FIELD NOTES FROM A PRESALES SOLUTIONS ARCHITECT

Postfix in a Chroot Cannot See Your MySQL Socket — Why Mail Stopped Both Ways

A field note about a self-hosted mail server where nothing moved in or out for weeks — the one master.cf column almost nobody reads — and the four checks I now run before touching a single password.

Daanial Farrukh · 08 November 2026 · https://quabin.com/postfix-chroot-mysql-socket-mail-stuck/

The first sign was not an error. It was silence.

No bounce. No angry reply. No "message could not be delivered". Just a mailbox that had stopped receiving anything, and outgoing mail that never arrived anywhere.

The server was a self-hosted grommunio box on Ubuntu 24.04. Postfix in front, MySQL behind it holding the mailboxes and aliases. A normal, modern stack.

I opened the queue.

```
postqueue -p
```

It was full. The oldest message was about six weeks old.

And the part that changed the whole investigation: some of the stuck messages were from one mailbox on the domain to another mailbox on the same domain. Same server. No internet in the path. No remote MX, no blacklist, no DNS.

If internal mail is stuck, the problem is not out there. It is inside the box.

That's the trap. A broken mail server usually shouts. This one only deferred — politely, forever.

What the Log Actually Said

The mail log had the same pair of lines over and over:

```
tail -n 50 /var/log/mail.log
```

```
postfix/cleanup: warning: mysql:/etc/postfix/grommunio-virtual-mailbox-alias-maps.cf lookup
error
... virtual_alias_maps map lookup problem ... message not accepted, try again later
```

Read that second line slowly. "Try again later" is a temporary failure. Postfix did not reject the mail — it asked every sender to come back. Remote servers obeyed and retried. Local submissions sat in the queue and retried. Nobody got a bounce, because a temporary failure is not a bounce until the queue lifetime runs out.

That is why it stayed quiet for weeks.

The first guess almost everyone makes from this log is the database: wrong password in the .cf file, MySQL down, the grommunio user lost its grant. So I tested exactly that — the same map, the same lookup, by hand:

```
postmap -q user@example.com mysql:/etc/postfix/grommunio-virtual-mailbox-alias-maps.cf
```

It answered. Instantly. Correct result.

Here's the part that trips people up. The lookup works from the shell and fails inside Postfix. Same file, same credentials, same database. **When the same lookup works for you and fails for the daemon, the problem is not the lookup — it is where the daemon is standing.**

What a Postfix Chroot Actually Is

A chroot is a jail: the process is told that one directory is the whole filesystem, and it cannot see anything outside it.

For Postfix, that directory is the queue directory, usually /var/spool/postfix. When a service in master.cf is chrooted, a path like /run/mysqld/mysqld.sock is really read as /var/spool/postfix/run/mysqld/mysqld.sock. That file does not exist. So the MySQL client cannot connect, and every mysql: map lookup fails.

The flow, inside the jail:

cleanup → mysql: map → host = localhost → Unix socket /run/mysqld/mysqld.sock → not found inside /var/spool/postfix → lookup error → "try again later"

And the same flow from your root shell:

postmap -q → mysql: map → host = localhost → /run/mysqld/mysqld.sock → found → answer

The cleanup daemon matters more than people think. It runs on every message, inbound, outbound and local, and it is where virtual_alias_maps is expanded. Break it and you break mail in every direction at once — which is exactly what the queue showed.

Where this setup is common:

- Debian and Ubuntu style master.cf files, where many services are marked chrooted.
- Mail stacks that keep users and aliases in MySQL or MariaDB: grommunio, Postfix admin panels, hand-built virtual mailbox servers.
- Servers rebuilt or upgraded where master.cf came from a package and the MySQL maps were added later.

Why Nobody Sees It in Testing

Three reasons, and they stack.

First, every manual test runs outside the jail. postmap -q, the mysql client, a PHP admin page — all of them see /run/mysqld/mysqld.sock and all of them pass. The test you trust is the one that cannot fail.

Second, the failure is a deferral. No bounce reaches a user, no alert fires on a queue that is "only" growing, and senders' servers keep retrying quietly.

Third, the fix that looks obvious makes it worse. From the log, the natural move is to edit the .cf map and change hosts = localhost to hosts = 127.0.0.1 — "use TCP instead of the socket". On this server that is a new failure, not a fix. The MySQL client treats localhost as "use the Unix socket" and 127.0.0.1 as a TCP connection, and MySQL matches accounts by host. The grommunio database user was granted on localhost. Change the host and the login itself stops matching. You have swapped a jail problem for an access-denied problem.

How to Actually Fix This

Everything up to Step 4 is read-only.

Step 1: Confirm mail is deferred, not rejected.

```
postqueue -p | tail -n 1
```

The last line gives the count of queued requests and their size. Then look at the oldest entries at the top of the full list. If messages are days or weeks old and include local-to-local mail, stop looking at DNS and remote servers.

Step 2: Read the exact error.

```
grep -E "lookup error|map lookup problem" /var/log/mail.log | tail -n 5
```

You are looking for mysql: named in the warning and map lookup problem with try again later. The map file named in the warning is the one to test next.

Step 3: Run the same lookup as root, outside Postfix.

```
postmap -q user@example.com mysql:/etc/postfix/grommunio-virtual-mailbox-alias-maps.cf
```

Use an address you know exists. If it returns a result, the credentials, the grant and the database are fine. This is the step that turns "database problem" into "Postfix environment problem".

If it fails here too, you do have a database or credentials problem, and the rest of this article is not your fix.

Step 4: Read the chroot column in master.cf.

```
grep -E '^(cleanup|smtp\s|smtpd|pickup)' /etc/postfix/master.cf
```

The columns are: service, type, private, unpriv, **chroot**, wakeup, maxproc, command. The fifth column is the one that matters. A y there means that service runs inside /var/spool/postfix.

Postfix can also print the value for every service in one go:

```
postconf -F '*/*/chroot'
```

On the broken server, cleanup, smtpd and smtp were all y. That plus a working postmap -q is the proof.

Step 5 (changes state): Turn chroot off for all services.

```
postconf -F '*/*/chroot=n'
```

This edits master.cf for you — no hand editing, no missed line. Then restart Postfix and flush the queue:

```
systemctl restart postfix
```

```
postqueue -f
```

postqueue -f asks Postfix to retry everything in the queue now instead of waiting for the next retry interval.

Step 6: Verify.

This step is where most tutorials stop and where mine starts.

```
postconf -F '*/*/chroot'
```

Every line must now end in = n.

```
tail -f /var/log/mail.log
```

You should see the old messages leave: status=sent lines for remote delivery, local delivery lines for internal mail, and no new lookup error warnings. Then watch the queue drain:

```
postqueue -p | tail -n 1
```

What success looks like: the count drops, and finally Postfix reports Mail queue is empty. What a failure looks like: the count stays flat and the lookup error lines keep appearing — in that case the restart did not happen, or a service still carries y.

Last, send one fresh test message from one local mailbox to another and one to an outside address. The old queue draining proves the fix; a new message proves the server is healthy now.

On this server, mail flowed again in both directions right after the restart.

Why Not Keep the Chroot?

A fair question. A chroot is a security feature, and turning it off is a trade-off, not a free win.

You can keep it, but then the socket has to exist inside the jail — for example by making MySQL's socket available under /var/spool/postfix/run/mysqld/. That works until a reboot, a package update or a systemd change removes it, and then you are back to a silent queue. And moving the maps to TCP needs matching database grants for the TCP host, which, as above, is exactly what this server did not have.

For a mail server whose users and aliases live in a database, I prefer the boring setup that cannot quietly break: no chroot on the Postfix services, the database socket where the client expects it, and the grants left alone. Postfix still drops privileges and runs its daemons as an unprivileged user; the chroot is one extra layer, not the only one.

The Checklist I Run Now

When a mail server goes quiet, in this order:

1. Is anything queued, and how old is the oldest message? postqueue -p
2. Is local-to-local mail stuck too? If yes, the cause is inside the box. Read the sender and recipient in the queue list.
3. Which map is failing? grep "lookup error" /var/log/mail.log
4. Does the same lookup work outside Postfix? postmap -q <address> <map>
5. Which services are chrooted? postconf -F '/:chroot'
6. After any fix: does the queue drain to empty? postqueue -p | tail -n 1

Plain takeaway: **a lookup that works as root and fails in the daemon is an environment problem, not a credentials problem.**

What I Tell Every Customer Now

Whenever someone runs their own mail on Postfix with a database behind it, I ask one question in the review:

"Who gets told when the oldest message in the queue is more than an hour old?"

If the answer is a named person and a working alert, they're in good shape. If the answer is "users would complain", we're not done yet — because in this failure, nobody complains. The mail just waits.

I write about the questions I ask in real customer rooms — the ones that change architectures and surface the problems no one else noticed. If you found this useful, drop your email below and I'll send the next one when it's ready.

FAQs

Why does postmap -q work but Postfix says "map lookup problem"?

postmap runs in your shell, outside any chroot, so it can reach the MySQL socket at /run/mysqld/mysqld.sock. A chrooted Postfix daemon such as cleanup looks for the same path inside /var/spool/postfix, where it does not exist. Same file and credentials, different view of the filesystem.

What does "virtual_alias_maps map lookup problem ... try again later" mean?

Postfix could not complete a lookup in the virtual alias map, so it refused the message with a temporary error instead of guessing. Senders retry and local mail stays queued, so nothing bounces for days. The fix is to make the lookup work again, then flush the queue.

How do I check if Postfix services run in a chroot?

Run postconf -F '/:chroot' to print the chroot setting for every service, or read the fifth column of each line in /etc/postfix/master.cf. A y means that service is chrooted to the Postfix queue directory.

Should I change hosts = localhost to 127.0.0.1 in the MySQL map files?

Not as a fix for this problem. With the MySQL client, localhost means the Unix socket and 127.0.0.1 means TCP, and MySQL accounts are matched by host. If the mail user is granted only on localhost, the change makes the login fail.

Is it safe to turn off chroot for Postfix?

It removes one layer of isolation, so it is a trade-off. Postfix still runs its daemons as an unprivileged user. For a database-backed mail server, a setup that cannot silently lose access to the database socket is usually the safer choice in practice.

How do I get the stuck mail delivered after the fix?

Restart Postfix so the new master.cf is read, then run postqueue -f to retry every queued message now. Watch /var/log/mail.log for delivery lines and run postqueue -p until it reports that the mail queue is empty.

Read online: https://quabin.com/postfix-chroot-mysql-socket-mail-stuck/