistory.com/97>
- <https://access.redhat.com/solutions/30564>

- http://jensd.be/468/linux/two-network-cards-rp_filter
- https://zetawiki.com/wiki/%EB%A6%AC%EB%88%85%EC%8A%A4_%EC%8A%A4%ED%83%9C%ED%8B%B1_%EB%9D%BC%EC%9A%B0%ED%8C%85_%EC%84%A4%EC%A0%95

From:

<https://atl.kr/dokuwiki/> - **AllThatLinux!**

Permanent link:

https://atl.kr/dokuwiki/doku.php/multiple_gateway_%EC%84%A4%EC%A0%95?rev=1713508773

Last update: **2024/04/19 06:39**

