Message-ID: <afRr2BAmpEL8GnJM@donburi.himad.notcom.org>
Date: Fri, 1 May 2026 12:25:09 +0300
From: Valtteri Vuorikoski <vuori@...com.org>
To: oss-security@...ts.openwall.com
Subject: CVE-2026-42167: SQL injection in ProFTPd prior to 1.3.9a

A third party advisory was posted announcing CVE-2026-42167, an SQL injection vulnerability in ProFTPd versions prior to 1.3.9a, at <https://zeropath.com/blog/proftpd-cve-2026-42167-auth-bypass-privesc-rce>. Exploitation requires use of mod_sql, which is apparently the case in many prepackaged shared hosting setups. (Reportedly this is also fixed in the pre-release version 1.3.10rc1.)

The official site <https://www.proftpd.org> seems to be down at the moment so I don't know if or how this has been officially announced. Quotes from the above advisory:

> ZeroPath Research discovered a SQL injection vulnerability in ProFTPD's mod_sql extension. Depending on configuration, the flaw can be exploited before authentication and may lead to authentication bypass, privilege escalation, or remote code execution.
>
> MITRE has assigned the flaw CVE-2026-42167 and rated it 8.1 on the CVSSv3 severity scale.
>
> [...]
>
> Timeline
>
> 2026-03-28 — Issue reported to ProFTPD maintainers
> 2026-04-07 — ProFTPD maintainers and ZeroPath work to verify patch
> 2026-04-24 — CVE-2026-42167 issued
> 2026-04-27 — Commit af90843baf7dcb8c6be1e5261be2d0b5b5850673 fixes issue
> 2026-04-27 — 1.3.9a released with fix
>
> [...]
>
> mod_sql
>
> ProFTPD comes bundled with the mod_sql extension. When enabled, this extension can power a wide range of functionality, from quota tracking, to ban lists, to authentication.
>
> Two capabilities are especially important for understanding CVE-2026-42167 and its impact.
>
> Authentication
>
> Using the SQLAuthenticate and SQLUserInfo directives, an admin can configure ProFTPD to authenticate users against a SQL table instead of the local /etc/passwd file. This can come in handy for use cases like web hosting — it's not necessary to maintain full valid unix users for every FTP user, and it's quick to update user records in a centralized database.
>
> Logging
>
> Using the SQLNamedQuery and SQLLog statements, an admin can configure ProFTPD to store its logs in a SQL database where they can be easily accessed and aggregated. Like its SQL authentication feature, this is especially useful for hosting services.
>
> The Flaw: CVE-2026-42167
>
> Logging: Key Attack Surface

```
    Admins configure what gets logged to SQL with statements like these:

    SQLNamedQuery log_activity INSERT "'%U', '%r', '%m'" activity_log
    SQLLog        *          log_activity
    SQLLog        ERR_*      log_activity

    SQLLog selects particular commands to log. In this case all commands are
    logged. SQLNamedQuery specifies how to insert log entries into the database.

    Critically, the SQLNamedQuery includes magic % expansions, which get
    replaced by data from the request. Many of these expansions are potentially
    attacker-controlled, including:

    Variable    Meaning
    %A  anonymous-login password string
    %J  command parameters (everything after the verb)
    %S  response message string (may include attacker input echoed back in errors)
    %U  original username from USER (set before auth, available even on failed login)
    %d  directory name (last path component)
    %l  RFC 1413 ident response (attacker-controlled if they run identd)
    %m  FTP method/verb (attacker chooses which command to send)
    %r  full FTP command (verb + args)
    %u  authenticated username
    %{basename} filename component of the path argument, no directory prefix

    The attack surface here is obvious. Can an attacker use any of the parameters
    they control to slip an injection into an admin-configured SQLNamedQuery logging
    statement?

[Abbreviated code samples demonstrating that the answer is yes]

    Impact: RCE, Auth Bypass, Privesc and More Access Necessary

    What an attacker can do with this vulnerability depends on the admin's ProFTPD
    config. If the admin hasn't enabled mod_sql at all, or has not configured
    mod_sql-based logging, their instances are not vulnerable.

    If mod_sql logging is enabled, the access an attacker needs to get up to
    mischief depends on how that logging is configured.

    If pre-auth verbs like USER are logged, and that logging includes
    attacker-controlled values like %U (the username), an attacker only needs
    network access to the ProFTPD instance.

    If post-auth verbs like STOR are logged in a way that includes
    attacker-controlled values, like the filename (%f), then the attacker must
    authenticate to exploit the issue (but this authentication can include anonymous
    FTP login if the server is configured for anonymous access).

    RCE

    When ProFTPD connects to Postgres with superuser privileges, the impact extends
    beyond database access. Existing Postgres command-execution primitives allow SQL
    injection to be escalated to remote code execution in this configuration. (See
    POCs for more details.)  Auth Bypass And Privilege Escalation

    In the more common case, where a non-Postgres datastore is used, or ProFTPD is
    not authenticating to Postgres with a superuser, attackers can bypass
    authentication or expand their privileges if mod_sql is configured for
    authentication via the SQLAuthenticate directive.

    The malicious user simply inserts a record into the users table with the
    privilege, home directory and password that they desire. They then login as this
    user with the password they set.

    If pre-auth input, like username, are logged, this means the attacker can bypass
    authentication altogether. Even in cases where the attacker can only insert a
    user record after authenticating, they can significantly expand their privilege
    – e.g. setting their home directory to / so that they can browse and download
    the entire filesystem, not just a constrained directory within it.

[...]

    Mitigation
```

```
        Upgrade ProFTPD to at least 1.3.9a
        If upgrade is not possible, disable logging via mod_sql
        Monitor ProFTPD instances for suspicious activity

[...]
```