New issue

Jump to bottom

Multiple soundness issues in Chunk and InlineArray #11

Closed **Qwaz** opened this issue on Sep 6, 2020 · 5 comments

Qwaz commented on Sep 6, 2020

Hello, we have noticed a soundness issue and/or a potential security vulnerability in this crate while performing a security scan on crates.io.

Description

Chunk:

- Array size is not checked when constructed with `unit()` and `pair()`.
- Array size is not checked when constructed with `From<InlineArray<A, T>>`.
- `Clone` and `insert_from` are not panic-safe; A panicking iterator causes memory safety issues with them.

InlineArray:

- Generates unaligned references for types with a large alignment requirement.

Demonstration

- Crate: sized-chunks
- Version: 0.6.2
- OS: Ubuntu 18.04.5 LTS
- Rust: rustc 1.46.0 (04488afe3 2020-08-24)
- Cargo flags: --release

```
#![forbid(unsafe_code)]

mod boilerplate;

use sized_chunks::{Chunk, InlineArray};
use typenum::*;

#[repr(align(256))]
struct LargeAlign(u8);

struct DropDetector(u32);

impl DropDetector {
    fn new(num: u32) -> Self {
        println!("Creating {}", num);
        DropDetector(num)
    }
}

impl Drop for DropDetector {
    fn drop(&mut self) {
        println!("Dropping {}", self.0);
    }
}

impl Clone for DropDetector {
    fn clone(&self) -> Self {
        if self.0 == 42 {
            panic!("panic on clone")
        }
        DropDetector::new(self.0)
    }
}

struct PanickingIterator {
    current: u32,
    panic_at: u32,
    len: usize,
}

impl Iterator for PanickingIterator {
    type Item = DropDetector;

    fn next(&mut self) -> Option<Self::Item> {
        let num = self.current;

        if num == self.panic_at {
            panic!("panicking index")
        }

        self.current += 1;
        Some(DropDetector::new(num))
    }

    fn size_hint(&self) -> (usize, Option<usize>) {
        (self.len, Some(self.len))
    }
}

impl ExactSizeIterator for PanickingIterator {}

fn main() {
    boilerplate::init();
```

```
    // Some of these cases will panic earlier than assert in debug build due to overflow detection,
    // but they still have the same error

    // https://github.com/bodil/sized-chunks/blob/40aa74b824688a4d4b1e1c65a50c679abb58b41e/src/sized_chunk/mod.rs#L153-L177
    boilerplate::test_case(
        "1. Array size is not checked when constructed with `unit()` and `pair()`.",
        || {
            let _ = Chunk::<usize, U0>::unit(123);
            let mut chunk = Chunk::<usize, U0>::pair(123, 456);

            // Moreover, we can push more elements because `is_full` is implemented as `len != capacity`
            chunk.push_back(789);

            println!("len: {}", chunk.len());
            assert!(chunk.len() <= U0::USIZE);
        },
    );

    // https://github.com/bodil/sized-chunks/blob/40aa74b824688a4d4b1e1c65a50c679abb58b41e/src/sized_chunk/mod.rs#L815-L829
    boilerplate::test_case(
        "2. Array size is not checked when constructed with `From<InlineArray<A, T>>`",
        || {
            let mut from = InlineArray::<u8, [u8; 256]>::new();
            from.push(1);
            from.push(2);
            from.push(3);
            from.push(4);
            from.push(5);

            let to = Chunk::<u8, U0>::from(from);
            println!("len: {}", to.len());
            assert!(to.len() <= U0::USIZE);
        },
    );

    // https://github.com/bodil/sized-chunks/blob/40aa74b824688a4d4b1e1c65a50c679abb58b41e/src/sized_chunk/mod.rs#L120-L134
    boilerplate::test_case("3-1. `Chunk::clone()` is not panic-safe", || {
        let mut chunk = Chunk::<DropDetector, U3>::new();
        chunk.push_back(DropDetector::new(42));
        chunk.push_back(DropDetector::new(43));

        // observe the difference between creating and dropping log
        // uninitialized memory is dropped while unwinding
        println!("=> Dropping uninitialized memory");
        let _ = chunk.clone();
    });

    // https://github.com/bodil/sized-chunks/blob/40aa74b824688a4d4b1e1c65a50c679abb58b41e/src/sized_chunk/mod.rs#L564-L617
    boilerplate::test_case("3-2. `Chunk::insert_from()` is not panic-safe", || {
        let mut chunk = Chunk::<DropDetector, U5>::new();
        chunk.push_back(DropDetector::new(1));
        chunk.push_back(DropDetector::new(2));
        chunk.push_back(DropDetector::new(3));

        println!("=> Double-free of `DropDetector(2)`");
        chunk.insert_from(
            1,
            PanickingIterator {
                current: 1,
                panic_at: 1,
                len: 1,
            },
        );
    });

    boilerplate::test_case("4. `InlineArray` generates unaligned references for types with a large alignment requirement.", || {
        let mut arr = InlineArray::<LargeAlign, [usize; 256]>::new();
        arr.push(LargeAlign(0));

        boilerplate::assert_aligned(arr.get(0).unwrap());
    });

    // Other issues that should be fixed but probably minor to include in the advisory:

    // https://github.com/bodil/sized-chunks/blob/40aa74b824688a4d4b1e1c65a50c679abb58b41e/src/sized_chunk/mod.rs#L564-L617
    // `insert_from` relies on the behavioral correctness of `ExactSizeIterator`.
    // However, `ExactSizeIterator` is a safe trait, which has the same safety guarantee with `size_hint()`.
    // Programs should not assume that they will yield a correct value in unsafe code.
    // From Rust std doc: "An incorrect implementation of `size_hint()` should not lead to memory safety violations."
    //
    // Applying `take(insert_size)` and adjusting `left` and `right` field based on the number of items that are actually moved
    // (instead of using `insert_size`) will fix the problem.

    // https://github.com/bodil/sized-chunks/blob/40aa74b824688a4d4b1e1c65a50c679abb58b41e/src/inline_array/mod.rs#L167
    // This states an actual contract, so it should be `assert!()` instead of `debug_assert!()`
    // From Rust std doc: "Replacing `assert!` with `debug_assert!` is thus only encouraged after thorough profiling, and more importantly, only in safe code!"
}
```

Output:

```
--------------------------------------------------------------------------------
| 1. Array size is not checked when constructed with `unit()` and `pair()`. |
--------------------------------------------------------------------------------
len: 334
thread 'main' panicked at 'assertion failed: chunk.len() <= U0::USIZE', src/main.rs:109:13
note: run with `RUST_BACKTRACE=1` environment variable to display a backtrace

--------------------------------------------------------------------------------
| 2. Array size is not checked when constructed with `From<InlineArray<A, T>>` |
--------------------------------------------------------------------------------
len: 18446744052167409156
thread 'main' panicked at 'assertion failed: to.len() <= U0::USIZE', src/main.rs:126:13

------------------------------------------
| 3-1. `Chunk::clone()` is not panic-safe |
------------------------------------------
Creating 42
Creating 43
=> Dropping uninitialized memory
thread 'main' panicked at 'panic on clone', src/main.rs:59:13
Dropping 150495608
```

```
Dropping 32764
Dropping 42
Dropping 43

------------------------------------------------
| 3-2. `Chunk::insert_from()` is not panic-safe |
------------------------------------------------
Creating 1
Creating 2
Creating 3
=> Double-free of `DropDetector(2)`
thread 'main' panicked at 'panicking index', src/main.rs:78:13
Dropping 1
Dropping 2
Dropping 2

------------------------------------------------------------------------------------------
| 4. `InlineArray` generates unaligned references for types with a large alignment requirement. |
------------------------------------------------------------------------------------------
thread 'main' panicked at 'Reference is not aligned - addr: 0x7ffc08f859e8, align: 0x100', src/boilerplate.rs:46:9
```

Return Code: 0

Qwaz mentioned this issue on Sep 6, 2020

sized-chunks: Multiple soundness issues in Chunk and InlineArray rustsec/advisory-db#381

Merged

This was referenced on Sep 6, 2020

RUSTSEC-2020-0041: Multiple soundness issues in Chunk and InlineArray witnet/witnet-rust#1515

Closed

RUSTSEC-2020-0041: Multiple soundness issues in Chunk and InlineArray benbrandt/d20#1254

Open

RUSTSEC-2020-0041: Multiple soundness issues in Chunk and InlineArray SierraSoftworks/git-tool#105

Closed

fanatid mentioned this issue on Sep 8, 2020

[RUSTSEC-2020-0041]: upgrade sized-chunks vectordotdev/vector#3764

Closed

ErichDonGubler commented on Sep 9, 2020 • edited

~~@Qwaz: Just wanted to confirm: should the advisory note that this affects the 0.6.x releases that are already live? Right now it only states that 0.5.3, and it doesn't match, but in this issue your OP states that this reproduces with 0.6.2 (the latest release at time of writing).~~

EDIT: tools like `actions-rs/audit-check` were what wrote `0.5.3` in reported issues, but it seems that the OP of those auto-filed issues have become outdated already.

jrconlin mentioned this issue on Sep 10, 2020

Chore/update 202009 mozilla-services/syncstorage-rs#819

Merged

jrconlin added a commit to mozilla-services/syncstorage-rs that referenced this issue on Sep 10, 2020

`f skip audit due to bodil/sized-chunks#11` ✓ 549428a

GeorgeHahn mentioned this issue on Sep 11, 2020

Soundness issues in dependency `sized-chunks` (via `im` crate) getsentry/sentry-rust#258

Closed

boozook mentioned this issue on Sep 29, 2020

Soundness issues in sub-dependency `sized-chunks` (via `sentry`, `im` crates) dfinance/dvm#182

Open

mzabaluev mentioned this issue on Oct 16, 2020

Depends on sized-chunks which has soundness problems bodil/im-rs#153

Open

github-actions bot mentioned this issue on Oct 23, 2020

RUSTSEC-2020-0041: Multiple soundness issues in Chunk and InlineArray comit-network/comit-rs#3323

Closed

vorner mentioned this issue on Nov 5, 2020

Fix soundness issues in sized chunks and ringbuffer #13

Merged

vorner commented on Nov 5, 2020 — Contributor

As **@bodil** seems to be busy right now (at least, there doesn't seem to be much activity on the profile), I've decided to give it a look.

May I ask how thorough audid of the code have you done? Should I try going over the code after I get through the other half (I need to figure out what the general idea with the `InlineArray` is, on a first glance it looks a bit fishy).

Qwaz commented on Nov 6, 2020 — Author

- 1 and 2 can be easily fixed by adding a bound check.
- 3-1 and 3-2 may need a guard object like `SetLenOnDrop` (adapted for each case).
- There are multiple solutions for 4 that I can think of. To add a padding byte seems to be the least intrusive solution, but it might still change the behavior of some existing code.

vorner commented on Nov 6, 2020 — Contributor

Hey, thanks for the suggestions, but that's not what I was asking :-) Actually, I've already submitted the pull request to fix 1-3 before that. I think I have an idea for 4 that doesn't require padding bytes (and considering padding bytes would make the capacity computation a bit harder), which I'm going to write now.

What I was asking was, how confident are you there are not more soundness issues? Were you thorough, or are these some things you've noticed, but there are possibly others you haven't looked for?

vorner added a commit to vorner/sized-chunks that referenced this issue on Nov 6, 2020

`Fix alignment issues of InlineArray` ··· a649978

vorner mentioned this issue on Nov 6, 2020

Fix alignment issues of InlineArray #14

Merged

vorner added a commit to vorner/sized-chunks that referenced this issue on Nov 6, 2020

`Fix alignment issues of InlineArray` ··· 9a2e789

Qwaz commented on Nov 7, 2020 — Author

I was focusing on specific type of bugs, so it is possible that there are other types of bugs still present in the codebase.

1

drunkirishcoder mentioned this issue on Nov 17, 2020

bump metrics lib to 0.13 capsule-rs/capsule#116

Merged

github-actions bot mentioned this issue on Nov 28, 2020

RUSTSEC-2020-0041: Multiple soundness issues in Chunk and InlineArray avast/cargo-depdiff#7

Closed

github-actions bot mentioned this issue on Dec 22, 2020

RUSTSEC-2020-0041: Multiple soundness issues in Chunk and InlineArray doraemon93/m--octo-succotash#1

Open

github-actions bot mentioned this issue on Jan 21, 2021

RUSTSEC-2020-0041: Multiple soundness issues in Chunk and InlineArray victor-iyi/project#1

Closed

This was referenced on Jan 30, 2021

RUSTSEC-2020-0041: Multiple soundness issues in Chunk and InlineArray rust-secure-code/cargo-geiger#186

Closed

RUSTSEC-2020-0041: Multiple soundness issues in Chunk and InlineArray maxjoehnk/node-based-mizer#9

Closed

kornelski added a commit that referenced this issue on Feb 12, 2021

`Stop evil iterators breaking insert_from` ··· cd92298

kornelski mentioned this issue on Feb 12, 2021

Soundness fixes for RUSTSEC-2020-0041 #15

Closed

kornelski closed this as completed on Feb 14, 2021

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

Successfully merging a pull request may close this issue.

Soundness fixes for RUSTSEC-2020-0041
bodil/sized-chunks

4 participants