

New issue

[Jump to bottom](#)Soundness issue in `Zip::next()` specialization #81740🔒 Closed Qwaz opened this issue on Feb 4, 2021 · 5 comments · Fixed by [#81741](#)

Labels A-iterators C-bug I-unsound P-high T-libs

Qwaz commented on Feb 4, 2021

Contributor

[rust/library/core/src/iter/adapters/zip.rs](#)
Lines 191 to 211 in [e708cbd](#)

```
191     #[inline]
192     fn next(&mut self) -> Option<(A::Item, B::Item)> {
193         if self.index < self.len {
194             let i = self.index;
195             self.index += 1;
196             // SAFETY: 'i' is smaller than `self.len`, thus smaller than `self.a.len()` and `self.b.len()`
197             unsafe {
198                 Some((self.a.__iterator_get_unchecked(i), self.b.__iterator_get_unchecked(i)))
199             }
200         } else if A::may_have_side_effect() && self.index < self.a.size() {
201             // match the base implementation's potential side effects
202             // SAFETY: we just checked that `self.index` < `self.a.len()`
```

[rust/library/core/src/iter/adapters/zip.rs](#)
Lines 395 to 396 in [e708cbd](#)

```
395     /// 2. If `self: !Clone`, then `get_unchecked` is never called with the same
396     ///     index on `self` more than once.
```

There is a panic safety issue in `Zip::next()` that allows to call `__iterator_get_unchecked()` to the same index twice. `__iterator_get_unchecked()` is called at line 204 and the `index` is updated at line 206. If line 204 panics, the index is not updated and the subsequent `next()` call will use the same index for `__iterator_get_unchecked()`. This violates the second safety requirement of `TrustedRandomAccess`.

Here is a [playground link](#) that demonstrates creating two mutable references to the same memory location without using unsafe Rust.

2

 Qwaz added the **C-bug** label on Feb 4, 2021

Qwaz commented on Feb 4, 2021

Contributor Author

The bug fixed by this PR seems to have the same consequence, but it didn't get the `unsound` label.

[#80670](#)

sdroege commented on Feb 4, 2021

Contributor

Seems like this would be fixed by simply doing the same as in the first branch of the `if` and incrementing the `self.index` beforehand. Are you creating a PR for this?

Qwaz commented on Feb 4, 2021

Contributor Author

Seems like this would be fixed by simply doing the same as in the first branch of the `if` and incrementing the `self.index` beforehand.

Agreed :)

Are you creating a PR for this?

Not in the meantime. Feel free to work on a PR if you are interested.

sdroege commented on Feb 4, 2021

Contributor

Not in the meantime. Feel free to work on a PR if you are interested.

Sure, why not :)

1

 sdroege added a commit to [sdroege/rust](#) that referenced this issue on Feb 4, 2021 Increment `self.index` before calling `Iterator::self.a.__iterator_ge...`

7d43986

sdroege mentioned this issue on Feb 4, 2021

Increment `self.index` before calling `Iterator::self.a.__iterator_ge...` [#81741](#)

🔒 Merged

 **jonas-schievink** added the `I-unsound` label on Feb 4, 2021

 **rustbot** added the `I-prioritize` label on Feb 4, 2021


 **camelid** added `T-libs` `A-iterators` labels on Feb 4, 2021

 **apiraino** added `P-high` and removed `I-prioritize` labels on Feb 5, 2021

apiraino commented on Feb 5, 2021


Assigning `P-high` as discussed as part of the [Prioritization Working Group procedure](#) and removing `I-prioritize`.

 **jonas-schievink** added a commit to `jonas-schievink/rust` that referenced this issue on Feb 11, 2021

 Rollup merge of [rust-lang#81741](#) - `sdroege:zip-trusted-random-access-s...` ...


`ff5a47d`

 **Dylan-DPC-zz** pushed a commit to `Dylan-DPC-zz/rust` that referenced this issue on Feb 11, 2021

 Rollup merge of [rust-lang#81741](#) - `sdroege:zip-trusted-random-access-s...` ...

`94d1441`

 **Dylan-DPC-zz** pushed a commit to `Dylan-DPC-zz/rust` that referenced this issue on Feb 12, 2021

 Rollup merge of [rust-lang#81741](#) - `sdroege:zip-trusted-random-access-s...` ...

`a79a1da`


 **Dylan-DPC-zz** pushed a commit to `Dylan-DPC-zz/rust` that referenced this issue on Feb 12, 2021

 Rollup merge of [rust-lang#81741](#) - `sdroege:zip-trusted-random-access-s...` ...


`0cfba2f`

 **bors** closed this as completed in [86a4b27](#) on Feb 13, 2021

 **TaKO8Ki** pushed a commit to `TaKO8Ki/rust` that referenced this issue on Feb 25, 2021

 Increment `self.index` before calling `Iterator::self.a.__iterator_ge...` ...

`21ff4a9`

 **Qwaz** mentioned this issue on Jun 18, 2021

Panic safety issue in `Zip::next_back()` `TrustedRandomAccess` specialization #86443

 Closed

 **the8472** mentioned this issue on Jun 21, 2021

fix panic-safety in specialized `Zip::next_back` #86452

 Merged

 **the8472** mentioned this issue on Jan 28

Use `TrustedRandomAccess` for loop desugaring #93243

 Closed

 **the8472** mentioned this issue on Feb 22

Stacked borrows fails on `{ChunksMut, ChunksExactMut}::__iterator_get_unchecked()` #94231

 Closed

Assignees

No one assigned

Labels

`A-iterators` `C-bug` `I-unsound` `P-high` `T-libs`

Projects


None yet

Milestone

No milestone

Development

Successfully merging a pull request may close this issue.

 Increment `self.index` before calling `Iterator::self.a.__iterator_ge...`
`sdroege/rust`

6 participants

