# What is SSHHole?

SSHHole (also written SSHole) is CVE-2026-123158, a password authentication bypass in OpenSSH sshd on Linux systems that use musl libc without PAM, such as Alpine and many embedded builds. musl's crypt() returns the string "*" when a password is longer than 256 bytes, and "*" is also what /etc/shadow stores for accounts with no password, so the comparison matches and the login succeeds. Fixed in Portable OpenSSH 10.6p2 on November 5, 2026; CISA added it to its KEV catalog on November 9. Setting PasswordAuthentication no closes it.

## HOW IT WORKS

### 01 How a long password opens a locked account

Without PAM, sshd checks a password itself. It reads the account's stored hash from /etc/shadow-, runs the submitted password through `crypt()` with that hash as the salt, and compares the two strings. If they match, you are in.

- The error value. musl's `crypt()` refuses to hash keys longer than 256 bytes and returns the string `*` to signal the error.
- The locked account. `*` is also what the password field holds for accounts that have no password: key-only users, service accounts and, on recent Alpine images, `root`.
- The match. sshd compares the error string `*` with the stored `*`, they are equal, and the password check passes.

In practice the attacker sends a password between 257 and 1,024 bytes for an account whose shadow entry is `*`. On Alpine the attack also needs at least one other account on the box with a real `$5$` or `$6$` hash, which almost every server has. Accounts locked with `!` instead of `*` are not affected. This is the same shape as many authentication bypass bugs: the code treats an error value as a valid answer.

### 02 Who is exposed

A server is exposed when all of these hold:

- It runs OpenSSH older than 10.6p2 on Linux with musl libc.
- sshd is built without PAM, or PAM is present but `UsePAM no` is set.

## HOW TO DEFEND

- Turn off password authentication now. Set `PasswordAuthentication no` in `sshd_config`, restart sshd and confirm with `sshd -T | grep -i passwordauthentication`. This closes the bug on every affected system, patched or not.
- Update OpenSSH. Upstream fixed it in 10.6p2. Alpine shipped fixed packages for every supported branch on November 5 (Alpine 3.21.10, 3.22.8, 3.23.8 and 3.24.4 include them). Upgrading the package does not restart a running sshd, so restart it.
- Lock accounts the safe way. Accounts that should never log in with a password are safer locked with `!` than left as `*`.
- Check for a past login. Search your auth log (on Alpine, `/var/log/messages`) for `Accepted password for` lines naming an account that has no password, such as `alpine`, `root` or a key-only user. If you find one, review `doas` and `sudo` use, new users, new files and authorized_keys entries, because a quick persistence step after the login is the obvious next move.
- Ask your device vendors. For appliances you cannot patch yourself, ask the vendor whether the firmware uses musl and password SSH, and keep the management interface off untrusted networks until they ship a fix.

- `PasswordAuthentication` is `yes`.
- At least one account has `*` in its password field.

On Alpine, `root` ships as `root:*`, but the OpenSSH default `PermitRootLogin prohibit-password` still blocks it, so root is only reachable where someone set `PermitRootLogin yes`. Other `*` accounts are the real risk: users created without a password for key-only login, directory-service users, and the `alpine` user in some Alpine cloud images, which also has passwordless `doas`, so a login there is root in one step. Containers that do not run sshd are not affected; appliance and IoT firmware usually is, and is the slowest to get a fix.

A working proof of concept circulated within days of the fix, scanning for it began soon after, and CISA added CVE-2026-123158 to its Known Exploited Vulnerabilities catalog on November 9, 2026.

## 03 Where SSHHole fits among named vulnerabilities

SSHHole is not one of the X2Shell vulnerabilities-, which end in code execution through a widely deployed component like PHP2Shell. It is an authentication bypass, and the result is closer to a valid login than to an exploit: the attacker gets an ordinary shell as the account it chose, with no crash or odd process to give it away. From there the usual Linux privilege escalation paths apply. The lesson is old and keeps coming back: an error value that looks like a valid answer is an authentication bypass waiting for the right input.

### SOURCES

[1] OpenSSH 10.6p2 release notes

[2] OpenSSH fix commit

[3] Alpine security tracker: CVE-2026-123158

[4] NVD: CVE-2026-123158

**Find the way in before an attacker does.**

`securelayer7.net/learn/web-exploitation/what-is-sshhole`

Open online