2. GRE over IPsec+보안 설정

2016년 지방 1과제

GRE over ipsec과 보안 설정 항목에 대한 풀이집

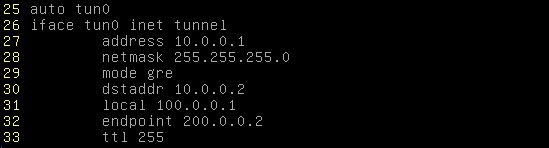
2016

이번엔 GRE over IPsec과 보안 설정을 진행해봅니다.

먼저 IPsec으로 암호화 하기 전에 SUN-R와 SUN-srv3에  
GRE인터페이스를 생성해보도록 하겠습니다.

<SUN-R>

root@SUN-R:~# **vi /etc/network/interfaces**

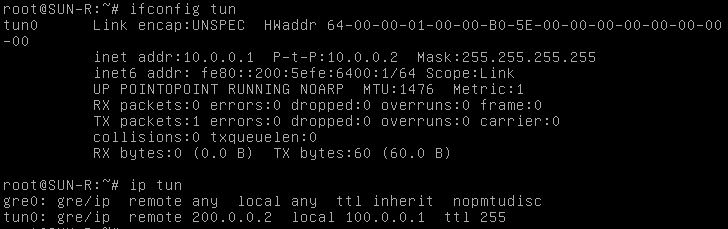


여기서 작성 요령은 아래와 같습니다.

auto tun0  
iface tun0 inet tunnel  
 address 터널IP  
 netmask 터널서브넷  
 mode gre  
 dstaddr 목적지 터널IP  
 local 터널을 연결할 실제 IP  
 endpoint 터널이 도착할 실제 목적지 IP  
 ttl 255

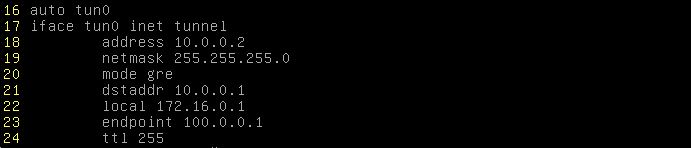
root@SUN-R:~# **systemctl restart networking**

인터페이스가 잘 생성되었는지 아래와 같이 확인합니다.



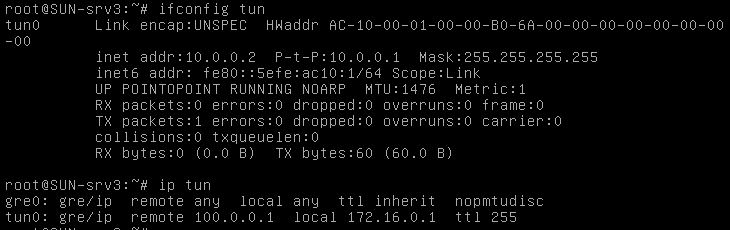
<SUN-srv3>

root@SUN-srv3:~# **vi /etc/network/interfaces**



root@SUN-srv3:~# **systemctl restart networking**

이번에도 잘 적용되었는지 확인해봅니다.



하지만 이와 같이 구성하면 GRE 터널링이 맺어지지 않습니다.

그 이유는 SUN-R는 Branch-R를 목적지로 두고 있기 때문입니다.

따라서 GRE 패킷을 SUN-srv3로 보내기 위해 Branch-R에  
아래를 참고하여 NAT를 작성합니다.

<Branch-R>

먼저 GRE 패킷을 넘겨주기 위해 모듈을 실행합니다.

root@Branch-R:~# **modprobe nf\_conntrack\_proto\_gre**

머신이 다시 시작될 때에도 이 모듈을 활성화하기 위해 모듈을 추가합니다.

root@Branch-R:~# **echo ‘nf\_conntrack\_proto\_gre’ >> /etc/modules**

이제 패킷을 NAT 합니다.

root@Branch-R:~# **vi /etc/rc.local**

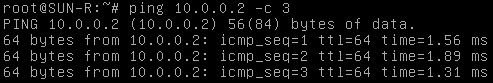


root@Branch-R:~# **/etc/rc.local**

터널링 구성에 성공했는지 통신을 보내어 테스트합니다.

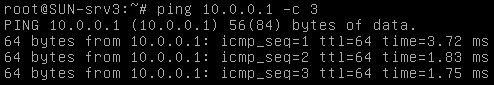
<SUN-R>

root@SUN-R:~# **ping 10.0.0.2 -c 3**



<SUN-srv3>

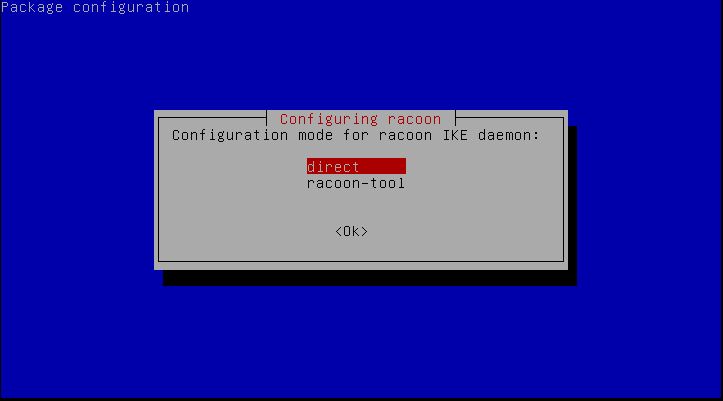
root@SUN-srv3:~# **ping 10.0.0.1 -c 3**



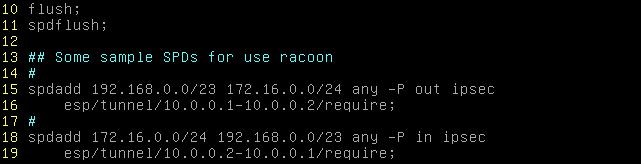
위와 같이 터널링 구성이 완료되면 IPsec를 위해 racoon 패키지를 설치합니다.

<SUN-R>

root@SUN-R:~# **apt-get install racoon -y**



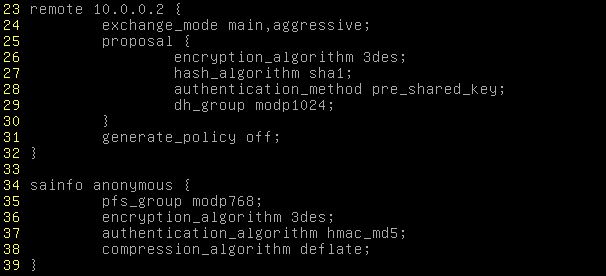
root@SUN-R:~# **vi /etc/ipsec-tools.conf**



위를 작성 시 주의 사항은 아래와 같습니다.

spdadd 다음은 각각 ‘출발지 대역’과 ‘목적지 대역’이며, any는 프로토콜을 의미합니다.  
또, in은 패킷이 들어올 때, out은 패킷이 나갈 때를 의미합니다.  
tunnel 뒤에 들어올 구문은 ‘연결할 출발지 IP’와 ‘연결할 목적지 IP’입니다.  
여기서 연결할 출발지와 목적지는 IPsec으로 묶을 구간을 의미합니다.

root@SUN-R:~# **vi /etc/racoon/racoon.conf**



위에서 remote는 위에서 작성했던 ‘연결할 목적지 IP’를 적으면 됩니다.

Sainfo 옆은 anonymous로 바꿔줍니다.

설정이 완료되었으면 교환할 키를 설정합니다.

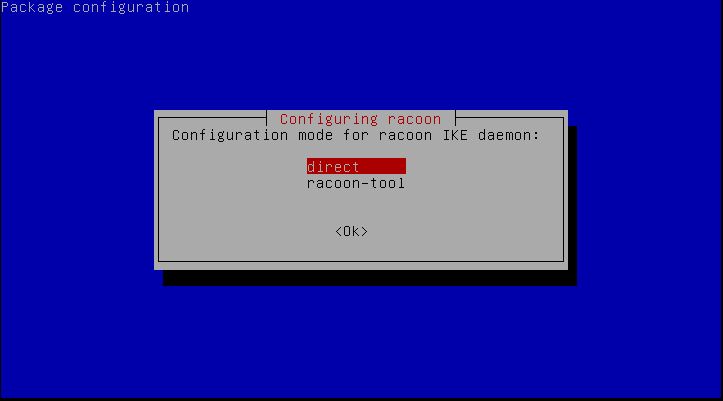
여기서 사용할 키는 정해져 있지 않기 때문에 ‘sun2016##’으로 해야 합니다.

root@SUN-R:~# **echo ’10.0.0.2 sun2016##’ > /etc/racoon/psk.txt**

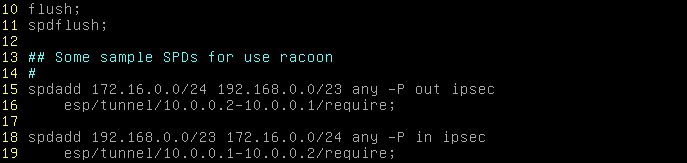
설정이 완료되면 패키지를 재 시작하지 말고 SUN-srv3를 설정합니다.

<SUN-srv3>

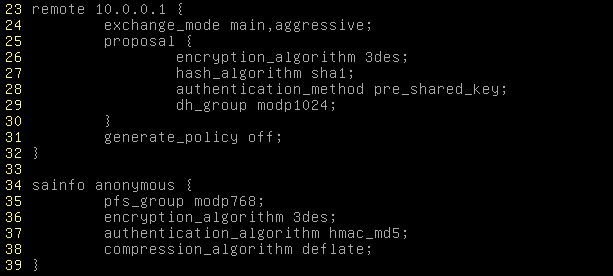
root@SUN-srv3:~# **apt-get install racoon -y**



root@SUN-srv3:~# **vi /etc/ipsec-tools.conf**



root@SUN-srv3:~# **vi /etc/racoon/racoon.conf**



root@SUN-srv3:~# **echo ’10.0.0.1 sun2016##’ > /etc/racoon/psk.txt**

두 머신의 패키지를 재 시작합니다.

<SUN-R>

root@SUN-R:~# **systemctl restart racoon.service setkey.service**

<SUN-srv3>

root@SUN-srv3:~# **systemctl restart racoon.service setkey.service**

이제 sun.com 도메인 클라이언트들이 서로 통신이 가능하도록  
하기 위해 라우팅을 진행합니다.

<SUN-R>

root@SUN-R:~# **vi /etc/rc.local**



root@SUN-R:~# **/etc/rc.local**

<SUN-srv3>

root@SUN-srv3:~# **vi /etc/rc.local**



root@SUN-srv3:~# **/etc/rc.local**

여기서 src 옵션을 붙이는 이유는 이 VPN에 SUN-srv3도 이용할 수 있어야  
하는데 위에서 맺어준 IPsec은 172.16.0.0/24 대역이기 때문에  
위와 같이 출발지를 지정해 준 것입니다.

이제 IPsec이 잘 맺어졌는지 확인하기 위해 tcpdump 패키지를 설치합니다.

<SUN-R>

root@SUN-R:~# **apt-get install tcpdump -y**

root@SUN-R:~# **tcpdump**

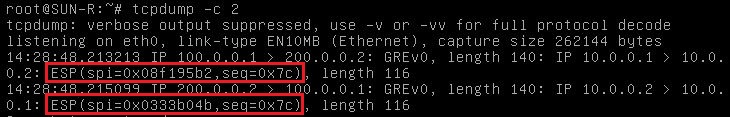
위 처럼 tcpdump를 실행한 후 SUN-srv2에서 SUN-srv3로 핑을 테스트합니다.

<SUN-srv2>

root@SUN-srv2:~# **ping 172.16.0.1 -c 1**



<SUN-R>

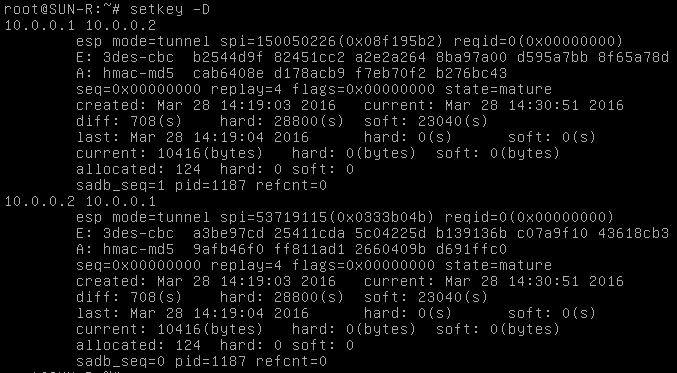


이와 같이 패킷이 정상적으로 암호화 되는 것을 확인할 수 있습니다.

이번엔 SUN-R에서 아래 명령어로 교환한 키를 확인해봅니다.

<SUN-R>

root@SUN-R:~# **setkey -D**



이번에는 SUN-srv3에서 핑 테스트를 진행하면서 SUN-R에서 패킷을 확인합니다.

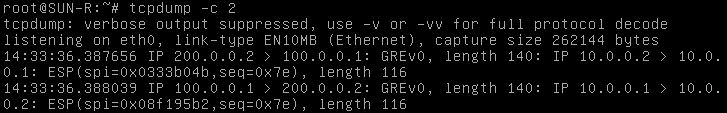
<SUN-srv3>

root@SUN-srv3:~# **ping 192.168.0.1 -c 1**



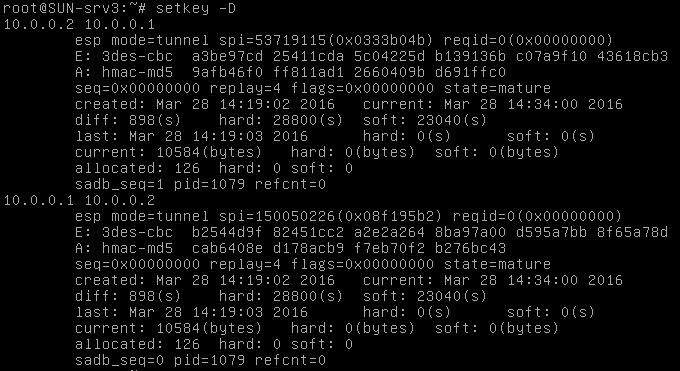
<SUN-R>

root@SUN-R:~# **tcpdump**



정상적으로 암호화가 되는 것을 확인할 수 있습니다.

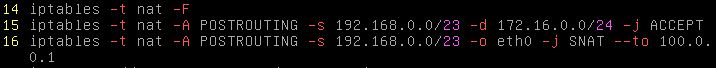
또, SUN-srv3에서 IPsec으로 교환한 키를 보면 정상임을 확인할 수 있습니다.



이제 내부 클라이언트들이 외부와 통신할 수 있도록 NAT 설정을 합니다.

<SUN-R>

root@SUN-R:~# **vi /etc/rc.local**



위 NAT는 아래와 같이 작동합니다.

이번에는 Branch-R에서 NAT와 mangle를 진행합니다.

여기서 mangle은 패킷의 TTL이나 전송 우선순위 값을 변경할 때 사용합니다.  
과제에서 HTTP 패킷을 외부로 보내라고 요구한 내용을 해결하기 위해 사용합니다.

iptables -t nat -F

* iptables의 nat 테이블을 모두 초기화 합니다.

iptables -t nat -A POSTROUTING -s 192.168.0.0/23 -d 172.16.0.0/24 -j ACCEPT

* 나가는 패킷 중 출발지가 192.168.0.0/23 대역이며, 목적지가 172.16.0.0/24 대역인 패킷은 NAT 하지 않고 그냥 통과시킵니다.

iptables -t nat -A POSTROUTING -s 192.168.0.0/23 -o eth0 -j SNAT –to 100.0.0.1

* 나가는 패킷 중 출발지가 192.168.0.0/23 대역이며, eth0 인터페이스를 통해 나갈 때 100.0.0.1 아이피로 NAT해서 나갑니다.

<Branch-R>

root@Branch-R:~# **echo ‘100 http’ >> /etc/iproute2/rt\_tables**

* Mangle을 사용하기 위해 테이블을 하나 생성합니다.(우선순위는 100을 사용했습니다.)

root@Branch-R:~# **vi /etc/rc.local**



* 여기서 16번 줄은 기존에 만든 것으로 무시합니다.

위 설정한 내용은 아래와 같습니다.

iptables -t nat -F

* 현재 설정된 nat 테이블을 모두 초기화합니다.

iptables -t mangle -F

* 현재 설정된 mangle 테이블을 모두 초기화합니다.

iptables -t mangle -A PREROUTING -p tcp --dport 80 -j ACCEPT

* 들어오는 패킷 중 tcp 프로토콜 도착지 포트 80인 패킷을 통과시킵니다.

iptables -t mangle -A PREROUTING -m iprange --src-range 172.16.0.101-172.16.0.121 -j MARK --set-mark 1

* 들어오는 패킷 중 172.16.0.101~172.16.0.121 사이의 패킷을 mark 1로 지정합니다..

Iptables -t nat -A PREROUTING -d 192.168.0.0/23 -p tcp --dport 80 -j DNAT --to 100.0.0.1

* 들어오는 패킷 중 192.168.0.0/23으로 가는 tcp 프로토콜 목적지 포트 80인 패킷을 100.0.0.1로 NAT합니다.

iptables -t nat -A POSTROUTING -s 172.16.0.0/24 -o eth0 -j SNAT –to 200.0.0.2

* 나가는 패킷 중 출발지가 172.16.0.0/24 대역이며 eth0으로 나가는 패킷을 200.0.0.2로 NAT합니다.

ip rule add fwmark 1 table http

* MARK 1의 IP 규칙을 테이블 http에 적용합니다.

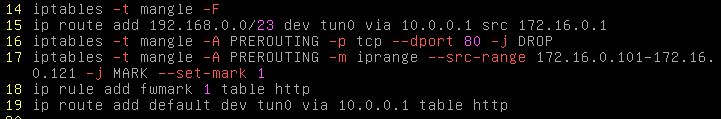
ip route add default dev eth1 via 172.16.0.1 table http

* 정적 라우팅을 생성합니다. 이때, eth1의 172.16.0.1 아이피로 패킷을 전달하며, 테이블 http에 설정된 규칙에 맞는 패킷만 전달합니다.

<SUN-srv3>

root@SUN-srv3:~# **echo ‘100 http’ >> /etc/iproute2/rt\_tables**

root@SUN-srv3:~# **vi /etc/rc.local**



* 여기서 15번 줄은 기존에 만든 것으로 무시합니다.

위 설정 내용은 아래와 같습니다.

위와 같이 설정한 후 SUN-manager에 임의로 172.16.0.101 아이피를 줍니다.

iptables -t mangle -F

* 현재 설정된 mangle 테이블을 모두 초기화합니다.

iptables -t mangle -A PREROUTING -p tcp –dport 80 -j DROP

* 들어오는 패킷 중 tcp 프로토콜 목적지 포트가 80인 패킷을 차단합니다.

iptables -t mangle -A PREROUTING -m iprange --src-range 172.16.0.101-172.16.0.121 -j MARK --set-mark 1

* 들어오는 패킷 중 172.16.0.101~172.16.0.121 사이의 패킷을 mark 1로 지정합니다..

ip rule add fwmark 1 table http

* IP 규칙을 테이블 http에 적용합니다.

ip route add default dev tun0 via 10.0.0.1 table http

* 정적 라우팅을 생성합니다. 이때, tun0의 10.0.0.1 아이피로 패킷을 전달하며, 테이블 http에 설정된 규칙에 맞는 패킷만 전달합니다.

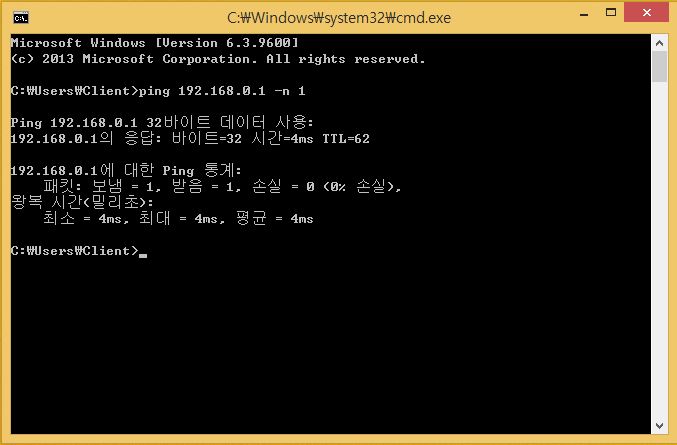
그 후 192.168.0.1로 통신을 시도해보며, SUN-R에서 tcpdump로  
캡슐화가 되는지 확인해봅니다.

또, 원활한 패킷 전달을 위해 포워드를 열어줍니다.

root@SUN-srv3:~# **echo ‘net.ipv4.ip\_forward=1’ >> /etc/sysctl.conf**

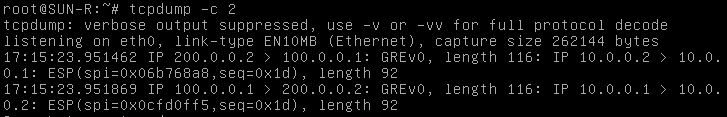
root@SUN-srv3:~# **sysctl -p**

<SUN-manager>



<SUN-R>

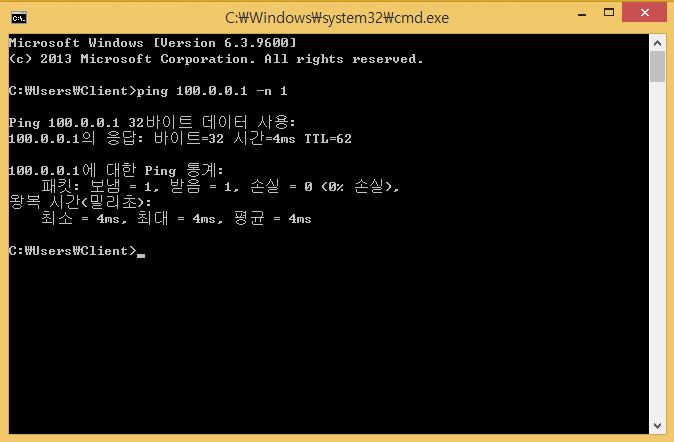
root@SUN-R:~# **tcpdump**



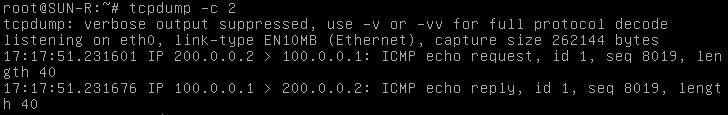
이번에는 SUN-srv3를 종료한 후 SUN-Client3에서 100.0.0.1로 통신합니다.

그리고 그 상황을 SUN-R에서 tcpdump로 패킷을 확인합니다.

<SUN-Client3>



<SUN-R>



패킷이 캡슐화 되지 않음이 확인되어야 합니다.

마지막으로 SUN-R에 172.16.0.0/24(SUN-manager를 위한 NAT) 대역에 대한 NAT를 생성하여 SUN-manager에서 외부로의 통신이 원활하도록 설정합니다.

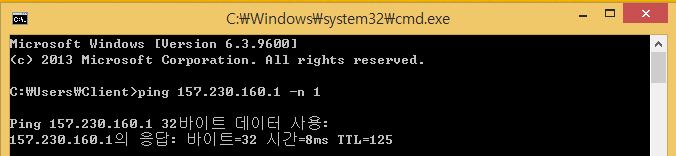
<SUN-R>



위와 같이 추가하여 SUN-manager의 핑 패킷이 외부로 NAT될 수 있도록 설정합니다.

SUN-manager가 외부와 정상적으로 통신이 되는지 테스트해봅니다.

<SUN-manager>



또한, SUN-srv1과 SUN-srv2에서 SUN-manager와 통신이 되어야 합니다.

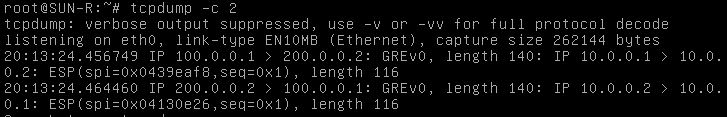
<SUN-srv2>

root@SUN-srv2:~# **ping 172.16.0.101**



<SUN-R>

root@SUN-R:~# **tcpdump**



또, 추가적으로 SUN-R 머신을 종료하면 SUN-manager에서  
외부로의 통신이 불가능해야 합니다.