Email archiving with piler

This documentation applies to version 0.1.24 of piler

Third Edition

Publication date Oct 4, 2013

Table of Contents

What is piler?

piler is an advanced open source (GNU GPL v3 licensed) email archiving solution with all the necessary features for your enterprise.

Email Archiving	Archives all incoming, outgoing and internal emails	✓
	Define email archiving rules	✓
	Attachment de-duplication	✓
	Message de-duplication	✓
	Archive compression & encryption	✓
	Live and scheduled archive backup	✓
Email Discovery	Search entire company, user emails	✓
	Easy search for beginners	✓
	Save and restore search queries	✓
	Search inside attachments	Word, Powerpoint, Excel, RTF, PDF, ZIP and OpenOffice attachments
	Import PST folders	✓
	Export emails en bulk to file	✓
	Restore emails en bulk to mail server	✓
	View emails and header information	✓
Email Compliance	Define retention policies	✓
	Tagging	✓
	Legal Hold (protect emails from being deleted by retention policies)	✓
	Digital signing and verification	✓
Web Console	Access to archive from web console	✓
	Active Directory authentication	✓
	LDAP, Google, IMAP authentication	✓
	Windows single sign-on authentication	✓
	Fine grained access control	✓
	Skinnable web based GUI	✓
	GUI can be detached from the archive	✓
System Interfaces	Inbuilt SMTP server	✓
	Commandline shell (SSH based)	✓
Monitoring	Search through detailed audit logs	✓
	Real-time status info	✓
	Real-time status charts	✓
MS Exchange	Multiple Exchange servers	✓

Import/Export	EML, MBOX, Maildir, PST, IMAP, POP3	✓
	Google Apps, Office 365	✓
	Export to .EML	✓
Performance	Fast search within seconds	✓
Security	Passwords hashed & encrypted	✓
	Support for STARTTLS	✓
	TCP_WRAPPERS support	✓
Other	Multiple language support (i18n is possible)	en, de, pt, hu
	Easy setup & configuration	✓
	Use of open standards, no proprietary formats	✓
	Appliance mode	✓
	Comprehensive documentation	✓
ISP / SaaS support	Multitenancy for SaaS providers	✓
Mail Server/s	Postfix, Sendmail, Qmail, Lotus Notes, iredmail, Zimbra, and others	anything that can forward a copy via SMTP
	Microsoft Exchange 2003, 2007, 2010, and 2013	✓
	Google Apps	✓ (the free edition is also supported)
	Office 365	✓
Operating Systems	Linux, Solaris	✓
	Tried and tested on VMware	✓
	VMware appliance (OVA)	✓

Benefits of email archiving

There are several reasons to archive your emails:

Business continuity and disaster recovery

piler gives you a secure central repository of emails providing the necessary information even if your mail servers are down.

Regulatory and compliance requirements

Governmental agencies and other regulatory organizations require you to preserve emails for several years. To meet these obligations piler saves your emails in its long term storage.

Legal discovery and investigations

piler helps you to provide relevant information in a timely manner in case of legal discovery, audit or other events.

Storage management

piler saves storage space by compression and smart deduplication. It also allows you to offload data from the mail server to piler, thus minimizes storage costs.

Your company has lots of knowledge and data in tons of email messages. A proper email archiving solution gives you the following benefits:

- fast response to eDiscovery requests
- compliance
- reduce mail server load
- save storage space
- protect against disaster

Installing piler

Download piler from http://www.mailpiler.org/en/download.html

Prerequisites

Mandatory

- openssl[1]
- MySQL 5.1+[2]
- sphinx search 2.0.7+[3]
- php 5.3.x+[4]
- a rewriting rule capable webserver, eg. apache[5], lighttpd[6], nginx[7], …
- TRE regex library[8]
- libzip
- iconv

Highly recommended

- memcached[9]
- xcache[10]

Optional

- clamav 0.97.x[11]
- libpst[12]

Preinstall tasks, recommendations

Since piler archives your emails it may need lots of disk space. To keep things tidy, I recommend you to put all piler related data (including your log files) under the /var partition. I strongly recommend using logical volumes.

1 http://www.openssl.org/
2 http://www.mysql.com/
3 http://sphinxsearch.com/
4 http://www.php.net/
5 http://httpd.apache.org/
6 http://www.lighttpd.net/
7 http://www.nginx.net/
8 http://laurikari.net/tre/
9 http://www.memcached.org/
10 http://xcache.lighttpd.net/
11 http://www.clamav.net/
12 http://www.five-ten-sg.com/libpst/

You may create the following layout:

```
/     - 2 GB
/tmp  - 128 MB
/var  - several hundreds of GBs (or even more)
swap  - 512 MB
```

The used filesystem is not a concern, however if you choose *XFS*, then be aware of a 32 bit vs. 1 TB issue[13].

Create a dedicated and non-privileged user to run piler

```
groupadd piler
useradd -g piler -s /bin/sh -d /var/piler piler
usermod -L piler
```

Unpack, compile and install piler

```
tar zxvf piler-0.1.24.tar.gz
cd piler-0.1.24
./configure \
        --localstatedir=/var \
        --with-database=mysql \
        --enable-starttls \
        --enable-tcpwrappers

make
su -c 'make install'
```

Hint: use gmake on FreeBSD.

Execute **ldconfig** as root to make sure that *libpiler.so* is recognised. On some systems you may need to add /usr/local/lib to /etc/ld.so.conf.d/piler.conf.

13 http://xfs.org/index.php/XFS_FAQ#Q:_Why_do_I_receive_No_space_left_on_device_after_xfs_growfs.3F

Postinstall tasks

After installing piler a series of postinstall tasks must be done. Execute the following command as *root*:

```
make postinstall
```

The postinstall utility asks a series of questions, then creates a database for piler, generates an encryption key, copies sphinxsearch configuration file, sets up some cron jobs, and finally it copies the gui directory.

To run the postinstall script, execute the following command *as root*:

```
make postinstall
```

IMPORTANT! Never run the postinstall utility if you upgrade! It's designed to use at a fresh install only.

If you prefer to do the postinstall tasks manually, then refer to http://www.mailpiler.org/en/post-install.html for further instructions.

The postinstall script makes an encryption key for piler, eg. /usr/local/etc/piler.key.

IMPORTANT! Make sure you never lose/overwrite the key otherwise you won't access your archive ever again. So whenever you upgrade be sure to keep your existing key file.

Starting sphinx search daemon and piler

Start both searchd and piler:

```
sh init.d/rc.piler start
sh init.d/rc.searchd start
```

You may copy these files to /etc/init.d and add them to the list of your scripts to start, eg. chkconfig, etc.

Create a dedicated virtual host for piler

This task depends greatly on what webserver you use. Basically you have to create a virtualhost dedicated to piler, and enable rewriting rules. The piler GUI expects to be in the DocumentRoot not in a subdirectory.

An example *Nginx* (eg. /etc/nginx/sites-available/piler) and *Apache* (eg. /etc/apache2/sites-available/piler) configurations are available in the contrib/webserver/ directory.

Customize piler.conf

After installing piler you may customize the piler configuration file, *piler.conf*.

Pay attention the "*mysqlsocket*" parameter, it may require you to adjust it to the correct socket path of your system.

STARTTLS support

If you need STARTTLS support, then set the following in piler.conf:

```
tls_enable=1
pemfile=/usr/local/etc/piler.pem
```

Then create a certificate and key for STARTTLS:

```
openssl genrsa 2048 > /usr/local/etc/piler.pem
chmod 600 /usr/local/etc/piler.pem
openssl req -new -key /usr/local/etc/piler.pem > 1.csr
openssl x509 -in 1.csr -out 1.cert -days 3650 -req -signkey /usr/local/etc/piler.pem
cat 1.cert >> /usr/local/etc/piler.pem
```

Finally restart piler:

```
/etc/init.d/rc.piler restart
```

TCP_WRAPPERS support

Piler supports tcp_wrappers to prevent archiving emails from unknown sources. To enable it just edit /etc/hosts.allow, and set the following:

```
piler: 192.168.1.100: ALLOW
sshd: your.client.ip.here: ALLOW
ALL: ALL: DENY
```

Then if anything other than 192.168.1.00 tries to connect to piler, it will get the following error:

```
telnet piler.yourdomain.com 25
Trying 192.168.1.100...
Connected to piler.yourdomain.com.
Escape character is '^]'.
550 You are banned by local policy
Connection closed by foreign host.
```

You can see the following in syslog:

```
Oct 29 10:35:32 archivehost piler[23336]: connection from
123.123.123.123
Oct 29 10:35:32 archivehost piler[23336]: denied connection from
123.123.123.123 by tcp_wrappers
```

Antivirus and antispam support

Piler supports clamd to scan for viruses in incoming emails. Whenever a virus is found, then it's dropped and piler syslogs the event.

If you need clamd support then recompile piler with –-enable-clamd, and set the clamd_socket parameter in piler.conf.

Piler itself doesn't have any antispam features, however if you use any antispam application, set the *spam_header_line* parameter in piler.conf, then piler can recognize when it receives a spam. If you use *SpamAssassin*, then you may use the following:

```
spam_header_line=X-Spam-Level: ********
```

However for performance reasons it's better if you filter for both viruses and spam on your MX servers, and forward only clean messages to piler.

Configure your mail server to forward a copy of each email it receives or sends to piler

Postfix

Add the following to *main.cf* then issue the *postfix reload* command:

```
always_bcc = archive@piler.yourdomain.com
```

Zimbra

Zimbra uses postfix internally, so you may use the postfix approach: find the zimbra version of main.cf, and apply the same as for postfix.

Office 365

Configure networking, and make sure that Office 365 can reach piler on port 25. This may require setting some firewall rules, etc.

Create a Journaling rule

1. Login to Office 365 as an administrator
2. Select "*service settings*"
3. Select "*mail*"
4. Select "*recipients*", then "*contacts*", then click on the Plus sign, and select "*Mail contact*"
5. Enter "*piler*" to First name, enter "*piler*" to Display name, enter "*piler*" to Alias, and enter "archive@piler.ypurdomain.com" as External email address.
6. Select "mail flow", then "rules", then click on the Plus sign, and select "*Create a new rule...*"
7. Enter "archiving" to Name, select "*[Apply to all messages]*" at "Apply this rule if...", select "*Bcc the message to ...*" at "Do the following", then select the contact named "piler" at the popup window coming up. Finally select "*Enforce*" at "Choose a mode for this rule", and click save.

Exchange

Basically you have to create SMTP journaling in Exchange, find the details on the different

Exchange vrsions below.

Note that you may have to force the GUI to rewrite the *Message-id* header field preventing Exchange to discard the restored message as a duplicate. Edit config-site.php, and set the following:

```
$config['REWRITE_MESSAGE_ID'] = 1;
```

Exchange 2003

Create SMTP Contact

1. Create an SMTP Contact
2. Set Use MAPI rich text format to Never
3. Enable Automatic Forwarding to allow automatic forwarding
4. Set Exchange rich-text format to Never use
5. Open Exchange System Manager (ESM)
6. Expand Administrative Groups, First Administrative Group (or appropriate group), Servers
7. Expand your server name
8. Expand First Storage Group (or relevant group)
9. Right click on the target mailbox store, select Properties
10. Check the option Archive all messages sent or received by mailboxes on this store
11. Click Browse
12. Select the name of the user account/mailbox that has been created as the journal mailbox
13. Click OK to finish
14. Restart the SMTP service

Exchange 2007

Create SMTP Contact

1. Open Active Directory Users and Computers.
2. Right-click the organizational unit in which you want to create the contact, point to New, and then click Contact.
3. Enter the following: First Name: *Journal1*, Last Name: *Archive*, Display Name: *Journal1 Archive*.
4. Click OK.
5. Open the Exchange Management Console on the Mailbox server.
6. Expand Recipient Configuration, right-click Mail Contact, and then click New Mail Contact.
7. Click Existing Contact, browse to and select the *Journal1 Archive* contact you just created, then click OK.
8. Click Next.
9. For the External Email Address field, click Edit, enter the archive email address, eg. *archive@piler.yourdomain.com*, then click OK. *piler.yourdomain.com* must be FQDN. i.e. it must be resolvable on Exchange server, eg. you should be able to ping it.
10. Click Next, then click New.

Configure Message Format Settings

1. Open your Exchange Management Console.
2. Expand Recipient Configuration, then select Mail Contact.
3. In the result pane, select the SMTP contact.
4. In the action pane, under the SMTP contact, click Properties.
5. On the General tab, from the Use MAPI rich text format list, select Never.

With this setting, journal reports are sent in MIME rather than S/TNEF.

Enable Standard Journaling

1. Open the Exchange Management Console on the Mailbox server on which you want to enable journaling.
2. Expand Server Configuration, then click Mailbox.
3. In the result pane, select the server for the mailbox database for which you want to enable journaling.
4. In the work pane, right-click the mailbox database, then click Properties.
5. On the General tab, select Journal Recipient.
6. For Journal Recipient, click Browse, select the *Journal1 Archive* recipient, then click OK.
7. Click OK

Exchange 2010

Create SMTP Contact

1. Click Start > All Programs > Microsoft Exchange Server 2010 > Exchange Management Console
2. Select the Microsoft Exchange On-Premises instance
3. Select Recipient Configuration
4. Below the Recipient Configuration, select Mail Contact
5. In the Action pane to the right, select New Mail Contact
6. In the create mail contact dialog that appears, select New Contact
7. Enter "journal" for the new contact name and fill out the rest of the required information.
8. Select Edit aside the new contact's external e-mail address
9. A dialog appears prompting for an external email address. Enter the address "*archive@piler.yourdomain.com"*
10. After entering the address, click OK. After returning back to the new contact dialog, select Next
11. Select new to create the New Mail Contact
12. After clicking New, the New Mail Contact will be created.
13. Click Finish to return to the Exchange Management Console.

Enable Standard Journaling

1. Click Organization Configuration, thereafter Mailbox
2. In the Database Management tab to the right, select the database for which journaling must be enabled.
3. Right click on the database and select properties
4. A new window appears. Select the maintenance tab and then select Browse
5. Click the Journal Recipient and then browse to the New Mail Contact created previously
6. Click OK

Create a Send Connector

1. Open the Send Connector tab under Server Management->Hub Transport option
2. Right-click in the empty space and select new Send Connector
3. Ensure intended use field is set to custom. Enter "*Journal1"* in name field then click "Next."
4. Click "Add" in the Address Space Window.
5. In the address field, enter the FQDN of piler, eg. *piler.yourdomain.com*
6. Keep the default setting to use DNS to route email to the address space and click "Next."
7. Depending on your specific case, you may select "Add" to select another Hub Transport Server
8. Click "Next" once the Hub Transport Server has been added
9. Verify that all of the settings are correct. Click "New" then "Finish" once it is complete.
10. The Send Connector should now be listed.

Exchange 2013

Setup Journaling

1. Login to the Exchange Control Panel by opening the browser at https://exchange.yourdomain.com/ecp (where *exchange.yourdomain.com* is the FQDN of your Exchange server)
2. From the Exchange Admin Center
3. Click compliance management in left menu
4. Click journal rules in top right menu
5. Click the + icon
6. Enter journal in the name field
7. In the field that says "journal the following messages..", select "All messages"
8. In Send journal reports to field, enter archive@piler.yourcompany.com
9. Click Save to change changes
10. Accept "Do you want this rule to apply to all future messages"

Create a Send Connector

1. Select mail flow and then send connectors
2. Click + to add a new send connector
3. Enter 'MailArchiva Connector' in the Name field.
4. Select Custom for the Type option
5. Click Next
6. Under Network settings, select Route mail to smart host
7. Click + to add a new smart host
8. Enter the FQDN of the piler server (e.g. *piler.yourdomain.com*)
9. Click Save
10. The new host should be listed under Smart Host in the New Send Connector Window
11. Click Next
12. Select None for the Smart host authentication option
13. Click Next
14. Click + to add a new address space
15. Enter *piler.yourdomain.com*
16. Click Save.
17. The FQDN should be listed under Address in the New Send Connector Window (e.g. *piler.yourdomain.com*)
18. Click Next
19. Click + to add mail server(s)
20. Select the mail server from the list and then click add. Repeat this step for any additional servers
21. Click ok. The mail servers should be listed under Source in the New Send Connector Window.
22. Click finish

Adjust Maximum Message Size

By default, the maximum send message size for the Connector is set to 10 MB. To change this:

1. Open the Exchange Management Shell
2. Enter the following command to the set maximum send message size: Set-SendConnector "Piler Connector" -MaxMessageSize "100 MB"
3. Enter the following command to verify the maximum send message size is 100 MB: Get-SendConnector "Piler Connector" fl MaxMessageSize

Disable Non Delivery Reports (optional)

1. Open the Exchange Management Shell.
2. Enter the following command to disable NDRs
3. Set-RemoteDomain <Domain> -NDREnabled $false (replace <Domain> with the FQDN of the piler server, e.g.piler.yourdomain.com

Using the GUI at the first time

Note that the web based piler GUI stores its configuration in config.php. You shouln't edit config.php, rather put every local site-specific settings to config-site.php. The settings in config-site.php overrides the defaults in config.php.

Log in as administrator using the following account, then change its password immediately:

```
admin@local:pilerrocks
```

Add your domains you archive emails for in the Administration/Domains menu. When you are done, make sure to restart or reload piler by clicking on the "*Apply settings*" red button.

The next thing is to introduce your users to the gui, see *User authentication methods* in the documentation.

Before using piler it's a good idea to establish archiving and retention policies. See *Rules and policies* in the documentation.

If you want to use https on the GUI, then edit config-site.php, and make sure you set https in the SITE_URL variable:

```
$config['SITE_URL'] = 'https://piler.yourdomain.com/';
```

User authentication methods

The piler GUI supports several methods for authenticating users.

Authenticating against the local piler database

Normally only special users exist in the local piler database, eg. admin@local, auditor@local, etc. and the regular users are queried from an external database, eg. LDAP, AD, etc.

It's possible to add manually the regular users manually, however I recommend it as a last resort when a central user database is not available.

Authenticating against an LDAP server

This is the preferred way to authenticate regular users. The details depend on what mail system you use. In general you have to enable the LDAP authentication (see the following paraggraph), then create a helper account that can query the LDAP directory.

```
$config['ENABLE_LDAP_AUTH'] = 1;
$config['LDAP_HOST'] = 'ldaphost.yourdomain.com';
// set this if you want to limit the scope of the ldap query
$config['LDAP_BASE_DN'] = '';
// the helper account, see the details below
$config['LDAP_HELPER_DN'] = 'cn=...';
$config['LDAP_HELPER_PASSWORD'] = 'xxxxxxx';
$config['LDAP_MAIL_ATTR'] = 'mail';
```

Active Directory

```
$config['LDAP_ACCOUNT_OBJECTCLASS'] = 'user';
$config['LDAP_DISTRIBUTIONLIST_OBJECTCLASS'] = 'group';
$config['LDAP_DISTRIBUTIONLIST_ATTR'] = 'member';
```

iredmail

```
$config['LDAP_HELPER_DN'] = 'cn=vmailadmin,dc=yourdomain,dc=com';
$config['LDAP_BASE_DN'] = 'o=domains,dc=yourdomain,dc=com';
$config['LDAP_ACCOUNT_OBJECTCLASS'] = 'mailUser';
$config['LDAP_DISTRIBUTIONLIST_OBJECTCLASS'] = 'mailList';
$config['LDAP_DISTRIBUTIONLIST_ATTR'] = 'memberOfGroup';
```

Lotus Domino

```
$config['LDAP_ACCOUNT_OBJECTCLASS'] = 'dominoPerson';
$config['LDAP_DISTRIBUTIONLIST_OBJECTCLASS'] = 'dominoGroup');
$config['LDAP_DISTRIBUTIONLIST_ATTR'] = 'mail';
```

Zimbra

```
$config['LDAP_ACCOUNT_OBJECTCLASS'] = 'zimbraAccount';
$config['LDAP_DISTRIBUTIONLIST_OBJECTCLASS'] =
'zimbraDistributionList';
$config['LDAP_DISTRIBUTIONLIST_ATTR'] =
'zimbraMailForwardingAddress';
$config['LDAP_HELPER_DN'] = 'uid=zimbra,cn=admins,cn=zimbra';
```

Note that the gui grants regular user permissions for everyone authenticated successfully against LDAP. If certain users need auditor rights, then create a group in LDAP, and put the auditors to this group. Then set this value to LDAP_AUDITOR_MEMBER_DN in config-site.php:

```
// the value is case sensitive!
$config['LDAP_AUDITOR_MEMBER_DN'] =
'CN=PilerAuditors,CN=Users,DC=your,DC=domain,DC=com';
```

Authenticating against an IMAP server

It's also possible to authenticate users against your IMAP server. Set the following in config-site.php:

```
$config['ENABLE_IMAP_AUTH'] = 1;
```

```
$config['IMAP_HOST'] = 'imap.yourdomain.com';
$config['IMAP_PORT'] =  993;
$config['IMAP_SSL'] = true;
```

Authenticating against Google Apps free edition

Register with Google

Visit the Google API console (https://code.google.com/apis/console), and register a "*Client ID for web applications*". You have to set the "*Redirect URIs*", and the "*JavaScript origins*" fields, and you are free to customise it, eg. adding your logo.

```
Redirect URIs: http://piler.yourdomain.com/google.php
JavaScript origins: http://piler.yourdomain.com/
```

Configure the GUI

Open config-site.php, and enable Google Apps support:

```
$config['ENABLE_GOOGLE_LOGIN'] = 1;
```

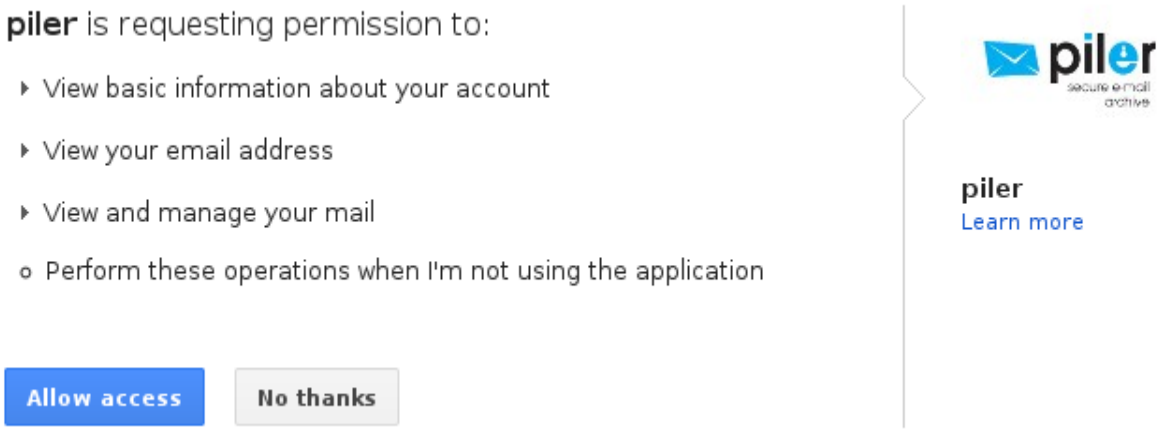
Also, set the following definitions (ie. replace the xxxxx...x characters) with the generated values on the API console:

```
$config['GOOGLE_CLIENT_ID'] = 'xxxxxxxxxxx';
$config['GOOGLE_CLIENT_SECRET'] = 'xxxxxxxxxxxxx';
$config['GOOGLE_DEVELOPER_KEY'] = 'xxxxxxxxxxxx';
```

Using the first time

Now have your users to visit your piler installation at http://piler.yourdomain.com/.

They will find a link saying "Login via Google account". Click on the link, enter your google account info to *Google*, then a notification is shown that "piler is requesting permission to ...". Let them accept it. Then they are redirected back to your site.



Add cron entries

The first time they won't see anything, since piler just got access to their emails. To enable the polling via IMAP, set the following crontab entry for user piler:

```
1 * * * * (cd /var/www/piler.yourdomain.com; /usr/bin/php
/usr/local/libexec/piler/gmail-imap-import.php .)
```

The cron job above will download emails to /var/piler/imap. The results are syslogged to maillog. Now we need another job to process these emails:

```
45 * * * * /usr/local/bin/pilerimport -d /var/piler/imap -r -q
```

This will import the downloaded emails to piler, then remove them. The importer script remembers the id of the last downloaded message, so it won't download from the first message when it runs again.

Additional notes

If you have several GBs of email in Google Apps Free Edition, then it takes time to download them all, so don't enable the cronjobs until the first run of the imap importing is done.

You can track the progress by viewing the google_imap table where piler puts helper beacons. It's sufficient to keep the freshest few hundreds or so per email addresses.

Single Sign On (SSO)

If your users login to a Windows network with Active Directory, then it's possible to login with single sign-on. It means that your browser negotiates your authentication credentials with the server running piler. The following example shows how to setup SSO on Ubuntu.

Install mod_auth_ntlm_winbind

```
apt-get install libapache2-mod-auth-ntlm-winbind
cd /etc/apache2/mods-enabled
ln -sf ../mods-available/auth_ntlm_winbind.load
```

Edit /etc/samba/smb.conf

```
[global]

workgroup = YOURDOMAIN
realm = YOURDOMAIN.YOURREALM
security = ADS
```

Restart winbind

```
/etc/init.d/winbind restart
```

Add the www-data user to the winbindd_priv group

```
uid=33(www-data) gid=33(www-data) groups=33(www-
data),125(winbindd_priv)
```

Restart apache

```
apache2ctl restart
```

Enable ntlm negotiation within the browser, and add the piler website

```
Firefox:
about:config
```

network.negotiate-auth.trusted-uris

Internet Explorer:
Tools
Internet Options
Security
Local Intranet
Sites

Last step: set the following in config-site.php

$config['ENABLE_SSO_LOGIN'] = 1;

Then whenever your users visit http://piler.yourdomain.com/, then they are redirected to sso.php, and logged in automatically, then redirected to the search page.

If anything goes wrong, then be sure to set "LogLevel debug" in apache to see what's going on.

Overview of how piler works

Piler is a secure email archive that keeps your messages in a central repository. It archives every email it receives, no matter if it's a spam or a user's private message. Whenever piler receives a message, then it parses, strips the attachments, deduplicates, encrypts the files it's about to store, and finally it acknowledges the delivery to the smtp sender.

Piler relies on heavily on an SQL server. Currently only mysql is supported. Piler stores all relevant information, such as metadata, attachment list, etc. in SQL tables. When the message is parsed then piler puts the parsed contents of the email to an SQL table for later indexing.

The message indexing is not real time. It's run from the crontab every half hour. It means that you cannot see your messages instantly in the GUI as soon as they are archived. It takes some time until they appear in the search hits.

When piler archives a message a few checksums are made to ensure that any tampering with the message will be reported to the users.

The GUI has a built-in access control to prevent a user to access other's messages. Auditors can see every archived email. Piler parses the header and extracts the From:, To: and Cc: addresses (in case of From: it only stores the first email address, since some spammers include tons of addresses in the From: field), and when a user searches for his emails then piler tries to match his email addresses against the email addresses in the messages. To sum it up, a regular user can see only the emails he sent or received.

This leads to a limitation: piler will hide an email from a user if he was (only) in the Bcc: field. This limitation has another side effect related to external mailing lists. You have to maintain which user belongs to which external mailing lists, otherwise users won't see these messages. Internal mailing lists are not a problem as long as piler can extract the membership information from openldap or Active Directory.

Fortunately both Exchange and postfix (and probably some other MTAs, too) are able to put envelope recipients to the email header, so the limitation mentioned above is solved.

Piler keeps track of your own domains you archive emails for, it's called mydomains. It's a similar concept to the mydomains variable of postfix. It defines a list of domains that you archive emails for. You can manage these domains in the GUI in the Administration / Domain menu.

If you want piler to archive only the emails in mydomains, then set the following in piler.conf, then reload piler:

archive_only_mydomains=1

Thus piler will drop every email where non of the sender or the recipients don't belong to mydomains.

Searching for emails

Users can access the archived emails using a browser. They have to authenticate themselves with any of their known email address and a password.

Quick search

Users can use two kinds of search interfaces. The default is the quick search, a long text input field. Just enter the search terms, and you get the last 1000 hits in a paged view. 1000 is a sphinx limit, so when you see that your search query has 1000 hits, it really means 1000+. The 1000 number can be incremented at the price of some performance penalty (edit sphinx.conf, and set your choice, then edit config-site.php and set $config['MAX_SEARCH_HITS'] to the same value)

Users may enter the search terms into a text field, and the web interface splits them into components, guesses the format of the components, and builds up a search query. If you type 2012-01-31 (2012.01.31 is also a valid date) then it knows it's a date. If it has a @ sign, then it's an email address. Otherwise it's a subject or body expression.

If you want to enter a finer search query, then you should use keywords that precisely identify the type of the given search word. You may specify the following keywords or fields:

from:	sender address
to:	recipient address
subject:	subject of the message
body:	body of the message
date1:	from ('not before') date (YYYY-MM-DD 00:00:00)
date2:	to ('not after') date (YYYY-MM-DD 23:59:59)
size:	size of the message in bytes
direction:	direction of the message, possible values: inbound, outbound, internal
d:	same as direction
attachment:	attachment type, possible values: word, excel, powerpoint, pdf, compressed, text, odf, image, audio, video, flash, other, any
a:	same as attachment

Quick search examples

Email from Gmail before 2012.02.29 00:00:00:

```
date2: 2012-02-28, from: @gmail.com
```

Email from the Matrix:

```
from: Agent Smith
```

Email to someone in Big company after 2012.01.31 00:00:00:

```
date1: 2012-01-31, to: @bigcompany.com
```

Email from jane@aaa.fu OR bill@aaa.fu on 2012.02.15 having any kind of attachment:

```
date1:2012-02-15, date2:2012-02-15, from: jane@aaa.fu,
bill@aaa.fu, attachment:any
```

Viagra spam bigger than 200 kB spoofing my email address as the sender, and having 'order', then 'now' in the body:

```
size:>.2M, subject: viagra OR cialis, body: order << now, from:
my@email.address
```

Price list to jenny@aaa.fu, in pdf attachment(s) smaller than 150 kB:

```
direction: inbound, size:<150k, attachment: pdf, subject: price
list, to: jenny@aaa.fu
```

The GUI relies heavily on the sphinx index data when searching. It also means that you may specify sphinxsearch operators (eg. |, &, <<, …) in your search query. The entered search phrases are in Boolean AND relation, eg. *cat dog* means that the documentation must contain both *cat* and *dog*. Some examples for sphinx operators:

`cat dog`	having cat and dog (order is not important)
`cat OR dog`	having cat or dog
`cat | dog`	having cat or dog
`"cat dog"`	having the expression "cat dog"
`!dog`	not having dog
`-dog`	not having dog
`"cat dog"~10`	proximity search
`cat << dog`	before operator: cat has to precede dog

See the *sphinx search* web site - “Boolean query syntax”[14] and “Extended query syntax”[15] - for more

14 http://sphinxsearch.com/docs/2.0.8/boolean-syntax.html
15 http://sphinxsearch.com/docs/2.0.8/extended-syntax.html

details.

Advanced search

The other search interface is the advanced search. It's a popup window you can get by clicking on either the down arrow in the right corner of the search field or the “*Advanced search*” button (depends on the used theme).

The quick search and the advanced search are equivalent, they give you the same possibilities to define a search query. The advanced search presents you a list of separated input fields giving a better visual view where you can specify the same fields (From, To, dates, etc.) as with the quick search.

Saving and loading search criteria

The GUI lets you to save then load search queries. It's especially handy if you entered a longer search expression of if you have to perform some regular searches. You can perform these tasks by clicking on the *Save* or *Load* buttons. You may remove a saved query by clicking on the Remove link next to the search expression.

Using the search results

If you have a search result then you can view any of the messages by clicking on the subject line or the serial number. The message is shown in the lower view pane.

You can also download the selected message as an EML file, or restore it to your mailbox via SMTP. In case of IMAP authentication you can restore it via IMAP protocol to the “Inbox” folder.

You may assign tags and notes to the selected emails. Sphinx indexes both of them, so you can use the “tag:” or “note:” keywords to search within tagged or noted messages. Note that the tags and notes are reindexed at periodic intervals, so there may be some delay when sphinx is aware of that a message is tagged.

It's also possible to download the search results from the current page as a zip file. To do so, select some messages then click on the download icon or button (theme dependent).

Wildcards

The sphinx search engine uses stemming to provide the possibility to search for word fragments, too. By default it starts stemming from the 6th character of the word. Let's take the following word: “estimation”.

Then the indexer creates the following fragments in addition to the original word: “estima”, “estimat”, “estimati”, “estimatio”.

Then it's enough to write *estima** to the search field, and sphinx will find all messages having any of the following words: “estima”, “estimat”, “estimati”, “estimatio” or “estimation”.

You may change the start of stemming in sphinx.conf, however consider that specifying a too low value (<5) produces really a lot of tokens for sphinx thus increasing its database and might yield to a poor performance.

Other notes

Note that the whole From: and To:/Cc: headers are indexed. It means that if the sender specified a name in addition to his email address, then you may search for the From name, just specify the from: keyword (from: jackson) or use the advanced search form.

Administering piler

Users, groups, and rights

piler allows you to access only your emails you either sent or got. It's often required that members of the same team should access each others' emails. To do so, piler lets you to create groups. Then you can assign email addresses to the group, and finally you can assign the groups to users.

Let's say you want the accounting employees to access each others' emails. Then create a group "*accounting*", and add the accounting employees' email addresses. Finally edit all accounting employees, and add the "*accounting*" group to their group membership.

Important note: there's a mutual relation within a group. If user1 and user2 are members of the "*accounting*" group, then it means that user1 can see user2's emails, and user2 can see user1's emails.

But what if the accounting manager wants to see all the accounting emails? If you add his email address to the "*accounting*" group, the employees will see his emails, too. So perhaps it's better if you don't add the manager's email address to the "*accounting*" group, but add the "*accounting*" group to his group membership data. This way the manager can see the accounting employees' emails, but they cannot see the manager's emails.

Note that the "group" feature works with local users only.

Administrator permissions

The administrator account (eg. admin@local) is used to administer piler only. It's not a super powerful account to see anyone's emails that's why it can't see the *Search* menu at all.

If you need a user who can access anyone's emails then grant him AUDITOR privileges on the user settings page.

The administrator role may see system statistics, accounting summary, edit user / group settings, policies, see audit logs, etc.

Rules and policies

Piler supports two kinds of policies or rulesets. The archiving rules specify what NOT to archive. The retention rules specifies how long to keep messages in the archive.

Archiving rules

You may define rules to prevent *piler* and *pilerimport* to archive certain messages, eg. having no subject, too big, recognized spam, etc.

You may specify the From:, To/Cc:, Subject:, message size, attachment type and size value. piler uses the *tre* regex library to test whether the given rule matches the processed message. If so, then neither *piler* nor *pilerimport* will archive it, but rather discard it after logging.

Here are some examples:

subject: ^ {0,}$	**no subject**
size: > 1000000, attachment type: video/	**messages bigger than 1 MB + having a video attachment**
size: < 500	**spam probes, not having any body**
subject: \[SPAM\]	**recognised spam emails**
BUY(NOW){0,1} (viagra|cialis)	**“buy” or “buy now” viagra/cialis spam**

You may test a single EML format message against your archiving rules by using pilertest, and it will print out the first matching archiving rule, eg.

```
pilertest message.eml
```

```
locale: hu_HU
message-id: <586387.60094.qm@web83907.mail.sp1.yahoo.com>
from: *Queen Oneil queenbaby3@att.net queenbaby3 att net
(att.net )*
to: *undisclosed recipients ()*
reference: **
subject: **
sent: 1293679136
hdr len: 3007
body digest:
0230af406d0e03e56ed63eeb8e0891ed6a97f49d297f42c0d2565cb770311323
```

```
rules check: from=,to=,subject=^$,size0,att.type=,att.size0
attachments:
direction: 0
spam: 0
```

When piler starts it reads the archiving rules and compiles them at startup. So if you change the archiving rules, be sure to click on the "*Apply settings*" red button to send a HUP signal to piler.

Retention rules

How long shall piler retain your messages? You may control it with the retention rules. The format is exactly the same as with the archiving rules: you can define the From:, To/Cc:, Subject:, message size, etc. and the retention period in days.

Every time piler archives a message, it assigns a retention period according to the retention rules or applying the default value (2257 days = 7 years + 2 days, unless you change it in piler.conf). After the retention period has expired the piler utilities will remove the message from the archive.

It's recommended to create your retention policy before deploying any archiving solution.

Note that the retention period is included in the per message verification digest, so the retention period should not be changed after the message has been archived.

piler also compiles a list from the retention rules, so whenever you change them, be sure to click on the "*Apply settings*" red button to send a HUP signal to piler.

What to do with spam

It's a waste to archive junk messages, so the best solution is to prevent them entering the archive. This can be achieved by redirecting spam messages to a quarantine, so only good emails can get to the archive.

If this setup is not feasible, you may create an archiving rule that keeps spam emails out of the archive. Or you may create a retention rule that no spam is to be retained beyond 30 days. Unfortunately these may be error prone, think about a false positive error.

Audit logs

The GUI keeps track of what users do and when they do. Every user's action involves an audit record that the gui stores into the audit SQL table.

Thus there's an audit trail of every user activity, eg. searching for a some messages, viewing them, downloading them, etc. Also the gui logs when a user logs in or logs out.

The following information is logged:

- timestamp
- username (email)
- action (eg. view, search, download)
- IP-address
- message serial number – if any
- optional description

Administrators and auditors are able to search within the audit logs, and even export the audit trail as a CSV file.

The search field is very similar to the quick search for emails, and the parser tries to figure out what you look for. If it has a format of a date (eg. YYYY.MM.DD or YYYY-MM-DD) then it's a date, if it has an @ symbol then it's an email, dotted quads are IP-addresses, and search|view|download|... are actions.

You may use wildcards by using the asterisk (*) character. Eg. 2013.08.* searches for the whole August of 2013. jack@yourdom* Searches for user jack in domains starting with “yourdom”. If you want to specify only the user part, then you still must include the @-symbol otherwise the parser won't recognize it as an email address: jack*@. IP-addresses can be “wildcarded” right after a dot, eg. 1.2.3.* or 1.2.*.*. Note that 123.123.12*.* is not a valid expression.

A slightly complex audit search to find the login, view and download attempts for the local auditor in this month from 31.0.0.0/8 address space:

audit*@ 2013.08.* login view download 31.*

If you don't want or need auditing, then turn it off:

```
$config['ENABLE_AUDIT'] = 0;
```

Common tasks

Localizing piler

Create the appropriate language directory under the language/ directory, eg. the following for Spanish:

```
cp -R language/en language/es
```

Then translate the *language/es/messages.php* file using UTF-8 encoding, and add it to *config.php*:

```
$langs = array(
             'hu',
             'en',
             'de'
             'pt',
             'es'
            );
```

Import emails

Use the *pilerimport* utility to import an EML message file, messages from an *mbox* file or emails from a directory containing EML format messages (1 message per file).

IMPORTANT! pilerimport doesn't change or delete the imported file, only reads it.

```
usage: pilerimport -e <eml file> -m <mbox file> -d <directory>
```

IMPORTANT! The *pilerimport* utility is setuid to the *piler* user, and to let it create some temporary files you have to *cd* to a directory where the *piler* user has write access. A few examples how to use the pilerimport utility:

Import a single EML file:

```
pilerimport -e message-from-jack.eml
```

Import a single EML file from STDIN:

```
pilerimport -e -
```

Import all messages from an mbox file:

```
pilerimport -m ~sj/201112.mbox
```

Import all messages from your Maildir:

```
pilerimport -d ~/Maildir/new
```

Import from IMAP server:

```
pilerimport -i <imap server> -u <username> -p <password> [-x comma,separated,skiplist]
```

By default pilerimport will skip the Junk, Trash, Spam and Draft folders (ie. -x junk,trash,spam,draft). If you want to import emails from all imap folders (including junk, spam,...), then specify -x ''.

To import all emails except from the Draft folder from IMAP server 1.2.3.4 with user account 'joe':

```
pilerimport -i 1.2.3.4 -u joe -p secret -x draft
```

To import all email from a POP3 server:

```
pilerimport -K pop3.yourdomain.com -u jack -p password
```

How to import a PST file?

pilerimport can't read it, however you can extract the contents of the PST file with the *readpst* utility from the *libpst* package[16], then use *pilerimport* to import the required messages.

Export emails

The *pilerexport* utility lets you export messages from the archive to EML files.

IMPORTANT! The pilerexport utility is setuid to the piler user, and to let it create the EML files you have to cd to a directory where the piler user has write access.

```
usage: pilerexport
```

16 http://www.five-ten-sg.com/libpst/

```
[-c|--config <config file>]
-a|--start-date <YYYY.MM.DD> -b|--stop-date <YYYY.MM.DD>
-f|--from <email@address> -r|--to <email@address>
-s|--minsize <number> -S|--maxsize <number>
-A|--all  -d|--dryrun
```

use -A if you don't want to specify the start/stop time nor any from/to address.

Note: if you use the *GNU libc* you may use the long option names, too.

If you want to see what messages piler would export, then use the -d (or –dryrun) command line option. It will print a list of IDs associated with each messages.

To prevent you dumping the whole archive you have to specify the from addresses and/or the recipient addresses and/or the start/stop dates. If you really want to export the whole archive, then use the -A (or –all) command line option. You may need lots of disk space to do so.

You may specify more email addresses, see the examples below. The conditions are in *Boolean AND* relation with each other.

export all emails >10kB in last December:

pilerexport -s 10000 --start-date 2011.12.01 --stop-date 2011.12.31

export all emails if size is between 100k and 2MB _AND_ the sender is any of aaa@aaa.fu,bbb@aaa.fu,ccc@ccc.fu _AND_ the recipient is aaa@gmail.com:

pilerexport -s 100000 -S 2000000 -f aaa@aaa.fu -f bbb@aaa.fu -f ccc@ccc.fu -r aaa@gmail.com

do a daily backup to the current directory:

pilerexport --start-date 2012.01.11 --stop-date 2012.01.11

Purge aged emails

piler gives you the chance to purge aged emails, ie. emails being in the archive for longer than their retention period. It's important to emphasize that you don't have to purge aged emails, it's not an obligation that piler forces, it's a possibility piler gives you.

You can purge the aged emails with the *pilerpurge* utility. It has a dry-run mode when it doesn't remove anything, rather it only prints what it would remove. To run *pilerpurge* in dry-run mode,

specify the '-d' command line option

```
pilerpurge -d
```

If you want to purge all aged emails then simply run the *pilerpurge* utility, and it will remove all aged emails.

```
pilerpurge
```

The pilerpurge utility honors the settings whether it's enabled or disabled to purge emails. If disabled then it quits immediately without purging anything.

Troubleshooting

Recreate sphinx index data

piler relies heavily on the *sphinx index data*. The *reindex* utility is for healing it if something goes wrong.

The *reindex* utility reads data from the *metadata* table, then retrieves the files from the */var/piler/store* directory. Then it parses them again, and puts data to be indexed to the *sph_index* table, so the sphinx indexer utility can re-index these messages.

```
Usage:
reindex -f <from id> -t <to id> [-p]

-f: start from index
-t: end with index
-p: put some dots as some sort of progress bar (it's optional).
```

So the following command will get all messages with metadata table id between 1 and 1000.

```
reindex -f 1 -t 1000
```

Before running *reindex*, please change to a directory where the *piler* user has read and write permissions.

Upgrade instructions

This section of the documentation provides detailed, step-by-step instructions on how to upgrade piler. It's important to follow the upgrade instructions precisely, don't just blindly execute commands otherwise you risk losing data.

It's also recommended that you upgrade piler at a time where you have less incoming emails.

Generic notes

If you upgrade from older releases then you should apply all the intermediate changes, ie. when upgrading from 0.1.18 to 0.1.20 please apply both db-upgrade-0.18-vs-0.19.sql and db-upgrade-0.19-vs-0.20.sql scripts.

0.1.18 to 0.1.19

Execute the following sql script on the piler database:

```
mysql -u piler -p piler < util/db-upgrade-0.1.18-vs-0.1.19.sql
```

Add the following variables to *config.php*:

```
define('TABLE_GROUP', 'group');
define('TABLE_GROUP_USER', 'group_user');
define('TABLE_GROUP_EMAIL', 'group_email');
```

0.1.19 to 0.1.20

Execute the following sql script on the piler database:

```
mysql -u piler -p piler < util/db-upgrade-0.1.19-vs-0.1.20.sql
```

0.1.20 to 0.1.21

Add the following variables to config.php:

```
define('ENABLE_FOLDER_RESTRICTIONS', 0);
define('ICON_NOTES', '/view/theme/default/images/notes.png');
define('ICON_PLUS', '/view/theme/default/images/plus.gif');
define('ICON_MINUS', '/view/theme/default/images/minus.gif');
define('ICON_EMPTY', '/view/theme/default/images/1x1.gif');
define('TABLE_FOLDER', 'folder');
define('TABLE_FOLDER_USER', 'folder_user');
define('TABLE_FOLDER_EXTRA', 'folder_extra');
define('TABLE_FOLDER_MESSAGE', 'folder_message');
define('TABLE_NOTE', '`note`');
define('SPHINX_NOTE_INDEX', 'note1');
define('LOAD_SAVED_SEARCH_URL', SITE_URL . 'index.php?
route=search/load');
define('MESSAGE_NOTE_URL', SITE_URL . 'index.php?
route=message/note');
```

Before running the following script, you should backup the 'tag' and 'note' tables then drop them, since the new version has a modified scheme for these tables.

```
mysql -u piler -p piler < util/db-upgrade-0.1.20-vs-0.1.21.sql
```

NOTE: the built-in admin@local account cannot see the users' emails any more. Instead the auditor role gives you the permissions to search in any mailbox. See *Administering piler* for more.

0.1.21 to 0.1.22

Execute the following sql script on the piler database:

```
mysql -u piler -p piler < util/db-upgrade-0.1.21-vs-0.1.22.sql
```

Rename LANG to DEFAULT_LANG in config.php (in a diff -u style):

```
-define('LANG', 'en');
+define('DEFAULT_LANG', 'en');
```

Add the following variables to config.php:

```
define('DECRYPT_ATTACHMENT_BINARY', '/usr/local/bin/pileraget');
define('ACTION_DOWNLOAD_ATTACHMENT', 15);
define('ACTION_UNAUTHORIZED_DOWNLOAD_ATTACHMENT', 16);
define('SEARCH_RESULT_CHECKBOX_CHECKED', 1);
define('REWRITE_MESSAGE_ID', 0);
define('ENABLE_SYSLOG', 0);
define('RESTRICTED_AUDITOR', 0);
```

Stop searchd, then initialize the new index called *note1*, and finally start searchd again:

```
# /etc/init.d/rc.search stop
# su - piler
$ indexer note1
$ exit
# /etc/init.d/rc.search start
```

If you upgraded sphinx.conf as well, then be sure to _remove_ the following lines from sphinx.conf:

sql_attr_uint = folder

and remove the 'folder' column from the sql queries in the 'main' and 'delta' sources.

0.1.22 to 0.1.23

Execute the following sql script on the piler database:

```
mysql -u piler -p piler < util/db-upgrade-0.1.22-vs-0.1.23.sql
```

Rename config.php to config-old.php, then copy the default config.php over your config.php. From now on, the variables are in an array with their default values. If you want to override any variable,

then place it to config-site.php with the new value, eg.

```
$config['SITE_NAME'] = 'archive.example.com';
```

An example config-site.php:

```
<?php

$config['SITE_NAME'] = 'demo.mailpiler.org';
$config['SITE_URL'] = 'http://demo.mailpiler.org/';
$config['DIR_BASE'] = '/var/www/demo.mailpiler.org/';

$config['ENABLE_AUDIT'] = 1;
$config['PILER_HOST'] = '1.2.3.4';
$config['SMTP_DOMAIN'] = 'demo.mailpiler.org';
$config['SMTP_FROMADDR'] = 'no-reply@demo.mailpiler.org';
$config['ADMIN_EMAIL'] = 'admin@demo.mailpiler.org';

?>
```

To support new versions of piler, you MUST relocate your store directory one level down:
Old store directory (for build <= 765):
$localstatedir/piler/store/

New store directory (from build >= 767):
$localstatedir/piler/store/<SERVER_ID>

By default server_id is 0 which translates to SERVER_ID: "00". if you set server_id=14 in piler.conf, then it translates to SERVER_ID: "0e". If you have a single piler node, then leave the default (0). The following example assumes this scenario.

Create the server_id top level store directory:

```
/etc/init.d/rc.piler stop
```

```
# download, compile, and upgrade the new version

cd /var/piler/store
mkdir aaa
mv * aaa
mkdir 00
chown piler:piler 00
mv aaa/* 00
rmdir aaa
/etc/init.d/rc.piler start
```

0.1.23 to 0.1.24

Execute the following sql script on the piler database:

```
mysql -u piler -p piler < util/db-upgrade-0.1.23-vs-0.1.24.sql
```