Grandine on Windows

Cross-platform engineering challenges
for Ethereum clients

Jin Mingjian

Ethereum Protocol Fellow, Cohort-Five

Challenges

Challenges

It is hard to run an Ethereum validator on native Windows.

Challenges

Problems	Comments
Jemalloc failed to build	Can be built for correct config, but it is hard to config correctly.
Stack overflow in debug mode	Windows’ stack size is conservative
Test failures in ssz spec tests	Git's default value of core.autocrlf on Windows is true, this will change some ssz binary artifacts
Reth can not boot after its first run	"Database commit error code: 998", but this not due to db file corruption

Challenges

Problems	Comments
System metrics are not supported	Missing in Both Grandine and Lighthouse.
Official windows APIs are unsafe	Unsafes are forbidden by many teams in Rust via lints.
Screensaver + sleeping	Common use cases for Windows.
Not easy to run a validator	Complex setup and configs for making staking + CL + EL.

Solution

Solution

Problems	Solution/Workaround
Jemalloc memory allocator failed to build	Disabled, in favor of the Windows built-in allocator --- Truth: Jemalloc is great for server side but neither necessary for edge nor better

Solution

Problems	Solution/Workaround
Stack overflow in debug mode	Run in release mode --- Reason: big on-stack allocation in debug mode in rust libp2p call stack. Reflection: 1. Limit statics and use a top singleton to manage others; 2. Rethink or may overhaul fundamental libs like libp2p

Solution

```
...
7:
once_cell::imp::impl$4::initialize::closure$0<array$<k25
6::arithmetic::mul::LookupTable,33>...
8: once_cell::imp::initialize_inner...
9:
once_cell::imp::OnceCell<array$<k256::arithmetic::mul::L
ookupTable,33> >::initialize...
10:
once_cell::sync::OnceCell<array$<k256::arithmetic::mul::
LookupTable,33> >::get_or_try_init...
11:
once_cell::sync::OnceCell<array$<k256::arithmetic::mul::
LookupTable,33> >::get_or_init...
12:
once_cell::sync::Lazy<array$<k256::arithmetic::mul::Look
upTable,33>...>::force...
...
```

More detail at [Grandine issue #41]

Libp2p -> discv5 ->

enr -> k256 ->

GEN_LOOKUP_TABLE lazy init

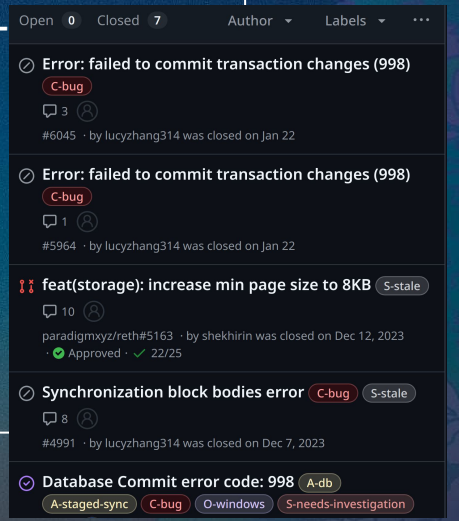
LookupTable is too big for the Windows stack

Solution

Problems	Solution/Workaround
Test failures in consensus spec ssz tests	Set core.autocrlf to false on Windows --- Lesson: Consistency and debuggability are so important for developers. Never overemphasis.

Solution

Problems	Solution/Workaround
Reth can not boot after its first run (Database commit error code: 998)	Run in WSL on Windows --- Lessons: * No Windows based tests in CI of Reth *Performance isn't all: mdbx is a double-edged sword



Solution

Problems	Solution/Workaround
System related metrics are missing	Implemented missing cpu and memory stats --- Points: 1. metrics have its API proposal from beaconcha.in team 2. Grandine now has better metrics support on Windows than Lighthouse

```rust
impl SystemHealth {
    #[cfg(not(target_os = "linux"))]
    pub fn observe() -> Result<Self, String> {
        Err("Health is only available on Linux".into())
    }

    #[cfg(target_os = "linux")]
    pub fn observe() -> Result<Self, String> {
        let vm: VirtualMemory = psutil::memory::virtual_memory()
```

Codes in Lighthouse

Solution

Problems	Solution/Workaround
Official windows APIs are unsafe	Upstream unsafes to winsafe. --- Pitfall: 'forbid' for lints just works for codes in current workspace but not for those in libraries

Solution

Problems	Solution/Workaround
Screensaver and sleeping	No problem.

Solution

Problems	Solution/Workaround
Not easy to run an Ethereum validator	* Staking * Engines interaction configs * BN, VC, EL configs ---------- Idea: I and ThreeHrSleep proposed CL + EL integration in cohort-5 (I for Grandin-Reth, ThreeHrSleep for Lighthouse-Reth)

Solution

Running an Ethereum validator shouldn’t be this hard.

Thoughts

- **Rust is great for cross-platform**
  - but we haven’t fully utilized it yet
- **Clients often share common implementations or libraries**
  - Significance of client diversity will be reduced

- **More best practices**
  - which from the wider Rust ecosystem definitely can benefit our Rust based Ethereum client ecosystem.

- **EPF is great**
  - In the process of EPF, I began to have a deep understanding of the protocol interaction process of Ethereum, and the development and source code of several core clients (Grandine, Lighthouse, Reth).

Acknowledgements

- Mentors
  - Saulius Grigaitis (Grandine)
  - weekday-grandine-io (Grandine)
- Ethereum Foundation
  - Tim Beiko
  - Josh Davis (EPF)
  - Mario Havel (EPF)

About Me

- PhD, 10+ years of database, 7 years of Rust, former CTO of DB startup and heads of data and engineering, creator of open source DB project with 1.4k Github stars
- Expertise and enthusiasm in Rust, DB, ZK and Ethereum core infra
- Open to any Rust + performance engineering + Ethereum opportunity


Thank you!
Jin Mingjian
Ethereum Protocol Fellow, Cohort-Five
Email: mj@mjzk.xyz
Github: https://github.com/mjzk