Small. Simple. Secure.

Alpine Linux is a security-oriented, lightweight Linux distribution based on musl libc and busybox.

2026-10-28

GITLAB SECURITY INCIDENT: ATTEMPTED EXPLOITATION OF CVE-2026-97229

Update October 30: GitLab is open again. See GitLab is back: what you need to do for what to set up again and the new SSH host key fingerprints.

On October 20 and 21, an attacker tried to exploit a GitLab vulnerability (CVE-2026-97229) on our instance, a week after the fix was released. The attack came close to working. We have found no evidence that it succeeded, but we cannot rule it out, so we treated the instance as compromised and rebuilt it from scratch. Our checks show that the aports repository, packages, and the October 21 releases were not tampered with.

GitLab is being tested before we open it to the public again. All passwords have been reset, so when it is back you will need to set a new one with “Forgot your password?” on the sign-in page. We will post exactly what you need to do when it reopens. In the meantime: if you used the same password for your gitlab.alpinelinux.org account anywhere else, change it there. We have seen automated login attempts against our GitLab with passwords leaked from other sites.

If you have questions or concerns please reach out to us at infra@alpinelinux.org.

WHAT HAPPENED

The vulnerability: CVE-2026-97229 is a critical (CVSS 10.0) vulnerability in how GitLab evaluates regular expressions in CI configuration. The underlying bug is in Onigmo, the regular expression engine built into Ruby. GitLab published fixed releases on October 12, but the fix itself is a set of patches to the Ruby that GitLab bundles in its own packages and container images. Upstream Ruby shipped the fix on October 23, in Ruby 3.3.13, 3.4.12 and 4.0.9.

The attack: On October 20 and 21, a newly registered account sent crafted CI configurations to our GitLab instance, which was still running 19.3.4 and did not have the October 12 fix. This crashed GitLab worker processes 8 times. In 4 of those crashes the attacker got partway through the exploit and failed at the next step. The attacker also downloaded our GitLab container image, which suggests they were tuning the exploit to our exact build. We have found no evidence that the attacker ran code on the server, but we cannot rule it out.

Our upgrade didn't fix the vulnerability: Upgrading to GitLab 19.3.6 on October 22 didn't fix the vulnerability. The fix isn't in GitLab's own code but in a patch to Ruby, which GitLab applies only in its official packages. Our image used standard Ruby, so it didn't get the patch.

If you run GitLab with your own Ruby (from source or in your own container image) rather than GitLab's packages, upgrading GitLab is not enough: you also need Ruby 3.3.13, 3.4.12, 4.0.9 or later.

If you use our alpinelinux/gitlab container image, pull `alpinelinux/gitlab:19.3.6` again. The image published on October 27 contains Ruby 3.3.13. Any 19.3.6 image you pulled before October 27, and all older tags, are vulnerable.

INCIDENT TIMELINE

Date	Event
Oct 29	GitLab reopened, upgraded to 19.3.7
Oct 28	GitLab rebuild complete; all passwords reset and GitLab's secret keys replaced
Oct 27	GitLab closed to everyone except developers and build servers; rebuild started
Oct 26	Found the instance still vulnerable after the upgrade
Oct 25 to 26	Attacker returns and tries to get back in; attempts fail
Oct 24	aports, packages and Oct 21 releases verified against outside records
Oct 23	Attacker's account blocked; Ruby releases the fix
Oct 22	Upgraded to GitLab 19.3.6
Oct 21	Main attack; GitLab crashes, 502 errors and failed CI jobs
Oct 20	Attacker registers an account and starts probing

Oct 13	CVE-2026-97229 published; we upgraded to GitLab 19.3.4 instead of 19.3.5 (which had the fix) by mistake
Oct 12	GitLab releases the fix

WHAT WE KNOW AND WHAT WE CAN'T RULE OUT

What we found

- **Alpine releases and packages were not modified.** The October 21 stable releases (3.21.9, 3.22.7, 3.23.7, 3.24.3), including their ISO images and the minirootfs tarballs the official `alpine` Docker images are built from, and all packages built from October 20 to 24 came from aports commits we verified against records outside GitLab. See the FAQ below for details.
- **No signing keys were exposed.** The keys that sign Alpine packages and releases are not, and never have been, stored on the GitLab server.
- No evidence that the attacker's code ran: timing probes showed no delays, and we found no unexpected files, keys, hooks or processes.
- No private data was returned to the attacker through any web or API request we can see.

Why we can't be sure

- A successful exploit would leave no crash and no trace in the web logs.
- Some logs from the attack day are gone (background job logs, the old container's filesystem).
- We had no logging of outbound network connections, and our hosting provider keeps none either.

What could have been exposed if the attack succeeded

- Email addresses, names and IP addresses of all users
- Password hashes
- 2FA secrets, CI/CD variables and runner tokens
- Webhook, integration and push mirror credentials
- Personal access tokens (stored as hashes)
- Private repositories, and confidential issues and merge requests

FAQ

WHY DID WE REBUILD GITLAB IF WE WERE ABLE TO MITIGATE THE ATTACK?

Because we cannot rule out code execution, and the evidence we would need to confirm it no longer exists, we took the harder route and rebuilt the GitLab instance from scratch. This removes any foothold a possibly successful exploit could have left behind.

While GitLab is closed, our package mirrors, cgit, and mailing lists/IRC are available as normal.

ARE ALPINE PACKAGES AND RELEASES AFFECTED?

We have found no sign of tampering in aports or the releases.

Every aports commit added October 20 to 22 arrived through a recorded push by a known developer. We checked this against records the attacker could not reach: a developer's clone and all 45 builders. Every package built from October 20 to 24, on every architecture, came from commits as published on git.alpinelinux.org. The October 21 releases (3.21.9, 3.22.7, 3.23.7, 3.24.3) were built from those same verified commits.

We ran these checks on October 24, covering commits from October 20 to 22 and packages built up to October 24. Our GitLab stayed vulnerable until we closed it on October 27, so we are extending the same checks to that date.

IS ALPINE'S RUBY PACKAGE AFFECTED?

The bug is in Ruby's regular expression engine, so any program that compiles regular expressions from untrusted input is exposed, not only GitLab. Alpine 3.22, 3.23, 3.24 and edge ship Ruby 3.4.12, and Alpine 3.21 ships Ruby 3.3.13; both include the fix. Run `apk upgrade` to get it.

WHAT ARE WE CHANGING IN RESPONSE TO THIS EVENT?

- **Fixed Ruby:** the rebuilt instance runs a Ruby with the fix (3.3.13 or later).
- **Faster patching:** apply critical GitLab security releases within a day, and check each release for Ruby or other component patches our own build must apply.
- **Better logging:** record outbound connections from the GitLab host and database connections, and keep logs longer than 14 days.
- **Signed release tags**, so releases can be verified independently of GitLab.

© Copyright 2026 Alpine Linux Development Team all rights reserved | Privacy Policy