LIGHTHOUSE

A SECURITY-FOCUSED ETHEREUM SERENITY CLIENT

sigma prime

AGENDA

- Introduction
- ETH 2.0 Roadmap & Timeline
- Lighthouse Client Architecture
- Security Considerations
- Achievements & Progress Updates
- Funding Status Update
- Challenges
- Conclusion


INTRODUCTION
sigma prime

- Sigma Prime (SigP) – Information security consultancy, focused on Blockchain tech, working mostly on Ethereum
  - *Security researchers, academics and software engineers working towards a secure and decentralised future*

- Some of our information security clients:

- Maintainers of Lighthouse, a Rust implementation of Ethereum 2.0

LIGHTHOUSE – BACKGROUND

- Passion for Proof-of-Stake designs
  - Game theoretical principles
  - Environmental impact
  - Finality
- Paul Hauner (@paulhauner) started working on a JS implementation of Casper TFG (*"Vlad's Casper"*)
- Moved to Casper FFG (*"Vitalik's Casper"*):
  - EIP-1011 (Hybrid PoW/PoS) – produce a Casper FFG voting smart contract on top of the existing PoW chain
    - Deprecated in late-June 2018
- Props to Danny Ryan for making us feel welcome!

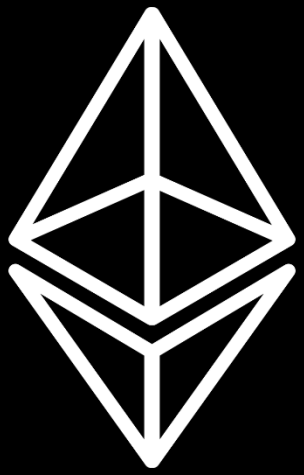
sigp/lighthouse

LIGHTHOUSE – MOTIVATION

- σ' ♥ ♦
- We believe PoS blockchains are the future
- Tests our understanding of fundamental concepts
- Massively expands our knowledge and capabilities
- Contributing - Someone’s gotta do it!


ETHEREUM SERENITY ROADMAP
sigma prime

ETHEREUM SERENITY DESIGN GOALS



Decentralisation

Allow standard consumer laptop to participate
Support participation of a large # of validators

Liveness

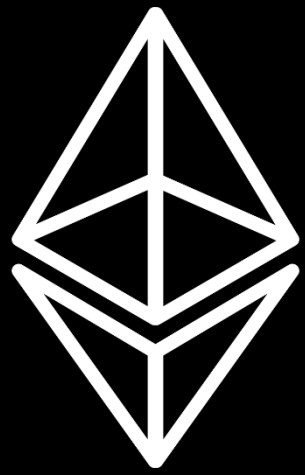
Network should remain live in a WWIII scenario

Simplicity

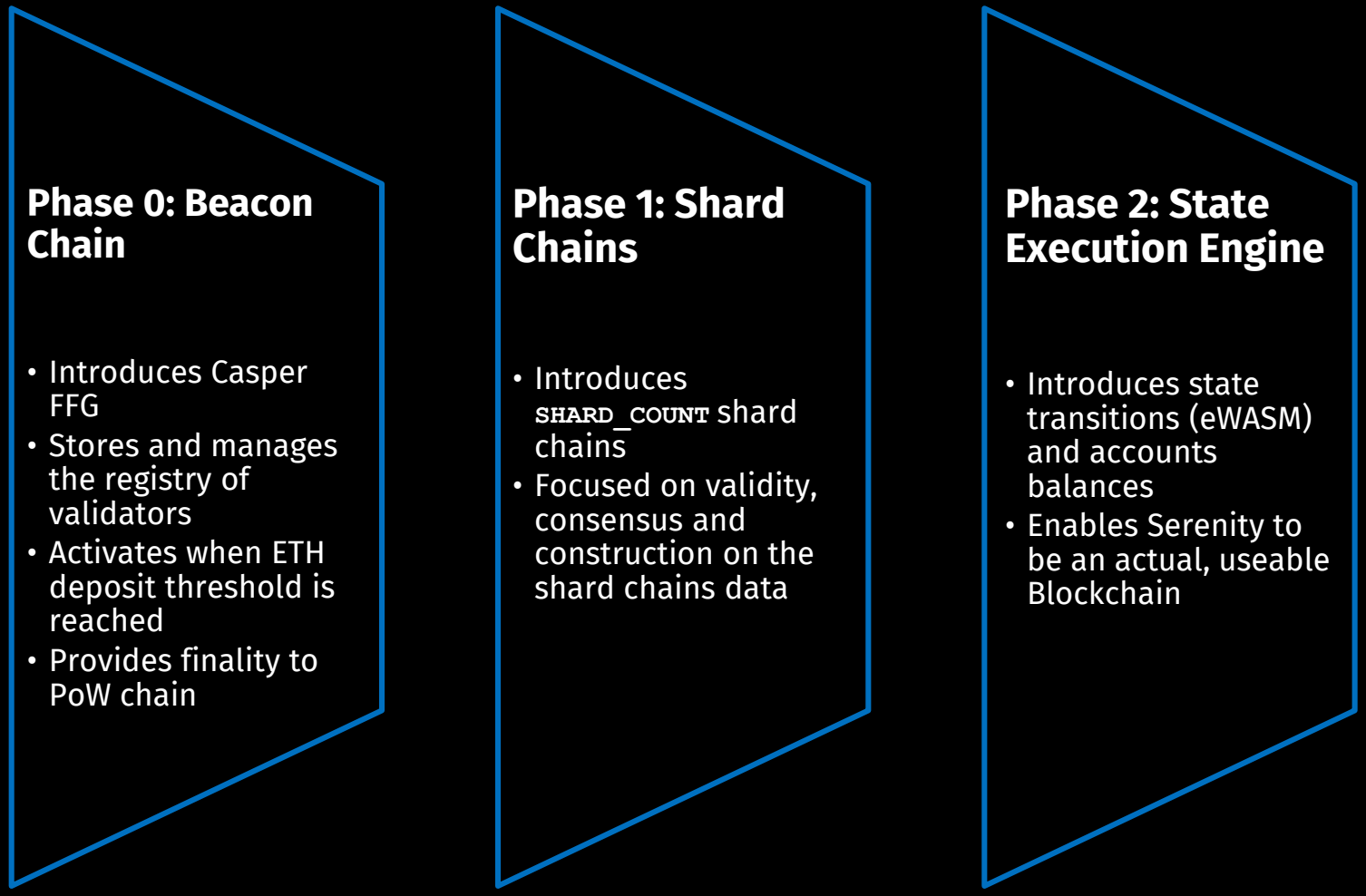
Minimize complexity even at the cost of efficiency

Security

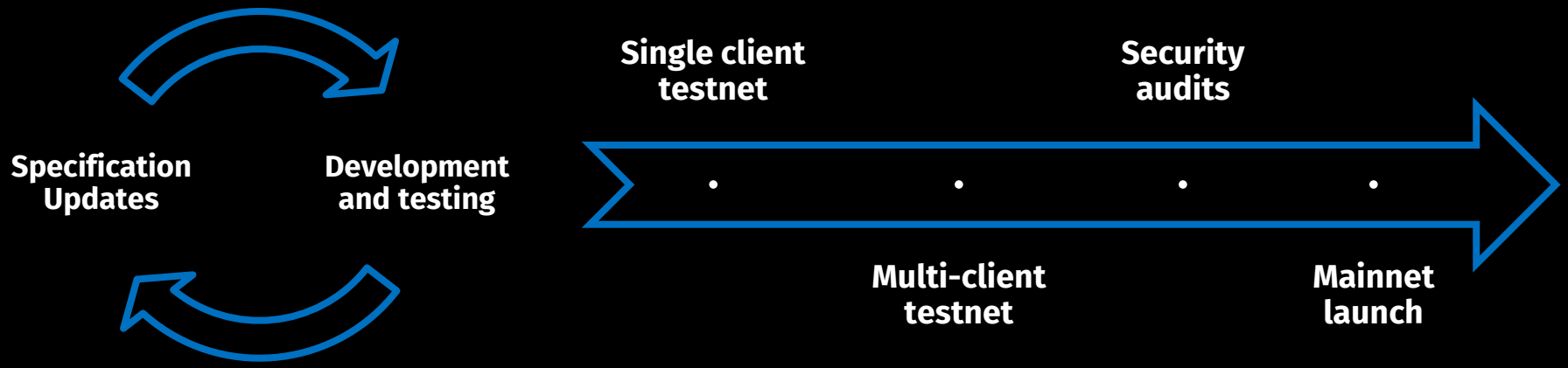
Use quantum secure cryptographic primitives where possible
Allow easy swapping of cryptographic components



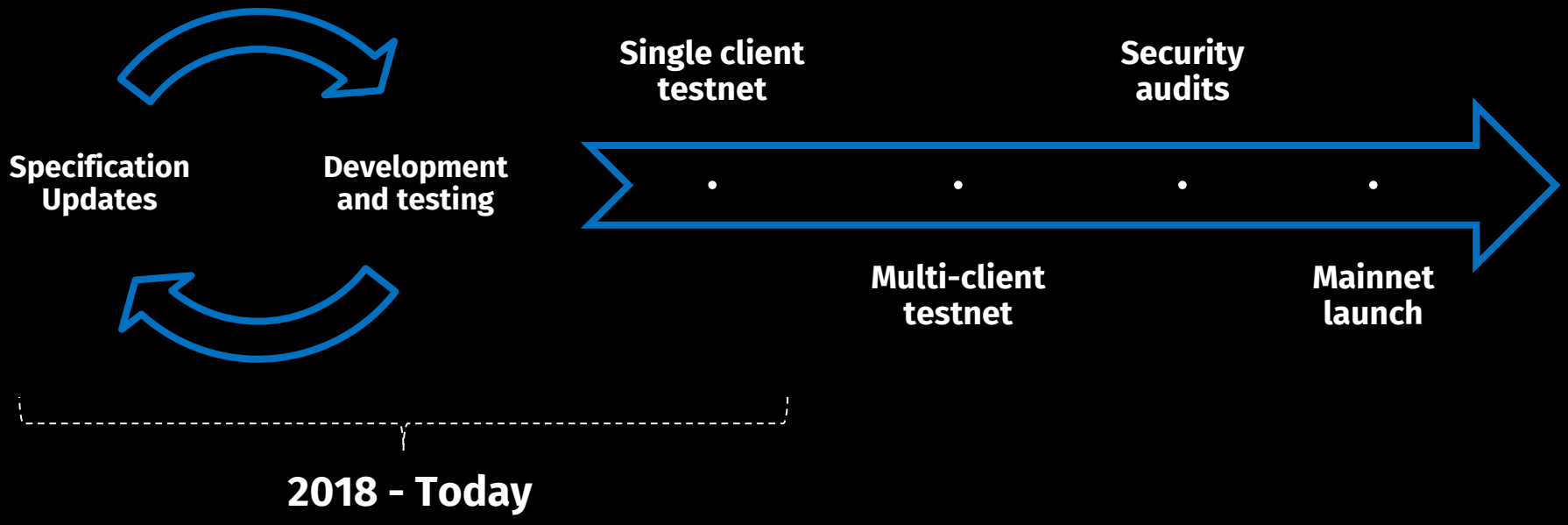
ETHEREUM SERENITY ROADMAP



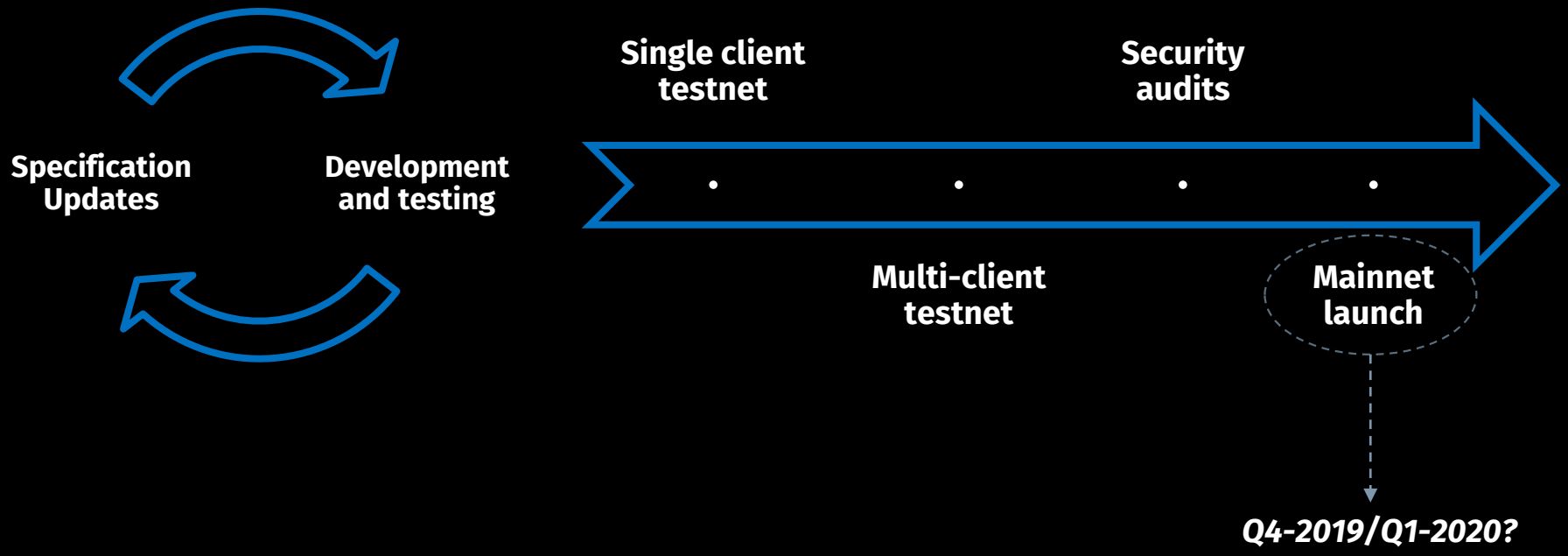
ETHEREUM SERENITY DEV CYCLE



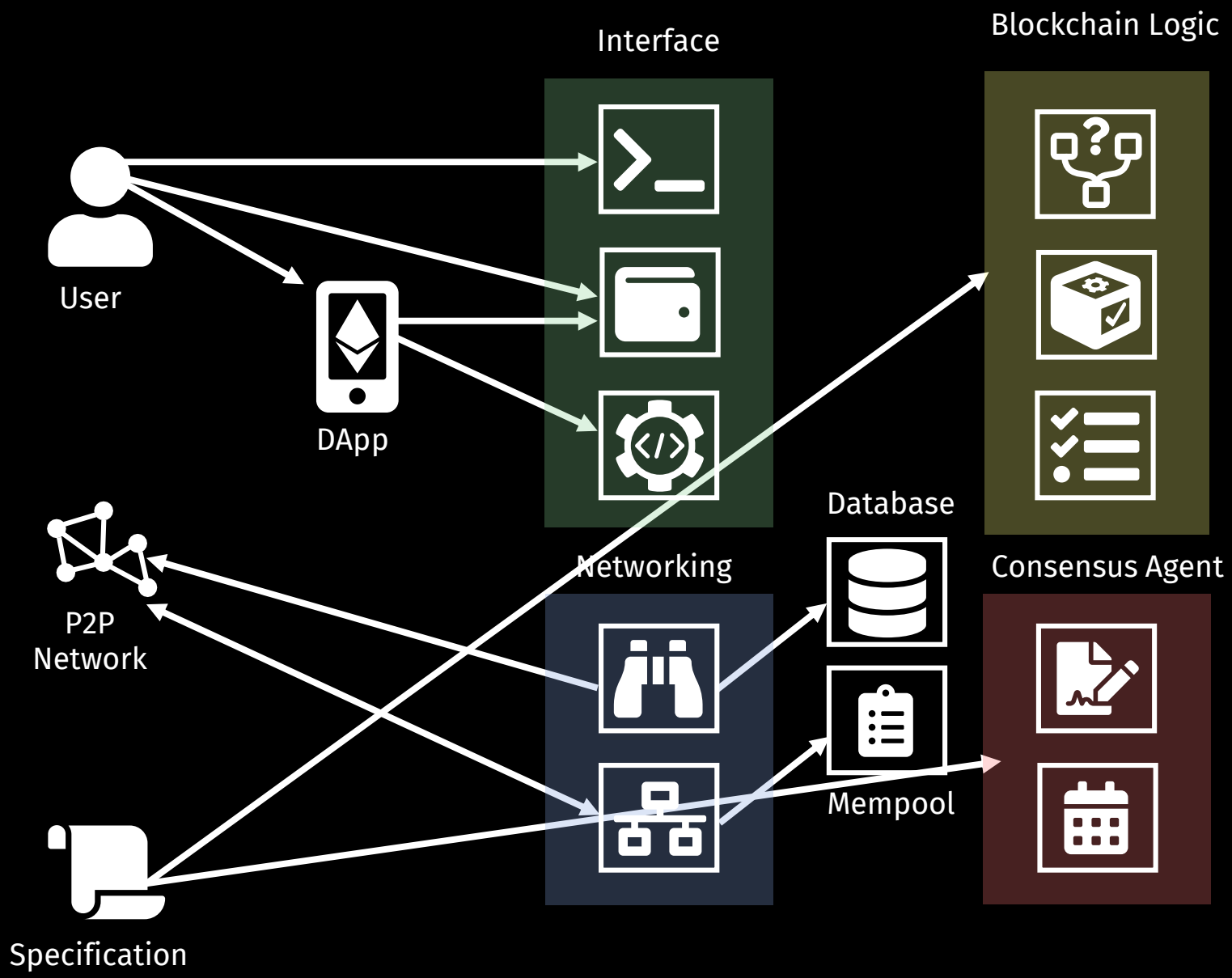
SERENITY PHASE 0 – TIMELINE



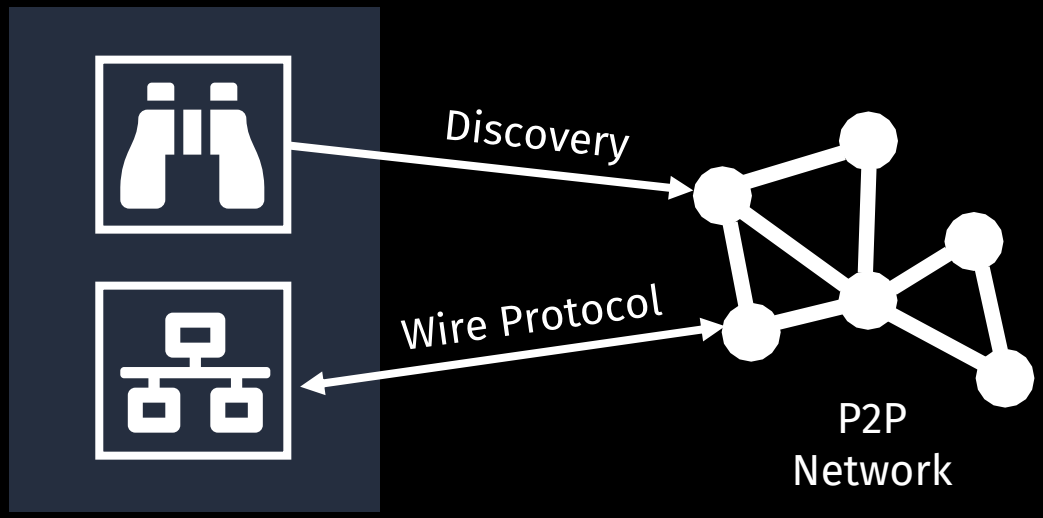
SERENITY PHASE 0 – TIMELINE




LIGHTHOUSE CLIENT ARCHITECTURE
sigma prime

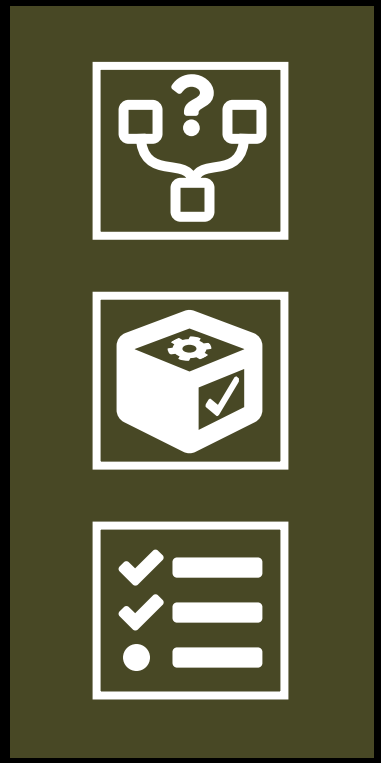

Interface
Blockchain Logic
User
DApp
Database
Networking
Consensus Agent
P2P
Network
Mempool
Specification

CLIENT ARCHITECTURE: NETWORK LAYER



Message Routing:
Gossipsub

CLIENT ARCHITECTURE: BEACON NODE



Fork Choice Rules

State/Block Processor

Tests

← All from the Spec.

CLIENT ARCHITECTURE: VALIDATOR

Signs Blocks & Attestations

Has “Duties”

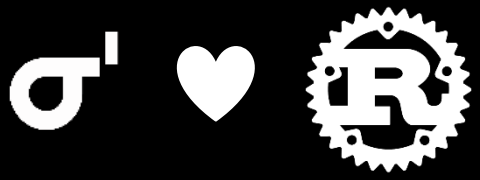
- Separate binary
- Connects to the Beacon Node
  - Requests blocks
- Ensures two signatures don’t conflict
  - Prevents “Slashing”


SECURITY CONSIDERATIONS
sigma prime

SECURITY – WHY RUST?

Rust is a systems programming language that is syntactically similar to C++

- Originally designed at Mozilla Research in ~2010
- Focused on safety, especially safe concurrency
- Fast & strongly-typed
- No automated garbage collector
- Lots of runtime errors are moved to compile-time errors
- *Innovative* memory management with *Ownerships*



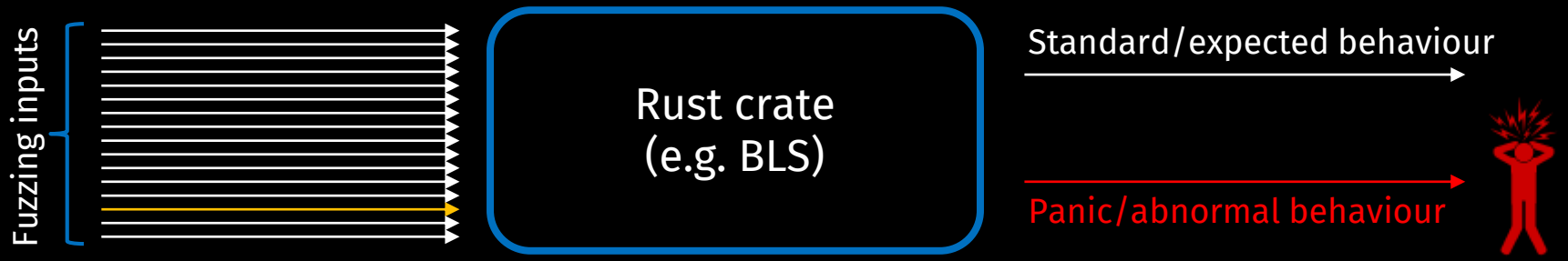
LIGHTHOUSE SECURITY PRINCIPLES

- One our key principles is *never panic!*
  - Follow the Rust philosophy, catch and safely handle all errors

- Follow strict coding guidelines (refer to ANSSI project):
  - e.g. If a sensitive object is to be freed, don't just drop it, zero out the related memory addresses

- Use *clippy* and *rustfmt*:
  - *clippy*: Rust linter
  - *rustfmt*: Tool for formatting Rust code
- Code reviews… we need you!
- Fuzz all our finalized crates… But what is Fuzzing?

FUZZING - INTRODUCTION




```
!8;`kZN(d94LGCS*GN}=j)k:,pWwM?+#znJ'7q@%ua'6bL@|{'{!~'~-`@-@&\{~'{:'`
```

2 fuzzers: Libfuzzer & AFL

LIGHTHOUSE SECURITY – NEXT STEPS

- Types of expected bugs:
  - Denial of Service attacks (Sybil attacks in libp2p?)
  - Consensus divergence due to client optimisations

- Before mainnet release, get an independent security review
- Looking at formal verification of a subset of lighthouse
  - Formal semantics for Rust in K are being developed

- Create a differential fuzzer and integrate other clients
  - Similar to diff fuzzer maintained by EF

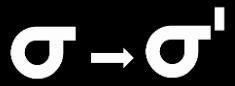

ACHIEVEMENTS &
PROGRESS UPDATE
sigma prime

ACHIEVEMENTS AND PROGRESS UPDATE

- Libp2p Gossipsub Rust implementation

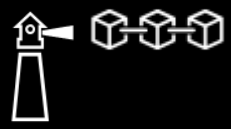
- Rust libraries:
  - SSZ, shuffling, BLS, and other critical functions

- Block processing and state transitions implemented



- Fork choice implemented

- Beacon chain with thousands of validators can build blocks



- Growing the team of lead/core devs

- Contributing to the spec by findings bugs


FUNDING STATUS UPDATE
sigma prime

FUNDING

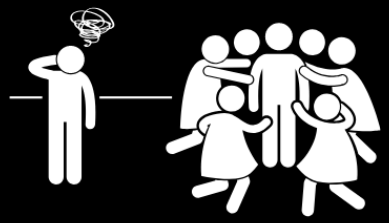
- **Sigma Prime**: Internal funding (~US$ 110k)
- **Ethereum Foundation**: Grant (US$ 150k)
- **Vitalik Buterin**: “YOLO” Grant (~US$ 100k)
- **Gitcoin**: Issue bounties

It’s a big project and funding is a challenge

Long term funding for infrastructure is unsolved

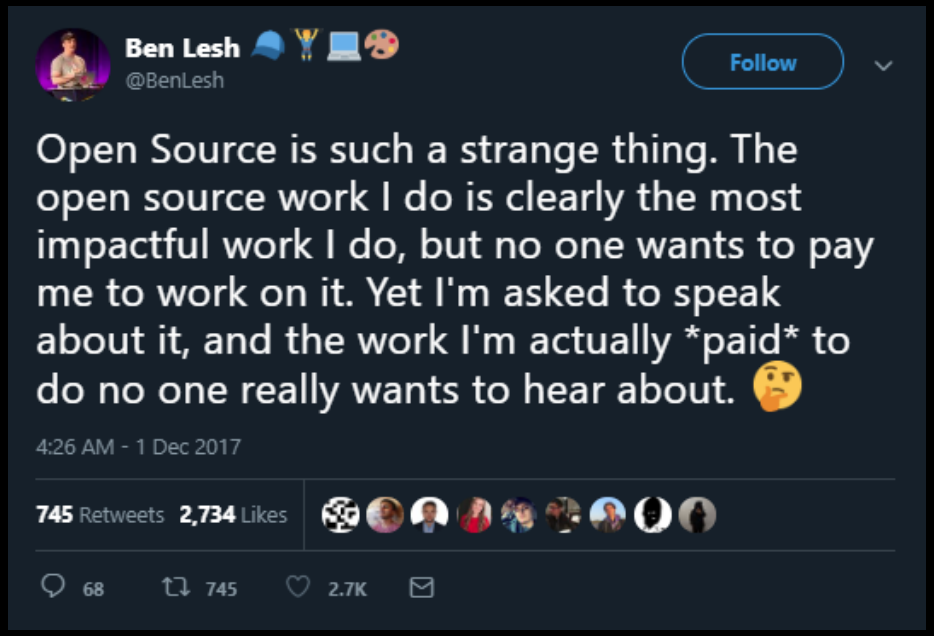
UNPOPULAR OPINION

Teams like ours shouldn’t hold grants in volatile crypto assets



FUNDING

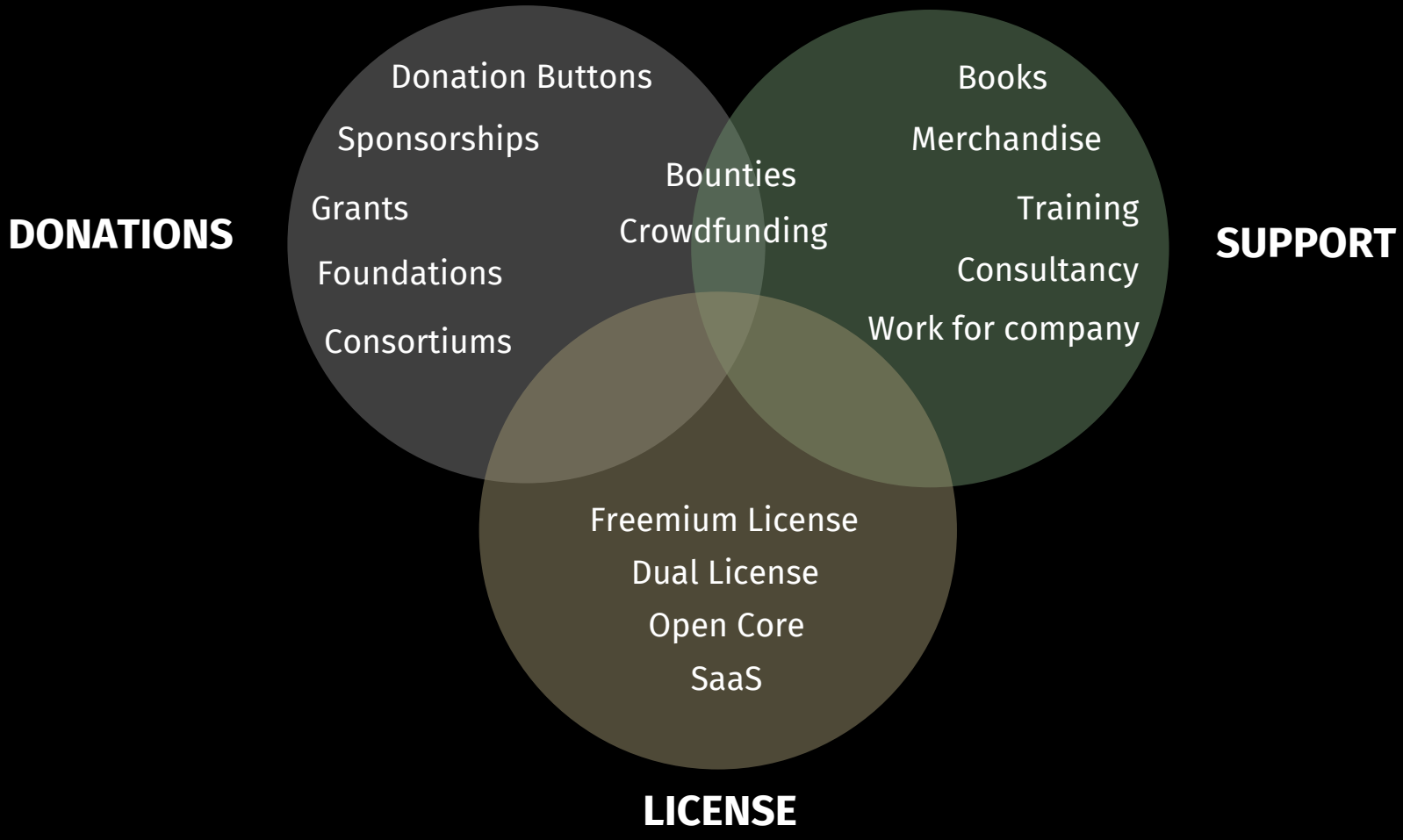
- This is obviously not a problem specific to Ethereum or the Blockchain space



- In fact, we're pretty *lucky* to have the support of the EF

FUNDING

- So how do projects outside of this space solve this problem?



COMMITMENT

Lighthouse will always be free and open source

FUNDING – CURRENT STATUS

- **Current lighthouse team size:**
  - 3x Senior/lead devs – full time
  - 1x Senior/lead dev – part time
  - 2x Junior devs – part time

- **Total yearly salary cost:** US$ 590,000

With the current set of grants and resource allocation, lighthouse is funded until July 2019


CHALLENGES
sigma prime

CHALLENGES – PAST AND PRESENT

- Temptation from working on other cool (and profitable) Ethereum stuff

- Ever-changing spec… much better with RCs!

- Naïve specification implementation is not viable

CONCLUSION

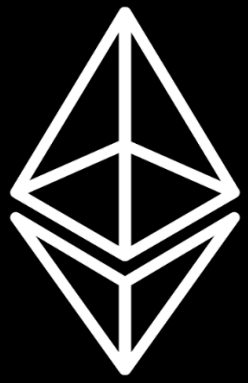
sigma prime

CONCLUSION

- Takeaways?
  - Should have we waited for a RC before starting development?
  - The research team is the PM team… Is this adequate? Can we actually do this differently in a decentralised context?

- Hope to see you all @ EDCON in Sydney!
  - @paulhauner and @agemanning to give a lighthouse update

- Help us get to Serenity! All contributions welcome



QUESTIONS?

mehdi@sigmaprime.io

@ethzed

sigma prime