# Secret Types in Rust

Daan Sprenkels and Diane Hosfelt

How timing side channels work
Why Rust isn’t suitable (right now)
How Rust can be side channel resistant

## How timing side channels work

## Timing side channels

- Side-channel: attack based on information gained from the implementation
- Timing side-channel: analyze the time taken to execute cryptographic algorithms
- Particular threat in a post-Spectre world
- Primarily used to attack long-lived secrets that are extremely valuable if compromised
- The fix: constant time code (“data-invariant”)

## Why Rust isn’t suitable (right now)

## Compilers are problematic

- Compilers are allowed to optimize anything
- For example: LLVM eliminating conditional moves that may load
- Example: https://godbolt.org/z/YWj7rr

## Why Rust

## How Rust can be side channel resistant

## Language level protections

- Can define newtype-style wrappers around integers
- Examples in the wild:
  - `subtle` crypto crate
  - `secret-integers` crate

- Don’t work to fix compiler-optimization issues

## How to trick the compiler

We can trick the compiler into doing it right :)

- Need to add “optimization barriers” on the secret data
- For example:
  - Empty `asm!()` directive
  - Do a volatile read of secret data <https://godbolt.org/z/abzdPY>

Both tricks not optimal and not 100% coverage

## RFC # 2859: Secret Types

## The Rust part

- Provide primitive data types for transient secrets
  - I.e. `secret_u8`, `secret_i32`, etc.
- Use `.declassify()` to mark something as public
- Additional secret types may be built on top of these primitives
- Only constant-time operations allowed
  - No `secret_isize`, `secret_usize` (don’t index based on secrets)
  - No branching on `secret_bool`
  - No division
  - No printing of values
- Combine secret with public ➜ secret

## Example error

```
error: cannot branch on secret_bool `cond`
 --> :2:46
  |
3 | if cond {
  | ^^ `cond` has a secret type, so this branch is unsafe
```

## The LLVM part

- Has been work on a sister RFC in LLVM
    - Currently not public (stale?)
- No branching on secret data
- No indexing with secret data
- No emission of variable-time instructions
- Memory zeroing is out of scope atm

## Questions?

- Authors:
  - Diane: https://diane.hosfelt.dev
    - @avadacatavra
  - Daan: https://dsprenkels.com
- Extra reading:
  - RFC: https://github.com/rust-lang/rfcs/pull/2859
  - More info on LLVM part: https://dsprenkels.com/cmov-conversion.html