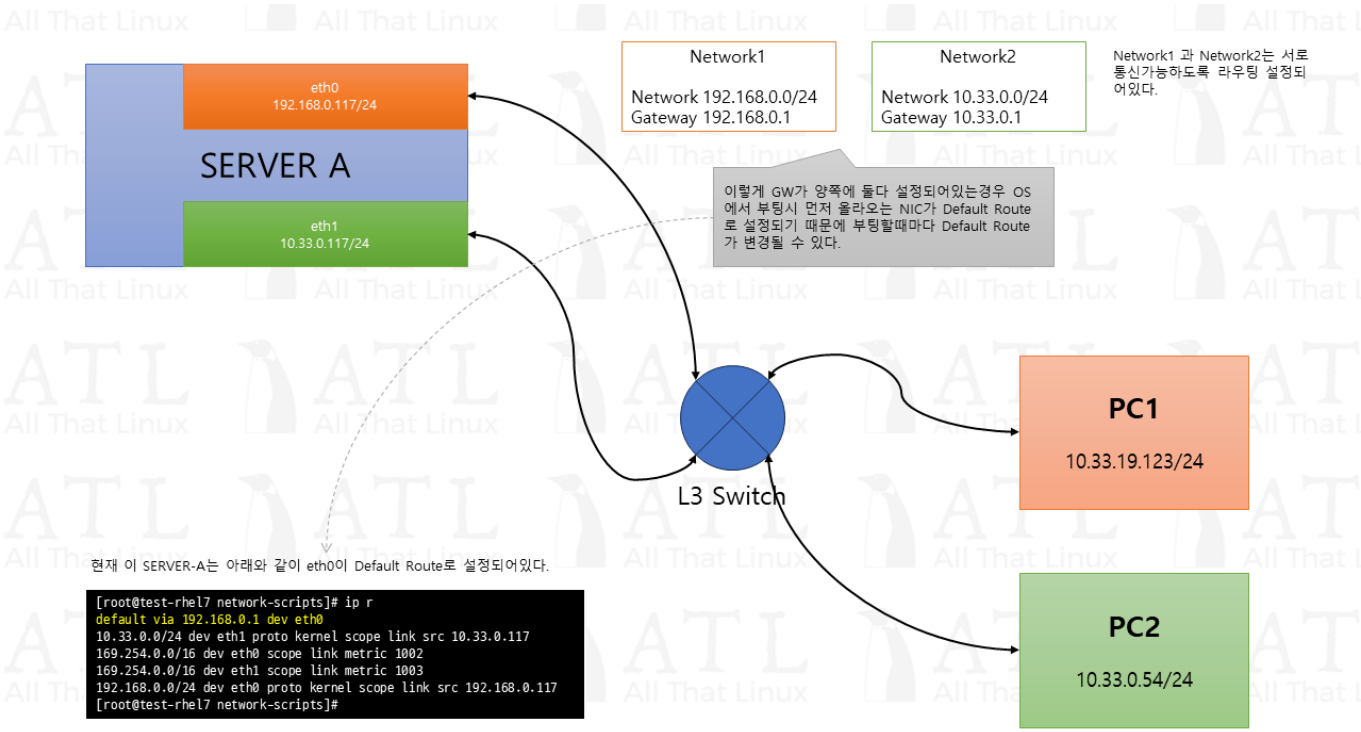
목차

Multiple Gateway 다중 게이트웨이 설정

— *이강우 2024/04/19 04:09*

두개의 NIC, 두개의 Default Gateway 설정방법

기본 환경

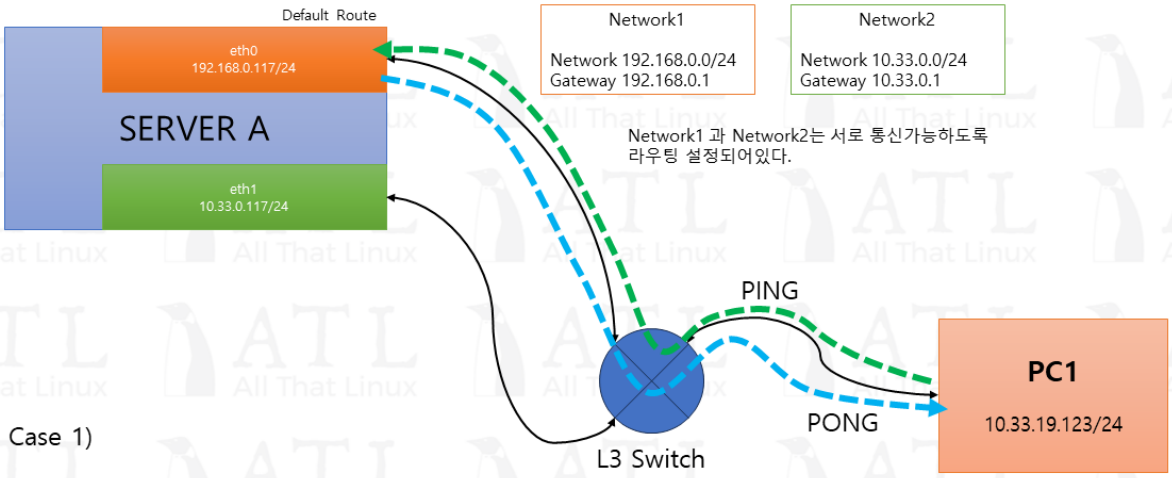


위 그림과 같이 기본 환경이 구성되어있다고 가정한다.

이때 SERVER-A의 라우팅 테이블 정보를 확인하면 아래와 같다.

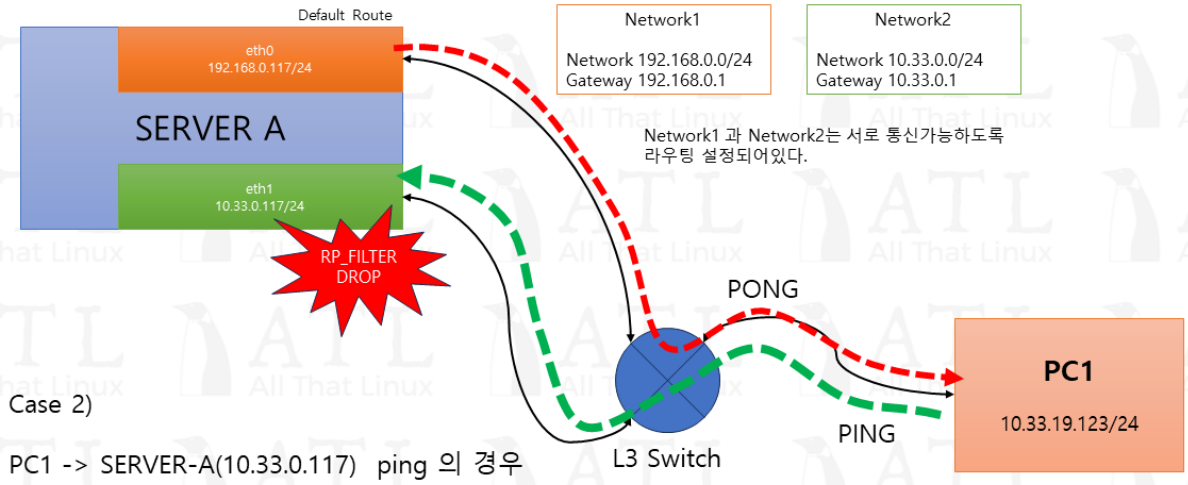
```
[root@test-rhel7 network-scripts]# ip r
default via 192.168.0.1 dev eth0
10.33.0.0/24 dev eth1 proto kernel scope link src 10.33.0.117
169.254.0.0/16 dev eth0 scope link metric 1002
169.254.0.0/16 dev eth1 scope link metric 1003
192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.117
[root@test-rhel7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

기본 라우팅 경로로 PING



PC1 에서 SERVER-A 의 `default route` 대역인 `eth0` NIC의 `192.168.0.117`로 핑을 시도하면 당연하게 정상적으로 핑 통신이 되는것을 확인할 수 있다.

두번째 NIC로 통신



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.

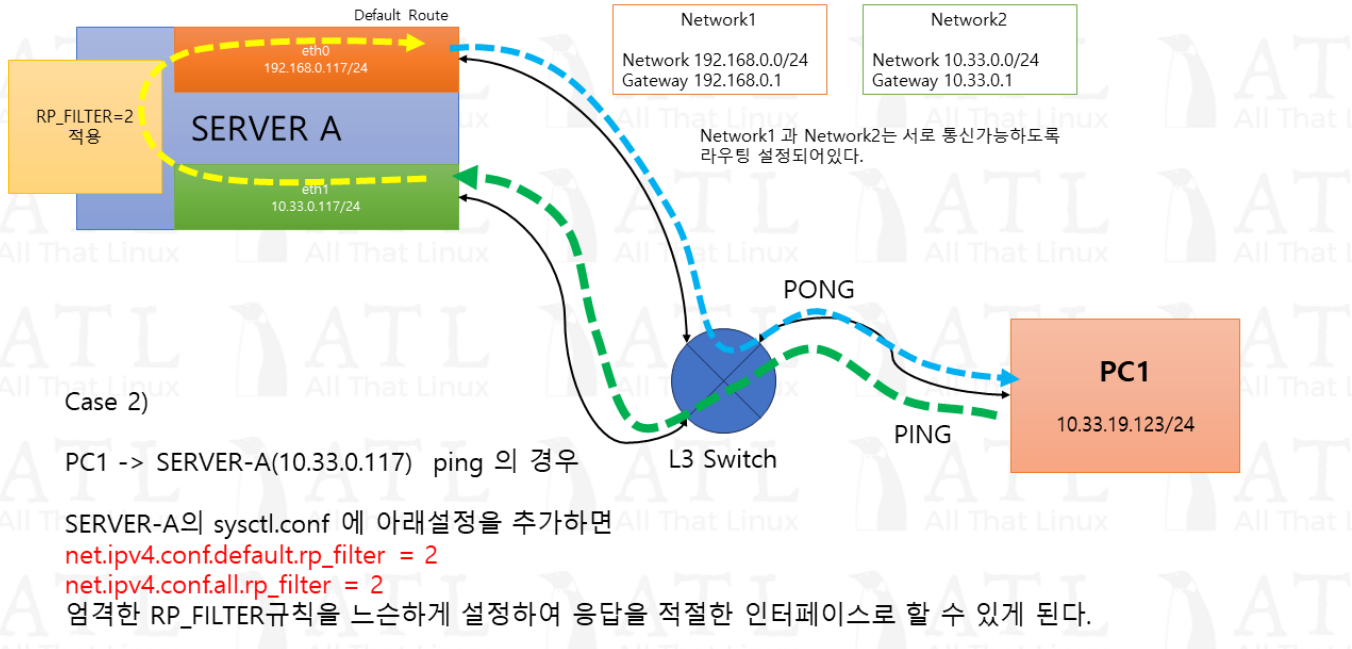
동일한 환경에서 SERVER-A의 두번째 NIC인 `eth1`의 `10.33.0.117`로 핑을 시도하면 핑이 되지 않는것을 확인할 수 있다.

이때 SERVER-A에서 `tcpdump`를 살펴보면 아래와 같다.

```
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
```

확인된 바와 같이 PING 패킷이 eth1으로 들어오지만 나가는것은 없다. 이것은 SERVER-A의 OS에서 응답자체를 안한다는 것인데 왜 그런가 하면 위의 그림에 설명된 바와 같이 요즘 OS(RHEL 7 이상)는 기본적으로 rp_filter가 RFC3704에 정의된 대로 Strict mode (=1)로 설정되어있기 때문이다.
rp_filter에 대한 설명은 아래를 참고한다.

이 상황을 해결하려면 아래처럼 rp_filter값을 설정하면 된다.



시스템의 rp_filter값을 loose mode (=2)로 설정하게 되면 패킷의 경로를 엄격하게 검사하지 않기 때문에 위 그림과 같이 통신이 가능하게 된다.

rp_filter설정 방법은 /etc/sysctl.conf에 아래 내용을 추가한다.

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

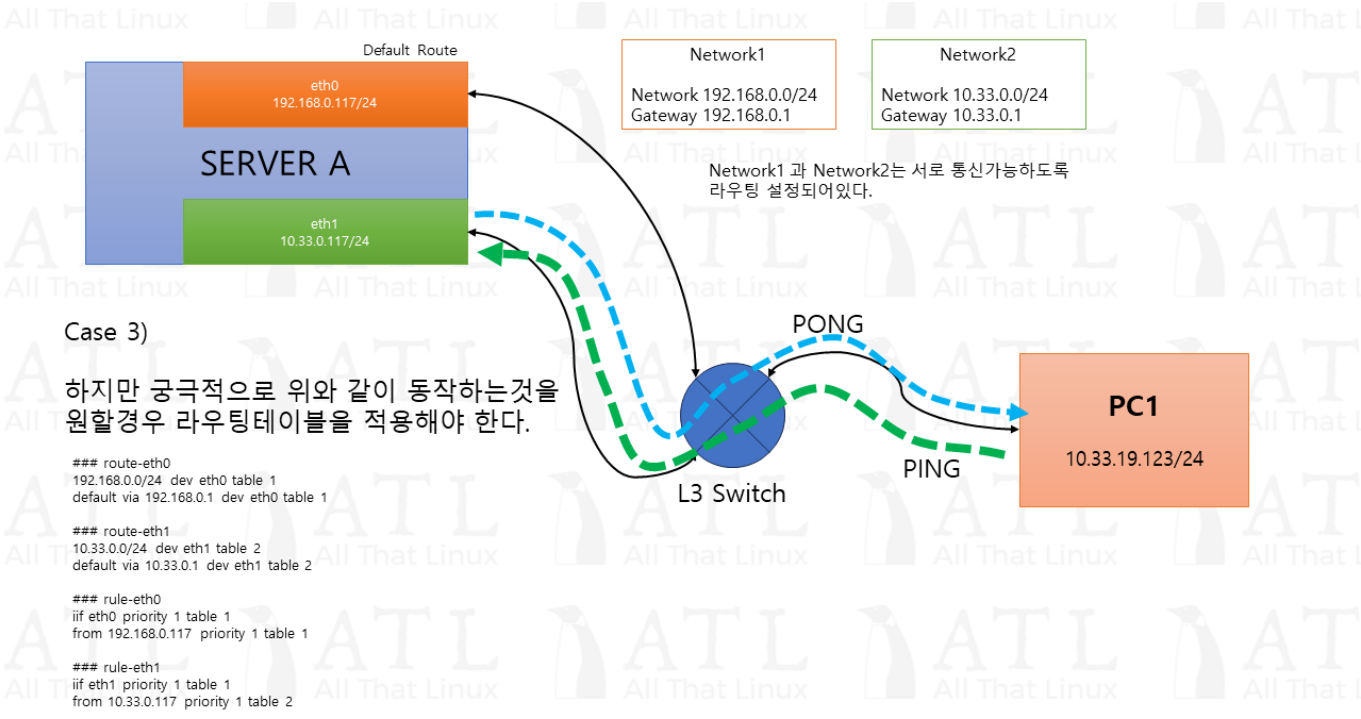
이렇게 하면 PING통신이 동작하는데 SERVER-A에서 tcpdump로 패킷을 확인해보면

```
[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
```

위와 같이 PING request는 eth1으로 들어오지만 응답은 eth0으로 하는것을 확인할 수 있다.

라우팅 테이블 등록

하지만 궁극적으로는 패킷을 받은 인터페이스로 응답을 하길 원하는경우가 있다.



위 그림처럼 PC1 → SERVER-A의 eth1(10.33.0.117)으로 PING을 시도할때 응답도 SERVER-A의 eth1을 통해서 응답이 오길 원하는것이다. 하지만 PC1의 IP가 10.33.19.123이므로 응답시 default route 인터페이스인 eth0을 통해 나가는데 들어온 인터페이스를 통해 응답하려면 아래처럼 별도의 라우팅 테이블 설정을 추가 하여야 한다.

먼저 각각의 인터페이스에 대해 라우팅 테이블을 따로 생성한다.

eth0의 라우팅 테이블 1번을 정의하고 해당 인터페이스의 기본 게이트웨이를 192.168.0.1로 설정한다. /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
```

마찬가지로 eth1의 라우팅 테이블 2번을 정의하고 해당 인터페이스의 기본 게이트웨이를 10.33.0.1로 설정한다. /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
```

eth0인터페이스에 대해 iif(income interface)가 eth0인 경우는 라우팅 테이블 1번의 rule을 따르도록 한다. 또는 eth0의 IP(192.168.0.117)에서 받는 요청은 라우팅 테이블 1번 rule을 따르도록 한다. /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
```

마찬가지로 eth1인터페이스에 대해 iif(income interface)가 eth1인 경우는 라우팅 테이블 2번의 rule을 따르도록 한다. 또는 eth1의 IP(10.33.0.117)에서 받는 요청은 라우팅 테이블 2번 rule을 따르도록 한다.

/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 값은 위 상황에선 생략 가능하다.

이렇게 설정하고 PING을 날려보면

```
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
```

위와같이 동일한 인터페이스 eth1으로 응답하는것을 확인할 수 있다.

NetworkManager 사용시

전통적인 ifconfig 파일이 아닌 NetworkManager를 사용하는경우는 설정 방법이 다르다. 기존의 ifcfg-eth0파일이나 route-eth0, rule-eth0과 같은 파일을 사용할 수 없다.

NetworkManager는 자체적으로 별도의 라우팅테이블을 지정하는 기능이 없으므로 dispatcher script를 이용하여야 한다.

위와 동일한 구성을 하려면 아래 파일을 생성하면 된다.

/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
```

그리고 **해당파일에 반드시 실행권한(+x)을 부여**해줘야 한다. 각 스크립트는 root가 소유한 **일반 실행 파일**이어야 합니다. 또한 그룹이나 다른 사람이 쓸 수 없어야 하며 setuid가 아니어야 합니다. 참조링크 : https://networkmanager.dev/docs/api/latest/NetworkManager-dispatcher.html

한방에 하려면 아래 chmod명령어 참고

```
chmod +x,g-w,o-w,-s,-t /etc/NetworkManager/dispatcher.d/10-multiple-gw
```

rp_filter

Reverse Path Filtering(rp_filter) 는 리눅스 커널이 라우팅 테이블을 참조하여 수신된 IP 패킷을 외부로 전달하는 기능을 하는 IP forwarding과 달리 RP Filtering은 **패킷이 들어오는 네트워크 인터페이스와 라우팅 테이블에 등록되어 있는 출발지 주소가 일치하지 않는 경우 자동으로 들어오는 패킷을 거절**하는 역할을 합니다

https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt 의 설명에 의하면 rp_filter의 역할은 다음과 같다.

```
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.
```

- 커널 공식 문서에는 이 값이 기본적으로 0으로 설정되어 있다고 하지만 리눅스 배포판마다 기본값을 다르게 설정하고 있습니다.
- RHEL 6 이상은 기본적으로 http://tools.ietf.org/html/rfc3704에 권장되는 엄격한 역방향 전달 필터링을 적용하도록 구성됩니다. 기본값 1

- Strict mode(엄격한 필터링)는 패킷이 시스템에 도착하면 커널이 패킷의 소스 IP를 가져와 라우팅 테이블을 조회하여 패킷이 도착한 인터페이스가 커널이 해당 IP에 패킷을 보내는 데 사용하는 것과 동일한 인터페이스인지 확인합니다. 인터페이스가 동일하면 패킷이 엄격한 필터링 테스트를 통과하고 정상적으로 처리됩니다. **인터페이스가 동일하지 않으면 패킷은 추가 처리 없이 폐기**되고 RHEL 7+에서는 IPReversePathFilter카운터가 증가됩니다.
- 엄격한 필터링의 주요 효과는 **지정된 원격 IP의 경우 시스템이 특정 인터페이스를 통해서만 통신한다**는 것입니다. 지정된 원격 IP 또는 네트워크에 응답하는 인터페이스를 제어하도록 정적 경로를 설정합니다.
- RHEL 5 이전버전에서 커널은 엄격한 필터링을 지원하지 않았습니다. 따라서 이러한 이전 릴리스에서 매개 변수에는 0(비활성화) 및 1(느슨한) 두 가지 가능한 값만 있습니다. 기본값은 1(느슨한)입니다.

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

영구적으로 적용하려면 /etc/sysctl.conf에 추가하도록 한다.

관련링크

- https://access.redhat.com/solutions/53031
- https://access.redhat.com/solutions/30564

참조링크

- 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