April 28, 2020

GHSL-2020-085: Open redirect vulnerability in Sourcegraph - CVE-2020-12283

Alvaro Munoz

Summary

An open redirect vulnerability has been found on Sourcegraph due to improper validation in the `SafeRedirectURL` method, as a consequence an attacker could potentially redirect a victim to any arbitrary URL and access their OAUTH token.

Product

Sourcegraph

CVE

CVE-2020-12283

Tested Version

Tested on master branch up to afcd7cf010d2508bfb8393fcf9c4497dc0099180

Details

An open redirect is a security vulnerability in which a website endpoint accepts a URL as input and and then redirects to the user-provided URL. This type of vulnerability can be used for e.g. phishing attacks in which an attacker abuses the trust relationship a victim has with the redirecting site to redirect to a malicious site. A more serious attack may occur when an application implements an oauth flow. In this scenario an attacker can abuse e.g. external service authentication (think “Login with Facebook/Google/GitHub/etc”) to redirect users to an attacker controlled site, where they will then steal the partial oauth token.

A good example of this type of vulnerability can be found in this OAuth2 advisory

`SafeRedirectURL` relies on `url.Parse` and `u.Path` to extract the relative path. `SafeRedirectURL` will transform any absolute URLs starting with `//` into `/` and any absolute URLs starting with `/\\` will be URL encoded into `/%5C`. However, when `url.Parse` parses the URL it will NOT normalize double slashes (e.g: `u.Path` will return `//bar` for `http://foo.com//bar`). An attacker can abuse this behavior by crafting an URL like `//foo//example.com` so that `u.Path` will return `//example.com` which, when sent to the browser, will make it visit the absolute URL `http://example.com`.

```
func SafeRedirectURL(urlStr string) string {
        u, err := url.Parse(urlStr)
        if err != nil || !strings.HasPrefix(u.Path, "/") {
                return "/"
        }
```

The vulnerability was found using the following CodeQL query:

```
/**
 * @name Open redirect due to sanitzation bypass
 * @kind path-problem
 * @problem.severity medium
 * @id go/example/hasprefix
 */

import go
import DataFlow::PathGraph

 predicate prefixCheck(StringOps::HasPrefix call, DataFlow::Node checked, Variable v, string prefix) {
   checked = call.getBaseString() and
   prefix = call.getSubstring().getStringValue() and
   v.getARead() = checked
 }

predicate insuffcientPrefixCheck(StringOps::HasPrefix call, DataFlow::Node checked, Variable v) {
    prefixCheck(call, checked, v, "/")  and
    (not prefixCheck(_, _, v, "//") or not prefixCheck(_, _, v, "/\\"))

}

class BadRedirectConfig extends TaintTracking::Configuration {
  BadRedirectConfig() { this = "BadRedirectConfig" }
  override predicate isSource(DataFlow::Node source) {
    source instanceof UntrustedFlowSource
  }
  override predicate isSink(DataFlow::Node sink) {
        insuffcientPrefixCheck(_, sink, _)
  }
}

from DataFlow::PathNode source, DataFlow::PathNode sink, BadRedirectConfig cfg
where cfg.hasFlowPath(source, sink)
select sink, source, sink, "Bad redirect check on untrusted data from $@", source, "this source"
```

Impact

The full impact of this vulnerability depends on the context of use. While open redirect issues can aid phishing attacks, it also seems that `SafeRedirectURL` is used in Sourcegraph’s OAuth flow which may lead to token hijacks.

Remediation

The vulnerability has been fixed in Sourcegraph v3.14.4 and v3.15.1

Coordinated Disclosure Timeline

- 2020-04-23: reported to security@sourcegraph.com
- 2020-04-23: acknowledge received from Sourcegraph security team
- 2020-04-24: vulnerability fixed and deployed sourcegraph/sourcegraph#10167
- 2020-04-27: CVE assigned CVE-2020-12283
- 2020-04-28: Patch releases for version 3.14.4 and 3.15.1.
- 2020-04-30: Sourcegraph issued a GitHub security advisory and notified all affected users.

References

- Sourcegraph advisory
- CVE-2020-12283
- Vulnerability fix

Credit

This issue was discovered and reported by GHSL team members @nicowaisman (Nico Waisman), @pwntester (Alvaro Munoz) and @sauyon (Sauyon Lee).

Contact

You can contact the GHSL team at `securitylab@github.com`, please include the `GHSL-2020-085` in any communication regarding this issue.