Writing Software That's Safe Enough To Drive A Car

Twitter: @shnewto
Blog: sheas.blog
GitHub: shnewto
Email: shnewto@gmail.com

How to interpret the title...

A discussion of what the process looks like today. Not a prescription.

Slide Deck

sheas.blog/talks

Functional safety is...

the absence of unreasonable risk due to hazards caused by malfunctioning behavior of E/E systems.

What does ‘safe enough’ mean today?

- **ISO 26262**: Standard for managing functional safety of road vehicles
- **MISRA** code guidelines (Motor Industry Software Reliability Association)

Ultimately: C code and static analysis (automated MISRA compliance)

Today’s “State of the Art”

ISO 26262:

- Manufacturers must implement any safety measure necessary to reduce risk available to the current state of the art.
- C with static analysis can be considered state of the art

...but how safe is that?

(let’s get technical for a minute)

DANGER: Mutable aliasing

```
int a = 0;    // data
int *b = &a;  // alias
int **c = &b; // alias

*c += 2048; // "corruption" (allowed by static analysis tools)
*b = 1; // crash
```

Alternatively: We can't destroy what we don't own

```
let mut a: i32 = 0;
let mut b: &mut i32 = &mut a;
let c: &&mut i32 = &mut b;

drop(*c); // "corrupt"
//~^ ERROR cannot move out of borrowed content

*b = 1;
//~^ ERROR cannot assign to `*b` because it is borrowed
```

Let’s just make our aliases immutable

Q: Have you ever hacked an API by modifying private variables?

Casting away the `const`

```
const int a = 0;
*((int *)&a) = 1;
```

Rust: Nope, still immutable

```
let a: i32 = 0;
*(&mut a) = 1;
//~^ ERROR cannot borrow immutable local variable `a` as mutable
```

And there's more:

- The ease of misusing `enum`s
- The data race blind spot

If it doesn't compile, it can't crash.

C is proven in use, why change?

Redefining “State of the Art”

MISRA-Rust?

Preliminary investigation shows that we get ~110 of 140 MISRA-C rules “for free.”

Examples of what’s left:

- Recursion?
- Require an `else` for every `else if`?
- No octal constants?

What’s next?

Resources

github.com/PolySync/static-analysis-argumentation (code)

polysync.io/blog

- *The Challenge of Using C in Safety Critical Applications (white paper)*
- *Should Safety-Critical Software be Written in C? (blog post)*

Twitter: @shnewto
Blog: sheas.blog
GitHub: shnewto
Email: shnewto@gmail.com