Bonsai: A Recovery Approach for Ethereum ERC-20 Transactions

Diogo Melita, David R. Matos, and Miguel L. Pardal
INESC-ID, Instituto Superior Técnico, University of Lisbon, Portugal
{diogo.melita, david.r.matos, miguel.pardal}@tecnico.ulisboa.pt

***Abstract*—Blockchain technology offers a mechanism for storing data with cryptographic links between blocks, creating a tamper-resistant ledger. Although this immutability ensures data integrity, it complicates recovery in cases of errors or intrusions. This work proposes Bonsai, an error and intrusion recovery system designed for token exchanges on Ethereum based applications. The system includes a custom ERC-20 token (`BON`) that maintains a one to one peg with ETH tokens while enabling transaction reversals through arbitration trials and an insurance mechanism to protect users against losses. Our experimental evaluation on Ethereum Sepolia and ZKsync Sepolia demonstrates that Bonsai can successfully trace and reverse token flows through up to five wallets in under 20 seconds, at an average cost of approximately $0.30 on ZKsync. Existing blockchain recovery approaches are slow and costly, with reversal operations taking up to 126 seconds, and some may not be able to complete the reversal. The system provides a practical solution for blockchain applications requiring error and intrusion correction capabilities while preserving the authentication, integrity, immutability, and non-repudiation properties of Blockchain.**

***Index Terms*—Blockchain, Immutability, Intrusion Recovery, Incident Response**

I. Introduction

Blockchain technology enables decentralized transaction records without trusted third parties through consensus protocols and cryptographic techniques ensuring authenticity, integrity, and non-repudiation. This creates immutable, tamper-evident ledgers where data modifications break hash chains and are detectable by all peers. Since Bitcoin was first introduced [1], blockchain has found applications in finance, offering decentralized alternatives to traditional banking; in supply chains, improving traceability and reducing fraud [2]; and in healthcare, securing patient data access [3]. Smart contracts, popularized by Ethereum, automate processes in insurance [4], real estate [5], and governance [6], reducing the reliance on intermediaries.

Immutability is a core property of blockchain systems that makes it extremely difficult to alter or revert transactions after their confirmation on the network. Although immutability offers benefits in applications such as asset transfers, where maintaining permanent and tamper-proof records is crucial for establishing trust and preventing fraud [7], it creates challenges for sectors that require operational flexibility and concerns with data protection regulations. If blockchain transactions are permanently unchangeable, then it is impossible to revert effects of errors or intrusions. Furthermore, vulnerabilities in smart contracts can create irreversible unauthorized states, as demonstrated by the 2016 Ethereum attack where nearly $150 million USD were stolen [8]. Traditional intrusion recovery systems for web applications [9]–[11], databases [12], file systems [13], [14], and virtual machines [15], [16] are not suitable for blockchains, whose immutability and continuous availability require new recovery solutions.

Some approaches address blockchain immutability by introducing reversal mechanisms [17], [18], though most require protocol modifications. Compensating transaction approaches [19]–[21] retain complete transaction history but suffer from slow execution (up to 126 seconds in Recoverable Token [20]) and high costs due to the blockchain's low throughput. Layer 2 solutions like sidechains [22] and rollups [23] offer higher throughput and lower latency [24], making them suitable for reducing recovery overhead. Blockchain layers separate responsibilities: Layer 1 blockchains, like Bitcoin and Ethereum, provide foundational security and consensus, while Layer 2 solutions can improve scalability, speed, or cost.

In this work, we present Bonsai [1], an approach for blockchain transaction recovery on EVM-based blockchains [2] intended to operate on a Layer 2 system. Bonsai requires no protocol modifications and introduces arbitration trials for reversal requests with an insurance mechanism for user protection. Our solution enables mutable transactions during a cool-down period while preserving security guarantees through a smart contract-managed log that tracks transactions until they become immutable, after the cool-down window expires.

This work proposes a new ERC-20 token, `BON`, and its support systems. The technical contributions are the following:

C1 Bonsai, an intrusion recovery system featuring the `BON` ERC-20 token, allowing mutable transactions during cool-down periods with arbitrator-controlled reversion.

Work supported by national funds through Fundação para a Ciência e a Tecnologia, I.P. (FCT) under projects UID/50021/2025 and UID/PRR/50021/2025 (INESC-ID), 2024.07494.IACDC (WELL), and Project Blockchain.PT – Decentralize Portugal with Blockchain Agenda, (Project no 51), WP 1: Agriculture and Agri-food, Call no 02/C05-i01.01/2022, funded by the Portuguese Recovery and Resilience Program (PRR), The Portuguese Republic and The European Union (EU) under the framework of Next Generation EU Program.

[1]Bonsai stands for Blockchain Operations Network for Secure Arbitrated Immutability. The name Bonsai draws an analogy to taking care of a bonsai tree — carefully shaping its growth. In this system, blockchain transactions can be adjusted within a defined time window through arbitration. After this period, the history becomes immutable.

[2]EVM stands for Ethereum Virtual Machine.

C2 An insurance mechanism to compensate users that are negatively impacted by transaction reversals, with payouts proportional to contributions to reduce cryptocurrency transfer risks.

C3 An implementation of the system, publicly available at Bonsai's GitHub repository [3], The prototype can be tested in both a local deployment of a blockchain or in a public testnet or mainnet.

C4 An experimental evaluation of the Bonsai system conducted on two public blockchain test networks, Ethereum Sepolia (a Layer 1 blockchain) and ZKsync Sepolia (a Layer 2 blockchain), which allows for a comparative analysis of latency and costs across both layers.

Our evaluation shows Bonsai can trace and reverse token flows across 5 wallets in under 20 seconds at a cost of approximately 2,186,187 gas units (about $0.30 on ZKsync).

II. Bonsai Overview

Bonsai proposes an alternative approach to prior works in this field to enable reversal of blockchain transactions, intending the system to be deployed on a Layer 2 network. To interact with this system, users must first get `BON` tokens in exchange for ETH ones. Afterwards, it is possible to transfer `BON` tokens between addresses. When a user intends to revert a transaction, they can request an arbitration trial. The arbitrators for the trial are chosen randomly, from all the users that hold `BON` tokens and are not involved in the dispute, and must make a decision on whether to revert the transaction. When a user no longer wishes to interact with Bonsai, they can get ETH tokens back. In Figure 1, we represent the lifecycle of tokens.

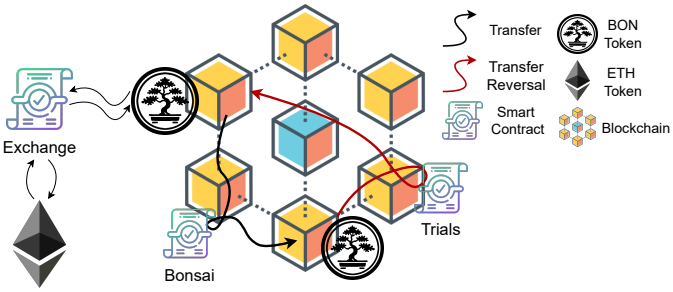

Fig. 1. Life cycle of tokens in the Bonsai environment.

A. Users

There are two types of users of the Bonsai system:

- **Blockchain users**: regular users that execute transactions to transfer `BON` tokens to other users.
- **Arbitrators**: special users responsible to arbitrate reversal requests and determine whether the reversal should be carried out. Every user in the network can be an arbitrator.

[3]https://github.com/inesc-id/Bonsai-Prototype

B. Threat Model

We define a threat as an intrusion, caused by a malicious actor, or an error, caused by an authorized user acting erroneously or under compromised credentials. We consider two main threat sources that can produce intrusions: malicious actors using stolen private keys to send transactions the rightful owner would not authorize under normal conditions; legitimate users submitting erroneous transactions, such as incorrect amounts or wrong recipient addresses.

C. System Model

a) State: The system's state is maintained on the blockchain, which stores confirmed transactions, contract states, user balances (in `BON` tokens), and trial data. Evidence for trials is stored on IPFS [4] to minimize on-chain data while maintaining references to off-chain information.

b) Assumptions: We assume: (A1) Each trial has a sufficiently large set A of potential arbitrators to select an anonymous committee $a \subseteq A$; (A2) Mixers are not used with Bonsai, as countermeasures [25], [26] are complex and beyond this study's scope; (A3) Trials conclude before transaction cooldown periods expire, preventing indefinite balance freezing.

D. System Architecture

Figure 2 shows the architecture of Bonsai, consisting of four main smart contracts plus a utility library (`BONLib`). The `Bonsai` contract handles `BON` token transfers, and stores mutable transactions and balances; the `Exchange` contract manages token swaps between ETH and `BON`; the `Trials` contract oversees the complete arbitration lifecycle, allowing parties to submit evidence and arbitrators to vote on reversal requests; the `Insurance` contract provides additional user protection by only interacting with `Bonsai` and `Trials` to manage insurance operations; and the `BONLib` library provides shared data structures and helper functions used consistently across all contracts.

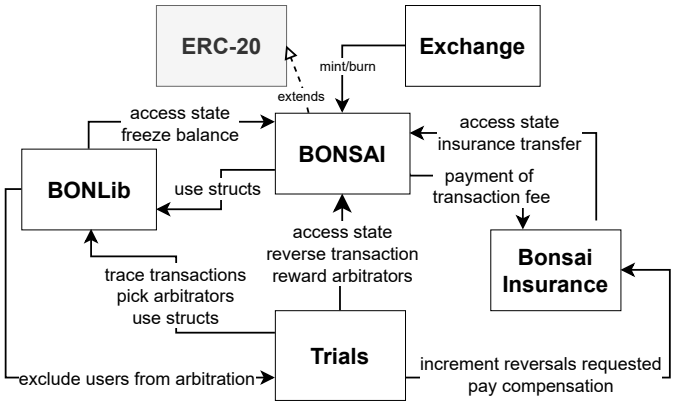

Fig. 2. Bonsai system architecture.

[4]https://ipfs.tech/

III. Bonsai in Depth

This Section details how user interaction, the expected process of a trial, key components, and design decisions.

A. Token Operations

We introduced `BON`, a new ERC-20 token, designed as a reversible ETH, maintaining a 1:1 value ratio with ETH. While `BON` could be developed as a reusable token interface, similar to ERC-20R [21] and Recoverable Token [20] proposals, we implemented it as a standalone token inheriting selected ERC-20 functionality to create a reversible ETH variant.

To interact with Bonsai, users must bridge ETH tokens to the network where Bonsai is deployed and then exchange ETH for `BON` tokens via the `getBonsai` method of the `Exchange` contract. The `Bonsai` contract logs mutable transaction IDs for tracking as `BON` transactions enter a cool-down period during which they remain mutable and reversible. If a reversal is requested during this period, an arbitration trial begins. An arbitration trial is a legally binding dispute resolution process where arbitrators review evidence and decide on the transaction reversal.

Users can exchange `BON` back to ETH using the `Exchange` contract's `getETH` method. However, only "stable" tokens (past the cool-down period) can be withdrawn. The `Bonsai` contract maintains an "unstable" token balance that increases proportionally when users receive tokens. Users must invoke `finalizeTransactions` to verify pending transactions and convert unstable tokens to stable ones after the cool-down period expires. The `Exchange` contract burns the stable `BON` tokens and transfers equivalent ETH to the user.

Unlike ERC20-R [21], Bonsai users maintain a single combined balance of stable and unstable tokens, enabling full network interaction while the unstable balance safeguards against improper withdrawals or token exchanges that could prevent recovery, reducing reliance on the Bonsai Insurance system, detailed in Section III-E.

B. Trials

If during the cool-down period the issuer of a transaction requests a reversal, an arbitration trial will be initiated. Reversal requests incur fees to reward arbitrators and incentivize good behavior. To initiate a trial, the issuer of a transaction triggers the `initTrial` method of the `Trials` contract with the transaction ID requiring reversal. Once the request is made, it is assigned with a unique ID and the `Trials` smart contract will freeze the transaction's amount on the destination address. This is done to prevent the spending of funds signaled to possible reversals. If this was not done, a destination party could move the funds after seeing a trial was initiated for a transaction where they are an interested party, impeding the reversal to be carried out.

Afterwards, the `Trials` contract will follow the trial initiation process by randomly selecting the arbitrators from network users, excluding the transaction's source and destination addresses. Due to blockchain determinism preventing true randomness, Bonsai draws inspiration from Chainlink Verifiable Random Function (VRF) [5] for secure random number generation, with future plans to integrate this service directly. We assume oracle existence for arbitrator notifications, potentially using protocols like Push [6], though implementation remains outside this work's scope. Every user in the network (i.e., anyone holding `BON`) is eligible to be part of the arbitrator committee. This decision allows the arbitrator committee to remain fully decentralized, instead of having a pool of possible arbitrators, as in the ERC-20R and ERC-721R proposals [21].

Once arbitrators are selected, a network event signals trial initiation with transaction and trial IDs. Both parties must upload evidence supporting their cases, presented as authentic documents (for example police reports or signed legal contracts) ensuring integrity and preventing tampering. While forged documents remain possible, the arbitration committee evaluates evidence fairly based on available information. Although verifying document authenticity could enhance trial fairness, it lies beyond the scope of intrusion recovery.

To submit evidence supporting a party's case, the user must invoke the `submitEvidence` method of the `Trials` smart contract, providing a payload with the required details. This payload must conform to the data structure shown in Figure 3. The submitting party is responsible for uploading the evidence to IPFS and ensuring that it remains accessible to the arbitrators via the provided URL.

```
{
 "txID": 0,
 "ipfsUrl": "https://ipfs.io/ipfs/
QmXoypizjW3WknFiJnKLwHCnL72vedxjQkDDP1mXWo6uco",
 "evidenceHash":
"2c26b46b68ffc68ff99b453c1d30413413422e4f173dc59c51f3d1d5e
e6f7f90",
 "description": "Theft report, witness statements, proof
of agreement"
}
```

Fig. 3. Evidence example. The actual content is stored in the IPFS link

Once both evidences are submitted, arbitrators review them and vote via the blockchain. The voting phase requires $2f+1$ votes (where f is the maximum number of faulty arbitrators and configurable before deployment), with at least $f+1$ identical responses. With only two options (`Approve` or `Deny`) and $2f+1$ being odd, one option will always have a majority, ensuring conclusive results. The voting phase ends at $2f+1$ votes to handle offline or non-participating arbitrators. Given a maximum of f faulty arbitrators, the total committee size is $N = 3f+1$.

To prevent vote influence and maintain secrecy, Bonsai employs commit-reveal voting. Arbitrators first commit vote hashes with chosen secrets using `commitVote`, then reveal votes via `revealVote` with trial ID, vote, and secret. The contract verifies hash matches before registering votes. After

[5] https://chain.link/vrf
[6] https://comms.push.org/docs/notifications/

all revelations, the guaranteed $f + 1$ equal responses enable faulty quorum majority decisions.

Upon reaching decisions, arbitrators invoke `finishTrial` to complete trials. If no action is required, arbitrators receive rewards and processes terminate. For approved reversals, the function updates issuer and receiver balances at the smart contract level without removing original transactions from the network. The `Bonsai` contract updates balances and emits events containing sender/destination addresses, amounts, reverted transaction IDs, and associated trial IDs.

The selected arbitrators will share the trial fee (further explained in Section III-E) equally among themselves. The current implementation of Bonsai does not include a mechanism to penalize arbitrators who exhibit faulty behavior, as this was considered outside the initial scope of this work. Later implementations of Bonsai can introduce penalties for such actions, such as withholding rewards and maintaining a block list of offending users. Blocked users would be excluded from serving as arbitrators in subsequent reversals, with the possibility of removal from the list either through an appeal process or after a specified period of time.

In Figure 4, we detail in a sequence diagram the expected trial process for an accepted reversal, from the transaction's sender perspective. Only one `Arbitrator` entity is represented as it was redundant to demonstrate the actions of the others as they are similar, except for the `finishTrial`, which is only invoked by one arbitrator.

C. Events

To inform trial participants of phase changes, the `Trials` contract emits events for evidence submission, vote commitment quorum, vote revelation, and trial finalization. The `Bonsai`, `Exchange`, and `Insurance` contracts emit events for token transfers, reversals, compensation payouts, and minting/burning of `BON` tokens. Future work could include a decentralized web application with an on-chain oracle to monitor these events in real-time, providing automated notifications and enhancing transparency.

D. Cool-down period

The cool-down period defines transaction reversal eligibility and must balance security against finalization speed. Networks prioritizing fast operations may choose shorter periods, like 1 hour, for quick completion. However, this weakens protection for users who rarely monitor accounts. Conversely, networks protecting less cautious or technically skilled users may implement longer periods, like 1 day, providing time to identify and respond to unauthorized transactions, though it reduces speed and attractiveness to users prioritizing immediacy.

Examining Ethereum's daily transaction patterns [7], we observe 2024 volumes generally fluctuating between 1,000,000 and 1,300,000 transactions with occasional spikes, peaking on January 14th before stabilizing within this range. Most transactions involve amounts less than 1 `ETH` [8]. Considering these factors, we set the cool-down period to four days, though this somewhat arbitrary choice can be adjusted before contract deployment.

Once a reversal request is made, the cool-down period continues uninterrupted until full duration elapses to prevent users from delaying transaction finalization through successive reversal requests. Users can make multiple reversal requests during the cool-down period, but the system stores only one request at a time, requiring the previous request's finalization before submitting a new one. Additionally, each request incurs proportionally increasing fees to prevent network spam.

E. Bonsai Insurance

Bonsai integrates a built-in insurance mechanism to protect users against transaction reversals. Every transaction includes a 5% insurance fee paid to a decentralized address. This fee covers the cost of the first reversal request; further requests require the user to pay, with the cost scaling linearly each time. Inspired by systems like Vinted [9], this insurance introduces a **Hard Cap** (maximum payout per claim) and **Reserve Requirements** (minimum balance retained by the insurance).

User compensation is reputation-based, calculated through a *plafond* score, which determines the maximum refundable amount. Users are eligible for compensation whenever they are negatively impacted by the system. Specifically, the compensation is paid in cases where a transaction reversal affects users who were not directly involved in the original transaction (i.e., neither the source nor the destination addresses) The plafond is computed as represented in Equation 1:

$$plafond = \frac{\min(membership,\ 10) \times \min(\sum payments_made \times diff_addresses,\ 100{,}000)}{\min(\#ReversalRequestsMadeLastYear,\ 0) + 1} \tag{1}$$

where:

- `membership`: years since the user's first payment (capped at 10, approximately Ethereum's age),
- `payments_made`: total insurance contributions,
- `diff_addresses`: number of unique addresses the user transacted with,
- `ReversalRequestsMadeLastYear`: reversal requests in the past year.

All insurance parameters (Fee, Hard Cap, and Reserve Requirements) can be configurable before deployment.

F. Freezing Assets

Reversing a transaction is simple when the recipient address holds enough balance. However, since Bonsai allows immediate transfers upon transaction confirmation, tokens are not frozen, allowing recipients to instantly spend them, potentially reducing their balance to zero or below the received amount.

We considered freezing assets during a cool-down period by tracking pending transactions in the smart contract. This would have allowed quick reversals by canceling unconfirmed

[7] Ethereum's transactions per day graph (https://etherscan.io/chart/tx)

[8] Etherscan transaction's list (https://etherscan.io/txs)

[9] Vinted is a peer-to-peer online marketplace where people can buy, sell, or swap second-hand fashion items. https://www.vinted.com/about

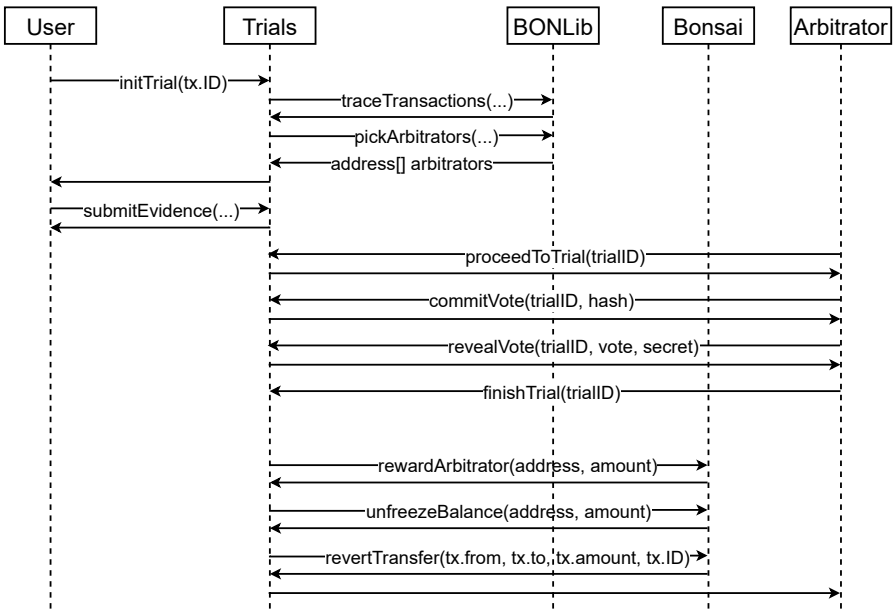

Fig. 4. Example of a successful reversal

transfers. However, requiring all transactions to wait for confirmation would introduce delays, degrading user experience and discouraging adoption, especially since we expect most transactions wouldn't need to be reverted.

Instead, Bonsai only freezes assets during trials to prevent further spending of tokens identified for reversal. This ensures the reversal process remains accurate and complete. We delve deeper into this mechanism in the following section.

G. Recursive Reversals

Bonsai may approve transaction reversals even when recipient addresses lack sufficient funds due to money laundering, scams splitting funds across addresses, or token spending. To recover maximum value, Bonsai employs a best-effort depth-first search (DFS) algorithm that identifies transaction chains likely moving funds, tracing from original recipients through newer transactions. All identified transactions become eligible for reversal and are processed from most recent to oldest, ensuring requester compensation while excluding all sender and receiver addresses from arbitrator eligibility to avoid conflicts of interest.

If destination addresses have sufficient balance, the required amount is frozen and returned; otherwise, the algorithm freezes maximum possible amounts, updates remaining deficits, and recursively searches newer transactions from those addresses until missing amounts are recovered or no paths remain. The algorithm then backtracks, updating missing amounts at each level. Algorithm 1 presents the pseudo-code: lines 3-12 lock maximum balance in destination addresses limited by missing amounts, returning if amounts match missing tokens; lines 15-30 use DFS-based recursive exploration of all transactions from destination addresses with higher IDs to locate missing tokens.

Since token burning and unstable token approval are prevented, tokens remain within the network in user balances, and transaction chains leading to current states comprise reversible transactions more recent than the original under trial. All users who are destination parties of identified reversal transactions (except the original transaction's destination) receive compensation for their losses, since they are not directly involved in the reversal. By recognizing that most transactions correspond to off-chain exchanges (tokens, goods, fiat currency, services), we believe compensating these indirectly affected users makes Bonsai more appealing. The Insurance compensates these users based on their system contributions using Equation 2.

$$\textit{compensation} = \min(tx.amount, plafond, HC, BI.balance - ResReq) \quad (2)$$

where: `tx.amount` refers to the reverted transaction amount; `plafond` is calculated with the formula on Equation 1; `HC` is the insurance's Hard Cap; `BI.balance-ResReq` is the maximum amount the Insurance can pay without having its balance below the Reserve Requirement limit.

IV. Evaluation

To evaluate Bonsai, we conducted unit and integration testing on Ethereum Sepolia (Layer 1) and ZKsync Sepolia (Layer 2) to compare performance. We chose ZKsync, a Zero Knowledge Rollup, for its enhanced security and instant verification, with batching mechanisms offsetting higher computation costs. Our choice was further reinforced by using Foundry [10] for development and testing, as it offers a specialized fork for the ZKsync environment.

We measured trial duration, reversal costs, and storage overhead across recursive reversals with chain lengths of

[10]Foundry's Book: https://book.getfoundry.sh/

Algorithm 1 traceTransactions Algorithm

```
function TRACETRANSACTIONS(txID, missing, txsToRevert, freezed)
    tx ← getTxByID(txID)
    if balanceOf(tx.dest) − frozenBalanceOf(tx.dest) ≥ missing then
        toFreeze ← missing
    else
        toFreeze ← balanceOf(tx.dest)
    end if
    txsToRevert.push(txID)
    freezed.push(toFreeze)
    if toFreeze = missing then
        return (txsToRevert, freezed)
    end if
    missing ← missing − toFreeze
    issuedTxs ← getIssuedTxs(tx.dest)
    for all t ∈ issuedTxs do
        newTx ← getTxByID(t)
        if newTx.id < txID then
            continue
        end if
        (txsToRevert, freezed) ← TRACETRANSACTIONS(newTx.ID, missing, txsToRevert, freezed)
        if txData.amount ≥ missing then
            missing ← 0
        else
            missing ← missing − txData.amount
        end if
        if missing = 0 then
            break
        end if
    end for
    return (txsToRevert, freezed)
end function
```

1 to 5 transactions, using seven user and four arbitrator accounts ($f = 1$). We ran 60 experiments per chain length on Layer 2 (meeting the Central Limit Theorem's sample size requirement) and 10 on Layer 1 due to higher costs. Unit tests used Foundry's Forge; end-to-end simulations used automated Bash scripts (available in the repository).

A. Source Code

To evaluate Bonsai's storage overhead, we measured *Deployed Size* (bytecode in bytes), *Gas* (deployment consumption), *LoC* (formatted source code lines), and *Deployment Cost* (estimated USD cost on Ethereum and ZKsync mainnets). Costs were estimated using average gas prices (1.62 gwei for Ethereum, 0.046 gwei for ZKsync) and an Ether price of $4,347.69 from August 2025 data [11]

[11] All historical data was gathered from Etherscan (https://etherscan.io/chart/gasprice) and ZKsync Blockscout (https://zksync.blockscout.com/stats)

Table I shows metrics for each contract, with `BONLib` providing data structures for `Trials`. All contracts achieved near 100% test coverage via Foundry and were analyzed by Slither[12] for vulnerabilities.

TABLE I
SOURCE CODE METRICS

Contract	Deployed Size	Gas	LoC	Ethereum ($)	ZKsync ($)
Bonsai	20870	4386889	355	$30.88	$0.87
Trials	21917	4700802	377	$33.06	$0.94
BONLib	8861	1848415	205	$13.00	$0.37
Insurance	5260	1098205	123	$7.74	$0.22
Exchange	2423	517301	40	$3.64	$0.10
Total	**59331**	**12551612**	**1100**	**$88.32**	**$2.50**

Deploying Bonsai on ZKsync is approximately 97% cheaper than Ethereum due to Layer 2's lower gas prices. However, Ethereum deployment remains viable as a one-time expense, with users controlling costs by adjusting maximum gas prices.

B. Time Consumption

In EVM systems, our latency metric reflects wall-clock time encompassing transaction broadcasting, validator inclusion, block propagation, and RPC response, with main variability arising from validator inclusion and network propagation due to testnet volatility and inconsistent block production.

Figure 5 shows average trial breakdown time across phases by chain length. The `initTrial` phase shows expected upward trends as longer chains need more effort to locate funds, with `finalizeTrial` unexpectedly decreases slightly between lengths 3 and 4 despite affecting more transactions. On `submitSenderEvidence` similar anomalies appear, likely attributed to network congestion or uncontrolled latency, though differences remain negligible to end users.

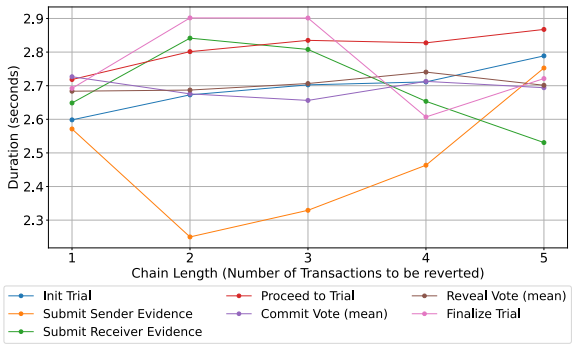

Fig. 5. Trial phase duration by chain length in ZKsync Sepolia

Figure 6 shows the mean total trial duration on the ZKsync Sepolia network, along with a 95% confidence interval. As observed, the total trial duration generally increases with chain length, reflecting the added complexity of reverting more transactions. However, with a chain length of 4 transactions, there is an unexpected decrease compared to length 3, which, consistent with earlier observations, can likely be attributed to network congestion during test execution or some uncontrolled latency between on-chain confirmation and the client response.

[12] Slither GitHub repository: https://github.com/crytic/slither

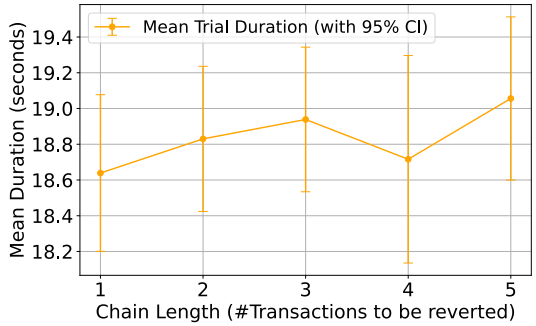

Fig. 6. Mean trial duration by chain length with 95% Confidence Interval in ZKsync Sepolia

Figure 7 shows mean trial phase durations by chain length on Ethereum Sepolia. Unlike ZKsync results, no clear relationship exists between chain length and `Init Trial` or `Finalize Trial` durations, and other phases show unexpected variability. These inconsistencies likely stem from network congestion and uncontrolled latency on the widely-used Sepolia testnet. However, maximum variation is around 1.5 seconds, likely negligible for end-users.

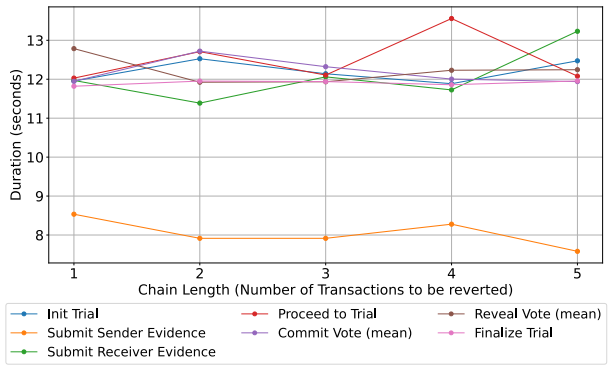

Fig. 7. Trial phase duration by chain length in Ethereum Sepolia

Figure 8 compares average phase durations for Bonsai execution between Ethereum Sepolia and ZKsync Sepolia across various chain lengths, demonstrating that ZKsync achieves approximately 4 times faster performance. This substantial improvement confirms that deploying Bonsai on a Layer 2 network significantly accelerates trial processing and reversal times, enabling more responsive smart contract interactions and faster resolution of user disputes, while also providing notable gas cost reductions.

C. Gas Consumption

Ethereum transactions require **gas**, a unit measuring computational effort on the blockchain. Each operation consumes a fixed amount of gas [13], while the *gas price* (ether per gas unit) varies with network conditions.

Figure 9 shows the gas cost per trial phase by chain length. As expected, the gas cost for both the `Init Trial` and `Finalize Trial` phases increases linearly with chain length, due to the need to locate and revert more transactions. The other phases are not dependent on the number of transactions to revert and therefore their gas cost is constant between all chain lengths, as anticipated.

[13]EVM opcodes list: https://github.com/crytic/evm-opcodes

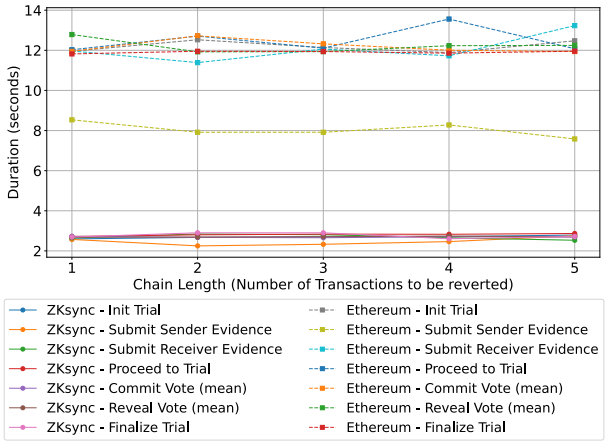

Fig. 8. Trial phase duration - ZKsync Sepolia vs Ethereum Sepolia

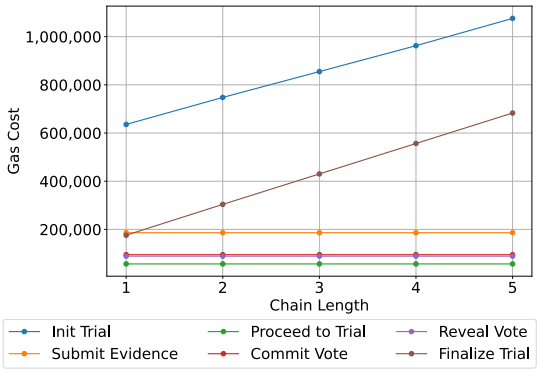

Fig. 9. Gas cost per trial phase by chain length

The formula to calculate phase cost in USD ($) is:

$$\text{PhaseCost}_{\text{USD}} = \text{gasUsed} \times \text{gasPrice} \times \text{EtherPriceUSD}$$

With the current implementation using four-arbitrator committees (three voting), two evidence submission phases, and a five-transaction reversal chain, a full trial costs approximately $0.30 on ZKsync compared to nearly $24 on Ethereum, with costs distributed among participants based on their active engagement. These estimates, using previously mentioned average gas and Ether prices, demonstrate ZKsync's significant economic advantage over Ethereum.

D. Comparison with Previous Work

Compared to Recoverable Token's 112 to 126 seconds on Ropsten testnet, Bonsai completes trials in roughly 80 seconds on average despite supporting recursive reversals, on-chain arbitrator selection, and commit-reveal voting. Sepolia's faster, more predictable block production than Ropsten partially explains this improvement.

While Recoverable Token consumed 820,591 to 833,656 gas for single reversals, Bonsai uses 1,238,780 gas for single-transaction chains and 2,186,187 for five-transaction chains. These higher gas costs are justified by enhanced functionality including recursive reversals and secure voting.

V. Related Work

Intrusion recovery mitigates attacks while preserving normal operations. In blockchain systems, recovery is complicated by ledger immutability. Although forks can undo harmful transactions, they may compromise chain continuity. Alternative approaches aim to bypass or selectively remove immutability.

Intrusion recovery systems improve traditional backups through selective re-execution [12], compensating transactions [11], and multi-versioning [13], [14] to reverse intrusions while preserving legitimate operations. While new cryptographic algorithms have emerged, theoretical and technical implementations of mutable systems maintaining blockchain security remain in early stages [17], [18], using strategies to either conditionally remove immutability or bypass it entirely.

Removal strategies include consensus-based approaches like hard forks, cryptographic modifications to hash functions [27], meta-transactions extending blockchain history [19], [28], and offline wallets [29]. While these address undesirable operations, they require significant computational power [30] and fundamental architectural changes. Therefore, we adopt an immutability bypass approach using compensating transactions to revert malicious or erroneous actions.

Bypass techniques manage blockchain data using decentralized judges or new structures to ensure GDPR and Right to be Forgotten compliance [31], [32], including off-chain storage [33], in-chain encryption [34], pruning [35], [36], and reversible tokens [20], [21].

Recoverable Token [20] enables transaction reversals through smart contracts using arbitrator consensus. Disputing parties submit claims, and approved reversals are executed. However, if tokens cannot be traced, recovery fails, and private arbitrator votes are not mentioned, potentially allowing influence between arbitrators.

ERC-20R and ERC-721R [21] introduce reversible tokens with limited reversion windows after confirmation. Issuers can request freezes with evidence to decentralized judges who decide on trial phases. The design introduces complexity by requiring both reversible and "normal" tokens, with conversion between them. Limitations include unclear token burning support (attackers could steal and burn tokens), unspecified token allowance behavior (enabling liquidity pool exploitation), unclear arbitration committee selection without conflict-of-interest exclusion, and uncertain private vote implementation. Additionally, when reversals occur, secondary users who legitimately acquired tokens through exchanges lose both tokens and assets, creating unfair outcomes.

Bonsai can trace and reverse transactions through 5 wallets in under 20 seconds on ZKsync and roughly 80 seconds on Ethereum Sepolia. In comparison, Recoverable Token only reverts single-transaction chains in 120 seconds on Ropsten testnet, while ERC-20R lacks experimental evaluation.

Table II presents how Bonsai differs from these approaches, highlighting the most important aspects of Bonsai's design.

TABLE II
Comparison between Bonsai and other approaches

Characteristic	Bonsai	RecToken	ERC-20R
Prevents unstable tokens' burning	Yes	No	Maybe
Keeps arbitrators' votes private	Yes	No	Maybe
Compensates victims of recursive reversals	Yes	No	No
Two different balances	No	No	Yes
Arbitrators' committees decentralized	Yes	No	No
Protects against liquidity pools	Yes	No	No

VI. Conclusion

This work proposed Bonsai, a system tailored to revert ERC-20 transactions on an EVM-based blockchain within a specified cool-down period using arbitration trials and an insurance feature for added security. We evaluated the system with two deployments, on two testnets, a Layer 1 and a Layer 2, and ran various scenarios to locate spent funds. Our results show that deploying Bonsai on a Layer 2 network offers advantages over previous work in both cost and time, thanks to lower gas fees and faster transaction confirmations.

This work shows how intrusion recovery on blockchains can introduce user opt-in mechanisms that allow immutability to be adjusted to meet application and legal requirements, while maintaining the core integrity guarantees of the technology.

While Bonsai demonstrates strong results, several limitations present opportunities for future work: fully integrating Chainlink VRF for verifiable randomness in arbitrator selection; implementing arbitrator pools for collective token staking; developing a notification system using on-chain event listeners for real-time updates; encrypting evidence submissions for enhanced privacy; establishing watch lists and block lists with appeal processes for managing malicious users and faulty arbitrators; automating transaction finalization through Chainlink Automation Job Scheduler [14] funded by insurance contract balances; and potentially creating a dedicated Layer 2 blockchain using rollups that maintains mutable transactions on Layer 2 while committing only finalized transactions to Layer 1 for optimized throughput and secure finality.

References

[1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," Whitepaper, 2008, self-published. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[2] M. M. d. Queiroz, R. Telles, and S. H. Bonilla, "Blockchain and supply chain management integration: a systematic review of the literature," *Supply Chain Management: An International Journal*, vol. 24, no. 6, pp. 757–794, 2019. [Online]. Available: https://www.emerald.com/insight/content/doi/10.1108/SCM-03-2018-0143/full/html

[3] D. R. Matos, M. L. Pardal, P. Adão, A. R. Silva, and M. Correia, "Securing Electronic Health Records in the Cloud," in *Proceedings of the 1st Workshop on Privacy by Design in Distributed Systems*. Porto Portugal: ACM, Apr. 2018, pp. 1–6. [Online]. Available: https://dl.acm.org/doi/10.1145/3195258.3195259

[14] https://blog.chain.link/chainlink-automation-job-scheduler/

[4] M. Raikwar, S. Mazumdar, S. Ruj, S. Sen Gupta, A. Chattopadhyay, and K.-Y. Lam, "A blockchain framework for insurance processes," in *2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, 2018, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8328731

[5] S. C. Goldstein, D. Goktas, M. Conn, S. P. T. Pitchuka, M. Sameer, M. Shah, C. Swett, H. Tu, S. Viswanathan, and J. Xiao, "BoLT: Building on Local Trust to Solve Lending Market Failure," 2020, unpublished draft. [Online]. Available: https://www.cs.cmu.edu/~seth/bolt/bolt-draft.pdf

[6] E. Tan, S. Mahula, and J. Crompvoets, "Blockchain governance in the public sector: A conceptual framework for public management," *Government Information Quarterly*, vol. 39, no. 1, p. 101625, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0740624X21000617

[7] B. Biswas and R. Gupta, "Analysis of barriers to implement blockchain in industry and service sectors," *Computers & Industrial Engineering*, vol. 136, pp. 225–241, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360835219303961

[8] Z. Pratap, "Reentrancy attacks and the dao hack explained," Chainlink Blog, 8 2022, blog post. [Online]. Available: https://blog.chain.link/reentrancy-attacks-and-the-dao-hack/

[9] R. Chandra, T. Kim, M. Shah, N. Narula, and N. Zeldovich, "Intrusion recovery for database-backed web applications," in *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*. Cascais Portugal: ACM, Oct. 2011, pp. 101–114. [Online]. Available: https://dl.acm.org/doi/10.1145/2043556.2043567

[10] D. R. Matos, M. L. Pardal, A. R. Silva, and M. Correia, "μVerum: Intrusion Recovery for Microservice Applications," *IEEE Access*, vol. 11, pp. 78 457–78 470, 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10190596/

[11] D. Nascimento and M. Correia, "Shuttle: Intrusion Recovery for PaaS," in *2015 IEEE 35th International Conference on Distributed Computing Systems*. Columbus, OH, USA: IEEE, Jun. 2015, pp. 653–663. [Online]. Available: http://ieeexplore.ieee.org/document/7164950/

[12] D. Matos and M. Correia, "NoSQL Undo: Recovering NoSQL databases by undoing operations," in *2016 IEEE 15th International Symposium on Network Computing and Applications (NCA)*. Cambridge, Boston, MA, USA: IEEE, Oct. 2016, pp. 191–198. [Online]. Available: http://ieeexplore.ieee.org/document/7778616/

[13] A. Goel, K. Po, K. Farhadi, Z. Li, and E. de Lara, "The taser intrusion recovery system," in *Proceedings of the Twentieth ACM Symposium on Operating Systems Principles*, ser. SOSP '05. New York, NY, USA: Association for Computing Machinery, 2005, p. 163–176. [Online]. Available: https://doi.org/10.1145/1095810.1095826

[14] D. R. Matos, M. L. Pardal, G. Carle, and M. Correia, "RockFS: Cloud-backed File System Resilience to Client-Side Attacks," in *Proceedings of the 19th International Middleware Conference*, ser. Middleware '18. New York, NY, USA: Association for Computing Machinery, Nov. 2018, pp. 107–119. [Online]. Available: https://doi.org/10.1145/3274808.3274817

[15] S. T. King and P. M. Chen, "Backtracking intrusions," in *Proceedings of the nineteenth ACM symposium on Operating systems principles*, ser. SOSP '03. New York, NY, USA: Association for Computing Machinery, Oct. 2003, pp. 223–236. [Online]. Available: https://dl.acm.org/doi/10.1145/945445.945467

[16] D. A. S. D. Oliveira, J. R. Crandall, G. Wassermann, S. Ye, S. F. Wu, Z. Su, and F. T. Chong, "Bezoar: Automated virtual machine-based full-system recovery from control-flow hijacking attacks," in *NOMS 2008 - 2008 IEEE Network Operations and Management Symposium*. Salvador, Bahia, Brazil: IEEE, 2008, pp. 121–128. [Online]. Available: http://ieeexplore.ieee.org/document/4575125/

[17] E. Politou, F. Casino, E. Alepis, and C. Patsakis, "Blockchain Mutability: Challenges and Proposed Solutions," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 4, pp. 1972–1986, Oct. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/8883080/

[18] D. Zhang, J. Le, X. Lei, T. Xiang, and X. Liao, "Exploring the redaction mechanisms of mutable blockchains: A comprehensive survey," *International Journal of Intelligent Systems*, vol. 36, no. 9, pp. 5051–5084, Sep. 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/int.22502

[19] F. Faria, S. Eisa, D. R. Matos, and M. L. Pardal, "EvoChain: a Recovery Approach for Permissioned Blockchain Applications," Nov. 2024, arXiv:2411.16976 [cs]. [Online]. Available: https://arxiv.org/abs/2411.16976

[20] F. F. Martins, D. R. Matos, M. L. Pardal, and M. Correia, "Recoverable Token: Recovering from Intrusions against Digital Assets in Ethereum," in *2020 IEEE 19th International Symposium on Network Computing and Applications (NCA)*. Cambridge, MA, USA: IEEE, Nov. 2020, pp. 1–9. [Online]. Available: https://ieeexplore.ieee.org/document/9306738/

[21] K. Wang, Q. Wang, and D. Boneh, "ERC-20R and ERC-721R: Reversible Transactions on Ethereum," Oct. 2022, arXiv:2208.00543 [cs]. [Online]. Available: http://arxiv.org/abs/2208.00543

[22] L. Yin, J. Xu, and Q. Tang, "Sidechains with fast cross-chain transfers," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 6, pp. 3925–3940, 2021.

[23] L. T. Thibault, T. Sarry, and A. S. Hafid, "Blockchain scaling using rollups: A comprehensive survey," *IEEE Access*, vol. 10, pp. 93 039–93 054, 2022.

[24] R. Neiheiser, G. Inácio, L. Rech, C. Montez, M. Matos, and L. Rodrigues, "Practical Limitations of Ethereum's Layer-2," *IEEE Access*, vol. 11, pp. 8651–8662, 2023, conference Name: IEEE Access. [Online]. Available: https://ieeexplore.ieee.org/document/10018958/?arnumber=10018958

[25] F. Béres, I. A. Seres, A. A. Benczúr, and M. Quintyne-Collins, "Blockchain is watching you: Profiling and deanonymizing ethereum users," in *2021 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*, Cambridge, MA, USA, 8 2021, pp. 69–78.

[26] Z. Wang, S. Chaliasos, K. Qin, L. Zhou, L. Gao, P. Berrang, B. Livshits, and A. Gervais, "On how zero-knowledge proof blockchain mixers improve, and worsen user privacy," in *Proceedings of the ACM Web Conference 2023*, ser. WWW '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 2022–2032. [Online]. Available: https://doi.org/10.1145/3543507.3583217

[27] G. Ateniese, B. Magri, D. Venturi, and E. Andrade, "Redactable Blockchain – or – Rewriting History in Bitcoin and Friends," in *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*. Paris: IEEE, Apr. 2017, pp. 111–126. [Online]. Available: https://ieeexplore.ieee.org/document/7961975/

[28] I. Puddu, A. Dmitrienko, and S. Capkun, "μchain: How to forget without hard forks," Cryptology ePrint Archive, Paper 2017/106, 2017. [Online]. Available: https://eprint.iacr.org/2017/106

[29] O. N. Challa, "Reversecoin," 2014. [Online]. Available: https://github.com/obulpathi/reversecoin

[30] C. Jentzsch, "Decentralized autonomous organization to automate governance," Slock.it, White Paper, 2016, slock.it White Paper. [Online]. Available: https://lawofthelevel.lexblogplatformthree.com/wp-content/uploads/sites/187/2017/07/WhitePaper-1.pdf

[31] European Union, "Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)," https://eur-lex.europa.eu/eli/reg/2016/679/oj, 2016, official Journal of the European Union, L 119, 4 May 2016, pp. 1–88.

[32] GDPR.eu, "Everything you need to know about the "Right to be forgotten"," Website, Nov. 2018, section: GDPR Overview. [Online]. Available: https://gdpr.eu/right-to-be-forgotten/

[33] J. Eberhardt and S. Tai, "On or Off Blockchain? Insights on Off-Chaining Computation and Data," in *Service-Oriented and Cloud Computing*, F. De Paoli, S. Schulte, and E. Broch Johnsen, Eds. Cham: Springer International Publishing, 2017, pp. 3–15.

[34] Y. Xiao, N. Zhang, J. Li, W. Lou, and Y. T. Hou, "Privacyguard: Enforcing private data usage control with blockchain and attested off-chain contract execution," in *Computer Security–ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part II 25*. Springer, 2020, pp. 610–629.

[35] J. D. Bruce, "The mini-blockchain scheme," Cryptonite, Technical Report, 2014, revision 3. [Online]. Available: http://cryptonite.info/files/mbc-scheme-rev3.pdf

[36] C. K. Pyoung and S. J. Baek, "Blockchain of Finite-Lifetime Blocks With Applications to Edge-Based IoT," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 2102–2116, Mar. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/8932408/