Roadwarrioring back to your mothership

Mobility with IPsec and proxy arp

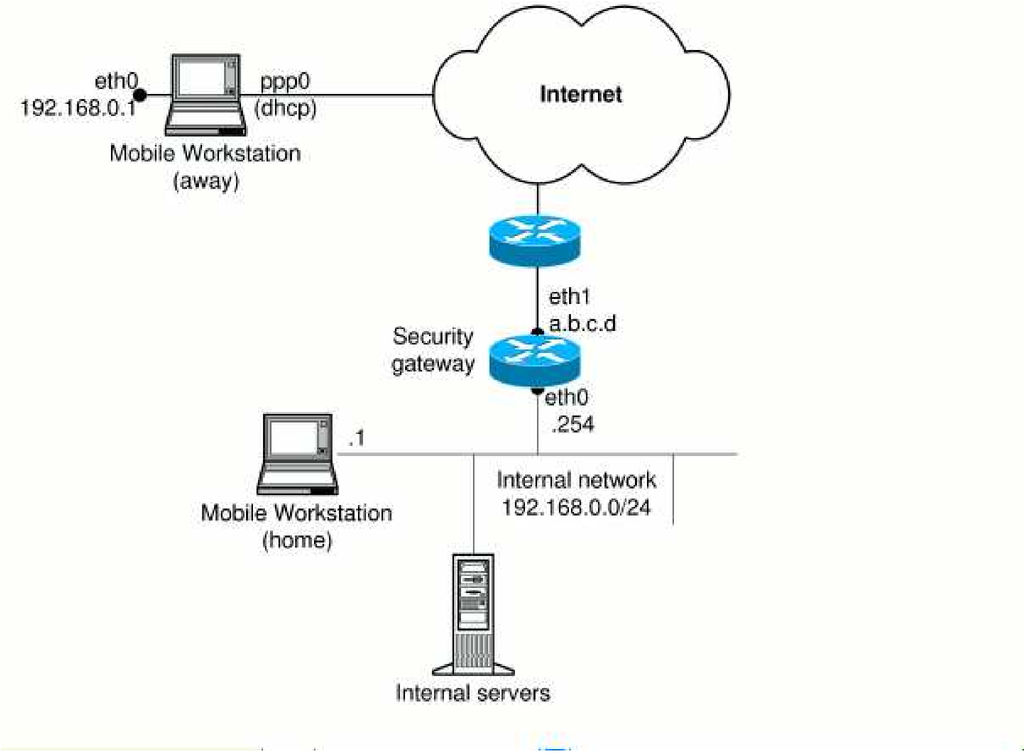
Abstract

When using a laptop at work, it is often cumbersome to use internal servers while away. Crossing the network gatewa might be hard, impossible, or involve painful and costly solutions. Also, these solutions never let you use your usual int IP address, and you then have to handle permissions problems and other access-lists burdens.
The setup presented here, derived from[3], lets you keep you private IP while connecting to your usual network. The remote connection is done via dialup/ppp and dynamic IP. But this setup can be applied to other connection types (DSI ISDN, WiFi, ...) without any modifications . The hosts are both Linux based, but at least at the security gateway, any IPsec enabled system (with X509 certificates) supporting proxy arp should do the trick. On the mobile station side, you need a station that can mangle source IP address for outbound packets.

Goal

Let's say you're usually sitting in the 192.168.0.0/24 network with your laptop (192.168.0.1).
Your default gateway (which we'll call the "security gateway") has internal IP 192.168.0.254, and external IP a.b.c.d.

When you're away from your network, you want to keep using your private IP (192.168.0.1) and appear to host in the 192.168.0.0/24 network like you're still connected on the LAN.
This situation is illustrated in the diagram below.



Requirements

Achieving this goal is quite straightforward but requires several elements to be set up. In the text below, prompts are l to show on which machine the commands are run (ca# for the certificate authority, gw# for the security gateway and ws# for the workstation).

Security gateway

Required first is an IPsec enabled security gateway. This gateway must be IPsec enabled, and should be able to do pro for our mobile station.
In this example, we'll use a Linux 2.6.3 kernel with the native IPsec stack and KAME tools[5] (a Fedora Core 2 test1[7] with 2.6.3 does a good job). You can reuse the existing gateway if you wish, or add a specific gateway between your in LAN and the internet. Note however that the security gateway needs to have a fixed public IP on the internet side.

Mobile workstation

The mobile workstation has to be IPsec enabled. We'll use FreeS/WAN[1] with X.509 patch (stock SuSE 9.0 FreeS/WAN). It also needs a working connection to the internet when away from the internal network. We'll assume ppp with a dyna IP, but adapting the setup to other methods should be straightforward. Setting up this connection is not covered here.

Certificate "server"

Also needed is a host to generate certificates, needed on the gateway and the mobile workstation. This can be any mac with openssl on it. This machine also needs to be secure, since your "security gateway" integrity will depend on the capability of the certificate server to keep the CA data safe.

Setup

Certificates

To get usable certificates for the security gateway and the workstation, you need to have a Certificate Authority (CA) s their certificates requests. If paying for it is not an option, you can create your own local CA and sign certificate reque yourself.

Certificate Authority

Thanks to CA.pl included with openssl distribution, setting up a CA is really easy. But take care about the host you'll el as the CA machine. Your IPsec setup (and everything based on certificates generated on this machine) will only be as secure as the CA host. If someone compromises your CA, he might generate certificates for his purpose and bad things could happen. With USB keys being widely available and inexpensive, it is recommended to use an isolated (not networ host and distribute keys via USB keys.

```
ca# /usr/share/ssl/misc/CA.pl -newca
CA certificate filename (or enter to create)

Making CA certificate ...
Generating a 1024 bit RSA private key
..........................++++++
.........................++++++
writing new private key to './demoCA/private/cakey.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:Rhone
Locality Name (eg, city) []:City
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Organization
Organizational Unit Name (eg, section) []:Erasme
Common Name (eg, YOUR name) []:ca.domain.tld
Email Address []:ca@domain.tld
ca#
```

Creating a Certificate Authority

This creates a "./demoCA" directory where the CA certificate (./demoCA/cacert.pem) and the CA key (./demoCA/private/cakey.pem) lies.
Now that the CA is set up, on to the client certificates.

gateway and workstation certificates

The process is the same for both machines, so only the workstation certificate generation will be showed here.
First, generate a certificate request for the workstation as below, substituting your own data.

```
ca# /usr/share/ssl/misc/CA.pl -newreq
Generating a 1024 bit RSA private key
.......................................................++++++
...................................++++++
writing new private key to 'newreq.pem'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:Rhone
Locality Name (eg, city) []:City
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Organization
Organizational Unit Name (eg, section) []:Erasme
Common Name (eg, YOUR name) []:mobile.workstation
Email Address []:workstation@domain.tld
```

```
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Request (and private key) is in newreq.pem
ca#
```

Creating a certificate request

This will leave a newreq.pem file (the certificate request) in the *current* directory (not somwhere below demoCA).
You then need to sign this request with the CA certificate.

```
ca# /usr/share/ssl/misc/CA.pl -sign
Using configuration from /etc/ssl/openssl.cnf
Enter pass phrase for ./demoCA/private/cakey.pem:
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 1 (0x1)
        Validity
            Not Before: Feb 28 18:13:18 2004 GMT
            Not After : Feb 27 18:13:18 2005 GMT
        Subject:
            countryName               = FR
            stateOrProvinceName       = Rhone
            localityName              = City
            organizationName          = Organization
            organizationalUnitName    = Erasme
            commonName                = mobile.workstation
            emailAddress              = workstation@domain.tld
        X509v3 extensions:
            X509v3 Basic Constraints:
            CA:FALSE
            Netscape Comment:
            OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
            E8:04:EB:61:43:F1:C7:E9:C6:AF:E3:00:77:D8:A1:55:4D:5A:5D:33
            X509v3 Authority Key Identifier:
            keyid:D1:F4:59:4E:A9:A4:11:99:55:E6:91:42:2E:13:48:55:95:2F:94:CB

DirName:/C=FR/ST=Rhone/L=City/O=Organization/OU=Erasme/CN=ca.domain.tld/emailAddress=ca@domain.tld
            serial:00

Certificate is to be certified until Feb 27 18:13:18 2005 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
Signed certificate is in newcert.pem
ca#
```

Signing the certificate request

CA.pl will ask for the CA private key, and ask confirmation before signing the certificate. After completion, you get a C signed certificate you can use on your workstation. Rename the certificate (newcert.pem) and key (newreq.pem) respectively to workstation-cert.pem and workstation-key.pem and put them on the worstation, along with the CA certificate (see below).
Repeat this process for the security gateway, and for any other mobile hosts needed.
The CA setup/cert request/cert signing process is well explained in great detail also [6]. So don't hesitate reading the howto for more accurate information.

Security gateway

IPsec

Copy the CA certificate to /etc/racoon/certs/ (more specifically to the "path certificate" specified in racoon.conf).
At the same place, put the security gateway certificate and key.

```
sg# cd /etc/racoon/certs
sg# ln -s cacert.pem `openssl x509 -noout -hash < cacert.pem`.0
```

You also need to 'unlock' your key, in order to have a passwordless key.

```
sg# openssl rsa -in security-gateway-key.pem -out security-gateway-key.pem
read RSA key
Enter PEM pass phrase: password
writing RSA key
sg#
```

Adjust the anonymous part of raccon.conf, in order to accept roadwarrior connections.

```
remote anonymous {
        exchange_mode main;
        generate_policy on;
        passive on;
        certificate_type x509 "security-gateway-cert.pem" "security-gateway-key.pem";
        my_identifier asn1dn;
        peers_identifier asn1dn;
        proposal {
                encryption_algorithm 3des;
                hash_algorithm md5;
                authentication_method rsasig;
                dh_group modp1024;
        }
}

sainfo anonymous
{
        pfs_group 2 ;
        lifetime time 28800 minutes ;
        encryption_algorithm 3des;
        authentication_algorithm hmac_sha1, hmac_md5 ;
        compression_algorithm deflate ;
}
```

/etc/racoon/raccon.conf extract

The important point here is "generate policy on", that will insert proper policies in the security database. Thus, no "set call is required (nor possible, since the workstation gets a dynamic IP), and you'll be able to roadwarrior.

For detailed instructions on how to setup 2.6 native IPsec stack, check out [2], [5] and [6].

proxy-arp

Under Linux 2.6, all that's needed to do proxy arp for our modile workstation is to insert a host route on the outbound (internet side) interface, and activate corresponding entries in /proc. So preparing our security gateway only requires

```
sg# echo 1 > /proc/sys/net/ipv4/ip_nonlocal_bind
sg# echo 1 > /proc/sys/net/ipv4/conf/eth0/proxy_arp
sg# ip route add 192.168.0.1 dev eth1
```

BTW, verify that routing is enabled :

```
sg# cat /proc/sys/net/ipv4/ip_forward
1
sg#
```

Mobile workstation

IPsec

FreeS/Wan has to be set up at the workstation.
First, you need to copy your certificate authority (CA) certificate in */etc/ipsec.d/cacerts/* and issue

```
ws# cd /etc/ipsec.d/cacerts
ws# ln -s MyCA.pem `openssl x509 -noout -hash -in MyCA.pem`.0
```

Copy the workstation key (workstation-key.pem) in the right place, usually */etc/ipsec.d/private/*
Then, copy the workstation certificate (workstation-cert.pem), in */etc/ipsec.d/certs/*

You can verify that your certificates setup is fine by issuing :

```
ws# openssl verify -CAfile MyCA.pem  ../certs/workstation-cert.pem
../certs/workstation-cert.pem: OK
ws#
```

Edit *ipsec.secrets* (lies usually in /etc), and add the workstation key to use for the certificate, with the key's password. A usual, the less people can read this file, the better...

```
a.b.c.d: RSA    /etc/ipsec.d/private/workstation-key.pem "password"
```

/etc/ipsec.secrets

Now, you need to edit the FreeS/Wan configuration file (*/etc/ipsec.conf*), and add an entry for the connection.
You can get the strings in leftid and rightid with the following command :

```
ws# openssl x509 -in /etc/ipsec.d/certs/workstation-cert.pem -subject | grep subject
subject=
/C=FR/ST=Rhone/L=City/O=Organisation/OU=domain/CN=mobile.workstation/emailAddress=workstation@domain.t

sg# openssl x509 -in /etc/racoon/certs/security-gateway-cert.pem -subject | grep subject
subject=
/C=FR/ST=Rhone/L=City/O=Organisation/OU=domain/CN=security.gateway/emailAddress=security@domain.tld
```

(Note the last command has been run on the security gateway)

Usually, there are no other relevant changes in the default configuration.

```
conn mothership
    # Left security gateway, subnet behind it
    left=a.b.c.d
    leftsubnet=192.168.0.0/24
    # Mobile workstation, subnet behind it
    right=%defaultroute
    rightsubnet=192.168.0.1/32
    leftrsasigkey=%cert
    rightrsasigkey=%cert
    rightcert=/etc/ipsec.d/certs/workstation-cert.pem
    authby=rsasig
    auto=add
    leftid="C=FR/ST=Rhone/L=City/O=Organisation/OU=domain/CN=security.gateway/emailAddress=security@doma

rightid="C=FR/ST=Rhone/L=City/O=Organisation/OU=domain/CN=mobile.workstation/emailAddress=workstation@
```

/etc/ipsec.conf

At this point, you can try to start the connection with your security gateway.
Fire your ppp connection to you provider and try :

```
ws# ipsec auto --up mothership
```

FreeS/Wan should reply with something like :

```
104 "mothership" #1: STATE_MAIN_I1: initiate
003 "mothership" #1: ignoring Vendor ID payload
106 "mothership" #1: STATE_MAIN_I2: sent MI2, expecting MR2
003 "mothership" #1: ignoring Vendor ID payload
108 "mothership" #1: STATE_MAIN_I3: sent MI3, expecting MR3
004 "mothership" #1: STATE_MAIN_I4: ISAKMP SA established
112 "mothership" #2: STATE_QUICK_I1: initiate
004 "mothership" #2: STATE_QUICK_I2: sent QI2, IPsec SA established
```

packet mangling

There are two things that remain to be done. You actually have the IPsec tunnel up and running, but you can't send anything thru until you tweak the source IP of the packets coming out from the workstation and directed to your interr network. That's why you need iptables to the rescue, addind source NAT to your netfilter rules :

```
ws# iptables -t nat -A POSTROUTING -o ipsec0 -j SNAT --to 192.168.0.1
```

Now, packets coming out via ipsec0 (and thus, directed to 192.168.0.0/24) will have their source IP changed to 192.16

Also, you cannot keep 192.168.0.1/24, since the home network will appear directly connected to eth0 in your routing tables.
So what can you do ? There are two ways.

"Normal" Way
What comes to mind first is to adjust the netmask so 192.168.0.0/24 doesn't appear directly connected anymore to eth

```
ws# ip address flush eth0
ws# ip address add 192.168.0.1/32 dev eth0
```

You can then check your routing table and verify that you have no route to 192.168.0.0/24.

Funny Way
Well, the other way to do it is just to flush any IP address belonging to the 192.168.0.0/24 network.

```
ws# ip address flush eth0
```

That's all ! But how can this work ? You don't have any IP in the 192.168.0.0/24 network now, so how can this stuff wo now ?
Well, remember you use SNAT for packets travelling ou thru ipsec0. So anything going out thru this interface will have source IP changed to 192.168.0.1?
It does the trick. As a side effect, since you don't need to keep 192.168.0.1 IP address anywhere, you just can use these instructions whatever your connection method is (network, DSL, ISDN, etc...). No need to create aliases or whatever. I just works as is.

Now, when you try to ping a home network friend (say, 192.168.0.10), you packets going thru ipsec0 will have your internal IP as source (and not your dynamically assigned ppp0 address, which Linux would use by default since it's the packet gateway), and you should get replies.

Firing it up

You can arrange the above stuff in a single script (let's say /usr/local/sbin/remote), that will be called when a ppp cone is established. On SuSE systems, you can call it from /etc/ppp/ip-up.local or /etc/ppp/ip-up.d/.

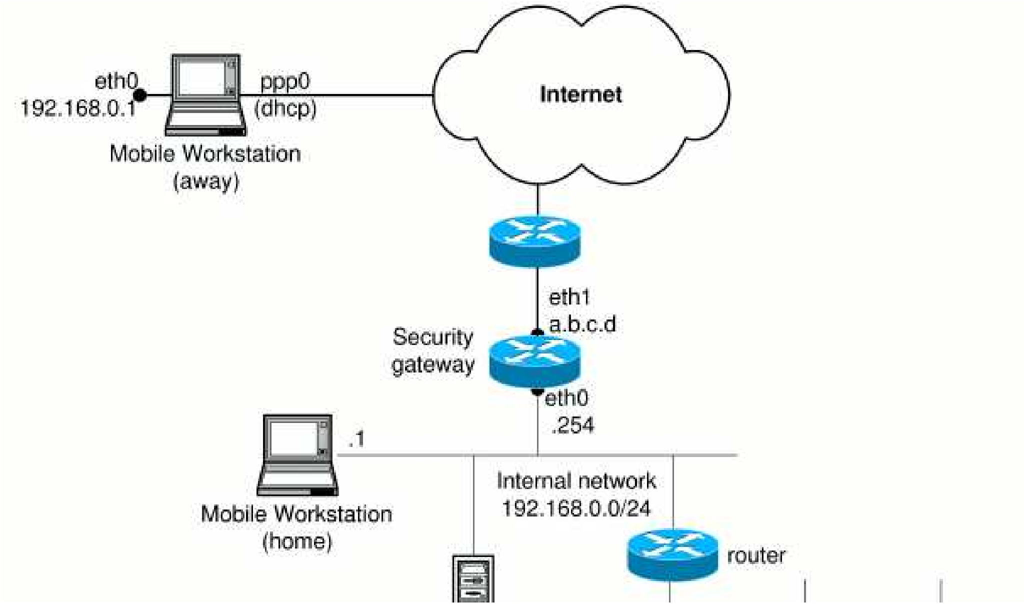
```
#!/bin/sh

FRIEND=192.168.0.10

case "$1" in
   start)
      # start FreeS/WAN
      /etc/rc.d/ipsec start

      # we need to wait a bit - starting tunnel too early would fail
      sleep 2

      # we get rid of eth0, or packets for home network will be routed via eth0
      /sbin/ip address flush dev eth0
      /sbin/ip link set eth0 down

      # bringing the tunnel up
      /usr/sbin/ipsec auto --up mothership && \
         /bin/logger -p local6.info -s -t remote-ipsec "mothership IPsec tunnel up"

      # SNAT rule insertion
      /usr/sbin/iptables -t nat -A POSTROUTING -o ipsec0 -j SNAT --to 192.168.0.1 && \
         /bin/logger -p local6.info -s -t remote-iptables "added SNAT rule"

      # connection test
      ping $FRIEND -c 1 -q > /dev/null
      ping $FRIND -c 1 -q > /dev/null && \
         /bin/logger -p local6.info -s -t remote "remote tunnel setup succeed" || \
         /bin/logger -p local6.info -s -t remote "remote tunnel setup failed"
      ;;
   stop)
      # flush SNAT
      /usr/sbin/iptables -t nat -F && \
         /bin/logger -p local6.info -s -t remote-iptables "flushed SNAT rules"

      # stop FreeS/WAN
      /etc/rc.d/ipsec stop

      # restart network to it's defaults
      /etc/rc.d/network start
      ;;
esac
```

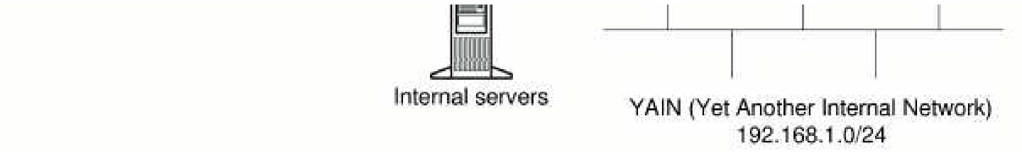
/usr/local/sbin/remote

You can now add "/usr/local/bin/remote start" and "/usr/local/bin/remote stop" repectively to "/etc/ppp/ip-up.local" and "/etc/ppp/ip-down.local"

More fun

Now, let's imagine you have another network behind your home network, and want to access it. Suppose this network 192.168.1.0/24, and is connected to your "home network" 192.168.0.0/24.





Accessing this network can be done in a snap, and only the workstation IPsec configuration has to be slightly modified the trick. Just add another connection (named 'yain' here), and you're done :

```
conn yain
    # Left security gateway, subnet behind it
    left=a.b.c.d
    leftsubnet=192.168.1.0/24
    # Mobile workstation, subnet behind it
    right=%defaultroute
    rightsubnet=192.168.0.1/32
    leftrsasigkey=%cert
    rightrsasigkey=%cert
    rightcert=/etc/ipsec.d/certs/workstation-cert.pem
    authby=rsasig
    auto=add
    leftid="C=FR/ST=Rhone/L=City/O=Organisation/OU=domain/CN=security.gateway/emailAddress=security@doma

rightid="C=FR/ST=Rhone/L=City/O=Organisation/OU=domain/CN=mobile.workstation/emailAddress=workstation@
```

/etc/ipsec.conf

The only change compared to 'mothership' is the 'leftsubnet' part (192.168.1.0/24). This will add a SPD entry in the IPs security gateway stack. This gateway will now also handle connections between 192.168.0.1/32 and the 192.168.1.0/24 network. To bring the tunnel up for this SPD, use the ipsec command on the workstation :

```
ws# ipsec auto --up yain
```

You can now access the 'yain' network as if you were connected locally. Of course, if you have restrictions and filtering 'router', you're are still subject to them.

Caveats

Proxy arping might break your LAN connection depending which of your security gateway and workstation replies to a requests first.
If your security gateway replies first, it will get you traffic even if your connected to your LAN, as illustrated in the example below (taken from another LAN workstation 192.168.0.2) :

```
# arp -na
#
# tcpdump -nieth0
[while tcpdump is runnning, a ping is issued in another terminal]
tcpdump: listening on eth0
12:14:12.457899 arp who-has 192.168.0.1 tell 192.168.0.2
12:14:12.458068 arp reply 192.168.0.1 is-at 0:8:2:64:3b:4d
12:14:12.458083 192.168.0.2 > 192.168.0.1: icmp: echo request (DF)
12:14:12.458281 192.168.0.1 > 192.168.0.2: icmp: echo reply
12:14:12.490133 arp reply 192.168.0.1 is-at 0:c:29:1a:71:55
12:14:13.456892 192.168.0.2 > 192.168.0.1: icmp: echo request (DF)
12:14:13.457095 192.168.0.1 > 192.168.0.2: icmp: echo reply

7 packets received by filter
0 packets dropped by kernel
# arp -na
workstation (192.168.0.1) at 00:08:02:64:3B:4D [ether] on eth0
#
```

You can see in the example above that two arp replies comes from the LAN : one from the workstation, another one fro the security gateway.
Since the workstation reply comes first, things run happilly, but it wouldn't be the case if the security gateway arp rep would come first : the workstation would be virtually non-existent on the ehernet segment.

There is no easy automatic workaround to this situation, since solutions involves :

- manually adding the proxy arp entry via ssh when you're away : you can put your ssh key on the security gatewa and automatically login via a FreeS/WAN triggerred script (but this requires to set-up a ssh server on the sec-gw which might not be good security-wise),
- switching to FreeS/WAN at the security gateway (FreeS/WAN can trigger script upong tunnel establishment, rac can not)
- write a daemon that will monitor tunnels going up (SPD changes) and add the correponding proxy-arp entry if needed
- add latency to security gateway arp replies (might be possible with dummynet under BSD, but AFAIK not possib with current linux IP stack)

Conclusion

Away from your home network this setup will let you connect to your internal servers in a secure manner. While it lack broadcast and multicast packet handling, you can still browse your samba fileservers for instance, by using their IPs directly, or by setting up the lisa daemon on your machine. Even connecting via ppp, you virtually sit in the middle of your home network, and can use it as you would via eth0.

Links

[1] FreeS/WAN project, http://www.freeswan.org/
[2] Ralf Spennenberg IPsec pages (check also his excellent articles in Sysadmin Magazine, unfortunately not all are online), http://www.spenneberg.com/
[3] L. Wolenczak pages and fancy IPsec setups, http://lwolenczak.net/
[4] Lisa daemon, http://**lisa-home**.sourceforge.net
[5] IPsec tools home page, http://ipsec-tools.sourceforge.net/
[6] IPsec HowTo, http://www.ipsec-howto.org
[7] Fedora, http://fedora.redhat.com/

Keywords : ppp, ipsec, freeswan, Linux 2.6, dynamic ip, no pptp
Author : mblanc@erasme.org
Version : 20040603
Revisions :
20040603 : added 'caveats' section
20040303 : fixed subject for security gateway
20040229 : fixed sysctl entries
20040228 : creation