Signal Shot: End-to-End Formal Verification of Signal's Cryptographic Stack

Liao Zhang and the BAIF team

Beneficial AI Foundation

May 16, 2026

Signal: Why Formal Verification?

Signal

- Secure messaging application
- Over **70 million users**
- Underpins end-to-end encryption

Verification of Signal's cryptography is crucial

- Subtle bugs in the crypto stack can leak users' information
- Testing covers only finite cases
- Formal verification gives more assurance

Challenges and Our Approach

Challenges in formally verifying Signal:

- Mathematics of cryptography is complex; significant expertise to formalize
- Formal verification is inherently labor-intensive
- Long-term maintainability requires modular, evolvable proofs

Signal Shot addresses these by leveraging:

- **Lean** as the proof assistant (mathematical expressivity)
- **Aeneas** (Rust→Lean transpiler) – verifiable link between security models and Signal's production code
- **AI integration** to improve verification efficiency

Contributions

1. **Signal Shot**
   A unified Lean-based framework that addresses the verification challenges of Signal.
2. **Large-Scale Verification**
   Completed formal verification of **curve25519-dalek**, a widely deployed Rust implementation of Curve25519 operations.
3. **AI Integration**
   Developed specialized coding agents tailored for formal verification.
4. **Benchmarking**
   Constructed a benchmark with ground-truth specs and proofs to evaluate AI models on real-world program verification tasks.

Technical Approach: Two Complementary Tracks

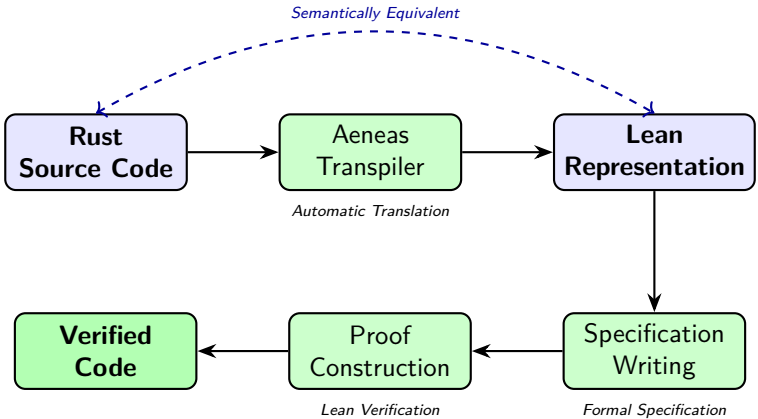
Protocol Correctness

- Formalize Signal's core cryptographic protocols in Lean:
  - X3DH, PQXDH
  - Double Ratchet
  - CKA
- Encode cryptographic papers into Lean
- Establish standard security properties

Function Correctness

- Verify that Signal's Rust implementations correctly realize developers' intentions
- Use Aeneas to obtain semantically equivalent Lean code
- Specifications + proofs in Lean

Lean Verification Pipeline



Completed: curve25519-dalek

A core cryptographic library written in Rust, implementing Curve25519 group operations.

Library Scope

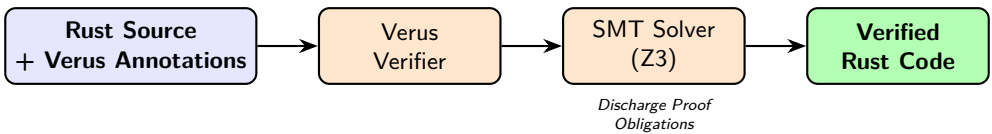
- **288 functions**, ∼**35,000 lines** of Rust
- Used in **Signal**, **Tor**, and **TLS**
- Covers finite-field & scalar arithmetic, elliptic-curve group operations, encoding layers, protocol-level primitives

First end-to-end formal verification of an elliptic-curve cryptographic library at this scale using Lean.

Two-Team Verification Approach

1. **Verus Team**
   Annotated the Rust source with compile-time Verus specifications; proof obligations discharged by SMT solvers.

2. **Lean Team**
   Used Aeneas to extract a semantically equivalent Lean representation and established its correctness in Lean.

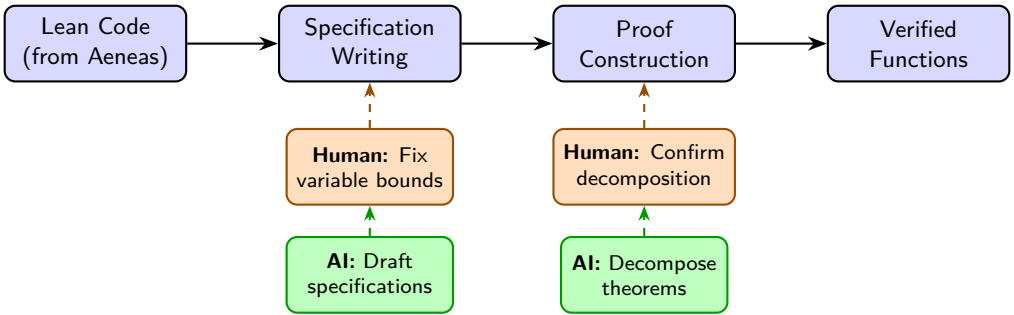


Technical Challenges Identified

- **Large proof states** generated by Aeneas
  – Aeneas-produced Lean code yields verbose monadic obligations that strain interactive proof.
- **Instability in SMT solving**
  – Small changes can flip success/failure; brittle for large-scale curve arithmetic.
- **Complexity of elliptic-curve reasoning**
  – Requires non-trivial mathematical infrastructure (modular arithmetic, group laws).

These challenges motivate improvements to both Aeneas and Lean's verification infrastructure.

AI-Assisted Verification Workflow



FVS: Formal Verification Skills

Custom formal-verification agents implemented as Claude Code skills.

- Easy installation and practical deployment in everyday workflows
- Specialized techniques tailored to formal verification
  (proof drafting, repair, decomposition, lemma search)
- Notable feature: **guidance for checking variable bounds** – a subtle aspect prone to errors in real-world verification

Significantly improved productivity during the curve25519-dalek effort, motivating continued investigation of AI in Lean-based verification.

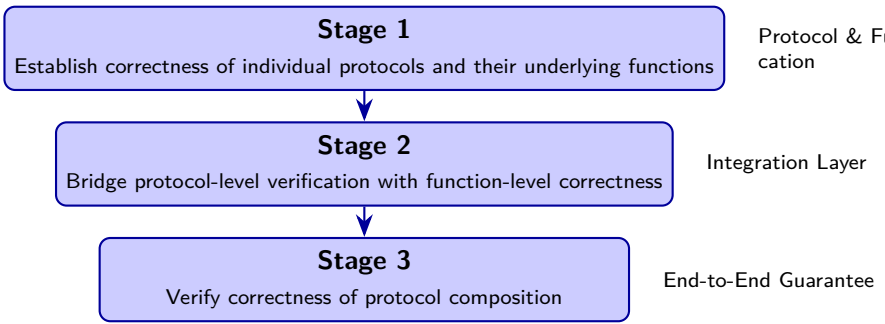
Benchmark Construction

A benchmark built from our verified Lean codebase.

- Ground-truth specifications and proofs for every entry
- Covers real-world cryptographic functions (not toy examples)
- Enables systematic evaluation of AI models for end-to-end verification

A foundation for measuring progress in AI-assisted formal verification.

Future Work: Three Stages



We're Hiring – Join Us!

Beneficial AI Foundation

- Research Engineers – software verification, math & AI agents
- Full-stack Engineers

`https://www.beneficialaifoundation.org/signal-shot`

Thank you!

Questions?

Scan to learn more

Opportunities at BAIF