New issue

Jump to bottom

Heap buffer overflow in `read_to_end_with_reservation()` #80894

Closed **Qwaz** opened this issue on Jan 10, 2021 · 3 comments · Fixed by #80895

Labels A-io C-bug I-unsound P-critical T-libs T-libs-api

Qwaz commented on Jan 10, 2021 Contributor

rust/library/std/src/io/mod.rs
Lines 358 to 403 in c97f11a

```
358     fn read_to_end_with_reservation<R, F>(
359         r: &mut R,
360         buf: &mut Vec<u8>,
361         mut reservation_size: F,
362     ) -> Result<usize>
363     where
364         R: Read + ?Sized,
365         F: FnMut(&R) -> usize,
366     {
367         let start_len = buf.len();
368         let mut g = Guard { len: buf.len(), buf };
369         let ret;
```

At line 393, the guard object's `.len` field is incremented by the value returned from a read implementation. If a questionable `Read` returns a value larger than the buffer size, it will take that value and set the length of the vector over the boundary.

This bug is reachable from `Read::read_to_end()` and `Read::read_to_string()`.

Here is a playground link that demonstrates the bug. It segfaults with `double free or corruption (out)`.

👍 4

Qwaz added the C-bug label on Jan 10, 2021

sfackler commented on Jan 10, 2021 Member

This is definitely broken - nice find.

A simple fix would be to just add an `assert!(self.len <= self.buf.len());` in the drop impl:

rust/library/std/src/io/mod.rs
Lines 302 to 306 in c97f11a

```
302     fn drop(&mut self) {
303         unsafe {
304             self.buf.set_len(self.len);
305         }
306     }
```

You can still get weird behavior in read_to_end_with_reservation, but it should be memory safe and it seems best to avoid a bunch of extra logic in the main loop.

sfackler added I-unsound T-libs-api labels on Jan 10, 2021

rustbot added the I-prioritize label on Jan 10, 2021

sfackler commented on Jan 10, 2021 Member

It looks like the bug was introduced in `ecbb896`.

sfackler mentioned this issue on Jan 10, 2021

Fix handling of malicious Readers in read_to_end #80895

Merged

camelid added T-libs A-io labels on Jan 10, 2021

taiki-e mentioned this issue on Jan 11, 2021

Heap buffer overflow in `read_to_end` rust-lang/futures-rs#2310

Closed

JohnTitor commented on Jan 12, 2021 Member

Assigning `P-critical` as discussed as part of the Prioritization Working Group procedure and removing `I-prioritize`.

JohnTitor added **P-critical** and removed I-prioritize labels on Jan 12, 2021

Qwaz mentioned this issue on Jan 13, 2021

Add advisories for standard library soundness bugs rustsec/advisory-db#539

Closed

Qwaz added a commit to Qwaz/advisory-db that referenced this issue on Jan 13, 2021

`Add report for rust-lang/rust#80894` a4bb400

m-ou-se added a commit to m-ou-se/rust that referenced this issue on Jan 14, 2021

`Rollup merge of rust-lang#80895 - sfackler:read-to-end-ub, r=m-ou-se ...` 09276b0

bors closed this as completed in ce48709 on Jan 14, 2021

Assignees

No one assigned

Labels

A-io C-bug I-unsound P-critical T-libs T-libs-api

Projects

None yet

Milestone

No milestone

Development

Successfully merging a pull request may close this issue.

Fix handling of malicious Readers in read_to_end
sfackler/rust

5 participants