

imappop - Getting PigeonDeliver handle IMAP and POP3

Carlo Contavalli

ccontavalli@masobit.net

Revision History

Revision 1.0.0 2006/02/24
First document revision

This document describes how to setup PigeonDeliver to handle IMAP and POP3 connections.

1. Introduction

As you probably may already know, PigeonDeliver was born as a modulare system able to easily handle and manage clustered mail solutions.

While the main role of a mail server is usually to deliver emails, any modern mailing system also offers POP3 and IMAP access to its own users.

Whenever you have a PigeonAir based server and you need to offer POP3 and IMAP services, you have two choices:

- install any imap/pop3 server you may like, like dovecot, cucipop, qpopper, or courier, and simply configure it (one way or another) to access the "same database", the same directory structure, and the same mailbox format as PigeonAir.
- use PigeonAir facilities to have a complete mailing system, with all the features integrated and an uniform interface to access all resources.

1.1. The same problems, over and over again...

The first choice is what usually happens with any other MTA software: you install the MTA, decide the directory structure for users' home directories, decide a common mailbox format and figure out, one way

or another, how to make the chosen pop3/imap server access that user database, directory structure and mailbox format.

The main disadvantage of this approach is that you often need to figure out how to make a given MTA with a well defined structure work well with a pop3/imap server with a completely different structure in mind. The main problems that are usually encountered are due to:

- Since most servers use either proprietary configuration files or databases, to make two different softwares work together you will probably end up using a shared database. While this is not bad by itself and even be something you were wishing, there are at least two drawbacks...
- ... first of all, your choice of database to use is limited to the databases working both with your MTA and POP3/IMAP server, which may well leave you just a couple choices...
- ... additionally, you need to figure out how two different softwares, which may use a completely different internal structure, can use one same database for their purposes. Usually, when configuring such a solution, problems like: "hey, the mta wants the full path of the home directory in one field...", "the pop3/imap server builds the path using two different fields...", "the uid / gid of the MTA is set from the configuration file...", "the pop3 / imap server wants two fields to be added...", "one crypts the passwords with ...", "the other wants..." and so on. Figuring out how to make things work smoothly in such conditions is not often easy, and usually leads to a database more complex than necessary and with weird relations/fields to be kept consistent/... to be used. Which means: something harder to maintain, and something harder to modify...
- ... even if you get a nice looking database without using hacks or SQL queries like CONCAT(field1, '/', ...) or strange constants, some softwares may just make different assumptions. Who creates the users' home directory? is that directory writeable by whom? if the directory is created automagically, how do we handle privileges? ...
- How do we handle privileges of different softwares having to access the same user base? What if some day or another you want to switch to a different software? What if the two softwares have slightly different semantics for things that should be standard? How do we handle quotas? ...
- How do we manage a cluster? how do we create one? how do we solve concurrency issues on a database that one software does not even lock when modifying it, or where relations are so weird that we cannot guarantee atomicity when updating user configurations... (and believe me or not, there are very few open source solutions that worry about the atomicity of updating user configurations).

1.2. The PigeonAir solution...

PigeonAir and PigeonDeliver internally provide support for using a couple of POP3 and IMAP servers.

At time of writing, support for the Courier IMAP and Courier POP3 server (<http://www.courier-mta.org>) has already been completed, while Dovecot support is on its way to completion.

The basic idea is quite simple: PigeonDeliver and PigeonAir must take care of handling all integration issues between the MTA and the POP3/IMAP server being used (user database, authentication, directory

structure, mailbox format) and must take care to abstract out clusters of mail servers like if they were a single machine both to the end user, to the administrator, and to the softwares being used.

From a more technical point of view, the idea is quite simple: PigeonDeliver internally implements a "modular" IMAP/POP3 proxy server. The proxy server itself takes care of authenticating users and fetching their configuration from a database, while its modular nature allows it to either transparently redirect the connections to other machines in a mail cluster or to *internally call and load* the mail server of your choice (courier or dovecot, at time of writing, Fri Feb 24 21:30:06 CET 2006), without any performance loss (and probably, with some gain). What's new about the PigeonDeliver IMAP/POP3 proxy modular structure is:

- authentication and fetching of users configurations are performed only *once*. Almost all IMAP/POP3 proxy we are aware of need to perform authentication twice: one on the machine the user connects to, and one on the machine where the user is redirected. And, on both machines, user configurations need to be fetched from the database.

The PigeonDeliver POP3/IMAP proxy can use a configurable authentication mechanism when talking to its own peers. One of these mechanisms (which is just one which you can choose among others...) establishes some kind of trust path based on cryptographic authentication of hosts. Once the host is authenticated, the IMAP/POP3 proxy can pass authentication data (is the user who claims to be? who is he?) and configurations directly to the other peer, avoiding any additional lookup. No sensitive data (like passwords, certificates or private keys) is ever sent on the network, not even on encrypted channels.

- when proxying is avoidable, *proxying is actually avoided*. At time of writing, the PigeonDeliver IMAP/POP3 proxy can be used just as a simple IMAP/POP3 authentication module. For example, the "courier-imapd" module allows PigeonDeliver to internally load the courier imapd daemon. When courier is loaded internally, however, authentication has been carried on already by PigeonDeliver and all user configurations have already been fetched from the database. So, no need to enable strange courier authentication modules, no need at all to configure any courier authentication or database. Everything is taken care efficiently by the PigeonDeliver module, which also caches connections to the database as much as possible. In short: by using PigeonDeliver IMAP/POP3 proxy you should see a speedup in user authentications. Both on single setup and on clustered solutions.
- If you start with one single machine, PigeonDeliver IMAP/POP3 proxy will just act as a very simple authentication module for your IMAP/POP3 server of choice, without adding any overhead to the standard mechanism used by Courier/Dovecot.

If you then switch to a mail cluster, PigeonDeliver IMAP/POP3 proxy will simply continue to handle "local connections" like before, by internally loading Courier or Dovecot, while redirecting connections that need to be handled by remote servers, thus becoming a very fast, smart and efficient POP3/IMAP proxy.

- This same modular structure also allows for many other mechanisms to be used. If you want a "normal and standard" IMAP/POP3 proxy, you can use the PigeonDeliver one just to avoid having to mess up with the database one more time. Right, because if you have a MTA, a POP3/IMAP server, an

IMAP/POP3 proxy based on different softwares, it is quite easy to end up with a good headache when configuring one single database to be used by all of them.

- Another nice feature is that it can really be *integrated* in any mail server. As an example, if you use the module "pd-forker-inet" you can run the IMAP/POP3 server from your inetd.conf file. If you use "pd-forker-stdio", you can load it from the command line. If you use "pd-forker-postfix", you can run it from the "master.cf" postfix configuration file, ending up with a single place for your logs and with all daemons involved with "mail management" monitored by the postfix master, which is extremely reliable and performant, and which should simplify the task of administering the machine. If you use "pd-forker-listen", you will be able to run the IMAP/POP3 server as a standalone daemon... and you are free to add more modules to support more advanced connection handling/forking mechanisms or to integrate PigeonDeliver IMAP/POP3 server in any other software you may like.

2. Using the PigeonDeliver IMAP/POP3 support

Using the PigeonDeliver IMAP/POP3 support is quite easy. You need to:

- configure PigeonDeliver with support:
 - for the POP3/IMAP authentication handling features (login-imap, login-pop3).
 - for the POP3/IMAP handler to be run from the mail server of your choice, or using the specified mechanism (forker-postfix, forker-stdio, forker-daemon, forker-inetd, ...). For example, if you use forker-postfix you will be able to add your new IMAP/POP3 server in your postfix "master.cf" configuration file, and leave the postfix "master" the duty of making sure it will always work. If you compile the forker-daemon module, you will be able to run the POP3/IMAP server as a standalone daemon, ...
 - for the IMAP/POP3 server that will be used. More than one server can be supported at the same time, even if, at time of writing, only one at a time can be used. In future versions of PigeonDeliver this limit may be as well removed.
- enable the POP3/IMAP handler.

2.1. Configuring PigoenDeliver with IMAP/POP3 support

Well, this step should be quite easy:

```
$ mkdir build.imap
$ mkdir build.pop3
$ cd build.pop3
$ sh ../examples/configure/configure.*.pop3
$ cd ..
$ cd build.imap
```

```
$ sh ../examples/configure/configure.*.imap
```

then, you can simply "make" and "make install" as usual. Instead of `configure.*.pop3` and `configure.*.imap`, you must choose one of the available examples. For example, I may chose to use `configure.linux.postfix.imap` and `configure.linux.postfix.pop3`. If you want to, you can open those files and modify them to better suit your own needs. Keep also in mind that this process involves "some duplication", which means that some libraries needed both by `pop3` and `imap` are compiled twice.

If you want to avoid the duplication, you can just take a look to the two files (`configure.*.pop3`, and `configure.*.imap`), and run one `../configure` instead of `sh ../examples[...]` with the parameters of both the files.

Keep also in mind that, at time of writing, Fri Feb 24 22:19:16 CET 2006, those two files will compile all the "forkers modules" available, which means: `forker-stdio` and `forker-listen`. In other words, you will be able to run the POP3/IMAP server both from the command line (`forker-stdio`), as a simple standalone daemon (`forker-listen`) and from the postfix `master.cf` file (`forker-postfix`). Keep also in mind that both the support for `courier` and `dovecot` will be compiled.

2.2. Configuring postfix to run the IMAP/POP3 service

If you compiled the "forker-postfix" module or you used the `configure.*.postfix.imap` or `configure.*.postfix.pop3` examples, you can run the POP3/IMAP proxy directly from the `master.cf` configuration file.

You just need to add the lines:

```
imap inet  n n n - - pd-forker-postfix -odscm_service=login-imap
pop3 inet  n n n - - pd-forker-postfix -odscm_service=login-pop3
```

and to restart postfix, with something like:

```
# postfix stop
# postfix start
```

or

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

Just a few notes about the IMAP/POP3 service security: both services run at fixed low privileges, as much as most of the other postfix daemons. If they are only proxying connections, you don't need to make them privileged and you don't need to make them run outside of your `chroot`, so, you can change the above lines in something like:

```
imap inet  n - - - pd-forker-postfix -odscm_service=login-imap
```

```
pop3 inet n - - - pd-forker-postfix -odscm_service=login-pop3
```

Those privileges are needed because:

- the daemon needs to access users home directories. Users home directories usually are not stored in the chroot, and they better not to be.
- the daemon eventually needs to create the home directory of the users. If you want to, and you set up group privileges correctly, you can set it up as an unprivileged component of postfix, with something like:

```
imap inet n y n - - pd-forker-postfix -odscm_service=login-imap
pop3 inet n y n - - pd-forker-postfix -odscm_service=login-pop3
```

Note, however, that the real imap/pop3 daemon (courier/dovecot) are run with *fixed low privileges*, exactly the same privileges that would be given directly to the user (or that you have configured with the MDA module). The security model is something like:

- very small login/authenticator process, wich runs with low privileges, but able to gain additional privileges to create user home directories.
- child process, which runs with fixed low privileges, able to really make the IMAP/POP3 services available.

In future releases of PigeonDeliver, the login/authenticator process will probably become an even simpler process using an authenticator module, which can then use an independent daemon for user authentication.

2.3. Configuring the POP3/IMAP service

Once you correctly installed the PigeonAir POP3/IMAP service, you *don't have to do anything else* to configure them. If you correctly set up mail delivery for PigeonDeliver those services will automatically use the configuration you already provied when configuring mail delivery: the correct MDA will be called to access the directory structure, the same users creation procedures will be used the first time a new user logs in, and the same database driver with the same parameters will be loaded.