

FIELD NOTES FROM A PRESALES SOLUTIONS ARCHITECT

Mail Stopped Both Ways — "451 Temporary Local Problem" and the One DNS Resolver Nobody Checked

A presales war story about a cPanel server that could neither send nor receive mail — no blacklist, no full disk, no broken config — and the single DNS line I now check before I open any mail log.

Daanial Farrukh · 26 October 2026 · <https://quabin.com/exim-451-temporary-local-problem-dns-resolver/>

The call was short.

"Email is down. Completely. We can't send, and nothing is coming in."

The server was a customer's cPanel/WHM box on Rocky Linux 8. It relayed outbound mail through their provider's smart host. It had been quiet and boring for a long time, which is exactly what you want from a mail server.

Users saw this in Roundcube every time they pressed Send:

SMTP Error (451): Failed to add recipient - Temporary local problem

The customer's IT lead had a theory ready.

"It must be the provider's relay. Or we got blacklisted again."

I asked one question.

"If it's the relay, why can't anyone send *to* you either?"

A pause. "...Both directions at the same time?"

Both directions at the same time. A relay problem breaks outbound. A blacklist breaks one side for some senders. Something that breaks everything, in and out, at once, usually sits underneath both.

The Trap, Named

That's the trap. "Temporary local problem" sounds like the mail server is sick. So people look at the mail server — the queue, the disk, the Exim config, the last cPanel update.

The mail server was healthy. It just could not ask a single DNS question.

And the reason was one line in a file that almost nobody reads during a mail outage.

What "Temporary Local Problem" Actually Means Here

Exim needs DNS for nearly every step of a mail transaction. Not only to find the recipient's MX. It needs DNS to find its own smart host, and it needs DNS to check that a sender is real before it accepts a message.

Outbound looks like this:

User → Roundcube → Exim → DNS lookup of the smart host → smart host → recipient

Inbound with sender verification looks like this:

Sender → Exim → DNS lookup of the sender's domain → callout to the sender's MX → accept or reject

If DNS does not answer, Exim cannot finish either flow. It does not bounce the mail permanently, because a DNS timeout might fix itself. It answers with a **temporary 451** and tries again later.

Roundcube shows that as "Temporary local problem". The sending server on the other side sees a deferral and keeps retrying.

A 451 with "local problem" often means the local server could not finish its own lookups — not that the message or the recipient is wrong.

The Exim log said it plainly once we looked:

```
temporarily rejected RCPT <user@example.com>: host lookup for smarthost.example.net did not complete (DNS timeout?)
```

And WHM › Email › Track Delivery was full of the same line:

```
Could not complete sender verify
```

Both are the same failure seen from two sides. The smart host lookup breaks outbound. The sender verify lookup breaks inbound.

This setup is common in three places:

- Servers built from a template where someone typed 8.8.8.8 as the only resolver because "it always works".
- Networks where outbound DNS to the internet is filtered or routed differently from web traffic.
- Servers with one resolver line and no second one, so there is no fallback when that one goes away.

Why Nobody Saw It

From the outside, a server in this state looks fine. WHM loads. SSH works. Websites keep serving, because answering a web request does not need the server to resolve anything.

The usual checks stay green too: disk, the Exim service, the last update. The queue grows, but a growing queue is a symptom, not a cause.

Here is the part that trips people up. **Resolver problems hide behind every other check.** A web request comes in, the server answers it, done — no outbound DNS needed. Mail is different. Mail is the one service on a cPanel box that makes outbound DNS lookups for every message.

The file that mattered was this one:

```
cat /etc/resolv.conf
```

It had exactly one line:

```
nameserver 8.8.8.8
```

One resolver. No second line. And from that network, 8.8.8.8 did not answer.

One dead resolver with no fallback stops all mail, in both directions, while everything else on the server keeps working.

Why 8.8.8.8 was unreachable from that network is a question for the network team, and I won't guess at it here. What mattered on the day was proving that the resolver, and only the resolver, was the problem.

How to Actually Fix This

Everything below is read-only until Step 3, which I mark clearly.

Step 1: Read the Exim log for DNS timeouts.

```
grep "did not complete" /var/log/exim_mainlog | tail -20
```

If you see host lookup for ... did not complete (DNS timeout?), stop reading other logs. You are looking at a DNS problem, not a mail problem.

Step 2: Prove it is the resolver, not DNS as a whole.

First, ask the resolver the server is configured to use:

```
host google.com 8.8.8.8
```

On this server it hung and then failed with a timeout. Then ask a different public resolver the same question:

```
host google.com 1.1.1.1
```

That one answered at once.

Same server, same network, same name. One resolver answers, one does not. **This error is not a bug. It is the verification.** It tells you the server's network and DNS in general are fine, and only the configured resolver is out of reach.

Step 3 (changes state): Point the server at resolvers it can reach.

Back up the file first:

```
cp /etc/resolv.conf /etc/resolv.conf.bak
```

Then set two resolvers instead of one. On this server we used:

```
nameserver 1.1.1.1
nameserver 1.0.0.1
```

Two lines, not one. A second resolver is the fallback the server never had.

Step 4 (changes state): Flush the queue.

The mail that piled up during the outage is still sitting in the queue. Check how much:

```
exim -bpc
```

Then force a delivery run of everything in it:

```
exim -qff
```

Step 5: Verify — in both directions.

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

Send from Roundcube to an outside mailbox. The 451 must be gone. Then send from an outside mailbox into the server, and follow the log:

```
tail -f /var/log/exim_mainlog
```

You want to see the smart host resolve and the outbound message delivered, and the inbound message accepted without a sender verify failure. Run `exim -bpc` again a few minutes later — the queue should be falling. In our case mail flowed in both directions as soon as the resolver changed.

Step 6 (changes state): Make it survive a reboot.

This is the step almost everyone skips, and it is the one that brings the outage back after the next reboot.

On Rocky Linux 8, `/etc/resolv.conf` is written by NetworkManager. Edit it by hand and it works — until the next reboot or network restart, when NetworkManager writes `8.8.8.8` straight back.

Find the connection name:

```
nmcli con show
```

Then set the resolvers on the connection itself:

```
nmcli con mod "<connection-name>" ipv4.dns "1.1.1.1 1.0.0.1"
```

Check that it is stored:

```
nmcli -g ipv4.dns con show "<connection-name>"
```

It should print `1.1.1.1,1.0.0.1`. Now the fix lives in the place that generates the file, not only in the file.

Plain takeaway: if a tool generates a config file, fix it in the tool, not in the file.

One Thing the DNS Fix Did Not Fix

One more failure was in the logs, and it had its own cause. Sender verify for the server's own no-reply address kept failing with "Mailbox is full / Blocks limit exceeded".

That mailbox was over its quota. A verify check against a full mailbox fails no matter how healthy DNS is.

I mention it because this is how outages hide a second problem. Once the big failure is fixed, look at what is still failing, and check whether it has its own cause. **A fix that clears the big failure has not cleared the small ones — it has made them easier to see.**

The Bigger Lesson — And a Caveat From My Last Article

A mail server is a DNS client that also happens to deliver mail. Treat its resolver as part of the mail design, not as a network detail.

There is one caveat I owe readers. In my last article on Spamhaus and the open-resolver error, the lesson was that a mail server should **not** send its blacklist lookups through big public resolvers, because Spamhaus refuses them. Switching to 1.1.1.1 was the right move on this day, because it brought mail back within minutes. But if the server checks Spamhaus blocklists, a public resolver will break those lookups. The long-term design for that case is a local recursive resolver on the server, with the network allowing it to reach the internet on port 53.

The two cases teach the same thing from opposite ends: **know exactly which resolver your mail server asks, and know that it will answer.**

Here is the short checklist I now run on any mail server I inherit:

- 1 **Count the resolvers.** `cat /etc/resolv.conf` must show at least two nameserver lines.
- 2 **Test each one directly.** `host google.com <resolver-ip>` for every line in the file. Each must answer.
- 3 **Search for DNS timeouts.** `grep -c "did not complete" /var/log/exim_mainlog` should be zero or close to it.
- 4 **Check who writes the file.** `nmcli -g ipv4.dns con show "<connection-name>"` must match what is in `resolv.conf`.
- 5 **Watch the queue.** `exim -bpc` on a normal day tells you what "normal" is, so a growing queue stands out.
- 6 **Check system mailbox quotas.** WHM › Account Information › List Accounts, then the account's Email Accounts page, for any no-reply or system address near its limit.

Each item takes seconds. Together they turn "email is down" into a two-minute diagnosis.

What I Tell Every Customer Now

When a customer shows me their mail server design, I ask one question:

"If the DNS resolver in `resolv.conf` stopped answering right now, what would happen to your mail?"

If the answer is "it would fall back to the second resolver, and we test both", they are in good shape. If the answer is "there is only one, and it's Google, so it never goes down", we're not done yet.

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

What does "SMTP Error (451) Failed to add recipient - Temporary local problem" mean?

It means the mail server could not finish its own work for that recipient and deferred the message instead of rejecting it. On cPanel servers a common cause is a DNS lookup that timed out, for example

the lookup of the smart host or of the sender's domain. Check the Exim log for "did not complete (DNS timeout?)".

Why does a DNS problem stop both sending and receiving?

Exim needs DNS to find the smart host or recipient MX for outbound mail. It also needs DNS for sender verification on inbound mail. If the only resolver does not answer, both flows fail at the same time, while websites on the same server keep working.

How do I test if my server's DNS resolver works?

Run `host google.com <resolver-ip>` for each nameserver listed in `/etc/resolv.conf`. Each one should answer at once. If one times out while another public resolver answers, the configured resolver is unreachable from that network.

Why did my resolv.conf change back after a reboot?

On Rocky Linux, AlmaLinux and RHEL 8, NetworkManager usually writes `/etc/resolv.conf`. A manual edit is overwritten on the next reboot or network restart. Set the DNS servers on the connection with `nmcli con mod "<connection-name>" ipv4.dns "..."` so the change persists.

Should a mail server use 8.8.8.8 or 1.1.1.1 as its resolver?

A public resolver can work for basic mail flow, but always list at least two and test them. If the server checks Spamhaus blocklists, public resolvers will break those lookups, so a local recursive resolver is the better long-term design.

What does "Could not complete sender verify" mean in Track Delivery?

The server tried to confirm the sender's address before accepting the message and could not finish the check. A DNS timeout causes it for every sender at once. A full or missing mailbox causes it for one address only.

Read online: <https://quabin.com/exim-451-temporary-local-problem-dns-resolver/>