New issue

Jump to bottom

[SECURITY] Fix Zip Slip Vulnerability #81

Merged

davemckain merged 1 commit into `davemckain:master` from `BulkSecurityGeneratorProjectV2:fix/JLL/zip-slip-vulnerability` 29 days ago

JLLeitschuh commented on Sep 8 Contributor

Security Vulnerability Fix

This pull request fixes a Zip Slip vulnerability either due to an insufficient, or missing guard when unzipping zip files.

Even if you deem, as the maintainer of this project, this is not necessarily fixing a security vulnerability, it is still, most likely, a valid security hardening.

Preamble

Impact

This issue allows a malicious zip file to potentially break out of the expected destination directory, writing contents into arbitrary locations on the file system.
Overwriting certain files/directories could allow an attacker to achieve remote code execution on a target system by exploiting this vulnerability.

Why?

The best description of Zip-Slip can be found in the white paper published by Snyk: Zip Slip Vulnerability

But I had a guard in place, why wasn't it sufficient?

If the changes you see are a change to the guard, not the addition of a new guard, this is probably because this code contains a Zip-Slip vulnerability due to a partial path traversal vulnerability.

To demonstrate this vulnerability, consider `"/usr/outnot".startsWith("/usr/out")`.
The check is bypassed although `/outnot` is not under the `/out` directory.
It's important to understand that the terminating slash may be removed when using various `String` representations of the `File` object.
For example, on Linux, `println(new File("/var"))` will print `/var`, but `println(new File("/var", "/")` will print `/var/`;
however, `println(new File("/var", "/").getCanonicalPath())` will print `/var`.

The Fix

Implementing a guard comparing paths with the method `java.nio.files.Path#startsWith` will adequately protect against this vulnerability.

For example: `file.getCanonicalFile().toPath().startsWith(BASE_DIRECTORY)` or `file.getCanonicalFile().toPath().startsWith(BASE_DIRECTORY_FILE.getCanonicalFile().toPath())`

Other Examples

- CVE-2018-1002201 - zeroturnaround/zt-zip
- CVE-2018-1002202 - srikanth-lingala/zip4j
- CVE-2018-8009 - apache/hadoop

➡️ Vulnerability Disclosure ⬅️

👋 Vulnerability disclosure is a super important part of the vulnerability handling process and should not be skipped! This may be completely new to you, and that's okay, I'm here to assist!

First question, do we need to perform vulnerability disclosure? It depends!

1. Is the vulnerable code only in tests or example code? No disclosure required!
2. Is the vulnerable code in code shipped to your end users? Vulnerability disclosure is probably required!

For partial path traversal, consider if user-supplied input could ever flow to this logic. If user-supplied input could reach this conditional, it's insufficient and, as such, most likely a vulnerability.

Vulnerability Disclosure How-To

You have a few options options to perform vulnerability disclosure. However, I'd like to suggest the following 2 options:

1. Request a CVE number from GitHub by creating a repository-level GitHub Security Advisory. This has the advantage that, if you provide sufficient information, GitHub will automatically generate Dependabot alerts for your downstream consumers, resolving this vulnerability more quickly.
2. Reach out to the team at Snyk to assist with CVE issuance. They can be reached at the Snyk's Disclosure Email. Note: Please include `JLLeitschuh Disclosure` in the subject of your email so it is not missed.

Detecting this and Future Vulnerabilities

You can automatically detect future vulnerabilities like this by enabling the free (for open-source) GitHub Action.

I'm not an employee of GitHub, I'm simply an open-source security researcher.

Source

This contribution was automatically generated with an OpenRewrite refactoring recipe, which was lovingly handcrafted to bring this security fix to your repository.

The source code that generated this PR can be found here:
Zip Slip

Why didn't you disclose privately (ie. coordinated disclosure)?

This PR was automatically generated, in-bulk, and sent to this project as well as many others, all at the same time.

This is technically what is called a "Full Disclosure" in vulnerability disclosure, and I agree it's less than ideal. If GitHub offered a way to create private pull requests to submit pull requests, I'd leverage it, but that infrastructure, sadly, doesn't exist yet.

The problem is that, as an open source software security researcher, I (exactly like open source maintainers), I only have so much time in a day. I'm able to find vulnerabilities impacting hundreds, or sometimes thousands of open source projects with tools like GitHub Code Search and CodeQL. The problem is that my knowledge of vulnerabilities doesn't scale very well.

Individualized vulnerability disclosure takes time and care. It's a long and tedious process, and I have a significant amount of experience with it (I have over 50 CVEs to my name). Even tracking down the reporting channel (email, Jira, etc..) can take time and isn't automatable. Unfortunately, when facing problems of this scale, individual reporting doesn't work well either.

Additionally, if I just spam out emails or issues, I'll just overwhelm already over-taxed maintainers, I don't want to do this either.

By creating a pull request, I am aiming to provide maintainers something highly actionable to actually fix the identified vulnerability; a pull request.

There's a larger discussion on this topic that can be found here: JLLeitschuh/security-research#12

Opting Out

If you'd like to opt out of future automated security vulnerability fixes like this, please consider adding a file called `.github/GH-ROBOTS.txt` to your repository with the line:

```
User-agent: JLLeitschuh/security-research
Disallow: *
```

This bot will respect the ROBOTS.txt format for future contributions.

Alternatively, if this project is no longer actively maintained, consider archiving the repository.

CLA Requirements

This section is only relevant if your project requires contributors to sign a Contributor License Agreement (CLA) for external contributions.

It is unlikely that I'll be able to directly sign CLAs. However, all contributed commits are already automatically signed off.

> The meaning of a signoff depends on the project, but it typically certifies that committer has the rights to submit this work under the same license and agrees to a Developer Certificate of Origin (see https://developercertificate.org/ for more information).
>
> - Git Commit Signoff documentation

If signing your organization's CLA is a strict-requirement for merging this contribution, please feel free to close this PR.

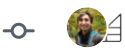
Sponsorship & Support

This contribution is sponsored by HUMAN Security Inc. and the new Dan Kaminsky Fellowship, a fellowship created to celebrate Dan's memory and legacy by funding open-source work that makes the world a better (and more secure) place.

This PR was generated by Moderne, a free-for-open source SaaS offering that uses format-preserving AST transformations to fix bugs, standardize code style, apply best practices, migrate library versions, and fix common security vulnerabilities at scale.

Tracking

All PR's generated as part of this fix are tracked here: JLLeitschuh/security-research#16



JLLeitschuh force-pushed the `fix/JLL/zip-slip-vulnerability` branch from **`efc5e41`** to **`1a46d6d`** 2 months ago

Compare

davemckain merged commit **`88d9786`** into `davemckain:master` 29 days ago

---

davemckain added a commit that referenced this pull request 29 days ago

`Tidied up pull request #81` ... `0319115`

davemckain commented 29 days ago — Owner

@JLLeitschuh: Many thanks for reporting this problem. I've merged in your pull request. (I've subsequently changed the handling of these bad ZIP entries to follow the existing "bad zip" handling.)

JLLeitschuh commented 29 days ago — Contributor — Author

Hi @davemckain,
Do you need any assistance with vulnerability disclosure and potential CVE issuance?

Reviewers

No reviews

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

None yet

2 participants