Arbitrary File Write via Archive Extraction (Zip Slip)

Affecting github.com/u-root/u-root/pkg/cpio package, versions **<0.9.0**

INTRODUCED: 1 SEP 2020 CVE-2020-7666 CWE-22 FIRST ADDED BY SNYK

Share

How to fix?

Upgrade `github.com/u-root/u-root/pkg/cpio` to version 0.9.0 or higher.

Overview

github.com/u-root/u-root/pkg/cpio is a package that provides Go versions of standard Linux tools and bootloaders. It also provides tools for compiling Go programs in a single binary and creating initramfs images.

Affected versions of this package are vulnerable to Arbitrary File Write via Archive Extraction (Zip Slip). It is vulnerable to leading, non-leading relative path traversal attacks and symlink based (relative and absolute) path traversal attacks in cpio file extraction.

PoC

// poc.go:

```
package main import ( "io" "log" "os" "github.com/u-root/u-root/pkg/cpio" ) func main() { archiver, err := cpio.Format("newc") if err != nil { log.Fatalf("Format -H newc not supported: %v", err) } var inums map[uint64]string inums = make(map[uint64]string) rr := archiver.Reader(os.Stdin) for { rec, err := rr.ReadRecord() if err == io.EOF { break } if err != nil { log.Fatalf(&amp;quot;error reading records: %v&amp;quot;, err) } if rec.Info.FileSize == 0 { if _, ok := inums[rec.Info.Ino]; ok { err := os.Link(inums[rec.Info.Ino], rec.Name) if err != nil { log.Fatal(err) } continue } } inums[rec.Info.Ino] = rec.Name if err := cpio.CreateFile(rec); err != nil { log.Printf(&amp;quot;Creating %q failed: %v&amp;quot;, rec.Name, err) } }
```

- Build the executable `go build poc.go`
- Run `./poc < archive.cpio` with "archive.cpio" being a cpio archive that includes at least one of the following:

1. file with filepath that uses leading or non-leading "../"
2. file symlink that point outside of the current directory (relative or absolute)
3. directory symlink that point outside of the current directory (relative or absolute) followed by a file under that directory

Details

It is exploited using a specially crafted zip archive, that holds path traversal filenames. When exploited, a filename in a malicious archive is concatenated to the target extraction directory, which results in the final path ending up outside of the target folder. For instance, a zip may hold a file with a "../../file.exe" location and thus break out of the target folder. If an executable or a configuration file is overwritten with a file containing malicious code, the problem can turn into an arbitrary code execution issue quite easily.

The following is an example of a zip archive with one benign file and one malicious file. Extracting the malicous file will result in traversing out of the target folder, ending up in `/root/.ssh/` overwriting the `authorized_keys` file:

```
+2018-04-15 22:04:29 ..... 19 19 good.txt

+2018-04-15 22:04:42 ..... 20 20 ../../../../../../root/.ssh/authorized_keys
```

References

- GitHub Commit
- GitHub PR
- GitHub PR

Snyk CVSS

Exploit Maturity	Proof of concept
Attack Complexity	Low
Integrity	HIGH

See more

NVD 7.5 HIGH

Do your applications use this vulnerable package?

In a few clicks we can analyze your entire application and see what components are vulnerable in your application, and suggest you quick fixes.

Test your applications

Snyk Learn

Learn about Arbitrary File Write via Archive Extraction (Zip Slip) vulnerabilities in an interactive lesson.

Start learning

Snyk ID	SNYK-GOLANG-GITHUBCOMUROOTUROOTPKGCPIO-570440
Published	1 Sep 2020
Disclosed	1 Sep 2020
Credit	Georgios Gkitsas of Snyk Security Team

Report a new vulnerability | Found a mistake?

RESOURCES

Vulnerability DB

Documentation

Disclosed Vulnerabilities

Blog

FAQs

COMPANY

About

Jobs

Contact

Policies

Do Not Sell My Personal Information

CONTACT US

Support

Report a new vuln

Press Kit

Events

FIND US ONLINE

TRACK OUR DEVELOPMENT

Registered in England and Wales. Company number: 09677925

Registered address: Highlands House, Basingstoke Road, Spencers Wood, Reading, Berkshire, RG7 1NT.