snakeyaml / snakeyaml / snakeyaml / Issues

Got StackOverflowError for many open unmatched brackets

Issue #525 RESOLVED

Andrey Somov created an issue 2022-04-26

No description provided.

clusterfuzz-testcase-minim...

Assignee	Andrey Somov
Type	bug
Priority	major
Status	resolved
Votes	0
Watchers	1

Comments (17)

Andrey Somov REPORTER

- attached clusterfuzz-testcase-minimized-YamlFuzzer-4626423186325504

2022-04-26

Andrey Somov REPORTER

- changed status to open

2022-04-28

Andrey Somov REPORTER

- changed status to resolved

it will be delivered in 1.31

2022-04-29

Maximilian Tews

Hi @Andrey Somov , this issue is related to CVE-2022-25857, right? Could you please share a little bit more information about the vulnerability such that others that use this library can assess if they are affected or not.

I am specifically interested to the following:

- The first thing I am wondering about is the file attached to this issue. It does not look like a yaml file to me. Could you provide a yaml file with that the issue could be reproduced?
- What needs to be done to exploit this vulnerability? Do I need to hand over a yaml file that specifies a list within a list within a list within a list... to org.yaml.snakeyaml.Yaml.load()?
- What are the consequences if the issue occurs? Is the only consequence a StackOverflowError that is thrown? That alone does not sound like a vulnerability to me. Are there other consequences like CPU/Memory consumption? Why was a CVE requested for this issue?

2022-09-05

Andrey Somov REPORTER

You should probably contact the `OSS-Fuzz` team. They provided the file with the message that SnakeYAML should fail quickly for such a file. It was of cause failing, but now it fails quicker, without an attempt to allocate the resources mentioned in the file.

2022-09-05

Robert Schaft

CVE-2022-25857 ist classified by the NIST with a CVSS 3.x base score of 7.5. The high score is based on the assumption, that it is used to parse Network data.

In most spring boot applications, snakeyaml is only used indirectly by spring to parse local trusted configuration files and would. So the warning is a false positive. But this is hard to prove without checking the whole code and configuration, that yaml format is indeed not used for untrusted data.

Nevertheless, the spring boot team has a stabiliy policy to not update the dependency to the fixed Version 1.31 to avoid breaking changes: CVE-2022-25857 - Upgrade to SnakeYAML 1.31 · Issue #32221 · spring-projects/spring-boot (github.com). This was already an issue with Version 1.26.

You can be proud, that your lib becomes a cornerstone of many applications in the internet. But that in turn requires some more support from you:

I kindly ask to overthink the release strategy of snakeyaml and provide long term support releases with hotfixes. For example version 1.30 could be marked as LTS release. And releasing this fix as Version 1.30.1 would allow the spring team to include it.

2022-09-06

Andrey Somov REPORTER

To give a binary another name or version does not have any effect on its contents.
The fact that a Spring team will accept 1.30.1 but does not accept 1.31 is not an issue for SnakeYAML.
There is no hotfix because **nothing was broken.**
Please correct me where I am wrong. I am disappointed with the quality of the NIST database and its classification.

2022-09-06

Robert Schaft

> There is no hotfix because **nothing was broken.**

I can see some commits regarding this issue.:

- `fc30078` - Restrict nested depth for collections to avoid DoS attacks
- `5150df1` - Add test for issue 525
- `bc7869b` - Make billionLaughsAttackTest.billionLaughsAttackExpanded() robust

They seem to fix something broken/vulnerable to me. I assume, that a release, that contains only the fixes for the worst issues and is declared as thus will be accepted by the spring team.

On the other hand, there are a few potential breaking changes in 1.31:

- `0468784` - force keyNode to be String for JavaBeans fixes issue #522
- `467bcc9` - expect node to be scalar when enforcings String keys
- `f46c45e` - fix build because some test dependency are Java 8 now

I don't see any hotfix releases in the history of snakeyaml. Are there issues related to creating a hotfix release?

> It was of cause failing, but now it fails quicker,

I would like to get more details on how it failed. An out of memory exception crashing the whole JVM is different to a Runtime Exception that is thrown when processing the document reached a certain reasonable CPU/Memory threshold. And "fail earlier" could be after one second instead of one hour CPU consumption.

I agree with you about the NIST classification. Since it is your component, you might be able to request an update to lower the severity. Unfortunately, I lack the experience on how to do that.

2022-09-06

Andrey Somov REPORTER

The fact that commits are present does NOT indicate that something was broken.
The message "make the test robust" does NOT improve the code.

fixed it. The fact that SnakeYAML added the restriction for depth does NOT improve the quality because users may set it to a very high value and still fail.
Can you please address the question to the appropriate team ?

2022-09-06

Maximilian Tews

Hi @Andrey Somov thanks for your answers. How do I contact the “OSS-Fuzz” team? It is unclear to me who the “OSS-Fuzz” team is and how it relates to snakeyaml. Who can I answer my questions?

- The first thing I am wondering about is the file attached to this issue. It does not look like a yaml file to me. Could you provide a yaml file with that the issue could be reproduced?
- What needs to be done to exploit this vulnerability? Do I need to hand over a yaml file that specifies a list within a list within a list within a list… to org.yaml.snakeyaml.Yaml.load()?
- What are the consequences if the issue occurs? Is the only consequence a StackOverflowError that is thrown? That alone does not sound like a vulnerability to me. Are there other consequences like CPU/Memory consumption? Why was a CVE requested for this issue?

2022-09-07

Andrey Somov REPORTER

1. I hope that a member of the OSS-Fuzz team can come and answer the questions.
2. The whole point is that the input is not YAML. It is trash. But trash may confuse the parser.
3. Most probably you are not affected at all (I do not know a single **real use case** when somebody is affected)

If you do not blindly open a socket to read any trash from any incoming input - you are totally safe.

2022-09-07

Robert Schaft

To my understanding, this finding here is the source of the CVE:

47024 - snakeyaml:YamlFuzzer: Uncaught exception in org.yaml.snakeyaml.composer.Composer.composeSequenceNode - oss-fuzz (chromium.org)

The CVE score seems to be based on the assumption, that you read any trash from incoming input, because it is marked as “Network accessible”, which is just wrong, because snakeyaml doesn’t provide or use a network interface (AFAIK). So the CVE score can use an update.

2022-09-07

Robert Schaft

> If you do not blindly open a socket to read any trash from any incoming input - you are totally safe.

But how should the code look like, that can decide, if a file is trash? The usual approach is, to feed it to a parser-library and check if the library yields a nice Object or throws a ThisIsTrashException.

The best choice of library for a spring-boot based project with yaml used input would be snakeyaml. But snakeyaml throws only an OutOfMemoryException (which shouldn’t be catched) - correct me if I am wrong. So snakeyaml is not totally free of liability to handle trash in an ordered fashion.

2022-09-07

2022-09-07

Robert Schaft

Looks similar to snakeyaml / snakeyaml / wiki / Billion laughs attack — Bitbucket. Maybe you can merge them.

But from security perspective it is not good to document those statements somewhere in a wiki.

A warning on the project entry page would help more. Something along:

> snakeyaml is not intended to process untrusted content. If fed with untrusted content, it might delete your data and kill all your processes!

After that, you can ask the OSS-Fuzz team to stop fuzzing around snakeyaml. Because fuzzing is all about generating ugly "untrusted" content.

2022-09-07

Andrey Somov **REPORTER**

I wish SnakeYAML is able to delete all the data and kill all the processes… Unfortunately, we are not there yet.

2022-09-07

Alexander Maslov

I thought this issue has been resolved already, but conversation seems to continue. @Robert Schaft , IMHO you are correct (I also thought about the point of how to decide if it is a trash or not before feeding it to parser) and I don't see a better place than the parser itself :)

We are not denying existence of those reported issues. But we are NOT GOING to rush fixing those, sorry. It maybe unfortunate for many, but it is what it is.
Whenever we have wiliness, time and courage to do something to remove those CVEs we do it.

But there is no ETA.

And, btw, any PRs are very much appreciated ;)

2022-09-08

Log in to comment