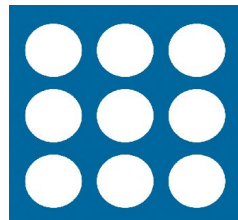


# **The Generic Gateway**

**- an ENUM driven PSTN-IP gateway  
Reference Implementation**

**Installation HowTo**



enum.at  
net.communications

© 2004 Internet Privatstiftung Austria (IPA)

<http://www.nic.at/ipa>

**Version history:**

<i>Version</i>	<i>Date/author</i>	<i>Changes</i>
0.1	04.11.2004/Klaus Darilion	initial version for GG tests
0.2	07.12.2004/Klaus Darilion	remove sensitive information, review for public release

## Table of Contents

1 The Generic Gateway Installation.....	4
1.1 Language Settings.....	4
1.2 Partition.....	4
1.3 Device Driver Modules.....	4
1.4 Network Configuration.....	4
1.5 Make System Bootable.....	6
1.6 Reboot the System.....	6
1.7 Password Options.....	6
1.8 Root Password.....	6
1.9 APT-Configuration.....	6
1.10 Tasks to install.....	6
2 Manual Installation .....	7
2.1 Configure the Virtual Interfaces.....	7
2.2 Install and configure bind9 name server.....	7
2.3 Configure the Helper PC to use the local bind9 as name server.....	8
2.4 Installing Perl Modules.....	9
2.5 Configuring Postgresql Database.....	9
2.6 Installing Radiator.....	13
2.6.1 Radius Authentication Server and Perl Call Routing Script.....	15
2.6.2 Radius Accounting Server.....	15
2.7 Installing procmail.....	15
2.8 Installing debug tools.....	16
2.9 Install the ENUM.pm Perl Module.....	16
2.10 Configuring exim.....	16
2.11 Install the sendfax script.....	17
2.12 Installing the DNS proxy.....	17
2.13 Time Synchronization.....	18
2.14 Manage Debian Distributions.....	18
3 Debugging.....	18
3.1 Services.....	18
3.2 Call Detail Records Analysis.....	19
3.3 Debugging on the Generic Gateway.....	19
3.3.1 On the Cisco Gateway.....	19
3.3.2 On the helper PC:.....	20

# 1 The Generic Gateway Installation

The original woody installation does not support new hardware, as new hardware is only supported by new kernels. In our case, the NW card was not supported by the Woody release, but they are supported by a modified Debian installer available at:

<http://wiki.osuosl.org/display/LNX/Debian+on+Dell+Servers>

Download the ISO image `debian-dell-2.4.26.iso`, burn it on CD and boot the PC with this CD. After booting of the kernel, the installation continues with the original Debian Woody CD 1.

This HowTo describes not all steps of the installation, but only the important parts which varies from the default settings.

For more details on how to install Debian please refer to the Debian Installation Manual at:

<http://www.debian.org/releases/stable/installmanual>

## 1.1 Language Settings

As language of the Debian installation choose “english”, as keyboard choose your keyboard!

## 1.2 Partition

The hard disc (in case of our PCs the hard disc was 36 GB) will be partitioned for easier maintenance. Use the partitioning tool to partition the hard disc and mount points according to the following table:

<i>partition</i>	<i>mount point</i>	<i>size</i>
/dev/hda1	/	1 GB
/dev/hda2	/usr	4 GB
/dev/hda3	/var	8 GB
/dev/hda5	/home	8 GB
/dev/hda6	swap	1 GB
		<b>22 GB (14 GB free)</b>

As file format use *ext3*.

## 1.3 Device Driver Modules

When asked for the device driver modules, select the modules for your hardware – e. g. choose `e1000` to add support for Intel based Gigabit Ethernet adapter.

## 1.4 Network Configuration

Enter the hostname according the network setup of the GG.

Configure the primary IP address of the Helper PC – virtual interfaces will be configured after the Debian installation. The IP addresses in the laboratory will be used according the following table:

<i>IP address</i>	<i>main purpose</i>	<i>hosted on</i>
10.0.0.230	dnsproxy,	Helper PC 1
10.0.0.231	Cisco Gateway	Cisco Gateway
10.0.0.232	main IP, postgresql DB, TFTP server,	Helper PC 1
10.0.0.233		Helper PC 2
10.0.0.234	bind9, radiator	Helper PC 1
10.0.0.235		Helper PC 2
10.0.0.236		Helper PC 2

Note: You have to use public IP addresses for your setup. Using private IP addresses will cause problem! Of course you can distribute the services over your IP addresses as you want :-)

When asked for nameservers, add the nameserver of your existing network infrastructure – these will be extended later by our own caching-only recursive nameserver.

**Note:** To ensure flawless operation of the GG make sure to add the domain names into the corresponding DNS zone and also add the corresponding revers lookup pointers. Suggested domain names for the helper PCs and the services are:

<b>Helper PC 1</b>	
ggpc1	ssh, standard services, tftp
ggns1	caching only recursive name server
ggdnsproxy1	dns proxy for NAPTR rewriting
ggpg1	postgresql DB
ggrad1	radius server for accounting and authentication (=ENUM lookup)
ggsmtpl	Email server for sending emails with tiff attachment
<b>Helper PC 2</b>	
ggpc2	ssh, standard services, tftp
ggns2	caching only recursive name server
ggdnsproxy2	dns proxy for NAPTR rewriting
ggpg2	postgresql DB
ggrad2	radius server for accounting and authentication (=ENUM lookup)
ggsmtpl2	Email server for sending emails with tiff attachment
<b>PSTN-IP Gateways</b>	
gggw1	first Cisco AS5300
gggw2	second Cisco AS5300

## **1.5 Make System Bootable**

Install the boot loader as suggested by the installer. Create a boot floppy as backup to boot the helper PC in case of boot troubles.

## **1.6 Reboot the System**

Now the helper PC should reboot and the second part of the Debian begins – in this part we will install the applications and manage users.

## **1.7 Password Options**

Enable MD5 passwords and shadow passwords for security reasons.

## **1.8 Root Password**

Set the root user password. The root password in the laboratory setup is “780enum”.

## **1.9 APT-Configuration**

The Advanced Packaging Tool will ask you for Debian mirrors. Select a mirror close to you and add download locations for the woody (stable) release and security downloads. Choose http as download protocol to avoid firewall issues.

## **1.10 Tasks to install**

Select the groups “C and C++” and “SQL database” to install the development environment and the postgresql database. Answer all following questions regarding the CVS installation with the default answer.

Exim configuration:

- Question 1: accept answer (1) (=default)
- visible mail names: FQDN of helper PC (e. g. ggpc1.domain.com)
- other incoming domains: faxrelay
- relay for: none
- relay IP: none
- root mail: none
- say “Yes” to save settings

Now the Debian installation is over and you can login as root using the chosen password.

## 2 Manual Installation

The following steps explain the installation parts which are not covered by the Debian installer.

### 2.1 Configure the Virtual Interfaces

Configure the system to **load the module for the network card** at startup by adding the module `e1000` to `/etc/modules` (if not done by the installer):

```
/etc/modules:
```

---

```
# /etc/modules: kernel modules to load at boot time.
#
# This file should contain the names of kernel modules that are
# to be loaded at boot time, one per line. Comments begin with
# a "#", and everything on the line after them are ignored.
e1000
```

---

**Configure the interface** in `/etc/network/interfaces` with the IP settings and the default gateway. Sub-interfaces can also be configured here.

```
/etc/network/interfaces
```

---

```
# /etc/network/interfaces -- configuration file for ifup(8),
ifdown(8)
```

```
# The loopback interface
auto lo
iface lo inet loopback
```

```
# The first network card - this entry was created manually
auto eth0
iface eth0 inet static
    address 10.0.0.232
    broadcast 10.0.0.255
    netmask 255.255.255.192
    gateway 10.0.0.254
```

```
auto eth0:1
iface eth0:1 inet static
    address 10.0.0.234
    broadcast 10.0.0.255
    netmask 255.255.255.192
```

```
auto eth0:2
iface eth0:2 inet static
    address 10.0.0.230
    broadcast 10.0.0.255
    netmask 255.255.255.192
```

---

### 2.2 Install and configure `bind9` name server

Installation a'la Debian:

```
# apt-get install bind9
```

The DNS server will be configured to accept requests only from certain clients, which will be all local services (e.g. the DNS proxy) and all other helper PCs. Furthermore, the IP address of the DNS server has to be configured.

The following shows the configuration of the helper PC 1 in the laboratory environment. The IP addresses here must be changed when the PCs will be installed in the production network.

```
/etc/bind/named.conf
```

---

```
options {
    ...
    // listen on the specified IP address
    listen-on { 10.0.0.234; };

    // allow requests from the following IP addresses
    allow-query { 10.0.0.231; localhost; localnets; };
    ...

    // for security reasons hide the version
    version "not available";
};
```

---

Restart the nameserver with the new configuration:

```
# /etc/init.d/bind9 restart
```

## 2.3 Configure the Helper PC to use the local bind9 as name server

The **hostname of the PC and the domain** will be configured in several files:

```
/etc/hostname for the name of the host
```

---

```
ggpc1
```

---

*/etc/resolver.conf* for the name servers and the search domain (typically the domain to which the host belongs)

---

```
# enter here the domain, e.g.
# search testlabs.enum.at
# your existing name servers
nameserver you.name.server.ip
nameserver you.name.server2.ip
# the local chaching only nameserver (for availability reasons)
nameserver 10.0.0.232
nameserver here.add.the.IPaddress.of.the.bind9.on.the.helper2pc
```

---

*/etc/hosts* includes the host name and resolves to the loopback device (127.0.0.1).

---

```
127.0.0.1    localhost
10.0.0.232   ggpc1.your.domain
```

```
# The following lines are desirable for IPv6 capable hosts
# (added automatically by netbase upgrade)

::1          ip6-localhost ip6-loopback
fe00::0     ip6-localnet
ff00::0     ip6-mcastprefix
ff02::1     ip6-allnodes
ff02::2     ip6-allrouters
ff02::3     ip6-allhosts
```

---

## 2.4 Installing Perl Modules

Install the following Perl modules:

```
# apt-get install libdigest-md5-perl libdbd-pg-perl libmime-perl
libmail-sendmail-perl libemail-valid-perl libmailtools-perl
```

The Debian woody distribution includes an old Net::DNS module (0.19-0.1) which does not support all needed features. Therefore we install the self packaged .deb package “libnet-dns-perl\_0.48-1\_i386.deb“ from the directory “libnet-dns-perl”.

```
# dpkg -i libnet-dns-perl_0.48-1_i386.deb
```

You could also compile the package yourself using the “dh-make-perl” application, which fetches the source code from CPAN and creates the Debian package (as described at: <http://perlmonks.thepen.com/133273.html>)

```
# apt-get install libtest-simple-perl
# apt-get install dh-make-perl
# dh-make-perl --build --cpan Net::DNS
# dpkg -i libnet-dns-perl_0.48-1_i386.deb
```

**Note:** You could also download the Net::DNS module manually from [www.cpan.org](http://www.cpan.org), enter the root directory of the un-packed archive and issue “dh-make-perl --build”.

## 2.5 Configuring Postgresql Database

**Note:** A mini-HowTo for postgresql can be found at:

[http://dev.panopticsearch.com/postgres\\_microhowto.html](http://dev.panopticsearch.com/postgres_microhowto.html)

The postgres database creates a Linux user called 'postgres'. Additionally it creates a db user called 'postgres'. There is no password for the Linux user postgres, therefore the Linux user postgres can not login, only root can **login as postgres via 'su'**.

as root:

```
# su - postgres
```

now, as user postgres we login to the database. The default password for the db user postgres is empty ("), therefore we **set a password**:

as user postgres:

```
postgres@ggpc1:~$ psql template1
Welcome to psql, the PostgreSQL interactive terminal.

Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help on internal slash commands
       \g or terminate with semicolon to execute query
       \q to quit

template1=# ALTER USER postgres WITH PASSWORD '780enum';
ALTER USER
template1=# \q
postgres@ggpc1:~$ exit
#
```

The next step is to **change the user authentication schema** of the DB from 'ident' to 'password'. As **root** edit the file `/etc/postgresql/pg_hba.conf` and change access rules (disable ident authentication and enable encrypted authentication):

Edit the line with access for the clients in the same subnet according to the production network.

```
/etc/postgresql/pg_hba.conf
```

---

```
#local  all                                ident sameuser
#host   all  127.0.0.1 255.0.0.0          ident sameuser
local   all                                crypt
host    all  127.0.0.1 255.0.0.0          crypt
host    all  10.0.0.192 255.255.255.192    crypt
host    all  0.0.0.0 0.0.0.0              reject
```

---

And now **restart the database** to accept the changes:

```
/etc/init.d/postgresql restart
```

Now, **create the database and the database user** for the accounting of the generic gateway. Log in as DB user postgres and create the database and the user:

```
ggpc1:~# psql template1 postgres
Password: 780enum
Welcome to psql, the PostgreSQL interactive terminal.

Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help on internal slash commands
       \g or terminate with semicolon to execute query
       \q to quit

template1=# create database ggacc;
CREATE DATABASE
template1=# create user gguser with password 'gguserpw';
CREATE USER
template1=# \q
```

```
ggpc1:~#
```

**Note:** You can find the SQL statements in the files **accounting.sql**, **fax.sql** and **grant.sql** in the folder **SQL-Files**.

Now, log in using the new ggacc database and **create the voice calls accounting table** and the **fax calls accounting table**:

```
ggpc1:~# psql ggacc postgres
Password: 780enum
Welcome to psql, the PostgreSQL interactive terminal.

Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help on internal slash commands
       \g or terminate with semicolon to execute query
       \q to quit

ggacc=> CREATE TABLE "accounting" (
ggacc(> "acctauthentic" varchar(512),
ggacc(> "acctdelaytime" varchar(512),
ggacc(> "acctinputoctets" varchar(512),
ggacc(> "acctinputpackets" varchar(512),
ggacc(> "acctoutputoctets" varchar(512),
ggacc(> "acctoutputpackets" varchar(512),
ggacc(> "acctsessionid" varchar(512) not null,
ggacc(> "acctsessiontime" varchar(512),
ggacc(> "acctstatustype" varchar(512),
ggacc(> "calledstationid" varchar(512),
ggacc(> "callingstationid" varchar(512),
ggacc(> "ciscoasport" varchar(512),
ggacc(> "nasipaddress" varchar(512),
ggacc(> "nasport" varchar(512),
ggacc(> "nasporttype" varchar(512),
ggacc(> "servicetype" varchar(512),
ggacc(> "timestamp" varchar(512) not null,
ggacc(> "username" varchar(512),
ggacc(> "ciscoh323callorigin" varchar(512),
ggacc(> "ciscoh323calltype" varchar(512),
ggacc(> "ciscoh323confid" varchar(512) not null,
ggacc(> "ciscoh323connecttime" varchar(512),
ggacc(> "ciscoh323disconnectcause" varchar(512),
ggacc(> "ciscoh323disconnecttime" varchar(512),
ggacc(> "ciscoh323gwid" varchar(512),
ggacc(> "ciscoh323remoteaddress" varchar(512),
ggacc(> "ciscoh323setuptime" varchar(512),
ggacc(> "ciscoh323voicequality" varchar(512),
ggacc(> "cavpacomlevel" varchar(512),
ggacc(> "cavpalerttimepoint" varchar(512),
ggacc(> "cavpcallid" varchar(512),
ggacc(> "cavpcallingpartycategory" varchar(512),
ggacc(> "cavpchargedunits" varchar(512),
ggacc(> "cavpcodebytes" varchar(512),
ggacc(> "cavpcodertyperate" varchar(512),
ggacc(> "cavpconnectprogress" varchar(512),
ggacc(> "cavpdisconnecttext" varchar(512),
ggacc(> "cavpearlypackets" varchar(512),
ggacc(> "cavpgapfillwithinterpolation" varchar(512),
ggacc(> "cavpgapfillwithprediction" varchar(512),
ggacc(> "cavpgapfillwithredundancy" varchar(512),
ggacc(> "cavpgapfillwithsilence" varchar(512),
ggacc(> "cavpgwfinalxlatedcgn" varchar(512),
ggacc(> "cavpgwrxdcdn" varchar(512),
ggacc(> "cavpgwrxdcgn" varchar(512),
```

```

ggacc(> "cavph323incomingconfid" varchar(512),
ggacc(> "cavph323ivrout" varchar(512),
ggacc(> "cavphiwaterplayoutdelay" varchar(512),
ggacc(> "cavpinfoftype" varchar(512),
ggacc(> "cavplatepackets" varchar(512),
ggacc(> "cavplogicalifindex" varchar(512),
ggacc(> "cavplostopackets" varchar(512),
ggacc(> "cavplowaterplayoutdelay" varchar(512),
ggacc(> "cavpnoiselevel" varchar(512),
ggacc(> "cavpontimervpplayout" varchar(512),
ggacc(> "cavppeeraddress" varchar(512),
ggacc(> "cavppeerid" varchar(512),
ggacc(> "cavppeerifindex" varchar(512),
ggacc(> "cavpreceivedelay" varchar(512),
ggacc(> "cavpreleasesource" varchar(512),
ggacc(> "cavpremotemediaaddress" varchar(512),
ggacc(> "cavpremotemediaudpport" varchar(512),
ggacc(> "cavpremotoudpport" varchar(512),
ggacc(> "cavproundtripdelay" varchar(512),
ggacc(> "cavpsessionprotocol" varchar(512),
ggacc(> "cavpsubscriber" varchar(512),
ggacc(> "cavptransmissionmediumreq" varchar(512),
ggacc(> "cavptxduration" varchar(512),
ggacc(> "cavpvadenable" varchar(512),
ggacc(> "cavpvoicetxduration" varchar(512),
ggacc(> CONSTRAINT "pk_accounting" PRIMARY KEY ("acctsessionid",
"acctstatustype", "calledstationid", "callingstationid", "timestamp",
"ciscoh323confid")
ggacc(> );
NOTICE: CREATE TABLE / PRIMARY KEY will create implicit index 'pk_accounting'
for table 'accounting'
CREATE
ggacc=>
ggacc=> CREATE TABLE "fax" (
ggacc(> "faxid" integer unique,
ggacc(> "timestamp" timestamp,
ggacc(> "fromnumber" varchar(512),
ggacc(> "fromnumber_fax" varchar(512),
ggacc(> "tonumber" varchar(512),
ggacc(> "toemail" varchar(512),
ggacc(> "status" varchar(512)
ggacc(> );
NOTICE: CREATE TABLE / UNIQUE will create implicit index 'fax_faxid_key' for
table 'fax'
CREATE
ggacc=>
ggacc=> CREATE SEQUENCE faxid;
CREATE
ggacc=> \p
CREATE SEQUENCE faxid;
ggacc=> \q
ggpcl:~#

```

Now allow the gguser to INSERT, UPDATE and SELECT from the accounting tables and the faxid sequence:

```

ggacc=# GRANT INSERT ON accounting TO gguser;
ggacc=# GRANT INSERT ON fax TO gguser;
ggacc=# GRANT INSERT ON faxid TO gguser;
ggacc=# GRANT UPDATE ON accounting TO gguser;
ggacc=# GRANT UPDATE ON fax TO gguser;
ggacc=# GRANT UPDATE ON faxid TO gguser;
ggacc=# GRANT SELECT ON accounting TO gguser;
ggacc=# GRANT SELECT ON fax TO gguser;
ggacc=# GRANT SELECT ON faxid TO gguser;
ggacc=> \q

```

```
ggpcl:~#
```

To **verify the accounting**, you have to login to the database and query the accounting table:

```
ggpcl:/etc/radiator# psql ggacc gguser
Password: gguserpw
...
ggacc=> select * from accounting;
ggacc=> select * from fax;
.....
```

The IP address, to which the DB binds can be configured with the `virtual_host` directive in `/etc/postgresql.conf`, but there can only one dedicated IP address to bind, e. g.:

```
#virtual_host = '127.0.0.1'
virtual_host = '10.0.0.232'
```

And now **restart the database** to accept the changes:

```
/etc/init.d/postgresql restart
```

You can verify that the DB binds to the correct interface using `netstat`:

```
# netstat -anp
```

This should show a line like:

```
tcp  0  0  10.0.0.232:5432  0.0.0.0:*  LISTEN  3998/postmaster
```

Also if all the components of the GG run on a the same PC, the communication should be configured to use the public IP address of the database server, not the loopback address. This will ease later the distribution of the components.

**Note:** If you have a lot of calls, the accounting tables will grow fast and querying the tables will take a lot of time. Therefore you should create indexes on some typical columns (phone numbers, date ...)

## 2.6 Installing Radiator

**Note:** This setup will use Radiator as Radius server. Radiator is a commercial software, but you could use any other Radius which allows execution of external scripts for authentication – e. g. `freeradius` and `xtradius`. Possibly you have to adopt the authentication script (which performs the enum lookup) and configure the Radius server for DB storage of the accounting records.

Nevertheless, you should install the generic gateway using a trial version of Radiator. Once the GG runs and you understand the functionality of the GG you can try to replace Radiator by another Radius server.

Alien will be used to transform the Radiator rpm package into a Debian package.

**Note:** The filenames of the Radiator packages may vary!

```
# apt-get install alien
# alien Radiator-Demo-3.9-2.noarch.rpm
```

Now install the Debian package:

```
# dpkg -i radiator-demo_3.9-3_all.deb
```

This will install the Radiator Perl files into `/usr/lib/perl5/site_perl/5.8.3/Radiator/`, which is not part of the Perl path, which will cause Radiator to fail working. Therefore, this directory must be moved to the “proper” location on Debian Woody systems:

**Note:** If the directory `/usr/local/lib/perl/5.6.1` does not exist, create it before moving the files:

```
# mkdir /usr/local/lib/perl
# mkdir /usr/local/lib/perl/5.6.1
```

Now move the files to the proper location:

```
# mv /usr/lib/perl5/site_perl/5.8.3/Radius /usr/local/lib/perl/5.6.1
```

Copy the radius configuration files, the authentication script and the radius hooks (`etc_radiator/*`) into the `/etc/radius` directory.

```
# cd etc_radiator
# cp -r * /etc/radiator
cp: overwrite `/etc/radiator/radius.cfg'? y
```

**Important:** The authentication-script must have execute permissions. If not, the Radius request is also successful, but no protocol indication (sip, h323, fax) will be returned to the gateway. Thus, set proper permissions:

```
# chmod a+x /etc/radiator/auth/auth-enum.pl
```

To define the IP address, the which a particular Radiator service should bind, the directive

```
BindAddress 10.0.0.234
```

must be used in the `/etc/radiator/radius.cfg` and `/etc/radiator/radius-auth.cfg` configuration file. Furthermore, the Radius clients and their secrets must be configured in the Radiator configuration files:

Set the IP address of the client in both configuration files to the IP address of the Cisco gateway and set the secret to the same secret as the Cisco GW is configured. If there are multiple Cisco GWs, then each of them must be configured as radius client in these configuration files.

```
<Client 10.0.0.231>
    Secret 780enum
</Client>
```

Now, the Radiator startup scripts must be installed by copying the radiator and radiator-auth files from `etc_init.d` into `/etc/init.d` and make them executable.

```
# cd ../etc_init.d/
# cp radiator* /etc/init.d/
cp: overwrite `/etc/init.d/radiator'? y
# chmod a+x /etc/init.d/radiator*
```

Furthermore, configure Debian to automatically start the Radiator processes.

```
# update-rc.d radiator defaults
# update-rc.d radiator-auth defaults
```

Now, (re)start the Radiator processes:

```
# /etc/init.d/radiator restart
# /etc/init.d/radiator-auth restart
```

**Note:** A detailed Radiator description is available at:  
<http://www.open.com.au/radiator/ref.html>

## 2.6.1 Radius Authentication Server and Perl Call Routing Script

### **Configuration**

Edit `/etc/radiator/auth/auth-enum.pl` if you want to **change the root-domain of the ENUM-tree**.

### **Maintenance**

The radius server logs requests and responses into the logfile `/var/log/radiator-auth/logfile`. The detail level of the logging can be configured in `/etc/radiator/radius-auth.cfg`.

The radius authentication server can be started/stopped by:  
`/etc/init.d/radiator-auth start|stop|restart`

## 2.6.2 Radius Accounting Server

Unsuccessful inserts into the database will be stored in SQL format in the file `/var/radiator/misssedaccounting`.

### **Configuration**

The configuration file (`/etc/radiator/radius.cfg`), the hooks (`/etc/radiator/hooks`) and the startup script (`/etc/init.d/radiator`) have to be installed. The files can be found in the directory `radiator+scripts`.

The accounting data are written into a postgresql database. Therefore, the database server, the database, the username and the password have to be configured in `/etc/radiator/radius.cfg`. The format is:

```
DBSource          dbi:Pg:dbname=NameOfTheDatabase;host=serverName
DBUsername        userName
DBAuth            userPassword
```

## 2.7 Installing procmail

Procmail will be used for local email delivery.

```
# apt-get install procmail
```

## 2.8 Installing debug tools

Tcpdump and ngrep will be used for packet sniffing and error debugging.

```
# apt-get install ngrep tcpdump
```

Typical usages:

Use ngrep to listen on all interfaces for DNS lookups and ICMP error messages

```
# ngrep -d any port 53 or icmp
```

Use tcpdump to write DNS, ICMP and radius authentication requests in the file debug.dump.

```
# tcpdump -i any -s 0 -w debug.dump port 53 or port 1545 or icmp
```

## 2.9 Install the ENUM.pm Perl Module

Get the ENUM module from <http://jprs.co.jp/enum/software/ENUM.pm> and locate it in the Perl path, i. e.: /usr/local/lib/perl/5.6.1.

```
# apt-get install wget
# cd /usr/local/lib/perl/5.6.1
# wget http://jprs.co.jp/enum/software/ENUM.pm
```

## 2.10 Configuring exim

The mail server must be configured to accept the emails from the gateway and deliver them to the local user *ifax*.

```
/etc/exim/exim.conf
```

---

```
#####
#                REWRITE CONFIGURATION                #
#####
*@ggpc1helper1    ${lookup{$1}lsearch{/etc/email-addresses}\
                  {$value}fail} frFs
*@faxrelay       ifax@ggpc1 T

end
```

---

If you have configured exim to run as standalone daemon, you have to (re)start the mail server. This step is not necessary if exim is run from inetd.

```
# /etc/init.d/exim restart
```

## 2.11 Install the sendfax script

The fax delivery will be performed as task of the local *ifax* user. The user can be created by (choose a password for the user):

```
# adduser --disabled-login ifax
```

This user can not login (for security reasons), therefore you have to “su” into this account.

```
# su - ifax
```

Next, create the `.procmailrc` file which delivers the fax emails to the sendmail fax. If the emails shouldn't be stored locally in the mailbox (pass-through only), the 'c' in the first line of the `.procmailrc` file has to be removed.

```
/home/ifax/.procmailrc  
:0 c  
|~/sendfax.pl
```

Copy the `sendfax.pl` script from the `home_ifax` folder into the home directory of the `ifax` user (`/home/ifax`). The exact destination of the script must be configured in the `.procmailrc` file. Furthermore, the access rights must be configured to allow the script to be executed.

```
$ chmod u+x sendfax.pl
```

Following data must be configured inside the `sendfax.pl` script:

- the database for the fax logging (server, database, user and password)
- the enum tree
- the format of the email (From, Envelope-from, Subject and the email body)

## 2.12 Installing the DNS proxy

The DNS proxy is implemented in Perl. As the UDP based query from the Cisco GW might miss some NAPTRs, the DNS proxy retrieves all NAPTRs using EDNS0 and TCP for lookup and returns only sip and h323 NAPTRs. Furthermore, the gateway rewrites the from new style (E2U+...) to Cisco old-style (...+E2U). This will reduce the number of NAPTRs and the risk of loss of NAPTRs. Furthermore the DNS proxy will send big UDP packets (fragmented) which will be accepted by the Cisco resolver.

Copy the perl script `dnsproxy.pl` from the directory `usr_local_bin` into `/usr/local/bin` and make the script executable:

```
# cp dnsproxy.pl /usr/local/bin/  
# chmod a+x /usr/local/bin/dnsproxy.pl
```

Copy the startup script from `etc_init.d` into `/etc/init.d` and make it executable:

```
# cp dnsproxy.pl /etc/init.d/  
# chmod a+x /etc/init.d/dnsproxy.pl
```

Configure the system to start the dnspoxy at startup:

```
# update-rc.d dnspoxy.pl defaults
```

Configure the startup script to tell the DNS proxy to which interface it should bind:

```
/etc/init.d/dnspoxy.pl  
...  
PARAM="-d -p 53 -i 10.0.0.230"  
...
```

**Note:** To find out all options of the DNS proxy execute `dnspoxy.pl -h`.

## 2.13 Time Synchronization

To keep the date and time on the helper PC up-to-date we install the ntpdate package.

```
# apt-get install ntpdate ntp-simple
```

If you want to use your own nameservers or have access to accurate nameservers configure them in `/etc/ntp.conf`.

**Note:** More info can be found on [http://www.togaware.com/linux/survivor/Using\\_NTP.shtml](http://www.togaware.com/linux/survivor/Using_NTP.shtml)

## 2.14 Manage Debian Distributions

Debian uses “stable” entries in `/etc/apt/sources.list` to refer to the current stable distribution. At the moment, the stable refers to “woody”. We want to avoid an automatic upgrade to “sarge” in the case of “sarge” becomes stable. Therefore, replace every occurrence of “stable” with the word “woody” in `/etc/apt/sources.list`.

# 3 Debugging

## 3.1 Services

To verify if a service is running use:

```
# ps -Af
```

To verify if the services binds to the proper interfaces use:

```
# netstat -anp
```

The following services must be started (on one or several PCs).

- Radius authentication server, e. g.:  
`/usr/bin/perl /usr/bin/radiusd -config_file /etc/radiator/radius-auth.cfg`
- Radius accounting server  
`/usr/bin/perl /usr/bin/radiusd -config_file /etc/radiator/radius.cfg`
- name server

```

root      172      1    0 11:33 ?          00:00:00 /usr/sbin/named
root      174     172    0 11:33 ?          00:00:00 /usr/sbin/named
root      175     174    0 11:33 ?          00:00:00 /usr/sbin/named
root      176     174    0 11:33 ?          00:00:00 /usr/sbin/named
root      177     174    0 11:33 ?          00:00:00 /usr/sbin/named
root      178     174    0 11:33 ?          00:00:00 /usr/sbin/named

```

- database server

```

/usr/lib/postgresql/bin/postmaster
postgres: stats buffer process
postgres: stats collector process
postgres: gguser ggacc 10.0.0.232 idle

```

- DNS proxy

```

/usr/bin/perl -wT /usr/local/bin/dnsproxy.pl -v -d -p 53 -i 10.0.0.230

```

- exim mail server (started via inetd)

```

/usr/sbin/inetd

```

## 3.2 Call Detail Records Analysis

The Gateway and the sendfax Perl script log to the database. A easy way to view this CDRs is the phppgadmin tool – a web based postgresql utility.

Therefore it is necessary to install a web server. We install apache-ssl to avoid sending login data in plaintext over the network.

```

# apt-get install apache-ssl
# apt-get install phppgadmin

```

Open `/etc/phppgadmin/config.inc.php` in an editor and configure phppgadmin to connect to the correct IP address:

```

$configServers[1]['host'] = '10.0.0.232';

```

Open the URL `https://10.0.0.232/phppgadmin/` in a web browser and login using `postgres/780enum`. Browse the *accounting* table for the GW accounting, and the table *fax* for the accounting.

Meaning of the *status* column in the *fax* table:

- received: the sendfax.pl script received an fax-email and begins the processing of this email
- sent: the sendfax.pl passed the new email to the local SMTP server, which accepted the email for delivery. **Note:** This does not mean, that the email was delivered to its destination

## 3.3 Debugging on the Generic Gateway

### 3.3.1 On the Cisco Gateway

terminal monitor

debug isdn q931

debug voice ivr

```
debug voice enum detail
```

```
debug radius ...
```

```
debug aaa ....
```

```
debug ccsip ....
```

```
debug cch323 ...
```

Details and help about the debug commands can be obtained by pressing the '?' key!

### 3.3.2 On the helper PC:

Use `ngrep` to listen on all interfaces for DNS lookups and ICMP error messages

```
# ngrep -d any port 53 or icmp
```

Use `tcpdump` to write DNS, ICMP and radius authentication requests in the file `debug.dump`.

```
# tcpdump -i any -s 0 -w debug.dump port 53 or port 1545 or icmp
```

Browse to `https://10.0.0.232/phppgadmin/` and login as `postgres/780enum`

Watch the logfiles:

```
/var/log/radiator/ (accounting)
/var/log/radiator-auth/ (auth+enum lookup)
/var/log/exim/ mailserver
/var/log/postgres.log
```

mit `su - ifax` als user `ifax` einloggen und `mutt` starten - dann seht ihr die kopien der emails die vom Cisco Gateway zum Helper PC kommen.

use `dig` to check if the NAPTRs are correct:

```
ggpc1:~# dig 1.0.0.4.0.8.7.3.4.e164test.labs.nic.at NAPTR

; <<>> DiG 9.2.1 <<>> 1.0.0.4.0.8.7.3.4.e164test.labs.nic.at NAPTR
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 11066
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL:
0

;; QUESTION SECTION:
;1.0.0.4.0.8.7.3.4.e164test.labs.nic.at.          IN NAPTR

;; ANSWER SECTION:
1.0.0.4.0.8.7.3.4.e164test.labs.nic.at. 120 IN NAPTR 100 11 "u"
"E2U+sip" "!.^.*$!sip:user1@10.0.0.214!" .

;; AUTHORITY SECTION:
```

```
e164test.labs.nic.at. 120 IN NS ns2.at43.at.
```

```
ggpc1:~#
```