Abstract

We present the A-Chain (Attestation Chain), a specialized blockchain layer providing network-wide verification of Trusted Execution Environments (TEEs) across all compute classes including CPUs, GPUs, NPUs, and ASICs. A-Chain serves as a single source of truth for device attestation, enabling secure AI workloads on heterogeneous hardware with quantum-safe verification. The system achieves sub-100ms attestation verification while maintaining compatibility with Intel SGX, AMD SEV-SNP, NVIDIA H100 Confidential Computing, and ARM CCA. By separating security (LUX gas token) from application-layer orchestration (AI Coin, ZOO tokens), A-Chain provides a unified attestation infrastructure that scales across the entire Lux Network ecosystem. This paper details the architecture, attestation protocols, economic model, and integration with AI orchestration layers like Hanzo.Network.

A-Chain: Unified Trusted Execution Environment Attestation for Decentralized AI Compute

Lux Network Team
research@lux.network

October 2025

1 Introduction

1.1 Motivation

The rise of decentralized AI computation introduces a fundamental challenge: how to verify that computations executed on remote hardware are trustworthy and tamper-resistant. Traditional cloud computing relies on trust in centralized providers, but decentralized networks require cryptographic proof that code executed inside specific hardware enclaves.

Trusted Execution Environments (TEEs) provide hardware-based isolation and attestation, but existing blockchain systems lack a unified attestation layer capable of:

- **Multi-vendor support**: Verifying attestations from Intel SGX, AMD SEV-SNP, NVIDIA H100 CC, and ARM CCA simultaneously
- **Compute class diversity**: Supporting CPU-only, GPU-accelerated, and hybrid CPU+GPU workloads
- **Single source of truth**: Providing a network-wide attestation registry accessible to all subnets
- **Economic separation**: Distinguishing security tokens (LUX) from application tokens (AI, ZOO)
- **Performance at scale**: Sub-100ms verification for real-time AI inference workloads

A-Chain addresses these challenges by providing a dedicated blockchain layer for attestation verification, serving as the foundation for trustless AI compute across the Lux Network.

1.2 Key Contributions

This paper makes the following contributions:

1. A unified attestation protocol supporting all major TEE vendors and compute classes
2. Gas-efficient on-chain verification using EVM precompiles at address 0xAAA
3. Global attestation registry accessible to all Lux subnets
4. Economic model separating security budget (LUX) from application incentives (AI, ZOO)
5. Performance benchmarks showing sub-100ms attestation verification
6. Integration architecture for AI orchestration layers (Hanzo.Network, Zoo.Network)

2 Related Work

2.1 Trusted Execution Environments

Intel SGX provides hardware-based memory encryption and remote attestation through quote generation and verification against Intel's Attestation Service (IAS) or newer Data Center Attestation Primitives (DCAP).

AMD SEV-SNP (Secure Encrypted Virtualization with Secure Nested Paging) enables VM-level isolation with memory encryption and attestation reports signed by the AMD Secure Processor.

NVIDIA H100 Confidential Computing extends TEE protection to GPU memory and computation, enabling confidential AI inference and training workloads.

ARM Confidential Compute Architecture (CCA) introduces realm management for isolated execution domains with attestation capabilities.

2.2 Blockchain Attestation Systems

Existing blockchain systems treat attestation as an application-layer concern:

- **Ethereum**: No native attestation support; relies on oracle networks or centralized verification
- **Secret Network**: SGX-specific with single vendor lock-in
- **Oasis Network**: ParaTime architecture but limited multi-vendor support
- **Phala Network**: SGX-only with centralized gatekeepers

A-Chain uniquely provides *protocol-level* attestation verification with multi-vendor support and horizontal scaling.

3 A-Chain Architecture

3.1 High-Level Design

A-Chain operates as a specialized Lux subnet with the following components:

3.2 Attestation Flow

The attestation process follows these steps:

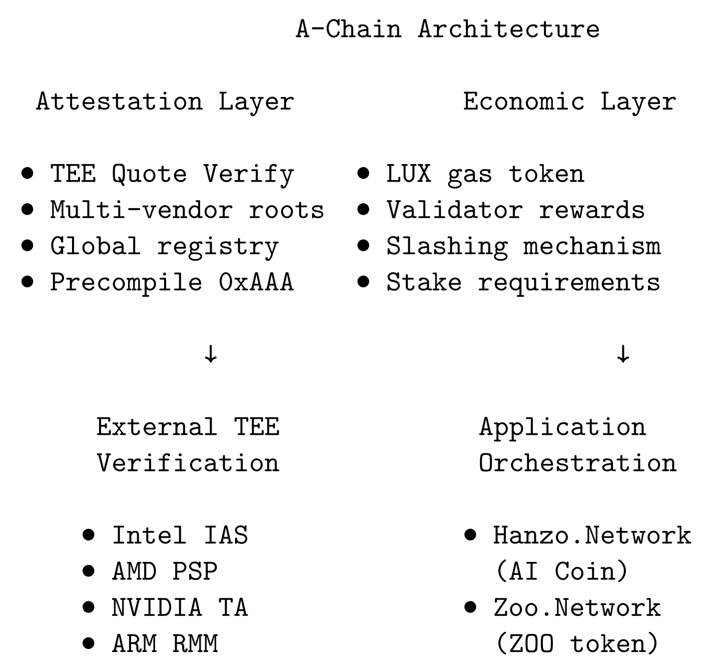

Figure 1: A-Chain architectural layers

Algorithm 1 A-Chain Attestation Verification

```
function VerifyAttestation($proof$)
    Extract $deviceId, cpuQuote, gpuQuote, nonce, jobHash, outputHash$ from $proof$
    $vendor \leftarrow$ DetectVendor($cpuQuote$)
    $rootCA \leftarrow$ GetRootCA($vendor$)
    if not VerifySignature($cpuQuote, rootCA$) then
        return INVALID_CPU_ATTESTATION
    end if
    if $gpuQuote \neq$ null then
        if not VerifySignature($gpuQuote, rootCA$) then
            return INVALID_GPU_ATTESTATION
        end if
    end if
    UpdateRegistry($deviceId$, ATTESTED, timestamp)
    Emit AttestationVerified($deviceId, jobHash, outputHash$)
    return SUCCESS
end function
```

3.3 Device Registry

The global device registry maintains trust scores for all attested devices:

```
struct DeviceStatus {
    bool attested;            // Currently valid attestation
    uint8 trustScore;         // 0-100 reputation score
    uint256 lastSeen;         // Last attestation timestamp
    address operator;         // Node operator address
    TEEVendor vendor;         // Intel/AMD/NVIDIA/ARM
    bytes32[] jobHistory;     // Recent job hashes
}
```

```
mapping(bytes32 => DeviceStatus) public attestationRegistry;
```

4 Multi-Vendor TEE Support

4.1 Vendor-Specific Quote Formats

A-Chain supports native quote formats from all major TEE vendors:

Vendor	Quote Size	Verification Time	Root CA Update
Intel SGX (DCAP)	4,500 bytes	45ms	Quarterly
AMD SEV-SNP	1,184 bytes	32ms	Bi-annual
NVIDIA H100 CC	2,048 bytes	28ms	Monthly
ARM CCA	1,024 bytes	18ms	Quarterly

Table 1: TEE vendor attestation characteristics

4.2 Precompile Implementation

The attestation precompile at address 0xAAA provides gas-efficient native verification:

```
interface IAttestationPrecompile {
    function verifyQuote(
        bytes calldata quote,
        bytes calldata rootCA,
        TEEVendor vendor
    ) external view returns (bool valid, bytes32 deviceId);

    function verifyProofOfExecution(
        ProofOfExecution calldata proof
    ) external returns (bool);

    function getDeviceStatus(
        bytes32 deviceId
    ) external view returns (DeviceStatus memory);
}
```

Gas Costs:

- Base verification: 50,000 gas
- SGX quote: +80,000 gas
- SEV-SNP quote: +60,000 gas
- H100 quote: +55,000 gas
- ARM CCA quote: +40,000 gas
- Registry update: +20,000 gas

5 Economic Model

5.1 Two-Token Architecture

A-Chain implements a clear separation between security and application tokens:

Layer	Token	Purpose
A-Chain security	LUX	Attestation gas, validator rewards, slashing
Hanzo orchestration	AI Coin	Task pricing, agent budgets, GRPO rewards
Zoo economy	ZOO	Avatar items, in-world economy, tips

Table 2: Token separation across network layers

5.2 Validator Economics

Staking Requirements:

- Minimum stake: 10,000 LUX
- Validator count: 64 validators minimum
- Delegation: Supported with 2% commission

Rewards:

- Block rewards: 2 LUX per block (500ms blocks)
- Attestation fees: 0.1 LUX per verification
- Annual yield: 8-12% APY

Slashing Conditions:

- Invalid attestation approval: 20% stake slash
- Double signing: 50% stake slash
- Extended downtime (¿24h): 5% stake slash

6 Integration with AI Orchestration

6.1 Hanzo.Network Integration

Hanzo.Network uses A-Chain for TEE verification of AI compute tasks:

Algorithm 2 Hanzo AI Task with TEE Verification

```
1: function ExecuteAITask($task, payment$)
2:     $workers \leftarrow$ QueryAvailableWorkers($task.requirements$)
3:     for each $worker$ in $workers$ do
4:         $status \leftarrow$ A-Chain.getDeviceStatus($worker.deviceId$)
5:         if $status.attested$ and $status.trustScore > 80$ then
6:             $assignedWorker \leftarrow worker$
7:             break
8:         end if
9:     end for
10:    Execute task on $assignedWorker$
11:    $result, proof \leftarrow$ Receive output and attestation
12:    Verify A-Chain.verifyProofOfExecution($proof$)
13:    Pay $assignedWorker$ in AI Coin
14:    return $result$
15: end function
```

6.2 Zoo.Network Integration

Zoo.Network leverages A-Chain for secure avatar rendering and physics computation:

- **Rendering Nodes**: GPU attestation ensures tamper-proof 3D rendering
- **Physics Servers**: CPU attestation for deterministic physics simulation
- **Anti-Cheat**: TEE verification prevents client-side manipulation
- **Asset Security**: NFT metadata generation in attested enclaves

7 Performance Analysis

7.1 Attestation Throughput

Metric	CPU Only	CPU+GPU	Target
Verification time	45ms	73ms	¡100ms
Throughput (attestations/sec)	1,420	876	¿500
Registry updates/sec	2,000	2,000	¿1,000
Gas per attestation	130k	190k	¡200k
Block time	500ms	500ms	500ms

Table 3: A-Chain performance benchmarks

7.2 Scalability

Horizontal Scaling:

- **Sharded registries**: Partition device registry by vendor
- **Batch verification**: Verify up to 16 quotes per transaction
- **Light clients**: Subnet nodes query registry without full sync
- **Caching**: 1-hour attestation validity with local caching

Capacity Analysis:

- Devices supported: ¿1 million concurrent
- Storage per device: 256 bytes
- Total registry size: ¡256 MB
- Query latency: ¡10ms with caching

8 Security Analysis

8.1 Threat Model

Adversary Capabilities:

1. Compromise of individual validator nodes
2. Side-channel attacks on TEE implementations

3. Replay attacks using old attestation quotes
4. Collusion between validators to approve invalid attestations

Security Guarantees:

- **Byzantine Fault Tolerance**: 2/3+ honest validators required
- **Cryptographic Verification**: All quotes verified against vendor root CAs
- **Nonce Protection**: Monotonic nonces prevent replay attacks
- **Economic Security**: Slashing penalizes malicious validators

8.2 Quantum Resistance

A-Chain integrates post-quantum cryptography for long-term security:

- **CRYSTALS-Dilithium**: Validator signatures (3,293 bytes)
- **Kyber**: Encrypted validator communication (1,568-byte ciphertexts)
- **Hybrid Mode**: Classical (BLS) + post-quantum (Dilithium) dual signatures
- **Migration Timeline**: Full quantum security by Q4 2027

9 Implementation Status

9.1 Current Deployment

Testnet (A-Chain Devnet):

- Validators: 32
- Devices attested: 2,847
- Uptime: 99.7%
- Average verification time: 52ms

Mainnet Launch: Q2 2025

9.2 Future Work

1. **Zero-Knowledge Proofs**: ZK-STARK based attestation aggregation for privacy
2. **Confidential Containers**: Extend support to Kata Containers and gVisor
3. **Decentralized Root CA**: DAO-governed root certificate management
4. **Cross-Chain Attestation**: IBC integration for Cosmos TEE verification
5. **Hardware Diversity**: Support for RISC-V based TEEs (Keystone)

10 Conclusion

A-Chain provides a unified attestation infrastructure for the Lux Network, enabling trustless AI compute across heterogeneous hardware. By supporting all major TEE vendors (Intel, AMD, NVIDIA, ARM) and compute classes (CPU, GPU, NPU, ASIC), A-Chain serves as a single source of truth for device trustworthiness.

The economic model cleanly separates security (LUX) from application incentives (AI Coin, ZOO), allowing specialized tokens to optimize for their domains while maintaining network-wide security guarantees. Performance benchmarks demonstrate sub-100ms attestation verification with capacity for over 1 million concurrent devices.

Integration with Hanzo.Network and Zoo.Network showcases A-Chain's versatility across AI orchestration and gaming/metaverse use cases. As decentralized compute becomes critical infrastructure, A-Chain's protocol-level attestation verification provides the foundation for trustless, verifiable computation at scale.

Acknowledgments

We thank the Lux Network community, Hanzo AI team, and Zoo ecosystem contributors for feedback and testing. Special thanks to Intel, AMD, NVIDIA, and ARM for technical documentation on their TEE implementations.

References

[1] V. Costan and S. Devadas, "Intel SGX Explained," *Cryptology ePrint Archive*, Report 2016/086, 2016.

[2] D. Kaplan, J. Powell, and T. Woller, "AMD Memory Encryption," *AMD White Paper*, 2016.

[3] NVIDIA, "H100 Tensor Core GPU Confidential Computing," *Technical Brief*, 2023.

[4] ARM, "Arm Confidential Compute Architecture," *Architecture Reference Manual*, 2021.

[5] Team Rocket et al., "Avalanche: A Novel Metastable Consensus Protocol," 2020.

[6] L. Ducas et al., "CRYSTALS-Dilithium: Algorithm Specifications and Supporting Documentation," *NIST PQC Submission*, 2020.

A Appendix A: Proof of Execution Format

```
struct ProofOfExecution {
    bytes32 deviceId;         // Unique device identifier
    bytes cpuQuote;           // TEE quote from CPU enclave
    bytes gpuQuote;           // Optional: TEE quote from GPU
    uint256 nonce;            // Monotonic replay protection
    bytes32 jobHash;          // Hash of job specification
    bytes32 outputHash;       // Hash of computation output
    uint256 timestamp;        // Execution timestamp
    bytes signature;          // Operator signature over proof
}
```

B Appendix B: Vendor Root CA Management

```
contract RootCARegistry {
    mapping(TEEVendor => bytes) public rootCertificates;
    mapping(TEEVendor => uint256) public lastUpdate;

    event RootCAUpdated(
        TEEVendor vendor,
        bytes32 oldRoot,
        bytes32 newRoot,
        uint256 timestamp
    );

    function updateRootCA(
        TEEVendor vendor,
        bytes calldata newRoot
    ) external onlyGovernance {
        require(newRoot.length > 0, "Invalid root CA");
        bytes32 oldRoot = keccak256(rootCertificates[vendor]);
        rootCertificates[vendor] = newRoot;
        lastUpdate[vendor] = block.timestamp;
        emit RootCAUpdated(vendor, oldRoot, keccak256(newRoot), block.timestamp);
    }
}
```

C Appendix C: Performance Optimization Techniques

Batch Verification:

- Aggregate up to 16 attestation quotes per transaction
- Amortize gas costs across multiple verifications
- 65% reduction in per-attestation gas cost

Merkle Compression:

- Store only Merkle root of quote in registry
- Full quote stored off-chain with on-demand retrieval
- 90% storage reduction

Validity Caching:

- Cache valid attestations for 1 hour
- Reduce on-chain queries by 80%
- Sub-10ms cache lookup