Debugging authentication problems

A common problem after installing the Courier authentication library is that authentication, using authtest, doesn't work. This document shows how to use courier's debugging features to pinpoint the problem.

1. Turn on debugging

For courier-imap, you need to set one of the following values in /usr/local/etc/authdaemonrc:

```
DEBUG_LOGIN=1    # turn on authentication debugging
DEBUG_LOGIN=2    # turn on authentication debugging AND show passwords
```

This setting is located at the very end of the configuration file.

After changing this setting, restart the authentication daemon by running the "authdaemond stop" and "authdaemond start" commands.

At this point, all debugging output goes to syslog at level 'debug', which is normally not shown. You will probably need to change your `/etc/syslog.conf` file to be able to see these messages. If you have an existing entry which says "mail.info" (which means facility 'mail', level 'info' or higher) then you can just change this to "mail.debug". Alternatively you can add a new entry like this:

```
*.debug                         /var/log/debug
```

Don't forget to create this file, and to send a HUP signal to syslogd to make it re-read its configuration:

```
# touch /var/log/debug
# killall -1 syslogd
```

If you don't want to mess around with your syslog configuration, you can also start `authdaemond` manually, and log its output to a file:

```
/usr/local/libexec/courier-authlib/authdaemond >filename 2>&1
```

2. Issue a manual login

You can use the authtest command to verify authentication, or go ahead and install Courier-IMAP.

For courier-imap, you will get much better information by not using a mail client and manually logging in using 'telnet'. The transcript of this telnet session may give useful information as to what is going on. If you are going to report a problem to the mailing list, you should certainly include this transcript as well as the corresponding debugging output.

```
-- to debug POP3 --
# telnet x.x.x.x 110
user USERNAME
pass PASSWORD
stat
quit

-- to debug IMAP --
# telnet x.x.x.x 143
a login USERNAME PASSWORD
a examine inbox
a logout

-- to debug POP3 over SSL --
# openssl s_client -connect x.x.x.x:995
(then use same commands as POP3 example)

-- to debug IMAP over SSL --
# openssl s_client -connect x.x.x.x:993
(then use same commands as IMAP example)
```

This isn't an option for sqwebmail of course - just login through the web interface and check the authentication debug log which is generated.

3. Interpret the debug output

First, a brief explanation of courier's authentication system. There are a number of standalone **authentication modules**. An authentication module exists for every authentication method. Each authentication module is installed as a shared library. When **authdaemond** starts, it attempts to load and initialize the authentication modules, logging the following messages to syslog:

```
Oct 17 11:25:37 commodore authdaemond: modules="authuserdb authpam authpgsql authldap authmysql authcustom", daemons=5
Oct 17 11:25:37 commodore authdaemond: Installing libauthuserdb
Oct 17 11:25:37 commodore authdaemond: Installation complete: authuserdb
Oct 17 11:25:37 commodore authdaemond: Installing libauthpam
Oct 17 11:25:37 commodore authdaemond: Installation complete: authpam
Oct 17 11:25:37 commodore authdaemond: Installing libauthpgsql
Oct 17 11:25:37 commodore authdaemond: libauthpgsql.so: cannot open shared object file: No such file or directory
Oct 17 11:25:37 commodore authdaemond: Installing libauthldap
Oct 17 11:25:37 commodore authdaemond: libauthldap.so: cannot open shared object file: No such file or directory
Oct 17 11:25:37 commodore authdaemond: Installing libauthmysql
Oct 17 11:25:37 commodore authdaemond: libauthmysql.so: cannot open shared object file: No such file or directory
Oct 17 11:25:37 commodore authdaemond: Installing libauthcustom
Oct 17 11:25:37 commodore authdaemond: Installation complete: authcustom
```

The first message lists all authentication modules that were compiled, and indicates that **authdaemond** will spawn five processes to handle all authentication

requests. This is followed by messages indicating that indicate which authentication modules were installed.

In this example, authdaemond did not load the authpgsql, authldap, and authmysql modules. That's because in this case the Courier authentication library is installed by the system's package manager. The LDAP, MySQL, and PostgreSQL support was placed into optional sub-packages which are not installed. Even though all of these modules were initially compiled, the optional authentication modules were not installed.

This is normal. authdaemond will simply ignore any authentication module it cannot find, and will activate only those modules that are available. When an authentication request comes in, all of the modules will be executed, one after the other, resulting in one of three conditions:

ACCEPT
: The user was authenticated successfully

REJECT
: The module did not know this username, or the user gave invalid credentials. The request is passed to the next module.

TEMPFAIL
: The module suffered an internal failure, such as inability to contact an external database. The login is rejected, and no further modules are tried.

In a typical Courier installation the authentication request is sent, via a filesystem socket, to a pool of **authdaemond** processes (note the extra "d" on the end) which perform the actual work. authdaemond, in turn, contains other authentication modules such as authpam, authmysql, and so on.

If `authdaemond` is running successfully, then it will in turn run each of the modules it is linked against. If any one returns REJECT then the next is tried; if any returns TEMPFAIL or ACCEPT then no further modules are tried.

So a typical example might look like this:

```
Apr 14 14:07:15 billdog authdaemond: received auth request, service=pop3, authtype=login
Apr 14 14:07:15 billdog authdaemond: authcustom: trying this module
Apr 14 14:07:15 billdog authdaemond: authcustom: nothing implemented in do_auth_custom()
Apr 14 14:07:15 billdog authdaemond: authcustom: REJECT - try next module
Apr 14 14:07:15 billdog authdaemond: authcram: trying this module
Apr 14 14:07:15 billdog authdaemond: cram: only supports authtype=cram-*
Apr 14 14:07:15 billdog authdaemond: authcram: REJECT - try next module
Apr 14 14:07:15 billdog authdaemond: authuserdb: trying this module
Apr 14 14:07:15 billdog authdaemond: userdb: opened /etc/userdb.dat
Apr 14 14:07:15 billdog authdaemond: userdb: looking up 'brian'
Apr 14 14:07:15 billdog authdaemond: userdb: entry not found
Apr 14 14:07:15 billdog authdaemond: authuserdb: REJECT - try next module
Apr 14 14:07:15 billdog authdaemond: authpam: trying this module
Apr 14 14:07:15 billdog authdaemond: authpam: sysusername=brian, sysuserid=<null>, sysgroupid=1001, homedir=/home/brian, address=brian, t
Apr 14 14:07:15 billdog authdaemond: pam_service=pop3, pam_username=brian
Apr 14 14:07:15 billdog authdaemond: dopam successful
Apr 14 14:07:15 billdog authdaemond: authpam: ACCEPT, username brian
```

What's happening here?

- The request was received by 'authdaemond'
- It tries 'authcustom' - this module does nothing unless you have customised it yourself, so it REJECTs the request
- It tried 'authcram', but since this was a request with authtype=login (rather than authtype=cram-md5, say), this module cannot handle it so it REJECTs
- 'authuserdb' has a go. In this case there is an /etc/userdb.dat file for it to look in, but the requested username 'brian' does not exist in there, so it REJECTs
- 'authpam' has a go. It finds the username and home directory in /etc/passwd, and then calls the PAM subsystem to authenticate. The authentication is successful.

So, in principle, debugging is straightforward. Watch the modules operate, search for the one which you *think* should be authenticating the user, and if it is not, check for REJECT (user not known or password mismatch) or TEMPFAIL (internal error) status. Additional messages should indicate why this status was returned.

4. Read the documentation

Most of the configuration files like authldaprc, authmysql are well documented with comments.

For the nitty-gritty details of authentication modules, see man authlib. There is probably a copy of this manpage installed on your system; if that command doesn't work, try one of these:

```
# man -M /usr/local/man authlib
or
# cd /path/to/sources
# cd authlib
# nroff -mandoc authlib.7.in | less
```

If you are using userdb authentication, you definitely need to read man makeuserdb, man userdb, and man userdbpw.

5. Use the mailing list

Please read through the common problems and solutions at the bottom of this document. The next thing to do, of course, is search the web to see if your particular problem has been seen before and solved. Google is very good for this.

If you still cannot work out what the problem is, then you can ask on the appropriate mailing list. But before you post, please gather together all the following information:

- The operating system and version you are running
- The versions of packages you have installed
- The `./configure` command line you gave to build it
- If you didn't build it yourself, where you got the package from (and if possible, find out from the packager what options they used to build it)
- The versions of any other relevant software which you are linking against, e.g. openldap, mysql, pgsql
- The transcript of the 'telnet' session you used to manually test server connections
- The corresponding debug output which was generated for that session
- The contents of relevant configuration files, e.g. authldaprc, authmysqlrc, imapd, pop3d
- A copy of the database entry you are trying to authenticate against: e.g. the line from your userdb file, an LDAP entry, a row from your mysql table, the line in

/etc/password, etc.

If you include all this, you are *much* more likely to get a helpful response.

Frequently seen authentication problems and solutions

See also the Courier MTA FAQ

When I try to login with POP3 using telnet, the server disconnects immediately after the "PASS" command, without giving a -ERR response

The reason for this error will probably be found in your mail logs. Usually it indicates either that the home directory does not exist (chdir failed), or the Maildir has not been created. See 'man maildirmake'.

PAM authentication says "pam_start failed, result 4 [Hint: bad PAM configuration?]"

Probably your PAM configuration is bad. If you have /etc/pam.d/other, then try simply removing /etc/pam.d/pop3 and /etc/pam.d/imap and see if it works (this is sufficient for FreeBSD). Otherwise, try copying one of your existing /etc/pam.d/xxx files to /etc/pam.d/pop3, imap or webmail respectively.
The result value is a PAM_XXXX constant from `/usr/include/security/pam_constants.h` (this file may be in a different location on your system). Under FreeBSD, 4 is PAM_SYSTEM_ERR.

When I connect on the SSL ports (995 or 993), the server accepts the connection but then immediately disconnects

You probably didn't install any SSL certificates. Courier-imap comes with scripts you can run to do this for you:

```
# /usr/local/sbin/mkimapdcert
# /usr/local/sbin/mkpop3dcert
```

I expected the authentication library to compile authmysql (or some other module), but it's not there

If the mysql authentication module did not compile, then perhaps ./configure was unable to find your mysql libraries (you can read through the file 'config.log' in the source directory to see what it found). You may need to force it to look in the right place, as follows:

```
# ./configure --with-authmsql --with-mysql-libs=/usr/local/mysql/lib  \
              --with-mysql-includes=/usr/local/mysql/include
```

On some systems (e.g. FreeBSD), the mysqlclient library depends on the math and compression libraries. For these systems, try:

```
# LDFLAGS="-lm -lz" ./configure --with-authmysql ... same as before
```

The POP3/IMAP server says "Temporary problem, please try again later" when a bad password is entered

`authdaemond` tries each of the configured authentication modules in turn, until either one accepts the login, or they have all rejected it (in which case the usual "Login failed" error is returned, and the user can try again).

However, if one of these modules is unable to run because some resource is not available, then it gives a "temporary failure" response and no further modules are tried. You should find the exact cause in your mail logs, but typically it means that you have a module like 'authmysql' in your module list, but the mysql database is not running.

So unless you actually do have account data in mysql (in which case you need to fix your mysql setup), you should remove 'authmysql' and any other modules you do not use from `authmodulelist` in `authdaemonrc`.