PAUSE

Protocol for Auditing and Underwriting Secure Exchanges

Technical Whitepaper

A Two-Phase Commit Protocol with Bayesian Probabilistic Risk Assessment for Secure, Non-Custodial Blockchain Transactions

Version 1.0 — February 2026
Author: Mithilesh Kumar
pausesecure.com | pausescan.com

Ethereum Mainnet | Live in Production | Non-Custodial | Open API

Table of Contents

1. Abstract

PAUSE (Protocol for Auditing and Underwriting Secure Exchanges) introduces a novel blockchain security ecosystem that solves the fundamental problem of irreversible, unverified cryptocurrency transactions. The system combines a Two-Phase Commit Protocol for non-custodial payments with a Bayesian Probabilistic Risk Assessment Engine that evaluates recipient addresses in real time using eleven independent signals.

In the PAUSE Commit Protocol, a sender creates a cryptographically signed payment intent off-chain at zero gas cost using EIP-712 typed data signing. The intent is shareable via link or QR code and revocable at any time. When the recipient commits the intent on-chain, the smart contract atomically verifies the signature, invokes the Bayesian Risk Engine to assess the recipient, and executes the token transfer if and only if the address passes the safety threshold. The entire operation is non-custodial: funds transfer directly from sender to recipient, never passing through any intermediary.

The Bayesian Risk Engine evaluates addresses through eleven independent signals — covering mixer exposure, wallet drainer patterns, scam graph connections, transaction anomalies, and more — combined using log-odds Bayesian scoring with correlation discounting. This produces calibrated probabilistic safety scores with confidence intervals, enabling principled risk-based decision making.

Additionally, the PAUSE ecosystem includes PAUSECard, a crypto-backed Visa card system that bridges blockchain and traditional payment networks through an optimistic authorization method, settling USDC on-chain asynchronously while meeting Visa's 2-second authorization requirement.

All components are deployed on Ethereum mainnet and live in production as of February 2026.

2. The Problem

2.1 Irreversibility Without Verification

Blockchain transactions are irreversible by design. Once a user signs and broadcasts a token transfer, the funds move permanently. There is no chargeback, no dispute resolution, and no central authority to reverse a fraudulent transfer. While this immutability is a feature of decentralized systems, it creates an asymmetric risk environment: the sender bears 100% of the risk in every transaction.

In 2023, cryptocurrency users lost over $5.6 billion to scams, phishing attacks, and malicious addresses. The fundamental cause is the absence of a pre-transaction verification layer that can assess recipient risk without introducing custodial intermediaries.

2.2 Technical Deficiencies in Current Systems

We identify five specific technical deficiencies in existing blockchain payment methods:

Deficiency	Description
Atomic Single-Phase	Standard ERC-20 transfers fuse intent and execution into one irreversible operation. No opportunity for pre-execution risk assessment.
No Intent Separation	No mechanism to express payment intent without simultaneously executing payment. The act of intending to pay and paying are indistinguishable.
Disconnected Risk	Existing address screening tools operate separately from the transaction flow. Users must manually check, interpret, context-switch, then transfer. Most skip verification entirely.
Custodial Solutions	Solutions providing payment safety (escrow, exchange custody) require a third party to hold funds, introducing counterparty risk and departing from non-custodial principles.
Binary Classification	Existing risk tools provide binary safe/unsafe verdicts or opaque scores without probabilistic foundations. Users cannot calibrate risk tolerance or understand assessment confidence.

Table 1: Technical deficiencies in existing blockchain payment systems

2.3 The Gap

There exists no system that simultaneously provides: (1) intent-execution separation at zero cost; (2) real-time probabilistic risk assessment embedded in the execution step; (3) unilateral sender revocation; (4) 100% non-custodial operation; and (5) calibrated Bayesian scoring with confidence intervals. PAUSE fills this gap.

3. System Architecture

3.1 Ecosystem Overview

The PAUSE ecosystem comprises three integrated subsystems:

Subsystem	Function	Technical Implementation
Risk Engine	Address risk scoring	11 Bayesian signals, log-odds scoring, correlation discounting. Python/Flask REST API. Deployed at pausescan.com.
Commit Protocol	Two-phase payments	Off-chain EIP-712 intent signing, on-chain atomic commit with risk gate. Solidity smart contract on Ethereum mainnet.
PAUSECard	Crypto-backed Visa	Optimistic authorization (<2s), async on-chain settlement. Node.js/Express with MySQL + Redis. Deployed at card-api.pausesecure.com.

Table 2: PAUSE ecosystem components

3.2 System Flow Diagram

The following diagram illustrates the complete transaction flow through the PAUSE ecosystem:

```
┌──────────────────────────────────────────────────────────┐
│                    PAUSE TRANSACTION FLOW                │
└──────────────────────────────────────────────────────────┘


PHASE 1: INTENT (Off-Chain, Zero Gas)
──────────────────────────────────────
Sender → Specifies: {to, token, amount, expiry}
       → EIP-712 typed data constructed
       → Wallet signs → (v, r, s) signature
       → Intent encoded as URL / QR code
       → Shared with recipient via any channel

       [Sender can REVOKE at any time before commit]

PHASE 2: COMMIT (On-Chain, Atomic, ~85k gas)
──────────────────────────────────────
Recipient → Opens intent link, clicks 'Claim'
          → Smart contract verifies EIP-712 signature
          → Checks: not expired, not revoked, not executed
          → RISK ENGINE: 11 Bayesian signals evaluated
          → Score computed via log-odds + correlation discount
          → IF score >= 40: safeTransferFrom(sender, recipient)
          → IF score < 40:  REVERT (funds stay with sender)
```

```
RESULT: Safe transfer, or blocked with reason. Non-custodial.
```

Figure 1: Complete PAUSE transaction flow

4. The Commit Protocol

4.1 Design Principles

- Non-custodial: The protocol smart contract never holds user funds. All transfers are direct sender-to-recipient via safeTransferFrom.
- Zero-cost intent: Creating a payment intent costs nothing. Only the commit phase consumes gas.
- Sender sovereignty: The sender retains unilateral revocation ability until the moment of commitment.
- Atomic execution: The commit either fully succeeds (signature valid, risk passed, transfer complete) or fully reverts (no state change).
- Risk-gated: No transfer executes without passing the Bayesian risk assessment.

4.2 Phase 1: Intent Creation

The sender constructs an EIP-712 typed data structure with the following schema:

```
EIP-712 Domain Separator:
  name:              'PAUSE Commit Protocol'
  version:           '2'
  chainId:           1 (Ethereum mainnet)
  verifyingContract: 0x604152D82Fd031cCD8D27F26C5792AC2CD328bF4

PaymentIntent Type:
  recipient:  address    // Ethereum address of recipient
  token:      address    // ERC-20 token contract (e.g., USDC)
  amount:     uint256    // Transfer amount in token base units
  nonce:      uint256    // Unique nonce to prevent replay
  expiry:     uint256    // Unix timestamp after which intent expires
```

The sender's wallet signs this typed data, producing an ECDSA signature (v, r, s). The signed intent is then serialized and encoded into a URL:

```
https://app.pausesecure.com/claim?intent=<base64-encoded-signed-intent>
```

This URL can be transmitted through any channel: email, SMS, messaging apps, QR codes, or embedded in web pages. The recipient does not need to have any prior relationship with the sender or the PAUSE platform to receive and claim the intent.

4.3 Phase 2: Atomic Commit

When the recipient opens the intent link and initiates the claim, the following steps execute atomically within a single Ethereum transaction:

```
function commit(SignedIntent calldata intent) external {
    // Step 1: Recover signer from EIP-712 signature
    address signer = ecrecover(hashIntent(intent), intent.v, intent.r, intent.s);
    require(signer == intent.sender, 'Invalid signature');

    // Step 2: Validate intent state
    bytes32 intentHash = keccak256(abi.encode(intent));
    require(!executed[intentHash], 'Already executed');
    require(!revoked[intentHash], 'Revoked by sender');
    require(block.timestamp <= intent.expiry, 'Expired');

    // Step 3: Risk assessment gate
    uint256 riskScore = riskEngine.assess(msg.sender); // recipient
    require(riskScore >= SAFETY_THRESHOLD, 'Risk threshold not met');

    // Step 4: Atomic non-custodial transfer
    executed[intentHash] = true;
    IERC20(intent.token).safeTransferFrom(
        intent.sender, msg.sender, intent.amount
    );

    emit Committed(intentHash, intent.sender, msg.sender, intent.amount);
}
```

Figure 2: Commit function pseudocode (Solidity)

4.4 Revocation

The sender may revoke any outstanding intent by calling:

```
function revoke(bytes32 intentHash) external {
    require(msg.sender == intents[intentHash].sender, 'Not sender');
    require(!executed[intentHash], 'Already executed');
    revoked[intentHash] = true;
    emit Revoked(intentHash, msg.sender);
}
```

Revocation is permanent and irrevocable. Once an intent is revoked, no party can execute it. This ensures the sender maintains sovereign control over their commitment at all times prior to execution.

5. Bayesian Probabilistic Risk Engine

5.1 Signal Pipeline

The Risk Engine evaluates a target Ethereum address through eleven independent risk signals. Each signal analyzes on-chain transaction data and returns a probability P(malicious | signal) ∈ [0.0, 1.0] with a confidence level C ∈ [0.0, 1.0]:

#	Signal	W	Detection Target
1	`mixer_exposure`	**0.45**	Transaction graph interaction with known mixing/tumbling services (Tornado Cash, Blender.io). Analyzes deposits to and withdrawals from mixer contract addresses within N-hop transaction radius.
2	`draining_pattern`	**0.30**	Wallet drainer behavior: rapid ERC-20 approval transactions (approve/increaseAllowance) followed by immediate transferFrom sweeps. Time-windowed pattern matching on approval-to-sweep latency.
3	`exchange_cluster`	**0.25**	Graph distance to known exchange hot/cold wallets. Addresses within 2-hop distance of major exchange infrastructure scored as lower risk (legitimate trading activity).
4	`sweep_pattern`	**0.25**	Fund consolidation detection: multiple inbound transfers from distinct addresses aggregated into single outbound transfers. Fan-in ratio analysis with time-decay weighting.
5	`tx_burst_anomaly`	**0.20**	Statistical anomaly detection on transaction frequency. Z-score analysis against address's historical baseline. Detects bot-like volume spikes indicating automated attack scripts or MEV extraction.
6	`scam_graph`	**0.20**	Graph connectivity to community-reported scam addresses (Chainabuse database). Evaluates direct connections (1-hop) and indirect connections (2-hop) with distance-decayed weighting.
7	`dusting_attack`	**0.15**	Micro-transaction pattern detection: outbound transfers below dust threshold ($0.01) to many distinct addresses. Characteristic of address deanonymization and UTXO linking attacks.
8	`ens_authenticity`	**0.15**	ENS domain ownership verification: registration age, forward/reverse resolution consistency, similarity to known brand names (typosquatting detection), and registration renewal history.
9	`rival_consensus`	**0.15**	Cross-referencing with external risk scoring providers. Aggregates assessments from multiple independent services using weighted voting with disagreement penalties.
10	`wallet_age`	**0.10**	First-transaction-date analysis. Addresses created within 30 days receive elevated risk. Score decays logarithmically with account age, plateauing at 1 year.
11	`balance_volatility`	**0.10**	Rolling-window standard deviation of balance over time. Rapid large inflows followed by immediate complete outflows indicate pass-through or laundering behavior.

Table 3: Complete signal pipeline with detection targets

5.2 Scoring Algorithm

The scoring engine combines the eleven signal outputs using a Bayesian log-odds approach with correlation discounting. This method provides theoretically correct evidence combination with principled handling of non-independent signals.

5.2.1 Algorithm Pseudocode

```
FUNCTION compute_safety_score(address):

  // Step 1: Base prior
  P_prior = 0.15                          // 15% baseline P(scam)
  LO = log(P_prior / (1 - P_prior))       // Convert to log-odds

  // Step 2: Evaluate all signals
  FOR each signal_i in [mixer_exposure, draining_pattern, ...]:
    P_i = signal_i.evaluate(address)       // P(malicious | signal)
    C_i = signal_i.confidence()            // Confidence [0, 1]
    W_i = signal_i.weight                  // Pre-configured weight

    // Convert signal probability to log-odds evidence
    LO_i = log(P_i / (1 - P_i)) * W_i * C_i

    // Step 3: Apply correlation discount
    D_i = get_discount(signal_i, active_signals)
    LO_i = LO_i * D_i                     // D=1.0 if no correlation

    LO = LO + LO_i                         // Additive in log-odds

  // Step 4: Convert back to probability
  P_final = 1 / (1 + exp(-LO))

  // Step 5: Map to safety score
  score = (1 - P_final) * 100              // 0-100, higher = safer

  // Step 6: Classify
  IF score >= 70: risk_level = SAFE         // P(malicious) < 30%
  ELIF score >= 40: risk_level = MEDIUM     // P(malicious) 30-60%
  ELSE: risk_level = HIGH                   // P(malicious) > 60%

  RETURN {score, risk_level, P_final, signals_breakdown}
```

Figure 3: Bayesian scoring algorithm pseudocode

5.2.2 Correlation Discount Matrix

Some signals are not fully independent. For example, mixer_exposure and sweep_pattern may both trigger on the same underlying laundering behavior. The correlation discount matrix prevents double-counting:

Signal A	Signal B	Discount	Rationale
`mixer_exposure`	`sweep_pattern`	**0.60**	Mixer usage often involves sweep consolidation
`draining_pattern`	`tx_burst_anomaly`	**0.70**	Drainer attacks produce burst anomalies
`scam_graph`	`mixer_exposure`	**0.75**	Reported scams often use mixers
`sweep_pattern`	`balance_volatility`	**0.65**	Sweeps cause volatility spikes

Table 4: Correlation discount matrix (lower-weight signal in each pair is discounted)

5.2.3 Worked Example

Consider address 0xABC...123 with the following signal evaluations:

```
Address: 0xABC...123 (recently created, interacted with Tornado Cash)

Signal Results:
  mixer_exposure:      P=0.85, C=0.90, W=0.45  → LO = 1.735 * 0.45 * 0.90 = 0.703
  draining_pattern:    P=0.10, C=0.80, W=0.30  → LO = -2.197 * 0.30 * 0.80 = -0.527
  exchange_cluster:    P=0.30, C=0.70, W=0.25  → LO = -0.847 * 0.25 * 0.70 = -0.148
  sweep_pattern:       P=0.70, C=0.75, W=0.25  → LO = 0.847 * 0.25 * 0.75 = 0.159
    (correlation discount with mixer_exposure: 0.159 * 0.60 = 0.095)
  tx_burst_anomaly:    P=0.20, C=0.60, W=0.20  → LO = -1.386 * 0.20 * 0.60 = -0.166
  scam_graph:          P=0.50, C=0.50, W=0.20  → LO = 0.000 (neutral)
  dusting_attack:      P=0.05, C=0.90, W=0.15  → LO = -2.944 * 0.15 * 0.90 = -0.397
  ens_authenticity:    P=0.50, C=0.00, W=0.15  → LO = 0.000 (no ENS, neutral)
  rival_consensus:     P=0.60, C=0.40, W=0.15  → LO = 0.405 * 0.15 * 0.40 = 0.024
  wallet_age:          P=0.75, C=0.95, W=0.10  → LO = 1.099 * 0.10 * 0.95 = 0.104
  balance_volatility:  P=0.40, C=0.65, W=0.10  → LO = -0.405 * 0.10 * 0.65 = -0.026

Base prior LO: log(0.15 / 0.85) = -1.735

Combined LO: -1.735 + 0.703 + (-0.527) + (-0.148) + 0.095 +
             (-0.166) + 0.000 + (-0.397) + 0.000 + 0.024 +
             0.104 + (-0.026) = -2.073

P_final = 1 / (1 + exp(2.073)) = 0.112  (11.2% probability malicious)
Score = (1 - 0.112) * 100 = 88.8

RESULT: Score 88.8 → SAFE (despite mixer interaction,
        strong age/exchange/no-drainer signals outweigh it)
```

Figure 4: Worked example — Bayesian score computation for a sample address

This example demonstrates how the Bayesian approach handles mixed evidence: the high mixer_exposure signal (P=0.85) is offset by strong negative evidence from draining_pattern (P=0.10), dusting_attack (P=0.05), and the correlation discount on sweep_pattern. The final score reflects the overall weight of evidence, not any single signal.

6. PAUSECard: Optimistic Authorization

6.1 The Timing Problem

Traditional payment networks (Visa, Mastercard) require authorization responses within 2 seconds. Ethereum blockchain transactions require approximately 12 seconds for confirmation, with settlement finality taking several minutes. This fundamental timing mismatch has prevented non-custodial crypto-backed card systems.

6.2 Optimistic Authorization Method

PAUSE introduces an optimistic authorization method that bridges this gap:

```
VISA AUTHORIZATION FLOW (< 2 seconds):

1. User swipes Visa card at merchant
2. Visa sends auth request to card-api.pausesecure.com
3. Backend checks Redis cache:
   - On-chain USDC balance (cached, refreshed every 30s)
   - On-chain USDC allowance to PAUSECard contract
   - Current spending limit and period
   - Sum of pending (unsettled) amounts
4. Available = min(balance, allowance, limit) - pending
5. IF amount <= available:
   - ATOMIC: Redis INCRBY pending amount (prevents double-spend)
   - MySQL: Insert transaction + enqueue settlement
   - RESPOND: APPROVE to Visa
6. IF amount > available:
   - RESPOND: DECLINE to Visa

SETTLEMENT FLOW (asynchronous, ~12 seconds):

7. Worker polls MySQL queue every 2 seconds
8. Dequeues item atomically (SELECT FOR UPDATE)
9. Calls PAUSECard.settle(cardHash, amount, settlementId)
10. On success: clear pending, confirm transaction
11. On failure: retry with exponential backoff (5 attempts)
    Backoff: 30s → 60s → 120s → 240s → 480s
12. After max retries: create alert, notify via email
```

Figure 5: Optimistic authorization and asynchronous settlement flow

6.3 Double-Spend Prevention

The atomic Redis INCRBY operation in step 5 is the critical safety mechanism. By atomically incrementing the pending amount before responding to Visa, the system ensures that concurrent authorization requests for the same card cannot both succeed if the combined amount would exceed the available balance. This is equivalent to a compare-and-swap operation on the available balance.

6.4 Crash Recovery

The MySQL-persistent settlement queue ensures no settlement is ever lost. If the backend crashes after approving a Visa transaction but before settling on-chain, the worker recovers all queued settlements on restart. A reconciliation job runs every 5 minutes to detect and re-queue any orphaned transactions.

7. Comparison with Prior Art

The following table compares PAUSE's capabilities with existing solutions in the blockchain payment and security space:

Capability	PAUSE	Chainalysis	Standard ERC-20	Payment Channels	Escrow	CEX P2P
Non-custodial	✓	N/A	✓	✓	✗	✗
Intent-execution split	✓	✗	✗	Partial	✓	✓
Integrated risk gate	✓	✗	✗	✗	✗	✗
Bayesian probabilistic	✓	✗	✗	✗	✗	✗
Sender revocation	✓	N/A	✗	Partial	✓	✓
Zero-cost intent	✓	N/A	✗	✗	✗	✗
Confidence intervals	✓	✗	✗	✗	✗	✗
Crypto-backed Visa	✓	✗	✗	✗	✗	✗

Table 5: Capability comparison with existing solutions

No existing solution combines all seven capabilities. Chainalysis and similar services provide risk scoring but as a disconnected service, not embedded in transaction execution. Standard ERC-20 transfers have no intent separation or risk assessment. Payment channels provide some intent-execution separation but no risk scoring. Escrow services provide safety but sacrifice non-custodial operation. Centralized exchange P2P provides safety through custody but requires trust in the exchange.

PAUSE is the only system that provides non-custodial, zero-cost intent creation with integrated Bayesian risk gating in the atomic execution step, combined with unilateral sender revocation and calibrated confidence intervals.

8. Security Model

8.1 Smart Contract Security

- ReentrancyGuard (OpenZeppelin): All fund-moving functions are protected against reentrancy attacks.
- Pausable: Contract owner can freeze all operations in case of discovered vulnerability.
- Non-custodial design: Contract never holds token balances. safeTransferFrom operates on sender's pre-approved allowance.
- Intent replay protection: Each intent hash is stored on-chain after execution, preventing replay.
- Expiry enforcement: Block timestamp comparison prevents execution of stale intents.
- Signature verification: EIP-712 ecrecover ensures only the declared sender can authorize the transfer.

8.2 Risk Engine Security

- Graceful degradation: If Etherscan API or Chainabuse API is unavailable, affected signals return neutral P=0.5, C=0.0 outputs that do not influence the score.
- Cache isolation: Results are cached per address with configurable TTL (default: 1 hour). Stale cache entries are never served.
- Rate limiting: Token-bucket rate limiting prevents abuse of the scoring API.
- No PII storage: The Risk Engine stores no personally identifiable information. Only Ethereum addresses and their scores are processed.

8.3 PAUSECard Security

- Atomic pending tracking: Redis INCRBY ensures concurrent authorization requests cannot double-spend.
- Persistent queue: MySQL settlement queue survives crashes. No settlement is ever lost.
- On-chain duplicate guard: Settlement IDs are unique on-chain. Contract rejects duplicate settle() calls.
- Exponential backoff: Failed settlements retry with increasing delays (30s to 480s) before alerting.
- Email alerts: Resend-based notifications for settlement failures, orphaned transactions, and daily digests.

9. Deployment and Production Status

All PAUSE ecosystem components are deployed and operational on Ethereum mainnet as of February 2026:

Component	Deployment
Commit Protocol	`Ethereum Mainnet: 0x604152D82Fd031cCD8D27F26C5792AC2CD328bF4`
PAUSECard Contract	`Ethereum Mainnet: 0x4BE2522662cEd5ba35553CC78dfE1Ca54a372eb7`
USDC Token	`Ethereum Mainnet: 0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48`
Risk Engine API	pausescan.com — Python/Flask, 11 Bayesian signals, SQLite
Card API	card-api.pausesecure.com — Node.js/Express, MySQL, Redis, Railway
Frontend	app.pausesecure.com — Next.js, wagmi, RainbowKit, Vercel

10. Conclusion

PAUSE introduces a fundamentally new approach to blockchain transaction safety. By separating payment intent from payment execution, embedding real-time Bayesian probabilistic risk assessment into the atomic commit step, and maintaining 100% non-custodial operation, PAUSE enables the safest possible way to transact cryptocurrency.

The Bayesian Risk Engine's eleven-signal pipeline with log-odds combination and correlation discounting produces calibrated probabilistic scores that enable principled risk-based decision making, replacing the binary blacklist approaches of existing solutions.

The Commit Protocol's two-phase design gives senders zero-cost intent creation, shareable payment links, and unilateral revocation — capabilities that do not exist in any current blockchain payment system.

The PAUSECard system extends this security infrastructure to the physical world, bridging blockchain and traditional payment networks through an optimistic authorization method that solves the fundamental timing mismatch between real-time card networks and blockchain confirmation times.

All components are deployed on Ethereum mainnet, live in production, and processing real transactions as of February 2026.

— — —

References

[1] EIP-712: Typed Structured Data Hashing and Signing. Ethereum Improvement Proposals. https://eips.ethereum.org/EIPS/eip-712

[2] OpenZeppelin Contracts. ReentrancyGuard, Pausable, SafeERC20. https://openzeppelin.com/contracts

[3] Chainabuse. Community-driven database of reported scam addresses. https://chainabuse.com

[4] Etherscan. Ethereum blockchain explorer and analytics platform. https://etherscan.io

[5] USDC (USD Coin). Circle. ERC-20 stablecoin. Contract: 0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48

[6] Bayes, T. (1763). An Essay towards solving a Problem in the Doctrine of Chances. Philosophical Transactions of the Royal Society of London.

Legal Notice

This whitepaper is published to establish prior art and public disclosure of the technical innovations described herein. The publication date of this document serves as the earliest public disclosure date for the described systems and methods.

Patent rights are reserved. A provisional patent application covering the systems and methods described in this whitepaper has been or will be filed with the United States Patent and Trademark Office.

The information in this document is provided for informational purposes. Nothing in this whitepaper constitutes financial, legal, or investment advice.