LoomCredit

Attested trade evidence and policy-constrained AI for supplier finance

Version 1.0 · 8 August 2026 · Creditcoin BUIDL CTC · Testnet prototype

> Verified trade events. Bounded AI. Testnet underwriting controls.

> Publication note: this Markdown is the current canonical whitepaper source. Regenerate and visually review the PDF after the corrected contract source, fresh deployment, and final evidence bundle are frozen. Existing receipts do not prove the newer lifecycle bytecode.

LoomCredit is a Creditcoin USC prototype for turning a buyer-backed trade event into a bounded underwriting action. It is a real testnet technical system in this checkout, but it is not a production lending product: the local browser policy lab uses explicitly labelled fixtures, the sandbox vault has no custody, and the recorded model-to-RiskGuard approval remains accounting-only testnet evidence.

Abstract

Small suppliers can have a legitimate purchase order and still lack the cash needed for materials, labour, packaging, testing, and logistics before delivery. The hard part is not producing another loan dashboard. It is establishing which commercial facts can be trusted, keeping the evidence current, and preventing an automated recommendation from exceeding policy.

LoomCredit tests a narrow protocol. A buyer or marketplace posts an order guarantee on a source chain. A worker waits for source finality and Attestcoin attestation, obtains a USC proof, and submits it to Creditcoin. `TradeEvidenceUSC` validates more than inclusion: it checks receipt success, the trusted emitter, the expected event, every business field, replay, and a scoped commercial fingerprint. A structured agent may then propose a facility quote. `RiskGuard`, not the agent, decides whether that quote can reserve accounting-only sandbox liquidity.

The prototype's contribution is a visible failure boundary. The local adversarial policy lab shows a safe 30% proposal passing and an unsafe 80% proposal being rejected against the deterministic cap. The recorded live bundle also includes a signed model quote and a backwards-compatible `QuoteApproved` receipt. The deployed bytecode does not expose the newer `QuoteDecisionAudited` event, and the bundle does not

claim a real loan, custody, physical delivery, legal enforceability, off-network duplicate financing, or borrower repayment ability.

Status vocabulary

The project uses status labels instead of blending a demo with a deployment.

Label	Meaning
`LOCAL_FIXTURE`	Deterministic browser data used to inspect the workflow. It is not a transaction.
`TESTNET`	A real testnet request or receipt with an address and explorer evidence.
`LIVE`	Production infrastructure with reviewed legal, security, privacy, and capital controls.
`PROPOSED`	A design or pilot requirement that is not implemented in this checkout.

The web console contains two explicit boundaries: the read-only live feed and live proof page use recorded testnet receipts, while the adversarial policy lab and local order route remain `LOCAL_FIXTURE`. Neither boundary invents a hash, proof, receipt, or model response.

1. The product wedge

The first user is not an anonymous borrower looking for a public lending pool. The initial customer is a marketplace, export aggregator, or trade platform that already knows the order context and wants a regulated lender to review supplier finance cases with stronger evidence and better lifecycle monitoring.

The operating model is:

```
buyer / marketplace
        ↓ buyer-backed order event
LoomCredit evidence and underwriting workflow
        ↓ verified packet + bounded recommendation
regulated lender or NBFC
        ↓ lender-owned agreement and disbursement
supplier
```

LoomCredit is infrastructure and workflow software in this model. A regulated capital provider owns lending, disbursement, collections, suitability, and jurisdictional compliance. Calling the current prototype a lender would be inaccurate.

2. Protocol flow

2.1 Source commitment

`OrderGuaranteeEscrow` records a minimal buyer-backed commitment and emits `OrderGuaranteed`. The event carries the order identifier, buyer, supplier, settlement token, order and guarantee amounts, delivery deadline, terms commitment, identity commitments, and nonce. The current source rejects zero buyer or supplier identity commitments so an order cannot silently omit both identity bindings. The contract also emits cancellation, dispute, and settlement events. On settlement, the specified amount goes to the supplier and any unused portion of the buyer's guarantee returns to the buyer, so the terminal escrow state has no residual balance.

The source contract does not make underwriting decisions. Its job is to emit purpose-built facts that the destination can validate.

2.2 Attestcoin and USC

The worker discovers the source event, waits for the required finality and attestation state, requests a Merkle and continuity proof, and submits the proof to Creditcoin. `TradeEvidenceUSC` calls Creditcoin's native query-verifier precompile at `0x0000000000000000000000000000000000000FD2`.

The precompile establishes transaction inclusion and finalized-chain continuity. The application contract separately requires a successful source receipt, the configured source contract, the exact event signature, and exact decoded values. Verification after attestation is synchronous in the Creditcoin transaction; the complete source-event-to-business-execution time still includes source finality, attestation, proof generation, submission, and inclusion. LoomCredit must measure that complete timeline rather than promise a universal fifteen-second unlock.

2.3 Evidence registration

The evidence registry stores only the normalized facts and commitments needed by policy. A replay key binds the source chain, block, transaction, log, and source contract. A commercial fingerprint binds the order and terms so the same obligation cannot create a second active facility inside participating LoomCredit integrations, even if it is recommitted in a new source transaction.

That lock is deliberately scoped. It cannot discover a facility originated outside integrations that share LoomCredit's registry.

2.4 Quote and reservation

The agent receives typed, attributable evidence and returns a schema-bound quote containing:

- decision and advance/fee basis points;
- expiry and risk tier;
- allowlisted reason codes;
- exact evidence identifiers;
- policy and model versions; and

- a signer-bound nonce.

`RiskGuard` checks the signer, evidence, facility state, expiry, nonce, policy version, guarantee ratio, advance cap, tenor, buyer concentration, and available sandbox liquidity. The settlement token is carried in the verified evidence, but the accounting-only vault does not custody or transfer tokens. Supplier exposure is tracked and released by the registry; no supplier concentration cap is enforced in this deployed demo. Only after the implemented checks pass can the accounting-only vault record a reservation.

The agent has no treasury key and cannot withdraw capital. Missing evidence or unavailable model configuration routes to `REFER`; an unsafe quote is rejected.

3. Policy envelope

The demonstration policy is intentionally simple and inspectable:

```
maximum advance              40% of order value
minimum buyer guarantee      10% of order value
maximum tenor                90 days
maximum buyer concentration  25% of sandbox capacity
quote lifetime               60-300 seconds; infrastructure-owned
```

These are control parameters, not a credit model and not a promise that a lender would use them in production. A production partner would need its own credit policy, affordability and fraud controls, KYB/KYC, legal review, model governance, and capital authorization.

4. Lifecycle and failure states

The facility lifecycle is evidence-driven:

```
EvidenceVerified → Quoted → Reserved → Settled
        │               │          ├──→ Disputed
        ├───────────────┴──────────┴──→ Cancelled
        └──────────────────────────────→ Settled
```

Cancellation, dispute, and settlement evidence can invalidate or close a facility, including direct `EvidenceVerified`/`Quoted` settlement and `Disputed` cancellation. The USC boundary compares the complete lifecycle payload, not just the event topic and order ID. Expired quotes and policy rejections remain off-chain decision outcomes; the current registry does not mutate them into terminal facility states. UI state is never authoritative; contracts and persisted worker records are. The worker stores event stages and a source cursor so a restart can reconcile rather than silently lose a case.

The negative path is part of the product, not a hidden test suite. The repository covers untrusted emitters, failed source receipts, proof replay, duplicate fingerprints, terms substitution, stale quotes, cancellation races, invalid transitions, and malicious agent recommendations.

5. Security and privacy boundary

The system separates trust responsibilities:

Boundary	What it establishes	What it does not establish
Attestcoin/USC	Inclusion and chain continuity	Receipt meaning, delivery, or legal enforceability
Source escrow	Allowed lifecycle event shape	Buyer identity or physical performance
Worker	Discovery, proof retrieval, retry, and submission	Permission to bypass contract validation
Agent	A structured recommendation	Authority over capital or policy
`RiskGuard`	Encoded action limits	Off-chain fraud that is absent from evidence
Regulated lender	Lending, agreement, and disbursement	Cryptographic truth without a verified packet

Full purchase orders, legal agreements, tax records, contact data, and KYB material should remain encrypted off-chain. Public commitments must be salted; predictable identifiers must not be hashed in a way that enables a cheap dictionary attack.

6. Product access: wallet connection plus verified sign-in

The browser exposes a real `Connect wallet` entry point using the injected EIP-1193 provider. After connection, `Sign in with wallet` requests a one-time server nonce and asks the wallet to sign a human-readable EIP-191 message. The server checks the exact stored message, chain, expiry, nonce replay state, and recovered address before binding the address to an account and server-controlled `viewer` or `operator` role.

Successful sign-in creates an opaque, expiring HttpOnly session token. The raw token is not stored in the database; only its SHA-256 hash is persisted. Sign-in, failed verification, and sign-out events are written to the authentication audit table. The current web surface has no capital-moving privileged action; future privileged handlers must require the session and call the shared audit helper before executing.

7. What is real in this checkout

Verified locally:

- source escrow lifecycle contracts compile and pass their Foundry suite;
- Creditcoin evidence, registry, `RiskGuard`, and sandbox vault contracts compile and pass their suite;

- the worker uses the official USC SDK shape, persists stages, and supports source-event watching with a cursor;
- the agent validates structured evidence, policy, model availability, and evidence binding;
- the Next.js product shell, demo lab, security boundary, order view, proof console, and server-verified wallet sign-in page build and run locally;
- the auth API issues one-time nonces, verifies EIP-191 signatures, persists accounts/sessions/audit events, and rejects replayed or mismatched signatures;
- browser smoke covers routes, responsive layout, demo interaction, API responses, and console errors.

Recorded testnet artifacts:

- Sepolia `MockUSDC` and `OrderGuaranteeEscrow` deployments with mined receipts;
- a successful `OrderGuaranteed` transaction at block `11443299`, with the source fields committed in `docs/deployments/source-order.json`;
- a successful proof-builder response matching the source block and proof shape;
- deployed CC3 `TradeEvidenceUSC`, `FacilityRegistry`, `RiskGuard`, and `SandboxCapitalVault` contracts with read-back wiring checks;
- a successful native USC verification receipt, on-chain evidence ID, and independent `EVIDENCE_VERIFIED` state read-back;
- a schema-valid Groq model quote, a dedicated signer, and a successful accounting-only `QuoteApproved` RiskGuard receipt for the same order and evidence ID. The later facility state is `RESERVED`.

Still not verified here:

- a fresh production model-provider request or production lending outcome;
- a live 80% RiskGuard rejection receipt;
- the newer `QuoteDecisionAudited` event on the currently deployed bytecode;
- a fresh live cancellation, dispute, or settlement worker receipt (worker routing and verifier calls are implemented and unit-tested);
- shared multi-instance production auth storage or persisted customer workspace data;
- regulated capital, lending agreements, custody, or disbursement.

The completed source-to-CC3 evidence bundle is recorded in `docs/demo-evidence.md`, including public transaction links, the evidence ID, and measured worker timings.

The honest product statement is therefore: **LoomCredit is a real technical prototype for evidence-bound underwriting controls, not yet a real lending product.** The next meaningful step is not another dashboard. It is one controlled pilot with a marketplace and a regulated lender, using redacted cases and an end-to-end testnet packet before any capital claim is made.

8. Evaluation and pilot gates

The technical demo succeeds when a judge can inspect one complete path and its failure modes:

1. a source order event is finalized, attested, proved, decoded, and registered;
2. the evidence is bound to the exact order terms and replay key;
3. the local policy lab accepts a safe 30% quote;
4. the local policy lab rejects an 80% quote against the deterministic cap;
5. replay, fake emitter, duplicate order, and cancellation cases fail closed;
6. worker restart preserves the case and retries safely;
7. stage-by-stage latency is published over at least ten runs; and
8. a credentialed release candidate supplies a live model request, signed quote, approval receipt, and unsafe rejection receipt.

The pilot gate adds commercial evidence: supplier need, buyer willingness to make commitments, lender review-time or evidence-quality improvement, privacy/KYB readiness, partner ownership of regulated activity, and written design-partner interest. Without those signals, the system remains a technically careful demo rather than a validated business.

9. Roadmap

Stage	Required outcome
Hackathon	One verified order path, bounded quote, attack suite, proof console, and honest evidence manifest
No-money pilot	Redacted real cases, one marketplace connector, one lender review workflow, durable auth and audit logs
Technical pilot	Live testnet/source integration, reliability metrics, privacy review, policy and model governance
Production facilities	Regulated partner, legal agreements, capital controls, security review, monitoring, and measured repayment outcomes

Features such as OCR, multi-model ensembles, public lending pools, tokenomics, arbitrary chain support, and complex dispute arbitration are intentionally below the critical path.

10. Conclusion

LoomCredit does not claim that a blockchain removes trade risk or that an AI model should control lending. Its narrower thesis is practical: verify the commercial evidence first, bind it to lifecycle state, let software propose an auditable action, and keep the final action inside deterministic policy.

Attestcoin proves the evidence. The agent proposes the action. RiskGuard controls the action. A regulated partner owns the loan.

Lineage and references

This whitepaper is a refined, implementation-aligned version of the project reference pack in `../loomcredit-assets`, especially its whitepaper, architecture/contract specification, threat model, technical-claims review, brand guide, and demo evidence manifest. The parent reference directory is used as source material and is intentionally not vendored into this repository.

- Creditcoin USC overview
- USC architecture overview
- USC query, proof, and verification
- USC migration guide
- Official USC testnet bridge examples