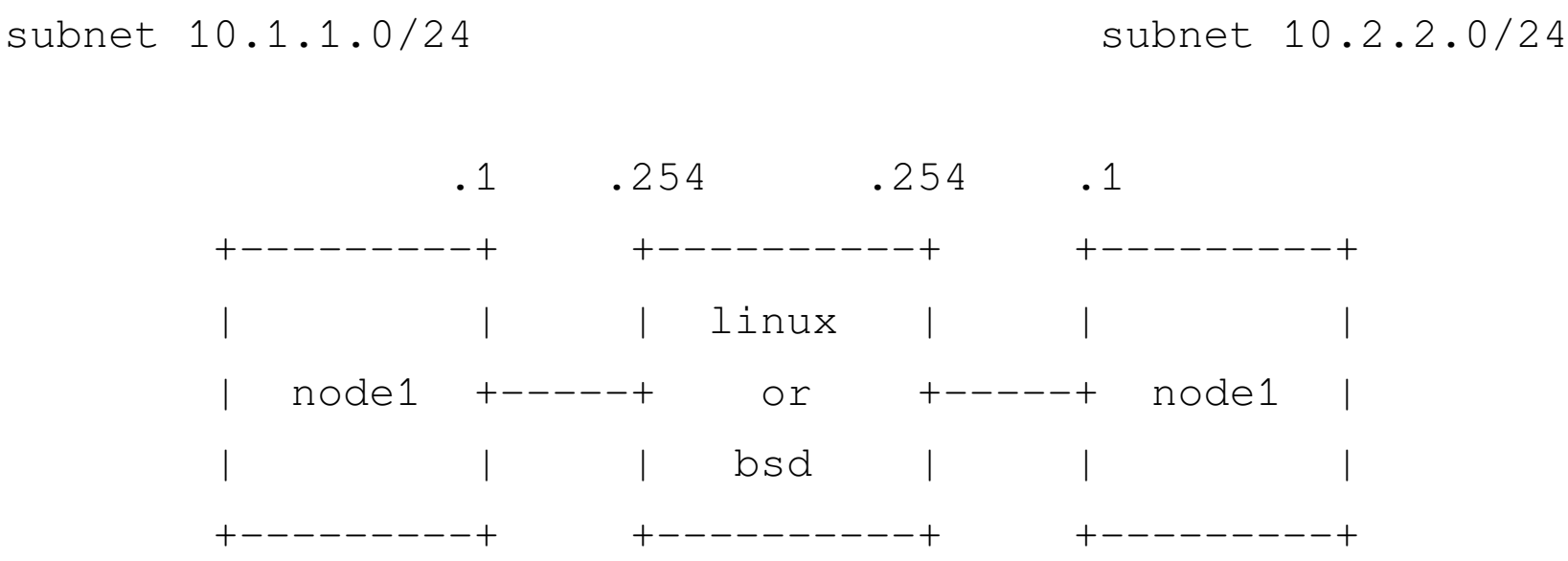
Happy-Happy L3

Networking

Pierre-Philipp Braun <pbraun@nethence.com>

Table of contents

MWE Routing



How to turn a UNIX system into a router?...

==> enable IP forwarding

GNU/Linux

```
#sysctl -w net.ipv4.ip_forward=1
echo 1 > /proc/sys/net/ipv4/ip_forward

#echo net.ipv4.ip_forward=1 >> /etc/sysctl.conf
#sysctl -p
```

BSD

```
sysctl -w net.inet.ip.forwarding=1
sysctl net.inet.ip.forwarding

echo net.inet.ip.forwarding=1 >> /etc/sysctl.conf
```

What to do next for the two subnets to talk to each other?…

==> enable static (or default) routes

configure the nodes to use the box/router as

- static route
- –or– default route

Note: both nodes need to be tweaked – otherwise the there would be no return path for e.g. an ICMP reply

Note: that works only from the next hop (not through the public network)

What's the most common scenario for a public network gw?…

==> NAT

Translating source or destination

- SNAT – outbound
  - traffic coming from internal subnet is translated to front-facing IP
  - not supposed to be reachable
- DNAT – inbound (port-forwarding)
  - traffic coming to front-facing IP gets translated to internal subnet
  - reachable by design

Do we absolutely need to enable firewalling for NAT to work?…

==> technically speaking, no

- Forwarding + SNAT is enough
- …and it is *almost* ok, as long as the gateway itself is clean
- …meaning it is not listening on any port on the front-facing interface

==> but sometimes, yes

- In case you need a firewall anyways to handle the internal network
- The gateway is ideal place to do it

And if you really need to enable Firewalling…

DO NOT *FULLY* DISABLE ICMP – IT IS USEFUL

```
==> /var/log/debug <==
Jan 16 06:30:17 slack9 dhcpd: ICMP Echo reply while lease
    10.1.1.145 valid.

==> /var/log/syslog <==
Jan 16 06:30:17 slack9 dhcpd: Abandoning IP address
    10.1.1.145: pinged before offer
```

Netfilter with IPTABLES

Second, *SNAT* on a static and front-facing IP

```
iptables -t nat -A POSTROUTING -o FACING-NIC -s INTERNAL-CIDR
    -j SNAT --to-source FACING-IP
```

–or– on a changing and front-facing IP

```
iptables -t nat -A POSTROUTING -o FACING-NIC -s INTERNAL/CIDR
    -j MASQUERADE
```

check

```
iptables -L -v -n -t nat
```

Netfilter with NFTABLES

with a STATIC IP

```
vi /etc/nftables.conf

...
    #SNAT
    chain postrouting {
            type nat hook postrouting priority 100;
            oifname eth0 masquerade
    }
```

with a DYNAMIC IP

```
...
    #SNAT
        chain my_masquerade {
           type nat hook postrouting priority srcnat; policy accept;
                oifname ”ppp0” masquerade
        }

systemctl reload nftables
```

NetBSD Packet Filter (NPF)

```
vi /etc/npf.conf

group default {
        pass in all
        pass out all
}

#SNAT
map xennet0 dynamic 10.1.1.0/24 -> 188.130.155.62

/etc/rc.d/npf reload
```

Besides, NPF is not vulnerable to NAT pivoting.

Now consider your home router, and let's say you want to do some peer-to-peer.

What do you need to enable here and what is it called?…

==> DNAT aka PORT-FORWARDING

DNAT with IPTABLES

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j DNAT --to-destination INTERNAL-IP
```

Note eventually against another port with `INTERNAL-IP:PORT`

DNAT with NFTABLES

```
vi /etc/nftables.conf

...
    #DNAT
    chain prerouting {
            type nat hook prerouting priority -100;
            iifname eth0 tcp dport 80 dnat x.x.x.x
    }

systemctl reload nftables
```

DNAT with NPF

```
vi /etc/npf.conf

map xennet0 dynamic proto tcp 10.1.1.x port xxxxx <-
188.130.155.62 port xxxxx

/etc/rc.d/npf reload
```

eBPF

LAB // dig into eBPF and PoC

// Questions on mwe routing?

Firewalling

Without firewall

- all interfaces,
- all subnets,
- and all tcp & udp ports can talk to each other

Possibly

- routed traffic
- across VLANs

Control inbound/outbound

L3 *aka* **ACLs**

- network interface
- src IP address ranges
- dst IP address ranges

L4 *aka* firewall

- tcp & udp port ranges

Default policy

```
 BLOCK IN ON <FACING-INTERFACE> ALL
(BLOCK OUT ON <FACING-INTERFACE> ALL)

PASS IN ON <FACING-INTERFACE> TCP PORT 80
PASS IN ON <FACING-INTERFACE> TCP PORT 443
...
```

No need to maintain tousands of blocking rules

Security by segmentation

Firewalls required only on routers/gateways

- smaller/dedicated and possibly open VLANs
- only filter between VLANs

No system firewall required

- as long as you do `netstat -ltup` frequently enough
- _even on the public network
- *but then SSHGuard may still be required*

WAN can break

(picture of twitter online-fr status)

BSDs are cool

TEST-NET-X… see links

DHCP

an extension to BOOTP

Boostrap Protocol (BOOTP)

the ancestor of DHCP

Used boot network-boot UNIX and early Windows stations. Also for diskless systems.

client `<-->` server

```
--> bcast BOOTP request
<-- BOOTP answer with ip/mask/gw (+ next-server?)
```

BOOTP relay

L2/l3 bcast don't pass through routers. Here's a solution

- router can listen on `udp/67` for client bcasts
- and relay those to the known bootp server
- forward the replay to the bootp client

DHCP products

Servers

- DHCP (ISC – like for BIND)
- dnsmasq
- udhcpd (busybox)
- *there's not a lot of choice*[1]

Clients

- dhclient (ISC – comes with DHCP)
- dhcpcd

[1] Comparison of DHCP server software, <https://en.wikipedia.org/wiki/Comparison_of_DHCP_server_software>

DHCP session layout

the theory…

client <--> server

```
--> DHCP Discover
    who's there?

<-- DHCP Offer
    I am here

--> DHCP Request
    gimme gimme

<-- DHCP ACK
    you got it
```

DHCP client discover

the practice…

Client remembers last IP address used, and eventually asks for it already

```
client bcast l2+l3 udp 68->67
provides host name
requests for specific parms e.g.
    mask / router / dns
    (host name)
    netbios name server
    netbios scope
    interface mtu
    ntp servers
```

DHCP server offer

the practice…

Server proposes an IP already

```
server dst client-ip already udp 67->68
client ip
next-server ip
lease time
mask / router / dns
```

Most of the informations are provided as options but the IP address and `next-server`

BONUS QUESTION // bootp relay works for DHCP?

DHCP client request

Client accepts ***the first*** offer (possibly among multiple DHCP servers)

BONUS QUESTION // not sure why client is still broadcasting

```
client bcast l2+l3 udp 68->67
```

and similar to DHCP discover but

```
confirms DHCP Server Identifier (IP address)
```

and asks (again) for proposed address

DHCP server ack

Server re-confirms everything just like in the original offer. there can be differences, though, e.g.

`host name`

(not offered but acknowledged)

Should multiple DHCP servers live on the same LAN?…

==> Not really, but it's possible:

- as long as deliver same subnet and use different range (split scope)
- the 80/20% range ratio for primary vs backup server was considered a good practice as you wouldn't have to serve more than 20% of the leases during a primary server outage

What happens then? How does the client choose?…

==> Client does NOT choose, it just takes the first answer it sees

Be ready to setup a rogue server!

DHCP clustering

- easy active/active using split-scope
- active/passive using traditional HA software? (the wrong way)
- active/passive using built-in feature! (the right way)[2]

LAB // active/passive DHCP buit-in fault-tolerance

LAB // active/active DHCP fault-tolerance? Not just split scope – maybe with VRRP/CARP?

[2] A Basic Guide to Configuring DHCP Failover, <https://kb.isc.org/docs/aa-00502>

DHCP security

What would an attacker want on this front?…

==>

- Rogue DHCP and get to **become the gateway** for MITM attacks
  - heavy traffic involved, need to hold-on
- Rogue DHCP and get to **become the DNS server**
  - doable for public records
  - how to handle internal records?

LAB // practice with a rogue DHCP PoC to get in the middle

LAB // target a precise victim and avoid getting the load of the whole network

LAB // how to prevent the genuine offer and ack to arrive and take precedence?

Mitigations?…

==> DHCP snooping

Multi-layer switch to drop rogue packets

- by authorized switch port to offer leases
- by DHCP server IP address
- further track MAC-IP bindings
- sanitize ARP
- mix with accounting

Alternative mitigation

One could also let it happen and simply IDS/sniff it but it's hard to find the *physical* location of the offender

DHCP & DNS

Use DynDNS updates to keep track of hostname leases and records

LAB // mixup DHCP and DNS?

DHCP vs. DHCPv6

Those are IPv4 only

How to deal with IPv6?…

==> just enable SLAAC if you got an RA router

==> DHCPv6 client with a DUID

One last thing about IP4

Allow 0.0.0.0/8 as a valid address range (kernel.org)
https://news.ycombinator.com/item?id=20430096

Allow 0.0.0.0/8 as a valid address range https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=96125bf9985a

Linux kernel to allow 0.0.0.0/8 as a valid address range
https://www.reddit.com/r/networking/comments/cd1957/linux_kernel_to_allow_00008_as_a_valid_address/

SLAAC

All you need is your kernel to listen for RAs…

Kernel settings

```
accept_ra
Accept Router Advertisements; autoconfigure using them.

accept_ra_pinfo
Learn Prefix Information in Router Advertisement.

autoconf
Autoconfigure addresses using Prefix Information in Router Advertiseme
```

Unsatisfying default

Accept Router Advertisements; autoconfigure using them *(default is 1)*

```
0 -- Do not accept Router Advertisements
1 -- Accept Router Advertisements if forwarding
    is disabled
2 -- Overrule forwarding behaviour. Accept Router
    Advertisements even if forwarding is enabled
```

Learn Prefix Information in Router Advertisement *(enabled)*

Autoconfigure addresses using Prefix Information in Router Advertisements *(enabled)*

Enforce SLAAC over forwarding

```
echo 2 >
/proc/sys/net/ipv6/conf/xenbr0/accept_ra

cat /proc/sys/net/ipv6/conf/interface/accept_ra_pinfo

cat /proc/sys/net/ipv6/conf/xenbr0/autoconf
```

DHCPv6 client

enable SLAAC + static IP at boot-time

```
/sbin/ifconfig xenbr0 inet6 add x::x/56 up

vi /etc/dhclient6.conf

interface ”xenbr0” {
        #/48
        send dhcp6.client-id xx:xx:xx:xx:xx:xx:xx:xx:xx:xx;
}

/sbin/dhclient -cf /etc/dhclient6.conf
# -6 -P xenbr0 -v
```

Operations

```
#netstat -rn --inet6 | grep ^::/0
```

```
ip -6 neigh show
```

```
ping6 ...
```

nota: ping6 comes with inetutils-ping while iputils-ping is PTR-capable

watch ICMPv6 router advertisments

```
tcpdump -vvvv -ttt -i xenbr0 icmp6
    and 'ip6[40] = 134'
```

Become an RA router

install RADVD http://www.litech.org/radvd/

enable forwarding on ALL INTERFACES

```
echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
```

setup the daemon to spit regularly on the internal network

```
vi /etc/radvd.conf

interface br0
{
    AdvSendAdvert on;
    AdvLinkMTU 1280;
    MaxRtrAdvInterval 300;
    prefix 2001:bc8:204a:101::/64
    {
        AdvOnLink on;
        AdvAutonomous on;
    };
};
```

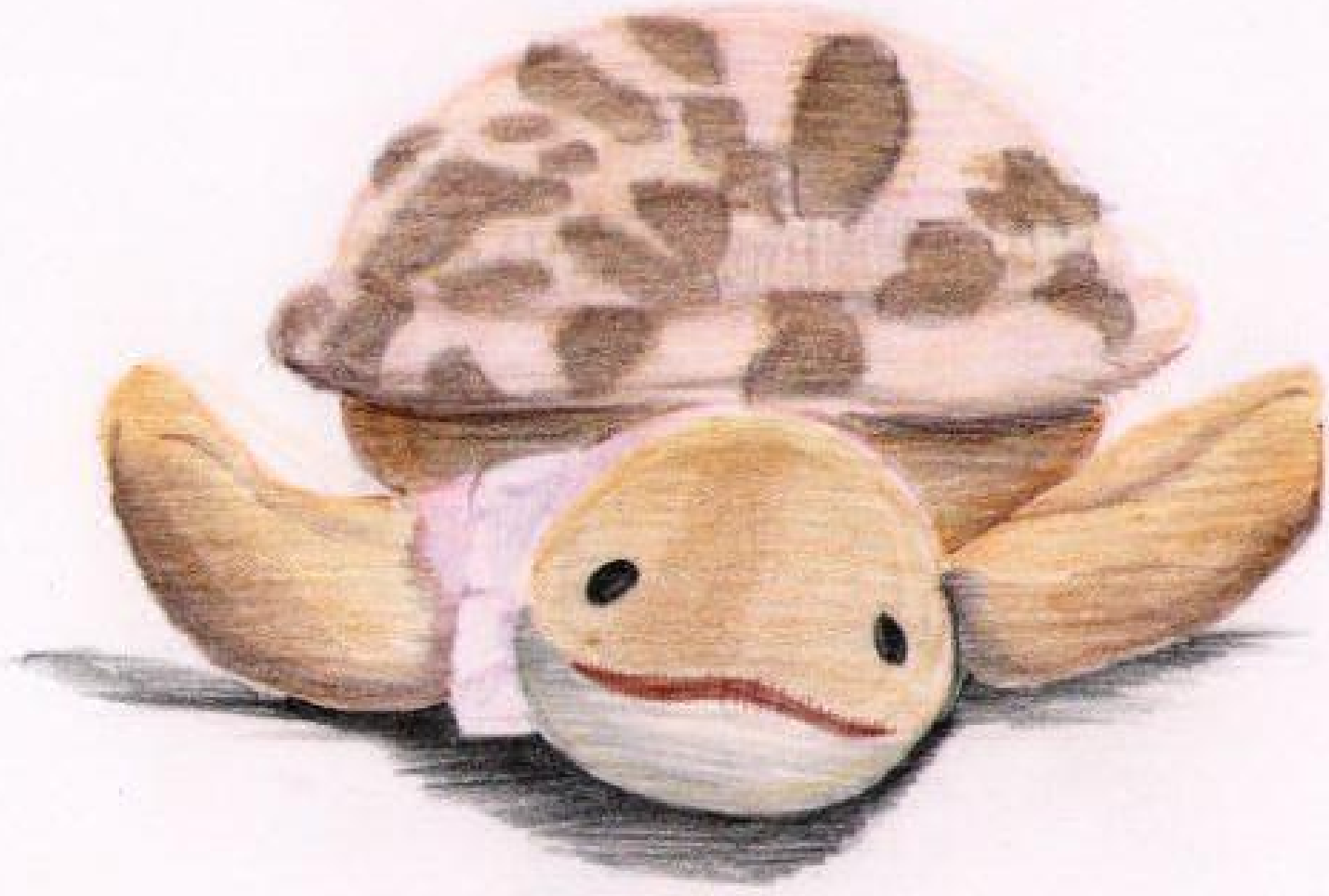
Run the daemon

```
/usr/local/sbin/radvd
```

Questions on DHCP and IPv6 address allocation?

Check on `www.kame.net` if you're IPv6

KAME not dancing



かめ かめ

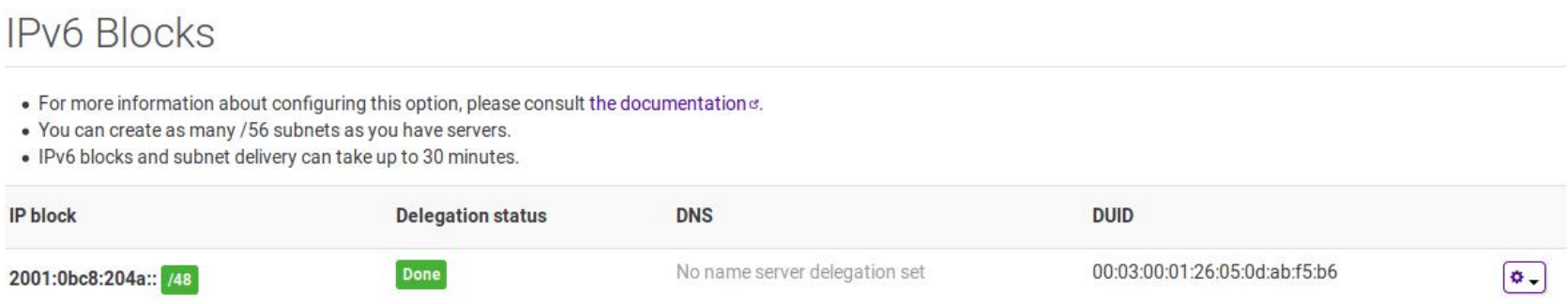
- …but *some* ISPs do provide it
- e.g. got a `/48` for free @Online/Scaleway

What if the server is IPv6-only? How to reach it if you are only IPv4?

==> proto-41 (6in4) tunnels e.g. `ipv6.ip4market.ru`

List of IPv6 tunnel brokers
https://en.wikipedia.org/wiki/List_of_IPv6_tunnel_brokers

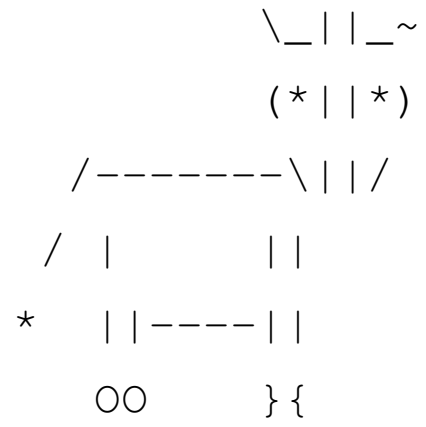
By the way…



Why would they want to delegate DNS?

==> Only for PTRs (reverse DNS)

IPSEC



Revision 2 (2020/21)

Pierre-Philipp Braun <pbraun@nethence.com>

The KAME project (Japan, 1998-2005)

- bring IPv6 and IPSEC to BSDs
- used onto freebsd netbsd dflybsd
- openbsd took only ipv6 and do their own ipsec
- linux took only ipsec & ipsec-tools (racoon)

NetBSD

- IPSEC kernel-enabled by default
- Make sure NAT works already
- IPSEC does not interfere
  - Tested in conjunction w/ ipfilter/ipnat
  - Tested in conjunction w/ NPF

Network Architecture

Fix `/etc/hosts` on both gateways (and two stations for better acceptance testing)

Nethence office (NAT)

```
212.83.171.255 ipsec.nethence.com ipsec
62.210.0.1     ipsecgw
10.6.6.254     ipsechb
10.6.6.253     ssdhb
```

OS3 office (NAT)

```
188.130.155.57 nbsd.os3.su nbsd
188.130.155.33 nbsdgw
10.1.1.252     nbsdhb
10.1.1.253     slack9
```

ipsec.nethence.com

```
vi /etc/ipsec.conf

add 212.83.171.255 188.130.155.57 esp 13245
    -E blowfish-cbc ”blowfishtest.001” ;
add 188.130.155.57 212.83.171.255 esp 13246
    -E blowfish-cbc 0xdeadbeefdeadbeefdeadbeefdeadbeef;
spdadd 10.6.6.0/24 10.1.1.0/24 any -P out ipsec
    esp/tunnel/212.83.171.255-188.130.155.57/require;
spdadd 10.1.1.0/24 10.6.6.0/24 any -P in ipsec
    esp/tunnel/188.130.155.57-212.83.171.255/require;
```

nbsd.os3.su

```
vi /etc/ipsec.conf

add 212.83.171.255 188.130.155.59 esp 13245
    -E blowfish-cbc ”blowfishtest.001” ;
add 188.130.155.59 212.83.171.255 esp 13246
    -E blowfish-cbc 0xdeadbeefdeadbeefdeadbeefdeadbeef;
spdadd 10.6.6.0/24 10.7.7.0/24 any -P in ipsec
    esp/tunnel/212.83.171.255-188.130.155.59/require;
spdadd 10.7.7.0/24 10.6.6.0/24 any -P out ipsec
    esp/tunnel/188.130.155.59-212.83.171.255/require;
```

Did anybody notice the difference?…

ipsec.nethence.com

```
cat > /etc/ipsec.conf.ipsec <<-EOF
...
EOF
chmod 400 /etc/ipsec*
ln -sf ipsec.conf.ipsec /etc/ipsec.conf
```

nbsd.os3.su

```
cat > /etc/ipsec.conf.ipsec <<-EOF
...
EOF
sed -r '3s/-P out/-P in/; 4s/-P in/-P out/' \
    /etc/ipsec.conf.ipsec > /etc/ipsec.conf.nbsd
chmod 400 /etc/ipsec*
ln -sf ipsec.conf.nbsd /etc/ipsec.conf
```

Apply

```
setkey -f /etc/ipsec.conf
```

And check (SAD vs SAS+SPD entries)

```
setkey -D
setkey -DP
```

Deliver to the other network

On both IPSEC/NAT gateways

```
sysctl -w net.inet.ip.forwarding=1
```

Reach the other network

ipsec.nethence.com

```
vi /etc/ifconfig.xennet1

inet 10.6.6.254/24 up
!/sbin/route -n add -net 10.1.1.0 -netmask 255.255.255.0 10.6.6.254
```

nbsd.os3.su

```
vi /etc/ifconfig.xennet1

inet 10.1.1.252/24 up
!/sbin/route -n add -net 10.6.6.0 -netmask 255.255.255.0 10.1.1.252
```

Do we need to provide the route for the internal nodes as well?

==> No

Nethence office already has `10.6.6.1` as default route

OS3 office already has `10.1.1.254` as default route

But that was just a PoC - so what if that was not the case?…

==>

on ssd (gnu/linux)

```
route add -net 10.1.1.0/24 gw 10.6.6.254
```

on slack9 (gnu/linux)

```
route add -net 10.6.6.0/24 gw 10.1.1.252
```

Acceptance Testing

From the Nethence office

```
tcpdump -n -vvv -i xennet0 host nbsd or host nbsdgw
ping -c1 nbsdgw
ping -c1 nbsd
ping -c1 nbsdhb
ping -c1 slack9
```

Which streams are ciphered?...

==>

- `nbsdgw` and `nbsd` are public (not ciphered)
- `nbsdhb` and `slack9` are internal (can see ESP packets)

Same resp. from the OS3 office

```
tcpdump -n -vvv -i xennet0 host ipsec or host ipsecgw
ping -c1 ipsecgw
ping -c1 ipsec
ping -c1 ipsechb
ping -c1 ssdhb
```

(Incl. from a node within the network)

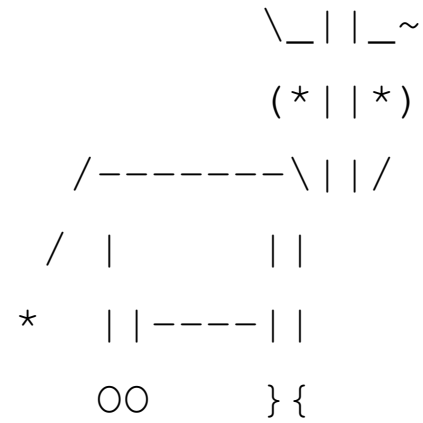
Deal with the keys… (Internet Key Exchange)

- IKEv1 (ipsec-tools/Racoon)
- IKEv2 (Racoon2, OpenBSD iked, …)
- DH vulnerable (Logjam)

==> **DIY** e.g. with SSH scripts

- within the pipe –or–
- outside the pipe as a failover

(SSHD hardening will be covered in the next lecture)



// Questions on IPSEC?