New issue

Jump to bottom

Use TreeMap in SimpleFacade to solve DoS vuln #390

Merged rossabaker merged 2 commits into `typelevel:main` from `nrktkt:patch-1` on Jan 2

nrktkt commented on Dec 15, 2021 Contributor

Scala's default `Map` is vulnerable to a DoS attack where an attacker basically fills up one leaf of the hash trie and causes resource exhaustion as more elements are added to the leaf with poor efficiency.
scala/bug#11203

`SimpleFacade` should be resistant to this as a "simple" implementation. Users can always implement their own `Facade` if they trust the input and need the performance edge.
Switching to `TreeMap` should solve this without any compatibility issues.

`Use TreeMap in SimpleFacade to solve DoS vuln` ✓ `e5ddb11`

rossabaker commented on Dec 16, 2021 Member

Thanks!

Circe uses a `LinkedHashMap`, which I think we'd have to wrap as an `immutable.Map` in `finish`. So does play-json. Even a wrapped `java.util.HashMap` is sufficient, if we're not trying to optimize iteration of the underlying map. I strongly suspect this would be faster than building `TreeMap`s. You are of course correct that users can implement their own, but maybe we can avoid a performance regression using the Java types.

There's also a `CollisionProofHashMap`. That may or may not be faster than the Java one, and could only work for Scala >= 2.13.

Seems like the mutable facade would also be vulnerable?

nrktkt commented on Dec 17, 2021 Contributor Author

I haven't looked deep at the implementations for how Circe or Play do that wrapping. But I'd guess that they turn the java map into some non-map type (I vaguely remember play is a Seq of tuples or something?), or have expensive mutations (cloning the underlying mutable map each time), or do something like `scala.collection.JavaConverters .toScala.toMap`, which would convert to the default HAMT and expose the vuln (I think Circe does on mutation).
None of the above are ideal; for my use case at least. Users would find the map follows the contract, but wouldn't perform as one would assume. So at the end of the day we need a safe immutable map and `TreeMap` is the only one I'm aware of.
Is there a way to more quickly build a different structure and then convert to `TreeMap`?

Unfortunately there's no immutable `CollisionProofHashMap`, so I think that leaves us in about the same situation as the java `java.util.HashMap` wrapped using `JavaConverters`.
I think the ideal solution would be an immutable `CollisionProofHashMap` in the standard library (and added to collections compat), but short of that `TreeMap` seems like what we're left with.

Let me know if that all sounds correct

 `the MutableFacade IS also vulnerable` ✓ `0707e25`

rossabaker approved these changes on Dec 30, 2021

View changes

 **rossabaker** left a comment Member

> Is there a way to more quickly build a different structure and then convert to TreeMap?

As long as the building can't be observed externally, using a `TreeMap.newBuilder` and converting at the end would probably be faster.

But we should fix the vulnerabilities first, and we can improve the performance second.

 **rossabaker** merged commit **19ceb95** into `typelevel:main` on Jan 2

View details

3 checks passed

Reviewers

 **rossabaker** ✓

Assignees

No one assigned

Labels

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

None yet

2 participants