Manual: How to Use GAA-API & IPSEC Integration

Li Zhou - 06/26/2002

PART ONE - INSTALLATION

We should have two gateway machines installed. We could call them "left node" and "right node". For our demo, left node will have GAA-FreeSWAN (the Integration of GAA-API & FreeSWAN) installed while right node is still running the original FreeSWAN version. Therefore, we should:

[On Right Node]
- Step 1: Install Linux Kernel
- Step 2: Install Original FreeSWAN

[On Left Node]
- Step 1: Install Linux Kernel
- Step 2: Install GAA-API
- Step 3: Install GAA-FreeSWAN

Here are the detailed procedures for each step:

Installation of Linux Kernel:

1. Download Linux kernel source to
   - [For GZ package] /usr/src/kernel-source-2.x.x.tar.gz
   - [For RPM package] /usr/src/kernel-source-2.x.x.rpm

   Kernel source packages available on http://www.redhat.com or http://www.kernel.org
2. Unpack the kernel source
   - [For GZ package] tar xzvf /usr/src/kernel-source-2.x.x.tar.gz
   - [For RPM package] rpm -vi /usr/src/kernel-source-2.x.x.rpm
3. If necessary, make a symlink linking kernel source you are using to /usr/src/linux
   - ln -s /usr/src/linux-2.x.x-xx /usr/src/linux
4. Compiling the Linux kernel
   - make dep; make clean; make bzImage; make modules; make modules_install

Installation of Original FreeSWAN:

1. Download FreeSWAN's source code to /usr/src/

   Code available on: http://www.isi.edu/~zhou/gaa-api/ipsec/freeswan-1.97.tar.gz
2. Unpack the FreeSWAN package:
   - tar xzvf /usr/src/freeswan-1.97.tar.gz
3. Compile & Install FreeSWAN:
   - [In X-Window] cd /usr/src/freeswan-1.97; make xgo; make kinstall
   - Within the kernel configuration dialog that popups, checked all options that relative to IPSEC in sub-menu "Network Configuration". And then, choose "Save and Exit" to go on.
4. Copy the compiled System.map file and kernel image file to /boot directory
   - cp System.map-2.x.x /boot/
   - cp vmlinuz-2.x.x /boot/
5. Add a new boot entry "ipsec-linux" into your configuration file:

[Using LILO]
   - a) Add the following lines into your /etc/lilo.conf file

     image=/boot/vmlinuz-2.x.x
     label=ipsec-linux
     read-only

root=/dev/hda1 #the linux root partition

b) Update new LILO configuration: /usr/bin/lilo

[Using Grub]

Add the following lines into your /boot/grub/grub.conf

title ipsec-linux

root=(hd0,5)

kernel /boot/vmlinuz-2.x.x ro root=/dev/hda7 #the linux root partition

6. Reboot your machine and choose "ipsec-linux" when prompted by LILO/Grub

Installation of GAA-API

1. Download GAA-API package onto /usr/src/gaa-api.tar.gz

Package available on: http://www.isi.edu/~zhou/doc/gaa-api-latest.tar.gz

2. Unpack the GAA-API package

tar xzvf /usr/src/gaa-api.tar.gz

3. Install the "lib2xml", "lib2xml-devel" and "libltdl" packages using GnoRPM. Make the following symlinks if they are not already there:

ln –s [xml shared library file path] /usr/lib/libxml.so.2

ln –s [xml header files directory path] /usr/include/lib2xml/

ln –s [ltdl shared library file path] /usr/lib/libltdl.so

4. If all GAA-API's shared libraries are already in /usr/src/gaa-api/lib directory. Then, generally, we don't need to compile it. But if we need to apply new modification on GAA-API and recompile it, please execute the following:

cd /usr/src/gaa-api; make

*5. If needed, copy the shared libraries and configuration files to /usr/local/:

cp /usr/src/gaa-api/lib/*.so /usr/local/lib/

mkdir /usr/local/gaa

cp /usr/src/gaa-api/test/config /usr/local/gaa/

cp /usr/src/gaa-api/test/eacls /usr/local/gaa/

cp /usr/src/gaa-api/test/demo /usr/local/gaa/

* If we are using latest version of LIBTOOL (3.0 or later), we may have to symlink the GAA-API shared libraries to "/lib" and "/usr/lib" because of unknown compatibility problems in LIBTOOL.

Installation of GAA-FreeSWAN:

1. Download GAA-FreeSWAN source code to /usr/src/gaa-freeswan.tar.gz

Code avaibale on: http://www.isi.edu/~zhou/gaa-api/ipsec/gaa-freeswan-latest.tar.gz

2-6. (same as "Installation of Original FreeSWAN")

PART TWO - FREESWAN CONFIGURATION

Generating RSA Keys (/etc/ipsec.secrets)

In FreeSWAN, we should RSA key peers for authentication. First of all, each FreeSWAN gateway should maintain the complete information of public/private keys in file /etc/ipsec.secretes. To generate RSA keys, we should do the following on both left node and right node:

1. Execute command: /usr/local/sbin/ipsec rsasigkey 2048

Note: 2048 is the length of RSA key (in bits).

2. Copy all text generated by the command into file /etc/ipsec.secrets. (Attention: we must preserve the original

indent of all these information.

Here is an example of ipsec.secrets file:

```
: RSA     {
          # RSA 2048 bits   rip   Fri Jun 14 10:43:46 2002
          # for signatures only, UNSAFE FOR ENCRYPTION
          #pubkey=0sAQNsFiwynyMo0qIK3ukcZcstMrn/SnXQEntDu8vrB…
          #IN KEY 0x4200 4 1 AQNsFiwynyMo0qIK3ukcZcstMrn/SnXQEntDu8vrBA93suaFD9Kx4VkDce4SxmT54y…
          # (0x4200 = auth-only host-level, 4 = IPSec, 1 = RSA)
          Modulus: 0x6c162c329f2328d2a20adee91c65cb2d32b9ff4a75d0127b43bbcbeb040f77b…
          PublicExponent: 0x03
          # everything after this point is secret
          PrivateExponent: 0x1203b2086fdb317870572526da10f732331effe1be4d5869e09f4ca72b57e948…
          Prime1: 0xd154fa4319db2c8bfd979e4e9c3ef45aca22adad1e2b5e63cae76c6536e54278a016e2be5…
          Prime2: 0x842ee8384eb50fee394467ca8a5ae65c7e1d0b0c2bc004527db552c2bbebbb45ef6ee9f9d…
          Exponent1: 0x8b8dfc2cbbe77307fe6514346829f83c86c1c91e141ce997dc9a484379ee2c506ab9ec…
          Exponent2: 0x581f457adf235ff4262d9a8706e7443da968b2081d2aad8c53ce372c7d47d22e9f9f46…
          Coefficient: 0x4249fe2c90c94d114faec275d784dae26b1e8412802e4b58fa0343ed624446803d9dc…
          # -- not filled in because ipsec.secrets existed at build time --
          }
# do not change the indenting of that "}"
```

Configuring FreeSWAN Connection (/etc/ipsec.conf)

For each IPSEC connection, we have left node & right node as FreeSWAN server. They both serve as the gateway of a specific LAN. The packages within the LAN is assumed reliable while is unreliable between these this two LAN. Therefore we could use two FreeSWAN gateways to authenticate and encrypt all communication between these two LAN. To make such mechanism work, we should add connection information into /etc/ipsec.conf on both left node and right node.

Here is an example configuration for connection named “LGW-GRW” in the “ipsec.conf” file.

```
conn LGW-RGW
    left=10.0.1.1
    leftsubnet=10.0.1.1/255.255.255.0
    right=10.0.2.1
    rightsubnet=10.0.2.1/255.255.255.0
    auth=rsa
    pubkeyleft=0sAQNsFiwynyMo0qIK3ukcZcstMrn/SnXQEntDu8vrBA93suaFD9Kx4VkDce4SxmT54yWRlISv2rM9OjKp0/afwo
      zS0KnucJ3tfkS4ouBls7VGoZTgMtOprBB/PHLKMyVvC9nQdX41RJbt0xS0BnpZqFsAUQxAosr3ggFwosL0eHL0/gP/SfY
      m7xr7ohAAptVTZp1dbHu5auM9xfJJMZxJhGyXjtDQCcZxtndcb+7MOTLnUd6zOV3R6tk6fH3AT7udrNjEtutP9cM4PZYm
      HtfGSaxt0CudII+7LXTMyxjWFlrSMvluxPDearBv16Taqofy8fmeKPz+Tk6wKUoK1zzr7uUb
    pubkeyright=0sAQNhUD8CTfksezfC8s7IdulnAThsou1ZQ7/Ec+6XkHfnXW+5UyMnyDzCS68c9TPWKDW9vraVg8N830khN
      F/L/GLkC9kgalWNkqqPEBh+9WYg90+r8M/DnNbIEo6RTBiuMErd5FB/7sj9MCZg6AaoCHs/iGObk3rerEVRHc8QpguQhs
      oVOPDUaaYlqwFH7g7Hiz9gEA80Q5fyBYdRGaYYnLpiTVP8G8D/wzE40oFe96h6tMIXD4byq/CZ37NbcXbFRKH6yzfan
      ZW2uGD2IA7yH+9Sva6XqP/PZe1VQ9zwk0/dmBciagN78VXhPnxbg6SsddYd7fD7q6omIU7vVq5Iedzl
    auto=add
```

Here, LGW-RGW is the user defined connection name. Directives left and right specify the IP address of left node and right node. Then, leftsubnet and rightsubnet define the LAN that they are serving for. Finally, the values of directive pubkeyleft and pubkeyright are copies from the pubkey fields in /etc/ipsec.secrets of left node and right node respectively.

We should also specify which network device FreeSWAN will run upon. For example, if we choose “eth0”, then the field “config setup“ should contain the line:

```
interface=“ipsec0=eth0”
```

Moreover, we need to register the connection we defined in “ipsec.conf” by the command:

```
/usr/local/sbin/ipsec auto --add LGW-RGW
```

To validate new configuration above, we should reboot the machine or execute the following command:

```
/etc/init.d/ipsec restart
```

Modifying Linux Configuration

To make FreeSWAN runnable, we need to change the following Linux configurations:

1. In file “/etc/sysconfig/network”, add this line to enable package forwarding:

```
ipforward=1
```

2. In the shell configuration file (.bashrc or .tshrc etc., it depends on which shell you are using.) of your home directory, add the following commands:

```
echo 0 > /proc/sys/net/ipv4/conf/eth0/rp_filter
echo 0 > /proc/sys/net/ipv4/conf/ipsec0/rp_filter
```

Here we suppose we are using “ipsec0=eth0” as the network device that FreeSWAN will run upon. If we are using other network device (such as eth1, ppp, lo), just substitute the field “eth0” with the right device name within the commands above.

3. To ensure the commands within .bashrc has been applied, first enter (re-enter) the shell

```
bash
```

And, then use the following commands to confirm that the values displayed is “0”.

```
cat /proc/sys/net/ipv4/conf/eth0/rp_filter
cat /proc/sys/net/ipv4/conf/ipsec0/rp_filter
```

PART THREE - EACL POLICY FOR GAA-FREESWAN

1. Add three more entries for condition evaluation callbacks into /usr/local/gaa/config/gaa.conf:

```
cond_eval cond_access_time DEFAULT {
  cond_eval libgaa_simple.so cb_check_cond_access_time
  idcred no
}
cond_eval cond_access_host DEFAULT {
  cond_eval libgaa_simple.so cb_check_cond_access_host
  idcred no
}
cond_eval cond_access_method DEFAULT {
  cond_eval libgaa_simple.so cb_check_cond_access_method
  idcred no
}
```

2. Add pathname of EACL files within <gaa:system> & <gaa:local> tags in file /usr/local/gaa/config/gaa.policy.loc

Here is an example of gaa.policy.loc:

```
<?xml version=”1.0”?>
<gaa:Namespace xmlns:gaa="http://www.isi.edu">
<gaa:Policy>
<gaa:system>
/usr/local/gaa/eacls/gaa_sys.xml
</gaa:system>
<gaa:sys_tag>
'Net*'
</gaa:sys_tag>
<gaa:local>
```

/usr/local/gaa/eacls/demo.eacl
</gaa:local>
</gaa:Policy>
</gaa:Namespace>

3. Specify EACL policy for FreeSWAN

To establish a FreeSWAN connection, there are two phases. The first phase is ISAKMP (also called OAKLEY mode) and the second phase is IPSEC (also called quick mode). We could pose access restrictions on both ISAKMP and IPSEC by GAA-API.

[Phase 1: ISAKMP]

For any policy for phase 1, the authority of request right should be “FREESWAN” and value should be “ISAKMP”:

```
{ pos_access_right | neg_access_right }  FREESWAN  ISAKMP
```

And we could have the following conditions specified:

```
cond_access_method  OAKLEY_ENCRYPTION  { DES | IDEA | BLOWFISH | RC5_R16_B64 | 3DES | CAST | AES | NONE }
cond_access_method  OAKLEY_HASH  { MD5 | SHA | TIGER | SHA2_256 | SHA2_384 | SHA2_512 | NONE }
cond_access_method    OAKLEY_AUTH    { PRESHARED_KEY  |  DSS_SIG  |  RSA_SIG  |  RSA_ENC  |  RSA_ENC_REV  |
        ELGAMAL_ENC | ELGAMAL_ENC_REV | AUTH_ROOF | NONE }
cond_access_method  OAKLEY_GROUP  { MODP768 | MODP1024 | GP155 | GP185 | MODP1536 | NONE }
cond_access_method  CONNECTION_NAME  [conn_name]
cond_access_time  { local | gmt }  [time_span]
cond_access_host  { LOCAL_IP | PEER_IP }  [host_set]
```

[Phase 2: IPSEC]

For any policy for phase 2, the authority of request right should be “FREESWAN” and value should be “IPSEC”:

```
{ pos_access_right | neg_access_right } FREESWAN  IPSEC
```

And we could have the following conditions specified:

```
pre_cond_access_method  IPSEC_AH_AUTH  { NONE | HMAC_MD5 | HMAC_SHA1 | DES_MAC | KPDK | UNKNOWN }
pre_cond_access_method  IPSEC_ESP_AUTH  { NONE | HMAC_MD5 | HMAC_SHA1 | DES_MAC | KPDK | UNKNOWN }
pre_cond_access_method   IPSEC_ENCRYPTION   { DES_IV64 | DES | 3DES | RC5 | IDEA | CAST | BLOWFISH | 3IDEA |
        DES_IV32 | RC4 | NULL | AES | NONE }
pre_cond_access_method  IPSEC_COMPRESS  { OUI | DEFLATE | LZS | V42BIS | NONE }
pre_cond_access_method  CONNECTION_NAME  [conn_name]
pre_cond_access_time  { local | gmt }  [time_span]
pre_cond_access_host  { LOCAL_IP | PEER_IP }  [host_set]
```

Here’s an example of FreeSWAN’s EACL policy

[In System Policy File: gaa_sys.xml]

```
eacl_mode 1                             # narrow mode
neg_access_right * *                    #this is a policy for all applications using GAA-API
pre_cond_stl        demo       =red     #deny access if the system thread level is red
```

[In Local Policy File: demo.eacl]

```
eacl_mode 2 # exact
pos_access_right      FREESWAN        ISAKMP
pre_cond_access_method CONNECTION_NAME   "B, A"
pre_cond_access_method OAKLEY_AUTH      "RSA_SIG, PRESHARED_KEY"

pos_access_right      FREESWAN        IPSEC
pre_cond_access_method IPSEC_COMPRESS   "NONE, DEFLATE"
pre_cond_access_time   local          "1/1/2002-12/31/2002 MON-FRI 6:00am-10:00pm"
```

PART FOUR - RUNNING GAA-FREESWAN

1. To bring up the IPSEC connection, we should execute:
    /usr/local/sbin/ipsec auto --up LGW-RGW
(LGW-RGW is the connection name we defined in /etc/ipsec.conf)
2. To bring down the IPSEC connection, we should execute:
    /usr/local/sbin/ipsec auto --down LGW-RGW
3. To restart IPSEC daemon and reload configuarion files (including ipsec.secrets, ipsec.conf and EACL policy), start the IPSEC daemon and stop the IPSEC daemon, we should execute:
    /etc/init.d/ipsec restart
    /etc/init.d/ipsec start
    /etc/init.d/ipsec stop

Important Note For Demo:

In the demo, we generally execute connection bring-up and bring-down only on left node. (the gateway with modified GAA-FreeSWAN.) Therefore, log information about GAA-API's policy evaluation will be printed to the terminal when connection is brought up.

If fatal error occurs and terminates the connection on left node improperly, the right node (the one with original FreeSWAN) may sometimes remain busy and then refuse any further connection. In this case we should execute connection bring-down on right node to reset its IPSEC daemon.