EARSmilter Documentation

Release 1.0

Larry G. Wapnitsky

September 05, 2012

Contents

CHAPTER 1

What is the EARS Milter?

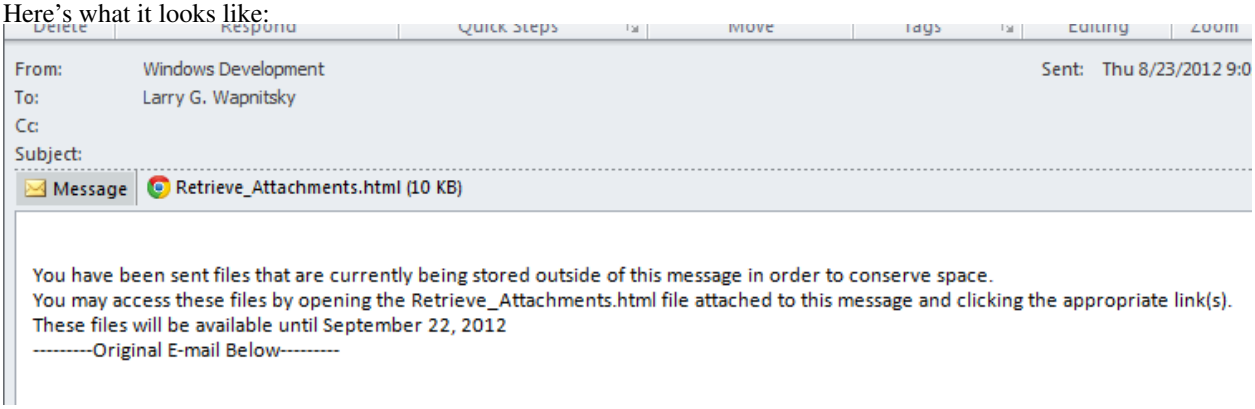
While wonderfully convenient, receiving files from clients and consultants through email has its drawbacks.

First there's the mailbox bloat. One 8MB email and wham!, your mailbox is full and you can't send. Then there's the PST file size limit to contend with. And if you need bigger files? Forgetaboutit! FTP? SendIT? There are ways around using email to receive files but the best of them aren't half as simple. Can't we just. . . .? Now, we can!

Introducing the *Email Attachment Removal/Retrieval System (EARS)*. Email attachment magic – automatically. Now, if someone outside the office sends you an email that exceeds a set threshold in size (currently 8MB), instead of being rejected, the message is passed to the Email Attachment Removal/Retrieval System. EARS saves any attachments larger than 160K to a temporary storage location, removes them from the email message and creates a new attachment named 'Retrieve_Attachments.html' before passing it on to your mailbox. The result is that you receive a nice small email with links to the original attachments that you can download and save to the location of your choice. You can even retrieve the attachments using your iPhone/iPad [1] or Outlook Web Access!

Incoming Email attachment size is no longer limited by WRT, only by the sender's email system.

Here's what it looks like:



Notice the subject line has been appended with "[Attachments Processed]". This tells you that the message was passed to the EARS system and attachments have been saved and removed. Also notice the 'Retrieve_Attachments.html' attachment – this is where you get your files back when you open it.

Open the 'Retrieve_Attachments.html' by double-clicking in Outlook and the page will open in your browser. Use the links to save your files to the location of your choice. But do it right away because the files will be deleted permanently (and un-retrievably) from the temporary location 30 days after the message was received whether you've saved them or not!

[1] iPhone/iPad compatibility is limited to files that are readable on your device based on installed apps.

EARS was conceived of by your IT team in response to direct feedback from you and was created by our one and only Larry Wapnitsky using FREE open-source software. No dollars were harmed during its development.

CHAPTER 2

EARS Milter Requirements

The EARS Milter is a Python-based mail-filter written for the postfix and sendmail mail transfer agents (MTAs).

Note: *Postfix* or *sendmail* must be installed on your system in order for this milter to function properly. It may work with any other MTA that supports milters but has only been tested with postfix and sendmail.

Note: *This milter has only been tested on Python 2.7 running on Debian Squeeze/Wheezy*

2.1 Server Requirements

The following software is required for the Milter to run:

- Debian Squeeze/Wheezy
- MySQL database
- Apache HTTP Server
- PHP 5.x
- postfix or sendmail
- Git - required to download the Milter from the development repository
- cifs-utils

2.2 Python Requirements

The following Python modules are required on the system in order for the EARS Milter to function:

- Python 2.7
- SQLAlchemy
- pymilter
- MySQLdb

- tnefparse
- mako

2.3 Recommended Software

While not necessary, this software should be installed as it will be referenced during the installation procedure:

- telnet
- rsyslog
- logrotate
- Webmin

2.4 Optional Software

The following software is optional, but is useful for diagnostics

- phpMyAdmin

CHAPTER 3

EARS Milter Installation

3.1 Server Installation

This milter has been tested on Debian Squeeze/Wheezy using a LAMP (Linux/Apache/MySQL/PHP)-based web server [1].

3.1.1 Debian Install

Installing a Debian Squeeze/Wheezy server is a fairly straight-forward procedure. The best method to use can be found via HowtoForge's The Perfect Server - Debian Squeeze (Debian 6.0) [ISPConfig 2] and following it up through **8 Change The Default Shell** on page 3.

Note: In WRT's environment, virtual linux servers are set up using Linux Containers (lxc).

To create a new system, log in to the LXC server as root and use the `lxc-prepare` script.

[1] Adapted from HowtoForge's Installing Apache2 With PHP5 And MySQL Support On Debian Squeeze (LAMP)

Make sure to edit `/var/lib/lxc/<machine name>/rootfs/etc/network/interfaces` and enter the appropriate static IP, routing and gateway information.

After logging in, make sure to install `anacron`, `cifs-utils`, `telnet`, `rsyslog` and `logrotate`.

```
aptitude install telnet rsyslog logrotate cifs-utils anacron
```

3.1.2 Web Server Installation

Note: Unless otherwise specified, all commands are run as the `root` user.

The web server needs to be set up in a LAMP configuration (Apache Web Server, MySQL, PHP)

1. Install MySQL

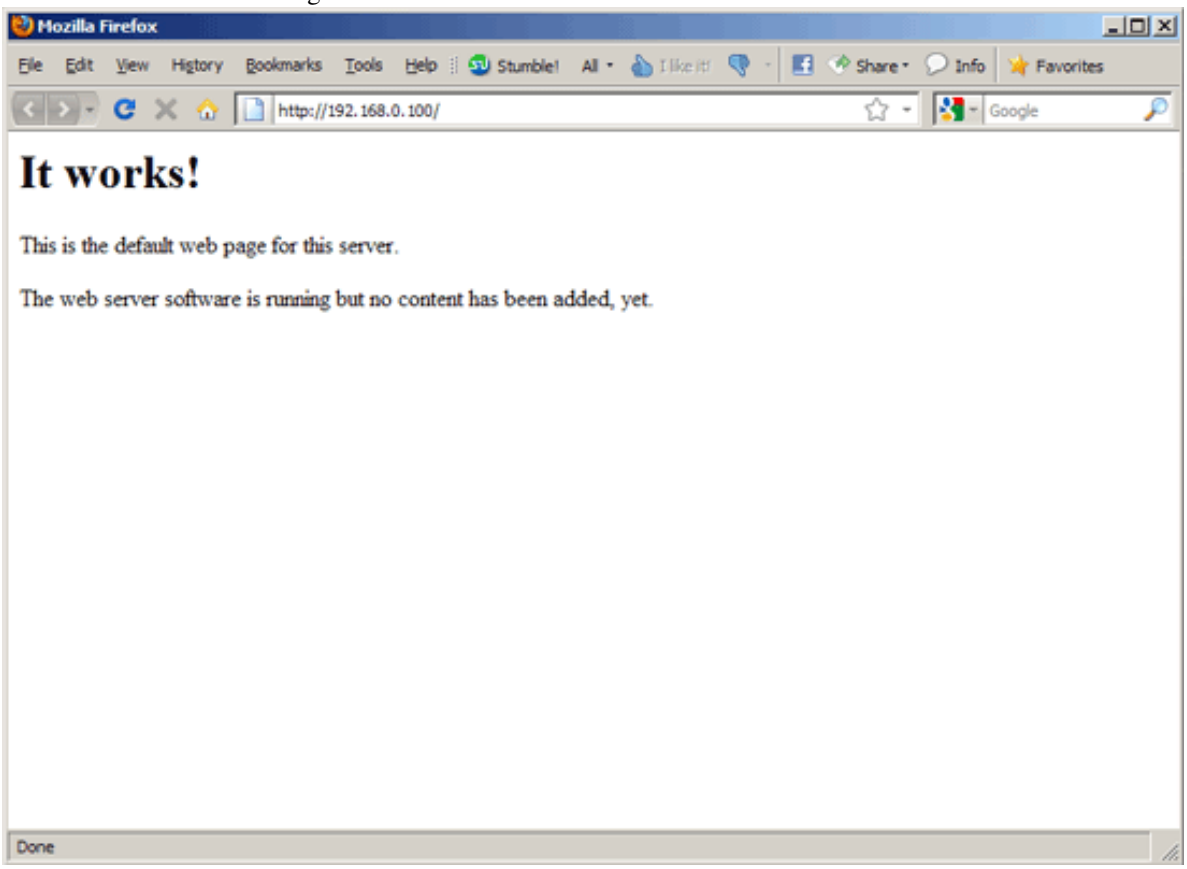
   ```
   % aptitude install mysql-server mysql-client
   ```

   Enter a password for the root MySQL user.

2. Install Apache2

   ```
   % aptitude install apache2
   ```

   Test to see that the web-server is running properly by visiting the IP address of this server in a web browser. You should see an image similar to this:



3. Install PHP5

PHP5 and the Apache PHP5 module are required to serve the EARS web-based code. Install them as follows:

```
% aptitude install php5 libapache2-mod-php5
```

Restart Apache:

```
% /etc/init.d/apache2 restart
```

4. Install MySQL support in PHP5

   To get MySQL support in PHP, we can install the *php5-mysql* package. It's a good idea to install some other PHP5 modules as well as you might need them for your applications.

   ```
   % aptitude install php5-mysql php5-curl php-pear php5-imagick \
      php5-mcrypt php5-memcache
   ```

   Note: You can search for available PHP5 modules like this:

   ```
   % apt-cache search php5
   ```

   Now restart Apache2:

   ```
   % /etc/init.d/apache2 restart
   ```

3.1.3 Git Installation

Git [2] is required to download the EARS Milter code from the development repository.

1. Install Git

   ```
   % aptitude install git
   ```

2. Configure git access

   - On the EARS Milter server, create a *ssh* key and copy it to the development repository server:

     ```
     ssh-keygen -t rsa
     ```

     Hit return at the prompts to create the key without passphrase authentication.

     ```
     % scp ~/.ssh/id_rsa.pub root@git:/root
     ```

   - Log in to the repository server and authorize the key:

     ```
     % ssh root@git
     % cd gitolite-admin
     % git pull
     % cp ~/id_rsa.pub keydir/root\@<milterservername>.pub
     % sed -i 's/\@.*$//g' keydir/root\@<milterservername>.pub
     % git add keydir/root\@<milterservername>.pub
     % git commit -a
     % git push
     % exit
     ```

   - On the EARS Milter server, test access to the repository server:

[2] Pro Git by Scott Chacon is available to read online for free.

```
% cd /tmp % git clone gitolite@git:gitolite-admin
```

If this fails, please verify all the steps in this section

3.1.4 Mail Server Installation

EARS requires an MTA. Please choose **either** postfix or sendmail.

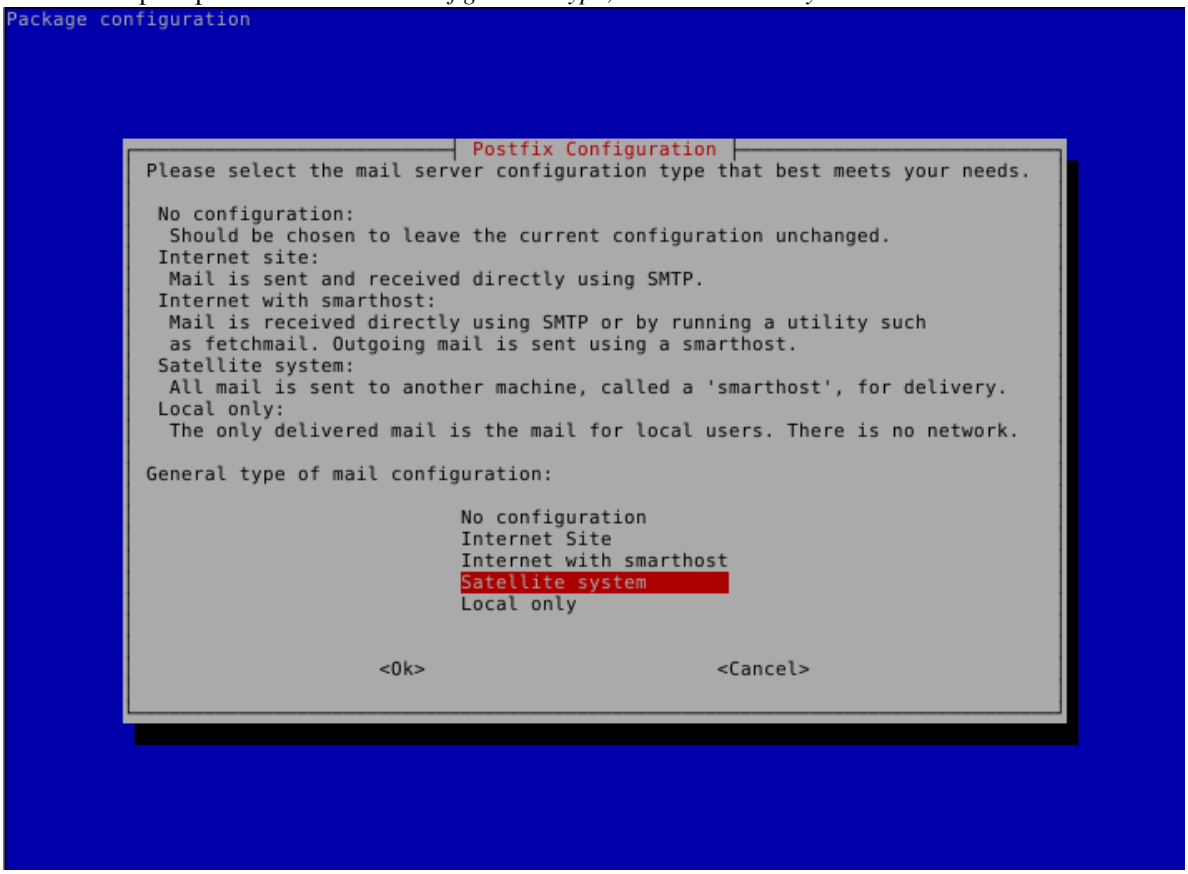
Postfix

1. Install postfix with PCRE support:

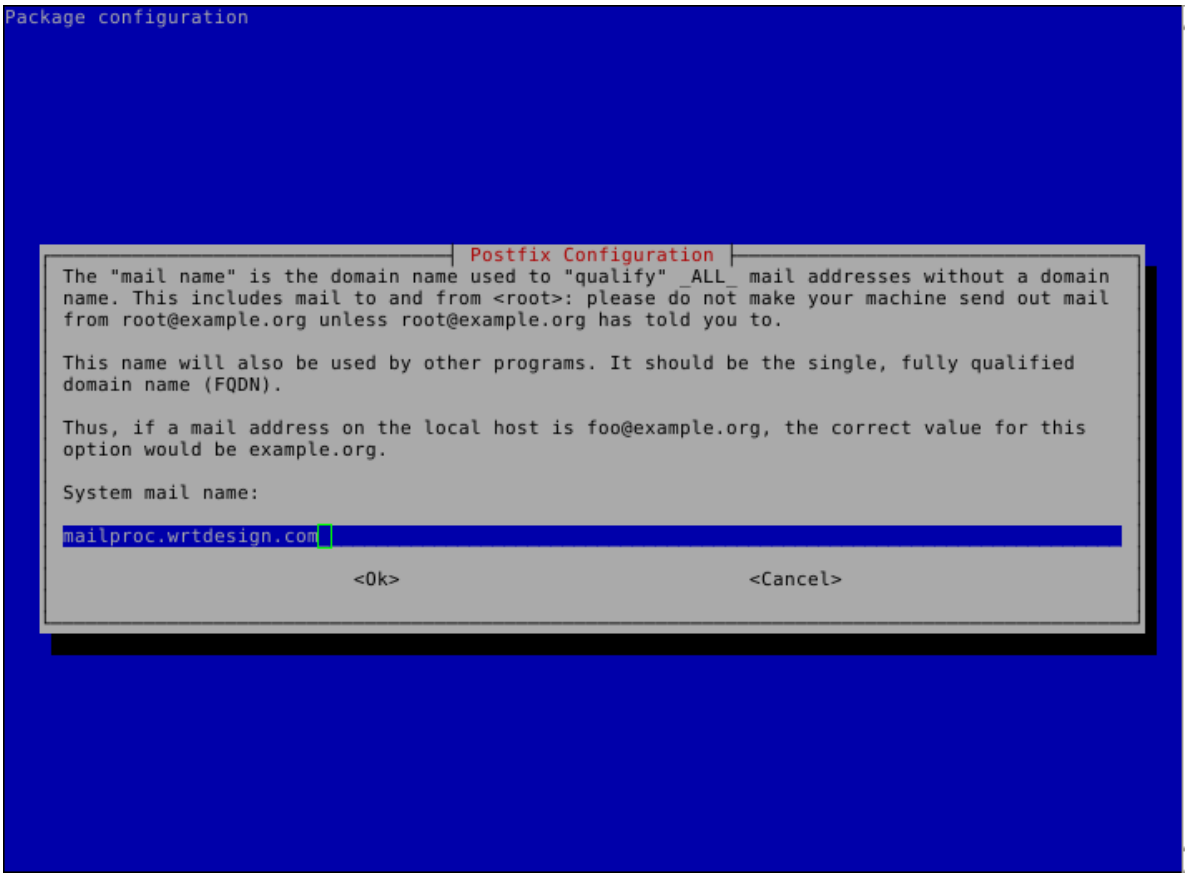
   ```
   % aptitude install postfix postfix-pcre
   ```

   If prompted to remove packages relating to `exim4` or `sendmail`, choose to *Accept the solution*.

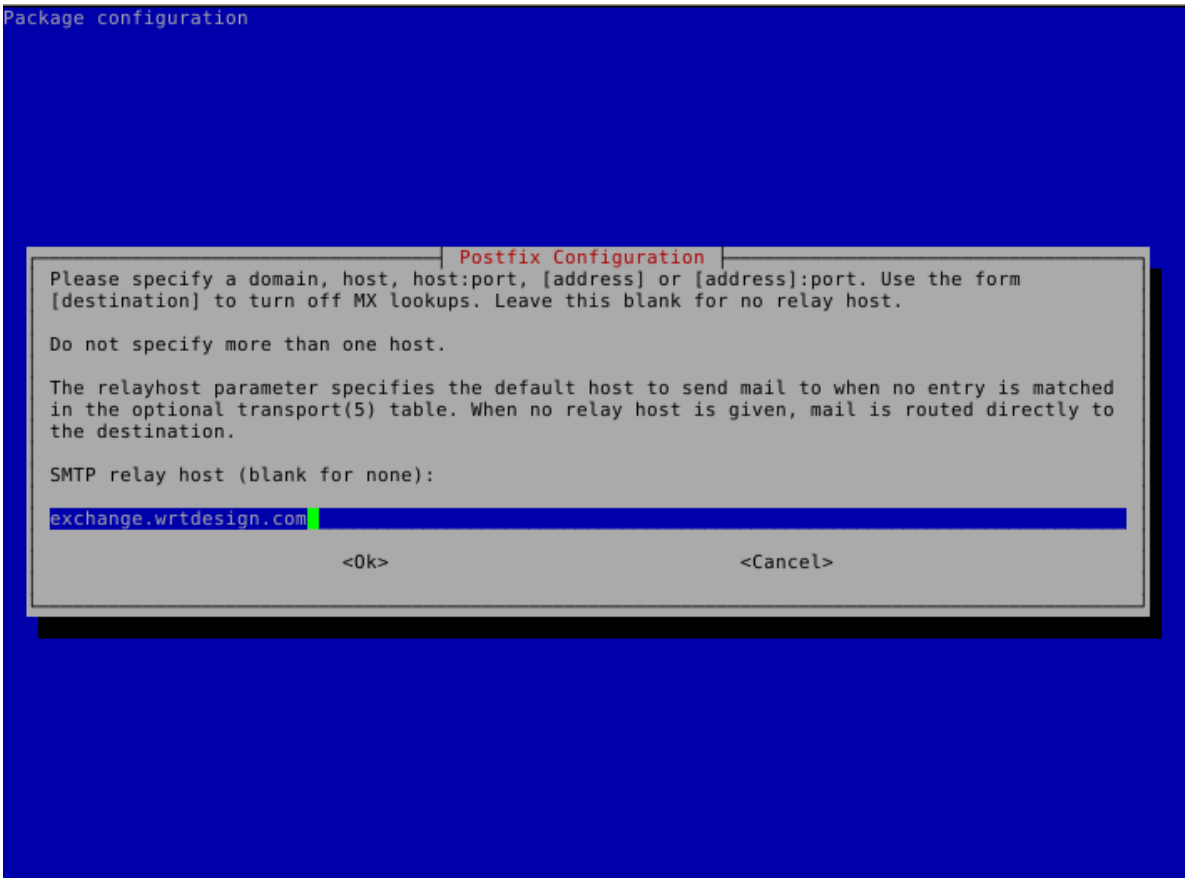
   When prompted for *mail server configuration type*, choose *Satellite System*:



   Enter a fully-qualified domain name in the form of *servername.wrtdesign.com*, where *servername* is the name of the EARS Milter server. Make sure that there is a DNS entry for this server and its corresponding IP address on the DNS server.



Enter the FQDN of the MS Exchange server when prompted for a relay host:



Accept the defaults for *Root and postmaster mail recipient*, *Other destinations to accept mail for* and *Force synchronous updates...*.

For *Local networks*, enter `10.102.0.0/16, 192.168.0.0/24, 127.0.0.1`. This will handle all of WRT's internal networks as well as the localhost.

Accept all the rest of the defaults.

2. Add the following lines to `/etc/postfix/main.cf`:

```
disable_vrfy_command = yes
smtpd_command_filter = pcre:/etc/postfix/bogus_commands
smtpd_recipient_restrictions = permit_mynetworks reject_unauth_destination
```

Remove the following line from the same file:

```
#inet_interfaces = loopback-only
```

Edit the following line to read:

```
inet_protocols = ipv4
```

3. Open up `/etc/postfix/master.cf` and uncomment the line:

```
#submission inet n       -       -       -       -       smtpd
```

Add the following lines (with indentation) to the same file:

```
scan      unix  -       -       n       -       10      smtp
 -o smtp_send_xforward_command=yes
 -o disable_mime_output_conversion=yes
 -o smtp_generic_maps=
```

Add the following (indented) after the line marked `relay`:

```
-o smtp_fallback_relay=
```

4. Create a file called `/etc/postfix/bogus_commands` and enter the following two lines:

```
/^[^ ]{3}\s.*/  NOOP
/^https{0,1}\:\/\/.*/ NOOP
```

5. Reload the configuration and send a test message:

```
% postfix reload
% telnet localhost 25
telnet localhost 25
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
220 ph-wks-lin01.wrtdesign.com ESMTP Postfix (Debian/GNU)
ehlo localhost
250-ph-wks-lin01.wrtdesign.com
250-PIPELINING
250-SIZE 10240000
250-ETRN
250-STARTTLS
250-ENHANCEDSTATUSCODES
250-8BITMIME
250 DSN
mail from: root
250 2.1.0 Ok
rcpt to: ph_test@wrtdesign.com
250 2.1.5 Ok
data
354 End data with <CR><LF>.<CR><LF>
test
.
250 2.0.0 Ok: queued as 4F00049F2A
quit
```

Sendmail

1. Install sendmail:

```
% aptitude install sendmail
```

If prompted to remove packages relating to `exim4` or `postfix`, choose to *Accept the solution*.

2. Open the file `/etc/mail/sendmail.mc` in an editor. Add the following lines above `MAILER DEFINITIONS`:

```
dnl # FEATURE('allmasquerade')dnl FEATURE('masquerade_envelope')dnl
FEATURE('accept_unresolvable_domains') FEATURE('accept_unqualified_senders')
define('SMART_HOST','exchange.domain.com')dnl
define('confDOMAIN_NAME','milterserver.domain.com') dnl #
```

Change these lines:

```
dnl
DAEMON_OPTIONS('Family=inet6, Name=MTA-v6, Port=smtp, Addr=::1')dnl
DAEMON_OPTIONS('Name=MTA-v4,Address=127.0.0.1,Family=inet,Port=smtp')
dnl
DAEMON_OPTIONS('Family=inet6, Name=MSP-v6, Port=submission, M=Ea, Addr=::1')dnl
DAEMON_OPTIONS('Name=MSP-v4,Address=127.0.0.1,Family=inet,Port=submission,Modifiers=aE')
```

to

```
dnl
DAEMON_OPTIONS('Family=inet6, Name=MTA-v6, Port=smtp, Addr=::1')dnl
DAEMON_OPTIONS('Name=MTA,Family=inet,Port=smtp')
dnl
DAEMON_OPTIONS('Family=inet6, Name=MSP-v6, Port=submission, M=Ea, Addr=::1')dnl
DAEMON_OPTIONS('Name=MSP,Family=inet,Port=submission,Modifiers=aE')
```

3. Open the file `/etc/mail/access`. Uncomment the following lines (according to your network):

```
Connect:10
RELAY GreetPause:10           0
ClientConn:10           OK
ClientRate:10           0
```

   Add additional lines for each FQDN of this IP address

4. Add all internal recipient domains to `/etc/mail/relay-domains`

   Example:

```
wrtdesign.com ph.wrtdesign.com
```

5. Recompile the `sendmail` files and restart the MTA and send a test message

```
% touch /etc/mail/access.new.db
% sendmailconfig
% telnet localhost 25
Connected to localhost.
Escape character is '^]'.
220 mailproc-test2 ESMTP
ehlo localhost
250-mailproc-test2 Hello localhost [127.0.0.1], pleased to meet you
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-EXPN
250-VERB
250-8BITMIME
250-SIZE
250-DSN
250-ETRN
250-DELIVERBY
250 HELP
mail from: root
250 2.1.0 root... Sender ok
rcpt to: ph_test@wrtdesign.com
250 2.1.5 ph_test@wrtdesign.com... Recipient ok
data
354 Enter mail, end with "." on a line by itself
test
.
250 2.0.0 q7VDE019017465 Message accepted for delivery
quit
```

3.2 Python Installation/Configuration

The default version of Python in Debian Squeeze/Wheezy is 2.7. This is what we will be installing, along with a Python package installer (pip) and some development libraries.

```
% aptitude install python python-pip python-dev libmilter-dev libmilter1.0.1  libmysqlclient-dev
```

Next we will need to install a number of Python modules. There are two ways to do this - the Debian way and the Python way. Each one has its advantages and disadvantages, but both are provided for instructional purposes.

The recommendation is to stick with one method instead of combining them.

3.2.1 The Python Method (preferred)

The Python Package Index (PyPI) is the most up-to-date resource for Python modules. Bugfixes and updates are regularly submitted for a majority of modules. The downside is that there is currenlty no way to automatically update the modules, but this can be considered a benefit as well since there is less chance of your code breaking.

```
% pip install SQLAlchemy pymilter MySQL-python Mako tnefparse
```

3.2.2 The Debian Method

Using debian's built-in package manager is very easy and convenient. When you do a full update on a Debian system, installed Python modules will be updated as well. The downside is that sometimes the modules in the Debian repositories can be out-of-date.

Here is the simple command to install the required modules, except for `tnefparse` which has to be installed via the Python method:

```
% aptitude install python-sqlalchemy python-milter python-mysqldb python-mako
```

3.3 Optional Software Installation

3.3.1 phpMyAdmin

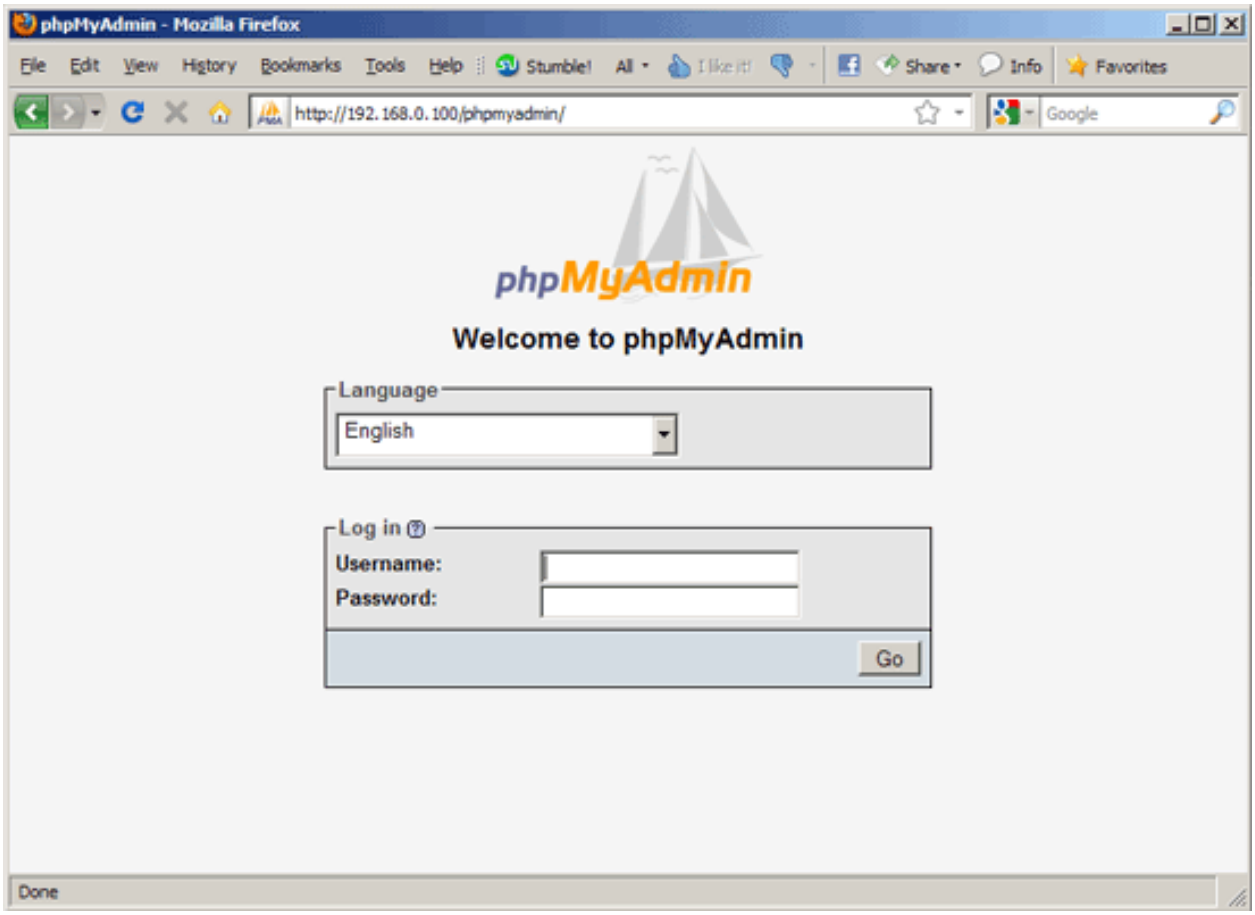
phpMyAdmin is a web interface through which you can manage your MySQL databases. It's a good idea to install it:

```
% aptitude install phpmyadmin php5-gd
```

You will see the following question:

```
Web server to reconfigure automatically: <– apache2
Configure database for phpmyadmin with dbconfig-common? <– No
```

Afterwards, you can access phpMyAdmin by going to `http://<serverIP>/phpmyadmin/`:



3.3.2 Webmin Installation

1. Create a file called `/etc/apt/sources.list.d/webmin.list`. Add the following lines, then save:

```
deb http://download.webmin.com/download/repository sarge contrib deb
http://webmin.mirror.somersettechsolutions.co.uk/repository sarge contrib
```

2. Download and install the security key, then update and install `webmin`:

```
% cd /root
% wget http://www.webmin.com/jcameron-key.asc
% apt-key add jcameron-key.asc
% aptitude update
% aptitude install webmin
```

3. Test the installation by going to `https://<serverIP>:10000`. Log in using the system root password.

3.4 Accquiring and configuring the Milter

1. Using *git*, clone **EARS** from the repository to the `/var/spool` folder:

```
% cd /var/spool % git clone gitolite@git:EARSmilter EARS % cd EARS
```

   (a) Copy EARS.sh to `/etc/init.d`. Make it executable and enable it at boot.

```
% cp /var/spool/EARS/EARS.sh /etc/init.d
% chmod +x /etc/init.d/EARS.sh
% update-rc.d EARS.sh enable defaults
```

2. Create a virtual host file for Apache in `/etc/apache2/sites-available/ears.conf` that contains the following (modify as necessary):

```
<VirtualHost *:80>
   ServerName ears.wrtdesign.com DocumentRoot /var/www/EARS
   Options -Indexes
</VirtualHost>
```

   Create a link to this file to make the site active:

```
% ln -s /etc/apache2/sites-available/ears.conf \
/etc/apache2/sites-enabled/ears.conf
```

   Create a folder called `/var/www/EARS`. Copy the files from `/var/spool/EARS/www` to this new folder and give Apache full rights to the folder. Restart Apache.

```
% mkdir -p /var/www/EARS % cp -R /var/spool/EARS/www/* /var/www/EARS
% chown -R www-data.www-data  /var/www/EARS
% chmod -x /var/www/EARS/*.php
% /etc/init.d/apache2 restart
```

3. Open the MySQL command-line utility

```
% mysql -u 'root' -p
```

   Create a blank database and associated MySQL user

```
mysql> CREATE DATABASE EARS; mysql> GRANT ALL PRIVILEGES ON
EARS.* TO "EARS"@"%" IDENTIFIED BY "password"; mysql> FLUSH PRIVILEGES; mysql> EXIT
```

   and change this line in `/etc/mysql/my.cnf`:

```
bind-address = 127.0.0.1
```

   to:

```
bind-address = 0.0.0.0
```

4. Set the appropriate permissions on `/var/spool/EARS` and its subdirectories based on the MTA installed.

   Postfix

```
% chown -R postfix.postfix /var/spool/EARS
```

   Sendmail

```
% chown -R smmta.smmta /var/spool/EARS
```

   You will also need to edit `/etc/init.d/EARS.sh` and replace **postfix** with **smmta**.

```
% sed -i.bak 's/postfix/smmta/g' /etc/init.d/EARS.sh
```

5. Create the log files:

```
% touch /var/log/EARSmilter.log
% touch /var/log/EARSmilter.err
% chmod 666 /var/log/EARSmilter.log
% chmod 666 /var/log/EARSmilter.err
```

6. Add/edit the following lines to the configuration file for the appropriate MTA:

Postfix - `/etc/postfix/main.cf`

```
milter_protocol = 6
smtpd_milters = unix:/var/spool/EARS/EARSmilter.sock milter_default_action = accept
```

Reload postfix - `postfix reload`

Sendmail - `/etc/mail/sendmail.mc`.

```
INPUT_MAIL_FILTER('EARS', 'S=unix:/var/spool/EARS/EARSmilter.sock, F=T, T=S:240s;R:240s;E:5m')dnl
```

Recompile the `sendmail` files and restart the MTA

```
% touch /etc/mail/access.new.db % sendmailconfig
```

Note: If/when you add additional milters to this sytem, make sure that **EARS** is the last one listed, as milters are processed in order.

---

1. Edit the database information in `/var/spool/EARS/EARSmilter/EARSmilter.py` with what is appropriate for your environment.

   This is listed under `EARSmilter.EARSmilter.milter.eom()`

   ```
   db = toDB( 'EARS', 'password', 'localhost', 'EARS' )
   ```

2. Create a folder called `/dropdir` :

   ```
   % mkdir -p /dropdir
   ```

   Mount it to a folder called `dropdir` on a FTP server. This example assumes the folder is on a MS Windows box:

   ```
   % echo '\\FTPSERVER\FTPSHARE\dropdir  /dropdir  cifs  workgroup=DOMAIN, \
      file_mode=0777,dir_mode=0777,password=PASSWORD,uid=1000,gid=1000, \
      username=USERNAME 0  0' >> /etc/fstab % mount -a
   % mount -a
   ```

3. Start the EARS milter:

   ```
   /etc/init.d/EARS.sh start
   ```

4. Add a `cron` job to run the `purgeEARSdb` script on a weekly basis.

   ```
   % crontab -e
   ```

   Add this line:

   ```
   /usr/bin/env python /var/spool/EARS/purgeEARSdb/purgeEARSdb.py -s \
      milter.domain.com -d EARS -u EARS -p password -q -x -v#purge EARS database
   ```

   For a description of the options, see **How To Use** under `purgeEARSdb()`.

1. Add the following lines to `/etc/logrotate.conf`:

   ```
   /var/log/EARSmilter.* {
     compress copytruncate
   }
   ```

CHAPTER 4

EARS Code

4.1 EARS.py

The main startup file for the EARS milter

EARS.**background**()
: This function starts the background threading of EARS so that multiple processes can handle incoming messages.

EARS.**main**()
: The main startup for the EARS milter.

Milter factory flags are set so that the milter can:

- Change the body of the message
- Change the headers of the message
- Add headers to the message
- Add/Delete recipients

4.2 EARS Milter main class/functions

EARS milter class definition and mail processing functions

- *EARSlog*
- *milter*
- *ProcessMessage*
- *FileSys*

4.2.1 EARSlog

class EARSmilter.EARSmilter.**EARSlog**
: Establishes logging for different error statuses: INFO, WARN, DEBUG, ERR

4.2.2 milter

class EARSmilter.EARSmilter.**milter**(*Milter.Base*)

Milter processing derived from pymilter.

Only functions that have been heavily modified from standard Milter processing have been documented

@Milter.noreply
connect(*IPname*, *family*, *hostaddr*)

Initializes when a new connection to the Milter is made via SMTP

@Milter.noreply
header(*name*, *hval*)

Processes headers from the incoming message and writes them to a variable for database storage

```
rgxSubject = re.compile( '^(subject)', re.IGNORECASE | re.DOTALL )
rgxMessageID = re.compile( '^(message-id)', re.IGNORECASE | re.DOTALL )
```

Regular Expression searches used to extract and log the Subject and Message ID of incoming messages

eom()

End-Of-Message milter function

Connection to the database is established here via the toDB class:

```
db = toDB( 'root', 'python', 'python.dev.wrtdesign.com', 'EARS' )
```

The variables above can be changed accordingly.

New messages are initialized in the database via:

```
db.newMessage( self.canon_from, self.Subject, self.headers, self._msg, self.R )
```

and are subsequently shipped to the ProcessMessage class to parse/remove attachments, then add the EARS link page to the message before being sent to the recipient(s)

If an error occurs, an Exception is thrown and logged to the <logname>.err file

4.2.3 ProcessMessage

class EARSmilter.EARSmilter.**ProcessMessage**(*_db*, *_log*, *_from*)

All message processing is done via this class.

- Attachments are parsed/removed
- Winmail.DAT files are extracted
- Retreive_Attachments.html file is added to original message
- Message information is stored in the MySQL database

__init__(*_db*, *_log*, *_from*)

- **_db** - toDB class
- **_log** - EARSlog class
- **_from** - sender's e-mail address

subChange is set to *False*. If no attachments are removed, **subChange** will be set to *True* and "Removed Attachments" will be removed from the subject line

ParseAttachments()
The message is passed through this function in order to analyze and potentially remove attachments

If the attached file is WINMAIL.DAT, it is sent for further processing so that the files it contains can be extracted and provided to the recipient(s)

Attachments are then analyzed for size and, if are above the set threshold, are detached.

If no attachments are extracted, the folder created in the **dropDir** is removed.

Otherwise, the "Retrieve_Attachments.html" notice is created and appended to the original message along with any attachments that remained below the threshold

extract_attachment(*data*, *fname*)
Used to extract attachments that are NOT in a WINMAIL.DAT file.

Filenames are unicode-corrected.

winmail_parse(*fname*)
If a WINMAIL.DAT file is attached to this message, parse and attempt to extract the files it contains.

For more information on WINMAIL.DAT and TNEF files, please see Transport Neutral Encapsulation Format

delete_attachments(*part*, *notice*)
For each attachment that needs to be removed from the message, edit the headers of the message and add in the "Retrieve_Attachments.html" file.

This only needs to be run once to avoid multiple copies of the HTML file being attached

mako_notice(*fnames*)
The "Retrieve_Attachments.html" file is generated.using the Mako Template Engine for Python.

The template is stored in the *www* subfolder of the main EARS milter folder. The CSS and image files should be stored on an externally-accessible web server.

Each filename is encoded in Unicode and parsed to be valid for URLs (hence, special characters like ampersands and percent signs can be part of valid filenames and still downloaded properly).

4.2.4 FileSys

class EARSmilter.EARSmilter.**FileSys**(*msg*)

__init__(*msg*)
msg - Message passed directly from the Milter

Variables that are established for this class are:

- **dropDir** - Base folder where attachment files will be stored
- **min_attach_size** - Minimum attachment size to detach from the message
- **remfile** - Name of the file with links to removed attachments

Dir()
Creates a unique folder in the **dropDir** based on the hash of the whole incoming message

hashit(*data*)
Generates a hash for **dropDir** folders and individual files

filesize_notation(*filesize*)
Determines proper filesize notation (K, M, G) based on mathematical calculation loop

unicodeConvert (*fname*)
Convert text to unicode.

This was designed to address issues with Latin-1 (iso-8859-1) encoding in addition to other foreign characters.

4.3 purgeEARSdb.py

EARS Database Purge

class purgeEARSdb.purgeEARSdb.**Options**
Establish the command-line options for purgeEARSdb.py

Note: For command-line usage, see *How to use*.

---

class purgeEARSdb.purgeEARSdb.**Purge**(*options*)

purgeAttachmentsMessages()

```
query = session.query( Attachment ).filter( Attachment.received <= delta ).all()
```

Searches attachments based on provided delta time (default of 7 days)

```
for q in query:
    for message in q.message:
        if message.dateReceived <= delta:
```

Only deletes messages older than the specified delta.

purgeSenders()
Designed to remove senders with no assocated messages from the database

query_yes_no(*question*, *default='yes'*)
Ask a yes/no question via raw_input() and return their answer.

"question" is a string that is presented to the user. "default" is the presumed answer if the user just hits <Enter>.

It must be "yes" (the default), "no" or None (meaning an answer is required of the user).

The "answer" return value is one of "yes" or "no".

4.3.1 How to use

Proper usage of purgeEARSdb.py

```
usage: purgeEARSdb.py [-h] [-q] [-v] [-D DAYS] [-H HOURS] [-M MINUTES] [-x] -s
                      SERVER -d DATABASE -u USERNAME -p PASSWORD

Purge old messages/attachments from EARS database

optional arguments:
  -h, --help            show this help message and exit

 -q, --quiet           Run without prompting
  -v, --verbose         Verbose output
```

```
Message/Attachment age options:
  -D DAYS, --days DAYS  Remove message older than x days (default = 7)
  -H HOURS, --hours HOURS
                        Remove message older than x hours (default = 0)
  -M MINUTES, --minutes MINUTES
                        Remove message older than x minutes (default = 0)

Sender Options:
  -x, --purge-senders   Purge senders from database that are no longer
                        associated with messages/attachments

Database Options (required):
  -s SERVER, --server SERVER
                        MySQL server name
  -d DATABASE, --database DATABASE
                        MySQL database name
  -u USERNAME, --username USERNAME
                        MySQL username
  -p PASSWORD, --password PASSWORD
                        MySQL password
```

Example **cron** job setup:

```
@daily /usr/bin/env python /var/spool/EARS/purgeEARSdb/purgeEARSdb.py -s mailproc.wrtdesign.com -d EARS -u EARS -
```

This setups a daily job to run the script and remove non-associated senders from the database using the default of 7 days

4.4 Database Definitions and Functions

4.4.1 SQLAlchemy.py - SQLAlchemy Database Definitions

```
from sqlalchemy.dialects.mysql import
        BIGINT, BINARY, BIT, BLOB, BOOLEAN, CHAR, DATE,
        DATETIME, DECIMAL, DECIMAL, DOUBLE, ENUM, FLOAT, INTEGER,
        LONGBLOB, LONGTEXT, MEDIUMBLOB, MEDIUMINT, MEDIUMTEXT, NCHAR,
        NUMERIC, NVARCHAR, REAL, SET, SMALLINT, TEXT, TIME, TIMESTAMP,
        TINYBLOB, TINYINT, TINYTEXT, VARBINARY, VARCHAR, YEAR
```

This import is necessary in order to define the specific field type for a MySQL database

database.SQLAlchemy.**mr_link = Table('mr_link', MetaData(bind=None), Column('recipient_id', Integer(), ForeignKey(**
Special table used to create relationships between messages and recipients/attachments. This table is never referenced specifically in the code other than in the table definition for :py:class:Message. Links are generated automatically.

database.SQLAlchemy.**ma_link = Table('att_link', MetaData(bind=None), Column('file_id', Integer(), ForeignKey('attac**
Special table used to create relationships between messages and recipients/attachments. This table is never referenced specifically in the code other than in the table definition for `Message`. Links are generated automatically.

class database.SQLAlchemy.**Attachment**(*filename*, *received*, *hash*, *data*)
Attachments are stored as binary blobs in the MySQL database for later retrieval

__init__(*filename*, *received*, *hash*, *data*)
Attachment data requires a filename, the date the attachment was received, the filehash and the binary data

class database.SQLAlchemy.**Message**(*subject*, *headers*, *body*, *dateReceived*, *raw_original*)
Messages contain automatically generated relationships to attachments and recipients via the `mr_link` and `ma_link` table definitions.

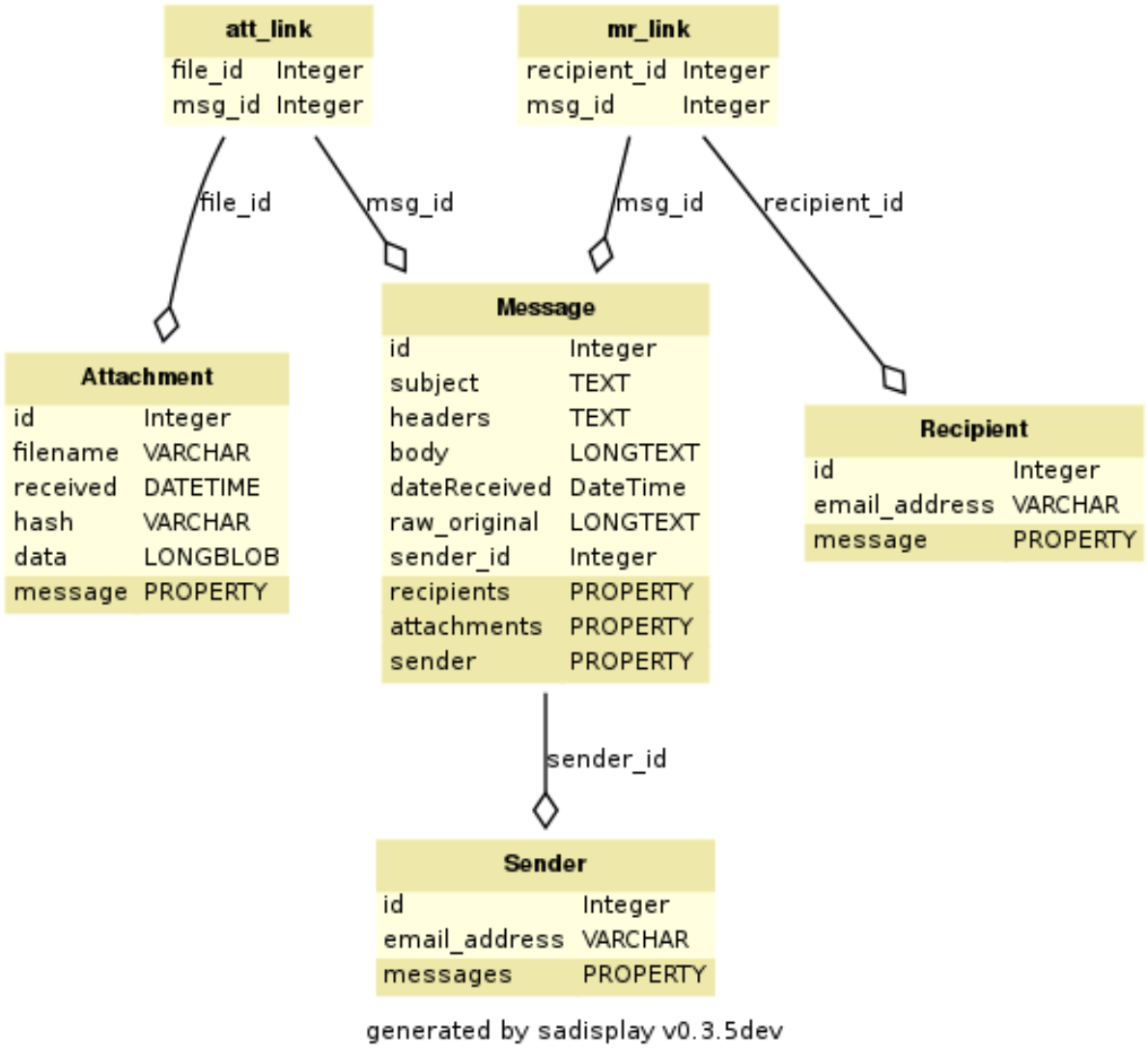
__init__(*subject*, *headers*, *body*, *dateReceived*, *raw_original*)
Message data requires a subject, the headers, message body, the date the message was received and a raw copy of the oricinal message

class database.SQLAlchemy.**Recipient**(*email_address*)
Recipient table

__init__(*email_address*)
A valid email address must be provided for the recipient

class database.SQLAlchemy.**Sender**(*email_address*)
Sender table

__init__(*email_address*)
A valid email address must be provided for the recipient



4.4.2 toDB.py - EARS Milter Database Population

class database.toDB.**toDB**(*username*, *password*, *server*, *database*)
This class holds all the functions to add/update new messages and attachments to the EARS database.

__init__(*username*, *password*, *server*, *database*)
To initialize the EARS database population class, you must provide a valid username, password, server and database names.

The database itself must exist prior to the first execution of EARS, but this class will automatically generate the necessary tables.

Currently, EARS is configured only to work with MySQL databases.

newMessage(*sender*, *subject*, *headers*, *raw_original*, *recipients*)
When a message is submitted to EARS, existing sender and recipient e-mail addresses are checked for in the databse. If they do not exist, they are added automatically.

addAttachment(*filename*, *filehash*, *data*)
When attachments are submitted to EARS, the filehash is compared with existing files in the database. If the file already exists, the datetime stamp is updated so that the attachment does not get deleted sooner than it should be when **purgeEARSdb.py** is run.

close()
Commits the message and its attachment(s) to the database

4.5 Customized Python Logging

Logger.py is a customized Python logging script that can be used in other applications

4.5.1 logger.py

class logs.logger.**logger**(*lname*)
logger class

__init__(*lname*)
Constructor - requires a filename prefix.

Logs are stored in /var/log. If the log files do not exist, or the permissions are incorrect for the executing user, you will receive the following message:

```
Please make sure that the log files exist by running the following commands as root:

    touch /var/log/EARSmilter.log
    touch /var/log/EARSmilter.err
    chmod 666 /var/log/EARSmilter.log
    chmod 666 /var/log/EARSmilter.err
```

start()
Starts the logger

CHAPTER 5

Indices and tables

- *genindex*
- *search*
- *modindex*

Index

Symbols

A

B

C

D

E

F

H

L

M

N

O

P

Q

R

S

T

U

W