Assure DeFi®

THE VERIFICATION **GOLD STANDARD**

Security Assessment

RollBackWallet

Date: 16/08/2025

Audit Status: PASS

Audit Edition: Advanced+

ASSURE DEFI®

THE VERIFICATION **GOLD STANDARD**

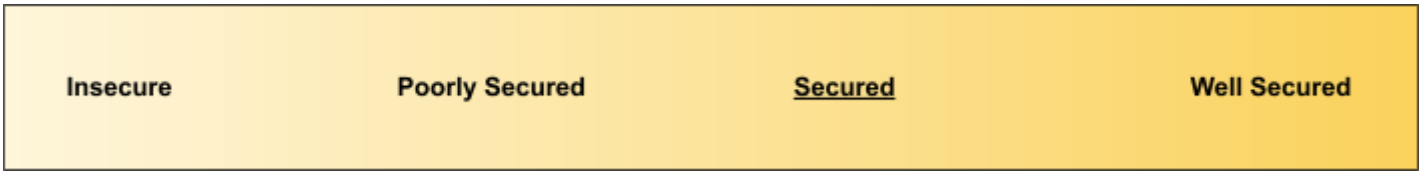
Risk Analysis

Vulnerability summary

Classification	Description
High	High-level vulnerabilities can result in the loss of assets or manipulation of data.
Medium	Medium-level vulnerabilities can be challenging to exploit, but they still have a considerable impact on smart contract execution, such as allowing public access to critical functions.
Low	Low-level vulnerabilities are primarily associated with outdated or unused code snippets that generally do not significantly impact execution, sometimes they can be ignored.
Informational	Informational vulnerabilities, code style violations, and informational statements do not affect smart contract execution and can typically be disregarded.

Executive Summary

According to the Assure assessment, the Customer's smart contract is **Secured.**



Scope

Target Code And Revision

For this audit, we performed research, investigation, and review of the RollBackWallet contracts followed by issue reporting, along with mitigation and remediation instructions outlined in this report.

Target Code And Revision

Project	Assure
Language	Solidity
Codebase	https://github.com/Rollback-Labs/Rollback_Contract/ 4e1825055e4dfdf1afea3d4f2832c71656392489 Fixed version: 918f10d15efd081afeeedc0d4089872ae034c63f
Audit Methodology	Static, Manual

Attacks made to the contract

In order to check for the security of the contract, we tested several attacks in order to make sure that the contract is secure and follows best practices.

Category	Item
Code review & Functional Review	• Compiler warnings. • Race conditions and Reentrancy. Cross-function race conditions. • Possible delays in data delivery. • Oracle calls. • Front running. • Timestamp dependence. • Integer Overflow and Underflow. • DoS with Revert. • DoS with block gas limit. • Methods execution permissions. • Economy model. • Private user data leaks. • Malicious Event log. • Scoping and Declarations. • Uninitialized storage pointers. • Arithmetic accuracy. • Design Logic. • Cross-function race conditions. • Safe Zeppelin module. • Fallback function security. • Overpowered functions / Owner privileges

AUDIT OVERVIEW

1. Emergency Recovery executes with zero recovery address [FIXED ✅]

Location: RollbackWallet.sol - requestVote(EMERGENCY_RECOVERY), _executeVote, transferFunds, Vote struct

Issue: requestVote never sets Vote.recoveryWallet for EMERGENCY_RECOVERY. At execution, _executeVote calls transferFunds(target, recoveryWallet) where recoveryWallet is the zero address. Standard ERC20s revert (permanent DoS of recovery); non-standard ERC20s may transfer to address(0) (token burn). Events signal success even if all transfers failed.

Recommendation: Extend requestVote to accept and store a non-zero recoveryWallet and validate it's active and not the target.

In _executeVote, verify transferFunds had at least one successful transfer before emitting 'Executed', otherwise revert.

Consider timelock + cancellation and exclude recoveryWallet from voting if it's an owner.

Fix: The recovery address is explicitly supplied, validated, stored, and excluded from approvals.

Execution reverts unless at least one token transfer succeeds, so 'false-positive' success events are avoided.

2. Obsolete owners retain admin privileges [FIXED ✅]

Location: RollbackWallet.sol: pause, unpause, setTreasury, updateMonitoredTokens, updateRandomization, updateFallbackWallet, updateWalletPriority (and any onlyOwner admin)

Issue: onlyOwner checks isOwner[msg.sender] but ignores walletOwners[msg.sender].isObsolete. Obsolete owners can still pause/unpause, change treasury, mutate monitored tokens, etc., enabling configuration capture or DoS.

Recommendation: Introduce onlyActiveOwner (owner && not obsolete) and use it for all administrative functions.

Prefer governance (vote) for sensitive config changes.

Fix: Obsolete owners can neither administer nor vote.

3. ERC721 reentrancy can shrink token arrays mid-loop [out-of-bounds revert/DoS] [FIXED ✅]

Location: RollbackWallet.sol: transferFunds, _transferNFTWithPartialApproval, updateMonitoredTokens (reentrancy target)

Issue: safeTransferFrom triggers onERC721Received on the recipient. A malicious recipient can call updateMonitoredTokens and shrink monitoredTokens while transferFunds is iterating by index, causing array

OOB and reverting future rollbacks/executions.

Recommendation: Snapshot monitoredTokens and monitoredTokenTypes to memory before iteration.

Add a mutation lock or reuse nonReentrant on callers of transferFunds.

Prefer a two-phase approach such as: collect items, then perform transfers.

Fix: transferFunds now snapshots monitoredTokens and monitoredTokenTypes into memory and caches tokenCount before iterating, so storage mutations during onERC721Received can't affect the loop. All top-level callers that lead to transfers (performRollback, confirmVote -> _executeVote, and requestEmergencyRecovery) are guarded with nonReentrant, preventing reentrant admin calls mid-execution.

4. Emergency Recovery requires only 2 approvals [FIXED ✅]

Location: RollbackWallet.sol - requestVote

Issue: For recovery votes, approvalsNeeded is set to 2 whenever at least two non-target voters exist, regardless of the total number of active wallets, enabling low-collusion fund movement (given approvals).

Recommendation: Use percentage-/supermajority-based quorum (for example ≥⅔ or ≥N-1 of actives), and/or scale with set size.

Fix: requestEmergencyRecovery now sets approvalsNeeded to all active owners minus the target and the recovery wallet (if active and distinct). Execution only occurs when approvalsReceived >= approvalsNeeded in confirmVote. No hardcoded '2' now it scales with the active set (unanimous among the remaining owners).

5. performRollback() irreversible obsolescence on failed transfers enables permanent DoS [FIXED ✅]

Location: RollbackWallet.sol - performRollback()

Issue: The function obsoletes the target wallet prior to transferring assets. When approvals/allowances are absent, transfers fail yet the obsolete flag persists, preventing any future rollback attempts and stranding assets.

Recommendation: Only set isObsolete = true after verifying transfers succeeded (all or at least one, per policy).

If no transferable assets (no approvals/allowances), revert and do not obsolete.

Optionally pre-check approvals/allowances and/or require transferFunds() to return success and enforce it with a require.

Fix: Obsolete flag is set only if transferFunds() succeeds inside _handleNonRandomizedRollback / _handleRandomizedRollback. On zero success, it emits a failure event and does not obsolete, allowing retries.

6. Treasury DoS via setTreasury [FIXED ✅]

Issue: Any single active owner can call setTreasury() and point it to a contract that reverts on receive. Because fee forwarding in initialize() and requestEmergencyRecovery() require(success), this bricks wallet creation/emergency recovery for everyone.

Recommendation: Gate setTreasury by governance (vote) or manager-only, alternatively switch to a pull pattern (no revert on forward and fees stored and later withdrawn).

Fix: setTreasury in the wallet is now onlyManager. Fee forwarding in initialize and requestEmergencyRecovery is non-reverting and failures accrue to pendingFees with a manager withdraw function.

7. Unilateral routing of rollbacks [FIXED ✅]

Issue: Any active owner can flip updateRandomization() and set priorities via updateWalletPriority() for any wallet. In non-random mode, getPriorityWalletNext() then routes assets to the lowest-priority wallet trivially steerable to the attacker.

Recommendation: Move these to governance and at minimum restrict owners to changing their own priority and require votes to change global policy (randomization flag).

Fix: updateRandomization is manager-only and updateWalletPriority is self-only (caller must be the same wallet, non-obsolete, and only in non-random mode). getPriorityWalletNext actually uses walletPriorities.

8. Admin override on creation (50% force-execute) [FIXED ✅]

Issue: forceExecuteWalletCreation() allows the manager owner to execute with only ≥50% signatures. If your requirements expect full consent, this is a high-risk centralization backdoor.

Recommendation: Remove force-execute or raise to the same threshold as standard creation (or require a governance vote / multisig on the manager).

Fix: forceExecuteWalletCreation is removed. finalizeWalletCreation requires ALL signatures and enforces the INITIALISATION_FEE and executeWalletCreation also checks full signature count.

1. Force execution bypasses initialization fee [FIXED ✅]

Location: RollbackWalletManager.sol : forceExecuteWalletCreation, executeWalletCreation, finalizeWalletCreation : RollbackWallet.sol initialize

Issue: forceExecuteWalletCreation calls executeWalletCreation(requestId, 0) after only 50% signatures;, Wallet.initialize doesn't enforce a fee. Treasury can be skipped.

Recommendation: Enforce INITIALISATION_FEE in forceExecuteWalletCreation or require feeAmount >= INITIALISATION_FEE inside executeWalletCreation and/or Wallet.initialize.

Fix: forceExecuteWalletCreation now enforces msg.value >= INITIALISATION_FEE and forwards it to executeWalletCreation.

2. Warning math underflow and incorrect remaining time [FIXED ✅]

Location: RollbackWallet.sol - checkAndSendWarning

Issue: Computes warningTime = inactivityThreshold - 30 days which underflows if threshold < 30 days. Emitted remainingTime = inactivityThreshold - timeSinceActivity underflows once past threshold.

Recommendation: Guard: if inactivityThreshold <= 30 days, skip warning logic.

Use saturating math for remainingTime (remaining = timeSince >= threshold ? 0 : threshold - timeSince).

Fix: Guard when threshold <= 30 days and use saturating math for remaining.

3. Manager-initiated agent change is inoperable (sender not an owner) [FIXED ✅]

Location: RollbackWalletManager.sol - requestAgentChange and RollbackWallet.sol - requestVote (onlyOwner)

Issue: Manager calls wallet.requestVote, but inside the wallet onlyOwner checks msg.sender (the manager), causing NotAuthorized. Function cannot be used as intended.

Recommendation: Call requestVote directly from owners, or add a dedicated manager role in the wallet, or adopt a trusted-forwarder / meta-tx pattern to preserve the original sender.

Fix: Manager calls wallet's managerRequestAgentChange (gated by onlyManager) instead of owner-only path.

4.Any active owner can reset any wallet's activity [DOS] [FIXED ✅]

Location: RollbackWallet.sol: resetActivity, activeOwnerOrAgent modifier

Issue: resetActivity allows any active owner (not just the wallet's owner) to reset another owner's lastActivity, keeping them perpetually ineligible for rollback.

Recommendation: Restrict resetActivity to msg.sender == walletAddress or msg.sender == agentWallet, or gate via vote/consent.

Fix: resetActivity now only allows self or agent to reset (explicit check on msg.sender).

5. 'Randomized' selection is fully agent-controlled (not random) [PARTIALLY FIXED ✅]

Location: RollbackWallet.sol: performRollback, _handleRandomizedRollback, _selectRandomWallet

Issue: Agent supplies randomSeed as _selectRandomWallet uses randomSeed % validCount. Agent can deterministically choose any recipient, defeating unpredictability claims

Recommendation: Use verifiable randomness (for example, VRF) or a commit–reveal scheme, or remove randomized mode and use deterministic policy governed by votes..

Fix: Agent-supplied seed removed, seed is derived internally (blockhash + context). Still pseudo-random (not VRF/commit-reveal).

1. Priority system is ineffective / unused [FIXED ✅]

Location: RollbackWallet.sol: updateWalletPriority, getPriorityWalletNext, ownersList.priorityPosition

Issue: updateWalletPriority sets walletPriorities, and Wallet struct has priorityPosition, but getPriorityWalletNext ignores both and just scans walletAddresses.

Recommendation: Apply the stored priority when selecting the next wallet (maintain a sorted list/index or compute min by priority).

Fix: getPriorityWalletNext now uses walletPriorities to pick the min-priority wallet.

2. Misleading events and observability gaps [FIXED ✅]

Location: RollBackWallet.sol: _executeSingleWalletAction (THRESHOLD_CHANGE), getVoteDetails, _executeVote (EMERGENCY_RECOVERY)

Issue: In single-wallet threshold change, event logs old and new as the same value (state updated before emit).

getVoteDetails reports approvals equal to "hasVoted" (no rejection tracking).

EmergencyRecoveryExecuted/VoteExecuted can emit even when all transfers fail.

Recommendation: Emit old before updating. Track and expose rejections separately.

Emit Executed only after confirming at least one successful transfer or otherwise revert.

Fix: Threshold change emits old to new correctly getVoteDetails tracks rejections emergency recovery emits only after at least one successful transfer (otherwise reverts).

3. No max length for walletList in proposals [FIXED ✅]

Location: RollbackWalletManager: proposeWalletCreation()

Issue: The function doesn't cap params.walletList length. Without seeing external "requirements," this isn't a security bug by itself. Very long arrays only increase the proposer's gas/storage cost (paid by the caller) and don't create an exploitable path against others. There's no direct impact on funds or authorization.

Recommendation: If product requirements mandate a limit (for example <10 owners), add:

```
require(params.walletList.length <= 10, "Too many owners");
```

Otherwise, consider a reasonable cap to avoid bloated storage.

Fix: Hard cap added (≤ 10 owners).

4. Duplicate / identical-parameter proposals allowed [FIXED ✅]

Location: RollbackWalletManager - proposeWalletCreation()

Issue: Users can open multiple creation requests with identical parameters prior to wallet creation. This is not a security vulnerability: execution still requires all signatures and the first executed request creates the wallet; subsequent ones can be ignored or will fail later logic. It can cause clutter/confusion, not fund loss.

Recommendation: If undesired, enforce uniqueness by rejecting open requests for the same userAddress, or maintain a mapping (userAddress => openRequestId) and require previous request closure/cancellation.

Fix: Rejects if an open request already exists for the same userAddress.

5. Not updating lastActivity / balanceHash after updateMonitoredTokens [FIXED ✅]

Location: RollbackWallet - updateMonitoredTokens()

Issue: Changing the monitored token set does not recompute each wallet's balanceHash or touch lastActivity. This can make previously emitted balanceHash values stale relative to the new token set. However, balanceHash isn't used in access control or fund-moving logic; lastActivity continues to govern eligibility.

Recommendation: On token set updates, consider recalculating balanceHash for all wallets and optionally resetting lastActivity (or emitting a dedicated "token set changed" event) so monitoring tools don't misinterpret staleness.

Fix: Recomputes balanceHash for all non-obsolete wallets, lastActivity intentionally unchanged.

6. NFT Transfer may stop after a single token [FIXED ✅]

Location: RollbackWallet - _transferNFTWithPartialApproval

Issue: Function returns true after transferring one NFT (either via isApprovedForAll first token or first individually-approved token), leaving additional NFTs untransferred even when approvals exist.

Recommendation: Iterate through all owned/approved tokenIds (bounded cap) and attempt multiple transfers and aggregate success.

Fix: Iterates (capped) and aggregates success across multiple NFTs instead of returning after the first.

INFORMATIONAL

1. Additional hardening recommendations (non-severity) [FIXED ✅]

Success accounting for transferFunds: return a boolean and enforce it in callers and collect per-token results also avoid emitting Executed on pure failure.

Snapshot arrays for all token-transfer loops to avoid mid-flight mutations even without reentrancy (defensive).

Use custom errors consistently rather than mixed string reverts.

As far as Gas/compatibility, interacting with non-standard ERC20s consider OpenZeppelin SafeERC20 to handle returnless tokens without relying on try/catch decode.

Fix: transferFunds returns a boolean and callers enforce it, arrays are snapshotted, custom errors used more consistently, ERC20 transfers wrapped in a safe low-level helper

2.Weak success accounting and token compatibility [FIXED ✅]

Location: RollbackWallet.sol: transferFunds, _transferERC20WithPartialApproval, _transferNFTWithPartialApproval, general token ops

Issue: Transfers rely on return values and try/catch but don't aggregate success/failure robustly; some ERC20s are non-standard.

Recommendation: Return a boolean aggregate from transferFunds and enforce it in callers.

Consider OZ SafeERC20 to handle non-standard tokens gracefully.

Log per-token outcomes to aid monitoring.

Fix: Aggregate success is returned and per-token failures are emitted. ERC20 call wrapper tolerates non-standard return values.

3. Smart contract Hardening [FIXED ✅]

Issue: requestEmergencyRecovery emits a fake vote ID (0)

When approvalsNeeded == 0 (instant execute), it emits VoteExecuted(0, …) even though no vote exists. Use a dedicated event (already emitted) and drop VoteExecuted, or emit with a real ID.

Fix: requestEmergencyRecovery() now early-returns on approvalsNeeded == 0 so it executes the transfer immediately, sets EmergencyRecovery.executed = true, emits EmergencyRecoveryExecuted, and does not create a vote or emit VoteExecuted.

Issue: emergencyRecoveries tracking is dead code

EmergencyRecovery[] public emergencyRecoveries is never populated. getEmergencyRecoveries() will always return an empty array. Either push entries on request/execute or remove the API.

Fix:The array is now populated:

On request path: pushed in requestEmergencyRecovery() (with executed set accordingly).

On normal vote execution: _executeVote() fills/marks the corresponding record using voteToRecoveryIdPlusOne.

getEmergencyRecoveries() returns real entries.

Issue: Priority exposed in Wallet struct is stale

updateWalletPriority() updates walletPriorities[] (mapping) but not walletOwners[].priorityPosition. Any caller of getAllWallets() will see an out-of-date priority. Update both or remove the struct field.

Fix: updateWalletPriority() updates both walletPriorities[wallet] and walletOwners[wallet].priorityPosition, so getAllWallets() shows the current value.

Issue: duplicate creation proposals allowed

proposeWalletCreation() doesn't block identical/pending requests for the same userAddress. You only block when a wallet already exists. Consider rejecting if an open request for that user is present.

Fix: RollbackWalletManager.proposeWalletCreation() now iterates existing creationRequests and reverts if there's an open request for the same userAddress (Open request exists for user).

Issue: Anyone can trigger a manager-routed agent-change vote

RollbackWalletManager.requestAgentChange() is permissionless and forwards to wallet.managerRequestAgentChange(). Functionally this lets any EOA open a vote (spam/noise). Gate it (like onlyOwner, or require the caller to be one of the wallet owners).

Fix: RollbackWalletManager.requestAgentChange() is gated so caller must be the manager owner or one of the wallet's owners (checked via rollbackToWallets[]), otherwise it reverts (NotWalletOwner).

Issue: Token list allows zero / non-contracts

updateMonitoredTokens() doesn't validate addresses. A 0x0 or non-contract token will revert later (for example computeBalanceHash/transferFunds calling balanceOf). Add require(token != address(0) && token.code.length > 0) per entry.

Fix: updateMonitoredTokens() enforces require(token != address(0) && token.code.length > 0) per entry, reverting with InvalidTokenConfig() on bad addresses.

Issue: RollbackPerformed event is ambiguous

It emits the full eligibleWallets batch even if only some wallet transfers succeeded. For clearer observability, emit the subset that succeeded (or include counts/flags)

Fix: performRollback() now builds a subset array of succeeded wallets (those actually obsoleted) and emits RollbackPerformed(succeeded,,…) with the trimmed list (length set via inline mstore), avoiding false positives.

Testing coverage

During the testing phase, custom use cases were written to cover all the logic of contracts. *Check “Annexes” to see the testing code.*

```
contract: RollbackWalletManager - 77.9%
  Ownable._checkOwner - 100.0%
  RollbackWalletManager.requestAgentChange - 100.0%
  RollbackWalletManager.setTreasury - 100.0%
  RollbackWalletManager.signWalletCreation - 100.0%
  RollbackWalletManager.finalizeWalletCreation - 93.8%
  RollbackWalletManager.forceExecuteWalletCreation - 91.7%
  RollbackWalletManager.proposeWalletCreation - 91.7%
  RollbackWalletManager.executeWalletCreation - 87.5%
  Clones.clone - 75.0%
  Ownable.transferOwnership - 75.0%
  RollbackWalletManager._findWalletForAddress - 0.0%
  RollbackWalletManager.getAllUsers - 0.0%
  RollbackWalletManager.getUserWallet - 0.0%
```

```
tests/test_rollback_wallet.py::test_initialize RUNNING
Transaction sent: 0x5e9b1372fa1df4b61949b2ef03cdd60d2aa71507c4f18b403a60d87ee289d8c6
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  ERC20Mock.constructor confirmed   Block: 1   Gas used: 523834 (4.37%)
  ERC20Mock deployed at: 0x3194c8DC3dbcd3E11a07892e7bA5c3394048Cc87

Transaction sent: 0x6603dac90376db9f528fc366f3ed9973f78a83ec9c50c0fd7f84403b537b1201
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  RollbackWalletManager.constructor confirmed   Block: 2   Gas used: 7416086 (61.80%)
  RollbackWalletManager deployed at: 0x602C71e4DAC47a042Ee7f46E0aee17F94A3bA0B6

Transaction sent: 0x89027353e44fd45d74baf826449f23d6a51842887e08a116640d12ea1be28931
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  RollbackWalletManager.proposeWalletCreation confirmed   Block: 3   Gas used: 454158 (3.78%)

Transaction sent: 0xae399d3516cf8995d94fa554fa8e697adb03578e406ce3b062534fa948c3c641
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  RollbackWalletManager.signWalletCreation confirmed   Block: 4   Gas used: 84452 (0.70%)

Transaction sent: 0x18c74cb7bc0d3c08b474369910b83285e2ba829f38d3d04524981c8f09403cc0
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  RollbackWalletManager.signWalletCreation confirmed   Block: 5   Gas used: 87836 (0.73%)

Transaction sent: 0x28c58f7b72d2e8934b567bd9f21c5325ba12b31abde569fad790a8a8c6dffda9
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  RollbackWalletManager.finalizeWalletCreation confirmed   Block: 6   Gas used: 985779 (8.21%)

Transaction sent: 0x97075f4ef56cbaac29142ede792b1dc8f075b564f2844d8c08e4ed9be04d4957
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  Transaction confirmed (InvalidInitialization: )   Block: 7   Gas used: 27968 (0.23%)

tests/test_rollback_wallet.py::test_initialize PASSED
```

```
tests/test_rollback_wallet.py::test_perform_rollback RUNNING
Transaction sent: 0x8c57ea74d8c5d12761d0996915d641cc83176dacbb271cec294ec97cf0b318d0
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 2
  ERC20Mock.constructor confirmed   Block: 8   Gas used: 523834 (4.37%)
  ERC20Mock deployed at: 0xE7eD6747FaC5360f88a2EFC03E00d25789F69291

Transaction sent: 0x9af29620830206a2f4cf18693f00a6fcb9b3061028eed5d1d26f4fd3f7144c65
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 3
  ERC20Mock.mint confirmed   Block: 9   Gas used: 65821 (0.55%)

Transaction sent: 0xc48f3163f572cb98935b552f31bffadda089fbd22369e3dad2b45ab453d296bc
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 4
  RollbackWalletManager.constructor confirmed   Block: 10   Gas used: 7416086 (61.80%)
  RollbackWalletManager deployed at: 0xe0aA552A10d7EC8760Fc6c2460391E698a82dDf9

Transaction sent: 0x15c54d62f87d26d547ce839ccfc43a1d9d00962c5d259ee8733acf834f9a4a1a
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  RollbackWalletManager.proposeWalletCreation confirmed   Block: 11   Gas used: 454146 (3.78%)

Transaction sent: 0x734de2d5391c2518ea49418986430773e8250e9b599c714e9af2fa4e5a130652
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  RollbackWalletManager.signWalletCreation confirmed   Block: 12   Gas used: 84452 (0.70%)

Transaction sent: 0x1df115063593c698eeed8f7f4e13519b4c709d5e79a07712e518db59d234b631
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  RollbackWalletManager.signWalletCreation confirmed   Block: 13   Gas used: 87036 (0.73%)

Transaction sent: 0x4ea18ef0679a9c7c44cabda05fce18dcc0bf656fc165e06c0b5a3bcaf4333eb9
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  RollbackWalletManager.finalizeWalletCreation confirmed   Block: 14   Gas used: 985779 (8.21%)

Transaction sent: 0xbc0148fc23dbb59f3932b3dab793d65c17a3aab5850dcf3851999a9a41a5bc0a
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 2
  Transaction confirmed (NotAgent: )   Block: 15   Gas used: 31163 (0.26%)

Transaction sent: 0x7cab0cc0dfee22d44e02c5dc5653cdc8569cff35a7eed30d2d3324308a78e0e2
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 0
  Transaction confirmed   Block: 17   Gas used: 132002 (1.10%)

Transaction sent: 0xb15f34b67116dda98fb0b88b8137d42c7b4e14235cd04730cfa6b2b0cc5b455e
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 2
  ERC20Mock.approve confirmed   Block: 18   Gas used: 44271 (0.37%)

Transaction sent: 0x60fd726c420c87d1b63d96b30f219127e83d1568a5692d296a5382bb021d7d88
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 1
  Transaction confirmed   Block: 19   Gas used: 37943 (0.32%)

tests/test_rollback_wallet.py::test_perform_rollback PASSED
```

```
tests/test_rollback_wallet.py::test_perform_rollback_2 RUNNING
Transaction sent: 0xc49505b62b8cf0c35d54614cea062792396e0e1eae36780a57c40186d8e850d5
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 5
  ERC20Mock.constructor confirmed   Block: 20   Gas used: 523834 (4.37%)
  ERC20Mock deployed at: 0x6b4BDe1086912A6Cb24ce3dB43b3466e6c72AFd3

Transaction sent: 0xa6a4acc93b5ced2922ec92f059e601ec9e6eb327bcdbb835c3dcc5c107063694
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 6
  ERC20Mock.constructor confirmed   Block: 21   Gas used: 523834 (4.37%)
  ERC20Mock deployed at: 0x9E4c14403d7d9A8A782044E86a93CAE09D7B2ac9

Transaction sent: 0xb974eecfe10c7089e4306d2a63f8087992d5cdb63bb263dd5bba1297941ca5d0
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 7
  ERC20Mock.mint confirmed   Block: 22   Gas used: 65821 (0.55%)

Transaction sent: 0xcd33332479be3a15a6c24a63bce4309daa6cb21ad88db3330df7dbba87e8e3b0
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 8
  RollbackWalletManager.constructor confirmed   Block: 23   Gas used: 7416086 (61.80%)
  RollbackWalletManager deployed at: 0x420b1099B9eF5baba6D92029594eF45E19A04A4A

Transaction sent: 0xba70bd4876d5c55489a31249278333afb24cf66eeb780c44b3ce72cdf0f4b941
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 3
  RollbackWalletManager.proposeWalletCreation confirmed   Block: 24   Gas used: 454158 (3.78%)

Transaction sent: 0x783d64e8944ae2aced3b599d14e95d11fc6aff430f894d067fb21fa16b0f3f7d
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 2
  RollbackWalletManager.signWalletCreation confirmed   Block: 25   Gas used: 84452 (0.70%)

Transaction sent: 0xde7e8adeb7998aaa72711a6a64203fa994f69f39a1e07162e9adb16e08ae5a18
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 2
  RollbackWalletManager.signWalletCreation confirmed   Block: 26   Gas used: 87036 (0.73%)

Transaction sent: 0x383ed508c24af66e15342bb2714c5128e5308fa9959e96db5ad6baf276f2b360
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 3
  RollbackWalletManager.finalizeWalletCreation confirmed   Block: 27   Gas used: 985779 (8.21%)

Transaction sent: 0xc4b364adccb99fe3279d8c831906d82ea3fc5afbca51a1e8c2af4ea57ee90e41
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 4
  ERC20Mock.approve confirmed   Block: 28   Gas used: 44271 (0.37%)

Transaction sent: 0xc3b65d659952cebbe86e4ec73a716fd78490b992baa52f7111f323238e9b7159
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 4
  Transaction confirmed (NotAgent: )   Block: 29   Gas used: 31163 (0.26%)

Transaction sent: 0x944c259752833a756067cea70787087c5513cb6b015ee12f27d0d9d33e4e768c
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 5
  Transaction confirmed (NotAuthorized: )   Block: 31   Gas used: 24590 (0.20%)

Transaction sent: 0x160417ffde9acbb3eed660b992f00698e968ac32aa2972778796e00188e41aa1
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 5
  Transaction confirmed (Length mismatch)   Block: 32   Gas used: 25419 (0.21%)

Transaction sent: 0xbe1c3da0816cd4a5919dbc47bcfb44ee09ed48a6397833719e875f4b7fe66001
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 6
  Transaction confirmed (Too many tokens)   Block: 33   Gas used: 28143 (0.23%)

Transaction sent: 0xe7850c94a85a7557fdc8ce2bb2d45307c9f11463b3812c8cfc53f003c6fc67b5
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 7
  Transaction confirmed   Block: 34   Gas used: 95042 (0.79%)

Transaction sent: 0x47d7028675ceebeec3d023ca1b2d2419d565654caa936e234b1a2ad20e5f7391
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 2
  Transaction confirmed   Block: 35   Gas used: 126498 (1.05%)

tests/test_rollback_wallet.py::test_perform_rollback_2 PASSED
```

```
tests/test_rollback_wallet_manager.py::test_constructor RUNNING
Transaction sent: 0x340bd2dfe8740cb5c913f33a2a4489a6f8730ccd7fcf877bcbc391ef9ca62611
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 9
  RollbackWalletManager.constructor confirmed (InvalidTreasury: )   Block: 36   Gas used: 659778 (5.50%)

Transaction sent: 0x362e78eb0a5fe09ee8e2f925ffb07b8e93f2af5732c680fb6c114b498030403f
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 10
  RollbackWalletManager.constructor confirmed   Block: 37   Gas used: 7416086 (61.80%)
  RollbackWalletManager deployed at: 0xb6286fAFd045132Oad6A8143089b216C2152c025

tests/test_rollback_wallet_manager.py::test_constructor PASSED
tests/test_rollback_wallet_manager.py::test_set_treasury RUNNING
Transaction sent: 0xafc289e6192fb10a8359edc37568b85a9066d5af05e08cd768c1ea063aa8eb8c
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 11
  RollbackWalletManager.constructor confirmed   Block: 38   Gas used: 7416086 (61.80%)
  RollbackWalletManager deployed at: 0x7a3d735ee6873f17Dbdcab1d51B604928dc10d92

Transaction sent: 0xdc05f49d208902484bacb121ddf8f1ceda4c113ba816870332a974ca6c2540af
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 6
  RollbackWalletManager.setTreasury confirmed (reverted)   Block: 39   Gas used: 22507 (0.19%)

Transaction sent: 0xe2606d373dfccfaa8ad713666892d9812cbfda414a746a6f39c28de9bb54ed67
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 12
  RollbackWalletManager.setTreasury confirmed (reverted)   Block: 40   Gas used: 22522 (0.19%)

Transaction sent: 0xbd4d5ab317582d0c9c5b6ee0b83bffa693e3d9653832b8c3501167b343cf7761
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 13
  RollbackWalletManager.setTreasury confirmed   Block: 41   Gas used: 30213 (0.25%)

tests/test_rollback_wallet_manager.py::test_set_treasury PASSED
tests/test_rollback_wallet_manager.py::test_propose_wallet_creation RUNNING
Transaction sent: 0x4e22034c1a19a12f1f61ffcd022188891Bae334db2601bae4570983795a8f555
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 14
  ERC20Mock.constructor confirmed   Block: 42   Gas used: 523834 (4.37%)
  ERC20Mock deployed at: 0xe65A7a341978d59d40d30FC23F5014FACB4f575A

Transaction sent: 0x91ce15f97aa4d927019e3c621073fe3710c3e8da249441d2ebe35f889a26ba5a
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 15
  RollbackWalletManager.constructor confirmed   Block: 43   Gas used: 7416086 (61.80%)
  RollbackWalletManager deployed at: 0x30375B532345B01cB8c2AD12541b09E9Aa53A93d

Transaction sent: 0x74e212aed175df40717b06424ed3587e41c1f344c124f08f17b0736c17d77d9d
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 7
  RollbackWalletManager.proposeWalletCreation confirmed (reverted)   Block: 44   Gas used: 27109 (0.23%)

Transaction sent: 0x6f3b33c395d8db90febf9ca028b43118dfa9f5247c9bc123576c6436d2137977
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 8
  RollbackWalletManager.proposeWalletCreation confirmed (reverted)   Block: 45   Gas used: 27897 (0.23%)

Transaction sent: 0xcfb62dffad0613c7c74b14494cc6f0ef2dc4780723a25acdc00202da3464dea8
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 9
  RollbackWalletManager.proposeWalletCreation confirmed (reverted)   Block: 46   Gas used: 27979 (0.23%)

Transaction sent: 0x777fcc5cc12bdb253c971697fb0d310643b323a8fc1488f7594252de6fe5cacc
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 10
  RollbackWalletManager.proposeWalletCreation confirmed (reverted)   Block: 47   Gas used: 29715 (0.25%)

Transaction sent: 0x8faf8b77e8f895470747l1d4edb26c41ea390b10d560c6bc3e7c8a32b198520a
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 8
  RollbackWalletManager.proposeWalletCreation confirmed   Block: 48   Gas used: 454158 (3.78%)

Transaction sent: 0x0f7d67b81dfd51b8e20aff843d262d1fc63d45b994e4abdaa1e9eea5c0dfc2c0
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 9
  RollbackWalletManager.proposeWalletCreation confirmed   Block: 49   Gas used: 395582 (3.30%)

Transaction sent: 0xde48fe3f32218d0b2803c394f22dc464d23938853fb1059675f5cbf35abfda03
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 11
  RollbackWalletManager.finalizeWalletCreation confirmed   Block: 50   Gas used: 646216 (5.39%)

tests/test_rollback_wallet_manager.py::test_propose_wallet_creation PASSED
```

```
tests/test_rollback_wallet_manager.py::test_sign_wallet_creation RUNNING
Transaction sent: 0x790950a51c789dc422b89d732704920e682f3942276ae107e0077a1bcd0ffd97
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 16
  ERC20Mock.constructor confirmed   Block: 51   Gas used: 523834 (4.37%)
  ERC20Mock deployed at: 0x26f15335BB1C6a4C0B660eDd694a0555A9F1cce3

Transaction sent: 0x9e0f27e4a45650a29e0eb65dd3b8ac574060b746f3880679ed93b03641ba883c
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 17
  RollbackWalletManager.constructor confirmed   Block: 52   Gas used: 7416086 (61.80%)
  RollbackWalletManager deployed at: 0xFbD588c72B438faD4Cf7cD879c8F730Faa2130a0

Transaction sent: 0x03e561fd471b730a828baa27940b175ad091eed75e15675e611094fd1caa07c4
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 10
  RollbackWalletManager.proposeWalletCreation confirmed   Block: 53   Gas used: 454158 (3.78%)

Transaction sent: 0x11aff5274d4d00b497cd5140ea070407eda8903d2561e1a74b3f48de80669c7b
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 12
  RollbackWalletManager.signWalletCreation confirmed (reverted)   Block: 54   Gas used: 22407 (0.19%)

Transaction sent: 0x00c656c4fb53d20b4d6d4bc4ccd29c1b3deaad67745e24eb46471972bdf288b0
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 13
  RollbackWalletManager.signWalletCreation confirmed (reverted)   Block: 55   Gas used: 32812 (0.27%)

Transaction sent: 0x02fd3283d8db709e289f7be63c391cdcd1d6bf9c4c122b3ca8252b4b117dea19
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 11
  RollbackWalletManager.signWalletCreation confirmed (reverted)   Block: 56   Gas used: 24202 (0.20%)

Transaction sent: 0x03d086e8e6c90e786ea084f7c2a584e6c8c790da3dacf7e16f8ab1152d85b4dc
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 3
  RollbackWalletManager.signWalletCreation confirmed   Block: 57   Gas used: 84452 (0.70%)

tests/test_rollback_wallet_manager.py::test_sign_wallet_creation PASSED
tests/test_rollback_wallet_manager.py::test_finalize_wallet_creation RUNNING
Transaction sent: 0x963d7522b42a164e176ef298bdd8267cac6c552340bc53e4d970ecc1d0851042
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 18
  ERC20Mock.constructor confirmed   Block: 58   Gas used: 523834 (4.37%)
  ERC20Mock deployed at: 0xed00238F9A0F7b4d93842033cdF56cCB32C781c2

Transaction sent: 0xb792c9767f6922ad23afd2d6e74e438a7f7c7e779aa5981f3767fadc8046626b
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 19
  RollbackWalletManager.constructor confirmed   Block: 59   Gas used: 7416086 (61.80%)
  RollbackWalletManager deployed at: 0xDae02e4fE488952cFB8c9517715Dl88647a0146

Transaction sent: 0xc3bc595b27f42a80a554d98854df3ce81c6886d0cd6543716194cb43379ada5d
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 12
  RollbackWalletManager.proposeWalletCreation confirmed   Block: 60   Gas used: 454146 (3.78%)

Transaction sent: 0x538576a7214b8faee30b35d64b33c4afc89abc6d8677c8d1043e54dc36b57339
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 14
  RollbackWalletManager.finalizeWalletCreation confirmed (reverted)   Block: 61   Gas used: 22429 (0.19%)

Transaction sent: 0x87268a24b8800f7fc4e86797279a403c1a8458f1d6abe87c54166e34809633e7
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 15
  RollbackWalletManager.finalizeWalletCreation confirmed (reverted)   Block: 62   Gas used: 24899 (0.21%)

Transaction sent: 0x5690645dedc346453fc9824781744bf0bedde47b36f85652c9eec3566cf7e8d7
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 4
  RollbackWalletManager.signWalletCreation confirmed   Block: 63   Gas used: 84452 (0.70%)

Transaction sent: 0x08ea2033a374830e2793b283bfea1946011348bec46fbcc90b8d56671abdb32e
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 3
  RollbackWalletManager.signWalletCreation confirmed   Block: 64   Gas used: 87036 (0.73%)

Transaction sent: 0x0ec8391d3ae77ee7c19b9fc5246de4991c200cbb7db4f70819b4283b4a4fcb0d
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 16
  RollbackWalletManager.finalizeWalletCreation confirmed   Block: 65   Gas used: 985779 (8.21%)

Transaction sent: 0xade9302ca1af1295a351e4aa35eaba1c649caa1116a3ed5f8b90899160d971dc
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 4
  RollbackWalletManager.signWalletCreation confirmed (reverted)   Block: 66   Gas used: 23239 (0.19%)

Transaction sent: 0xfeabc7333edb5a06e2399787ac89d00dc8b517849e3abd4f113c06c24b5c94a9
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 17
  RollbackWalletManager.finalizeWalletCreation confirmed (reverted)   Block: 67   Gas used: 23261 (0.19%)

tests/test_rollback_wallet_manager.py::test_finalize_wallet_creation PASSED
```

```
tests/test_rollback_wallet_manager.py::test_force_wallet_creation RUNNING
Transaction sent: 0x1d9f175cfbad5c36ee837b4d93ce6253e75d1accdfb50fa03ff246efdcc8e8d4
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 20
  ERC20Mock.constructor confirmed   Block: 68   Gas used: 523834 (4.37%)
  ERC20Mock deployed at: 0xdCF93F11ef216cEC9C07fd31dD801c9b2b39Afb4

Transaction sent: 0x8a9761b9ce2a8a796d862854cf127fa0e4aa2512130ecf3fc7368fc31dbb2a33
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 21
  RollbackWalletManager.constructor confirmed   Block: 69   Gas used: 7416086 (61.80%)
  RollbackWalletManager deployed at: 0xBcb61491F1859f53438918F1A5aFCA542Af9D397

Transaction sent: 0x51c3bb32490099a291b7d4cc9b0c34f387e02e9cb7ff69947bca2ba34dbcb68e
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 13
  RollbackWalletManager.proposeWalletCreation confirmed   Block: 70   Gas used: 454158 (3.78%)

Transaction sent: 0xa71d9011572f80052a00e31fc09fcc68891d7c642e484b632cldda145f2f5507
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 18
  RollbackWalletManager.forceExecuteWalletCreation confirmed (reverted)   Block: 71   Gas used: 22374 (0.19%)

Transaction sent: 0x0ab0f063f2056b85170cf7531d2b2573fe3d104f4e808d7660370f80fe668e4e
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 22
  RollbackWalletManager.forceExecuteWalletCreation confirmed (reverted)   Block: 72   Gas used: 23249 (0.19%)

Transaction sent: 0x4320c338c4713ceb54be00cbe3c1bc5bfe0ff51ab919f1fa919b3e679a002b67
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 23
  RollbackWalletManager.forceExecuteWalletCreation confirmed (reverted)   Block: 73   Gas used: 28342 (0.24%)

Transaction sent: 0x1db0656134c8e7993f8cc10a70a86adc7c622dabcc6e7928ed44979db0c25ada
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 5
  RollbackWalletManager.signWalletCreation confirmed   Block: 74   Gas used: 84452 (0.70%)

Transaction sent: 0x292b18874c4013d809e5ae042724956389142O6a85e1e39e6edb0d2bee160ed9
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 5
  RollbackWalletManager.signWalletCreation confirmed   Block: 75   Gas used: 87036 (0.73%)

Transaction sent: 0xebf01556455b9455fa26f0c62c67c175c30791c60d2d00cd8418cf5d4120dd9a
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 24
  RollbackWalletManager.forceExecuteWalletCreation confirmed   Block: 76   Gas used: 966229 (8.05%)

Transaction sent: 0x2f8ffb5cae21c6fd1babebb061192526bf8b838e2eb5ba03bd88f3044250c8fb
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 25
  RollbackWalletManager.forceExecuteWalletCreation confirmed (reverted)   Block: 77   Gas used: 24081 (0.20%)

Transaction sent: 0x12ac0d7183c6a1e750445d058e84b44839fdc3f38c3d47d4d95859e8cfd54457
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 19
  RollbackWalletManager.transferOwnership confirmed (reverted)   Block: 78   Gas used: 22757 (0.19%)

Transaction sent: 0x98b24189d0ab303624aadb6177c135c9ba387ee22ab375590e09920ccbd4ec42
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 26
  RollbackWalletManager.transferOwnership confirmed   Block: 79   Gas used: 30163 (0.25%)

tests/test_rollback_wallet_manager.py::test_force_wallet_creation PASSED
```

```
tests/test_rollback_wallet_manager.py::test_request_agent_change RUNNING
Transaction sent: 0x33f51a2f915b499e1371ccf027b9175a8e501865b6d285048ea4171b9f9f7c7e
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 27
  ERC20Mock.constructor confirmed   Block: 80   Gas used: 523834 (4.37%)
  ERC20Mock deployed at: 0x832698Daec363C9A7aB036C224Af5B21280b3AC6

Transaction sent: 0x648b405bb92dc15dc615940c8681f4ca8a8baffe4406bc22efef28b48d8e66e0
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 28
  RollbackWalletManager.constructor confirmed   Block: 81   Gas used: 7416086 (61.80%)
  RollbackWalletManager deployed at: 0xA95916C3D979400C7443961330b3092510a229Ba

Transaction sent: 0x20586f2d74ba37d899f07a90e92afac956882b461fb83bf243430b8c6aabddae
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 14
  RollbackWalletManager.proposeWalletCreation confirmed   Block: 82   Gas used: 454158 (3.78%)

Transaction sent: 0x40c4bbe796b9a62eca5e42654c0653d4d02519d0f806f4a8c673e181019ed85c
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 6
  RollbackWalletManager.signWalletCreation confirmed   Block: 83   Gas used: 84452 (0.70%)

Transaction sent: 0x95ca0bb40130a72add2063d14dde11e0250c64a648883d09e746e8fc6b230ab4
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 6
  RollbackWalletManager.signWalletCreation confirmed   Block: 84   Gas used: 87036 (0.73%)

Transaction sent: 0x58a5361dde1b35c7e62cc8a171ddf1fec8b691905e9f99ccd26d316b8aaded0c
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 29
  RollbackWalletManager.forceExecuteWalletCreation confirmed   Block: 85   Gas used: 966229 (8.05%)

Transaction sent: 0x7f2951ace65ce58ad1df697315c2f02d0b7e03a5ba8210155c4edb451a3b3b58
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 20
  RollbackWalletManager.requestAgentChange confirmed (Wallet not found)   Block: 86   Gas used: 23340 (0.19%)

Transaction sent: 0xf389842ba5a49e51e4b0c52c0f22c24627f726f988c39602ce1ad5ad52b49abd
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 21
  RollbackWalletManager.requestAgentChange confirmed (reverted)   Block: 87   Gas used: 23086 (0.19%)

Transaction sent: 0x514ffde3e00930edd24014cd979251dd550f44c9dd867d7f55de856dc671990b
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 22
  RollbackWalletManager.requestAgentChange confirmed (reverted)   Block: 88   Gas used: 27083 (0.23%)

Transaction sent: 0x536815d5e573996f3146ac2e5c9f558e6aa338802d0733a1ab5990c29fa47ee7
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 15
  Transaction confirmed   Block: 89   Gas used: 212859 (1.77%)

Transaction sent: 0x22eb7b7085a93b59e830d7fa453b85ed66763491118023 18f33f1833239d11e3
  Gas price: 0.0 gwei   Gas limit: 12000000   Nonce: 7
  Transaction confirmed   Block: 90   Gas used: 84063 (0.70%)

tests/test_rollback_wallet_manager.py::test_request_agent_change PASSED
```

Annexes

Testing code:

Rollback_WalletManager:

```
from brownie import (
    reverts,
    RollbackWallet
)

from brownie.network.contract import Contract

from scripts.helpful_scripts import (
    ZERO_ADDRESS,
    DAY_TIMESTAMP,
    get_account,
    get_timestamp,
    get_chain_number,
    increase_timestamp
)

from scripts.deploy import (
    deploy_erc,
    deploy_erc_721,
    deploy_rollback_manager
)

INITIALISATION_FEE = 0.00264e18
EMERGENCY_ROLLBACK_FEE = 0.00528e18
```

```python
def test_constructor(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)
    treasury = get_account(8)

    with reverts("InvalidTreasury: "):
        deploy_rollback_manager(owner, ZERO_ADDRESS)

    deploy_rollback_manager(owner, treasury.address)

def test_set_treasury(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)
    treasury = get_account(8)

    manager = deploy_rollback_manager(owner, treasury.address)
    with reverts():
        manager.setTreasury(ZERO_ADDRESS, {"from": other})
    with reverts("InvalidTreasury: "):
        manager.setTreasury(ZERO_ADDRESS, {"from": owner})

    assert manager.treasury() == treasury.address
    manager.setTreasury(extra, {"from": owner})
```

```
    assert manager.treasury() == extra


def test_propose_wallet_creation(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)
    treasury = get_account(8)

    usdc_token = deploy_erc(owner, "USDC", "USDC")
    manager = deploy_rollback_manager(owner, treasury.address)

    # Mock addr
    user = "0x9e9A4dead36999144e836136F9B8b167f6F3036B"
    wallet_lst = [
        user,
        "0x58DF75fAFc603B69c2e05A2bBDa22Fa5Cf396CDb",
        "0x03a9f18887dBB54A8684E2478e2Ce67d38eCB51d"
    ]
    fallback = "0xBF224703cAAf2B9219023edF6bdEc5Ebf282Ed68"
    agent = "0x44C5A8483c9D8cc3e317E897dA1306586F34a9A6"

    with reverts("InvalidUser: "):
        manager.proposeWalletCreation(
            [
                user, [other], get_timestamp(10),
                [usdc_token.address], [0],
                False, fallback, agent
```

```
        ], {"from": other}
    )
with reverts("InvalidUser: "):
    manager.proposeWalletCreation(
        [
            ZERO_ADDRESS, wallet_lst, get_timestamp(10),
            [usdc_token.address], [0],
            False, fallback, agent
        ], {"from": other}
    )
with reverts("InvalidAgent: "):
    manager.proposeWalletCreation(
        [
            user, wallet_lst, get_timestamp(10),
            [usdc_token.address], [0],
            False, fallback, ZERO_ADDRESS
        ], {"from": other}
    )
with reverts("NotWalletOwner: "):
    manager.proposeWalletCreation(
        [
            user, wallet_lst, get_timestamp(10),
            [usdc_token.address], [0],
            False, fallback, agent
        ], {"from": other}
    )


tx = manager.proposeWalletCreation(
```

```
        get_wallet_creation_params(usdc_token.address),
        {"from": user}
    )
    assert tx.events['WalletCreationProposed'][0]['requestId'] == 1
    assert tx.events['WalletCreationProposed'][0]['proposer'] == user

    tx = manager.proposeWalletCreation(
        [
            user, [user], get_timestamp(10),
            [usdc_token.address], [0],
            False, user, user
        ], {"from": user}
    )
    assert tx.events['WalletCreationProposed'][0]['requestId'] == 2
    assert tx.events['WalletCreationProposed'][0]['proposer'] == user
    manager.finalizeWalletCreation(2, {"from": other, "value": INITIALISATION_FEE})


def test_sign_wallet_creation(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)
    treasury = get_account(8)

    usdc_token = deploy_erc(owner, "USDC", "USDC")
    manager = deploy_rollback_manager(owner, treasury.address)

    params = get_wallet_creation_params(usdc_token.address)
```

```
    user = params[0]
    wallets = params[1]

    tx = manager.proposeWalletCreation(
        get_wallet_creation_params(usdc_token.address),
        {"from": user}
    )
    request_id = tx.events['WalletCreationProposed'][0]['requestId']

    with reverts("InvalidRequestId: "):
        manager.signWalletCreation(10, {"from": other}
    )
    with reverts("NotWalletOwner: "):
        manager.signWalletCreation(request_id, {"from": other}
    )
    with reverts("AlreadySigned: "):
        manager.signWalletCreation(request_id, {"from": wallets[0]}
    )
    tx = manager.signWalletCreation(request_id, {"from": wallets[1]})
    assert tx.events['WalletCreationSigned'][0]['requestId'] == 1
    assert tx.events['WalletCreationSigned'][0]['signer'] == wallets[1]


def test_finalize_wallet_creation(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)
    treasury = get_account(8)
```

```
    usdc_token = deploy_erc(owner, "USDC", "USDC")
    manager = deploy_rollback_manager(owner, treasury.address)

    params = get_wallet_creation_params(usdc_token.address)
    user = params[0]
    wallets = params[1]

    tx = manager.proposeWalletCreation(
        get_wallet_creation_params(usdc_token.address),
        {"from": user}
    )
    request_id = tx.events['WalletCreationProposed'][0]['requestId']
    with reverts("InvalidRequestId: "):
        manager.finalizeWalletCreation(2, {"from": other, "value": INITIALISATION_FEE})
    with reverts("NotEnoughSignatures: "):
        manager.finalizeWalletCreation(request_id, {"from": other, "value":
INITIALISATION_FEE})

    # Sign wallet
    manager.signWalletCreation(request_id, {"from": wallets[1]})
    manager.signWalletCreation(request_id, {"from": wallets[2]})

    tx = manager.finalizeWalletCreation(request_id, {"from": other, "value":
INITIALISATION_FEE})
    assert tx.events['WalletCreated'][0]['user'] == user

    with reverts("RequestAlreadyExecuted: "):
        manager.signWalletCreation(request_id, {"from": wallets[2]})
```

```
    with reverts("RequestAlreadyExecuted: "):
        manager.finalizeWalletCreation(request_id, {"from": other, "value":
INITIALISATION_FEE})


def test_force_wallet_creation(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)
    treasury = get_account(8)

    usdc_token = deploy_erc(owner, "USDC", "USDC")
    manager = deploy_rollback_manager(owner, treasury.address)

    params = get_wallet_creation_params(usdc_token.address)
    user = params[0]
    wallets = params[1]

    tx = manager.proposeWalletCreation(
        get_wallet_creation_params(usdc_token.address),
        {"from": user}
    )
    request_id = tx.events['WalletCreationProposed'][0]['requestId']

    with reverts():
        manager.forceExecuteWalletCreation(5, {"from": other})
    with reverts("InvalidRequestId: "):
        manager.forceExecuteWalletCreation(5, {"from": owner})
```

```
    with reverts("NotEnoughSignatures: "):
        manager.forceExecuteWalletCreation(request_id, {"from": owner})

    # Sign wallet
    manager.signWalletCreation(request_id, {"from": wallets[1]})
    manager.signWalletCreation(request_id, {"from": wallets[2]})

    tx = manager.forceExecuteWalletCreation(request_id, {"from": owner})
    assert tx.events['WalletCreated'][0]['user'] == user

    with reverts("RequestAlreadyExecuted: "):
        manager.forceExecuteWalletCreation(request_id, {"from": owner})

    with reverts():
        manager.transferOwnership(extra, {"from": other})
    manager.transferOwnership(extra, {"from": owner})

def test_request_agent_change(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)
    another = get_account(3)
    treasury = get_account(8)

    usdc_token = deploy_erc(owner, "USDC", "USDC")
    manager = deploy_rollback_manager(owner, treasury.address)
```

```
    params = get_wallet_creation_params(usdc_token.address)
    user = params[0]
    wallets = params[1]

    tx = manager.proposeWalletCreation(
        get_wallet_creation_params(usdc_token.address),
        {"from": user}
    )
    request_id = tx.events['WalletCreationProposed'][0]['requestId']
    manager.signWalletCreation(request_id, {"from": wallets[1]})
    manager.signWalletCreation(request_id, {"from": wallets[2]})
    tx = manager.forceExecuteWalletCreation(request_id, {"from": owner})
    wallet = tx.events['WalletCreated'][0]['wallet']

    with reverts("Wallet not found"):
        manager.requestAgentChange(extra, another, {"from": other})
    with reverts("InvalidAgent: "):
        manager.requestAgentChange(user, ZERO_ADDRESS, {"from": other})
    with reverts("NotAuthorized: "):
        manager.requestAgentChange(user, extra, {"from": other})

    rollback_wallet = Contract.from_abi("RollbackWallet", wallet, RollbackWallet.abi)
    tx = rollback_wallet.requestVote(0, extra, 0, {"from": user})
    assert tx.events['VoteInitiated'][0]['voteId'] == 0
    assert tx.events['VoteInitiated'][0]['voteType'] == 0
    assert tx.events['VoteInitiated'][0]['initiator'] == user

    tx = rollback_wallet.confirmVote(0, True, {"from": wallets[1]})
```

```
    assert tx.events['VoteConfirmed'][0]['voteId'] == 0
    assert tx.events['VoteConfirmed'][0]['voter'] == wallets[1]
    assert tx.events['VoteConfirmed'][0]['approved'] == True
    assert tx.events['VoteConfirmed'][0]['currentApprovals'] == 2


def get_wallet_creation_params(token):
    '''
    struct CreateWalletParams {
        address userAddress;
        address[] walletList;
        uint256 timeThreshold;
        address[] tokensToMonitor;
        RollbackWallet.TOKEN_TYPE[] tokenTypes;
        bool isRandomized;
        address fallbackWallet;
        address agentWallet;
    }
    '''
    # Mock addr
    user_addr = "0x9e9A4dead36999144e836136F9B8b167f6F3036B"
    wallet_lst = [
        user_addr,
        "0x58DF75fAFc603B69c2e05A2bBDa22Fa5Cf396CDb",
        "0x03a9f18887dBB54A8684E2478e2Ce67d38eCB51d"
    ]
    time_threshold = 864000
    tokens_to_monitor = [token]
    token_types = [0]
```

```
    is_randomized = False
    fallback = "0xBF224703cAAf2B9219023edF6bdEc5Ebf282Ed68"
    agent = "0x44C5A8483c9D8cc3e317E897dA1306586F34a9A6"


    wallet_params = [
        user_addr, wallet_lst, time_threshold,
        tokens_to_monitor, token_types,
        is_randomized, fallback, agent
    ]
    return wallet_params
```

Rollback_Wallet:

```
from brownie import (
    reverts,
    RollbackWallet
)


from brownie.network.contract import Contract


from scripts.helpful_scripts import (
    ZERO_ADDRESS,
    DAY_TIMESTAMP,
    get_account,
    get_timestamp,
    get_chain_number,
    increase_timestamp
)
```

```python
from scripts.deploy import (
    deploy_erc,
    deploy_erc_721,
    deploy_rollback_manager,
    deploy_rollback
)

INITIALISATION_FEE = 0.00264e18
EMERGENCY_ROLLBACK_FEE = 0.00528e18

def test_initialize(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)
    treasury = get_account(8)

    usdc_token = deploy_erc(owner, "USDC", "USDC")
    manager = deploy_rollback_manager(owner, treasury.address)

    rollback_wallet, _ = get_new_rollback_wallet(other, manager, usdc_token.address)
    with reverts("InvalidInitialization: "):
        rollback_wallet.initialize(get_wallet_params(usdc_token.address), {"from": extra})

def test_perform_rollback(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
```

```
    extra = get_account(2)

    another = get_account(3)

    treasury = get_account(8)

    user = "0x9e9A4dead36999144e836136F9B8b167f6F3036B"


    usdc_token = deploy_erc(owner, "USDC", "USDC")

    # mint some tokens

    usdc_token.mint(user, 1e18)

    manager = deploy_rollback_manager(owner, treasury.address)


    rollback_wallet, params = get_new_rollback_wallet(other, manager, usdc_token.address)

    agent = params[-1]

    wallet_lst = params[1]


    with reverts("NotAgent: "):

        rollback_wallet.performRollback([extra, another], 53, {"from": other})


    increase_timestamp(DAY_TIMESTAMP * 15)


    tx = rollback_wallet.performRollback(wallet_lst, 53, {"from": agent})

    assert tx.events['TokenTransferFailed'][0]['token'] == usdc_token.address

    assert tx.events['TokenTransferFailed'][0]['from'] == user

    assert tx.events['TokenTransferFailed'][0]['to'] == wallet_lst[1]

    assert tx.events['TokenTransferFailed'][0]['amount'] == 1e18


    usdc_token.approve(rollback_wallet, 1e18, {"from":
"0x9e9A4dead36999144e836136F9B8b167f6F3036B"})

    tx = rollback_wallet.performRollback(wallet_lst, 53, {"from": agent})
```

```
    assert "Transfer" is not tx.events


def test_perform_rollback_2(only_local):
    # Arrange
    owner = get_account(0)
    other = get_account(1)
    extra = get_account(2)
    another = get_account(3)
    treasury = get_account(8)
    user = "0x9e9A4dead36999144e836136F9B8b167f6F3036B"

    usdc_token = deploy_erc(owner, "USDC", "USDC")
    new_usdc_token = deploy_erc(owner, "USDC", "USDC")
    # mint some tokens
    usdc_token.mint(user, 1e18)
    manager = deploy_rollback_manager(owner, treasury.address)

    rollback_wallet, params = get_new_rollback_wallet(other, manager, usdc_token.address)
    agent = params[-1]
    wallet_lst = params[1]
    usdc_token.approve(rollback_wallet, 1e18, {"from": user})

    with reverts("NotAgent: "):
        rollback_wallet.performRollback([extra, another], 53, {"from": other})

    increase_timestamp(DAY_TIMESTAMP * 15)

    with reverts("NotAuthorized: "):
```

```
        rollback_wallet.updateMonitoredTokens([new_usdc_token.address], [1], {"from":
other})
    with reverts("Length mismatch"):
        rollback_wallet.updateMonitoredTokens([new_usdc_token.address], [], {"from": user})
    with reverts("Too many tokens"):
        rollback_wallet.updateMonitoredTokens(

[new_usdc_token.address,new_usdc_token.address,new_usdc_token.address,new_usdc_token.addres
s,new_usdc_token.address,new_usdc_token.address],
            [0,0,0,0,0,0],
            {"from": user})


    rollback_wallet.updateMonitoredTokens(
            [new_usdc_token.address],
            [0],
            {"from": user})


    tx = rollback_wallet.performRollback(wallet_lst, 53, {"from": agent})
    assert tx.events['RollbackAttemptFailed'][0]['wallet'] == user
    assert tx.events['RollbackAttemptFailed'][0]['reason'] == "No transfers succeeded"


def get_new_rollback_wallet(account, manager, token):
    params = get_wallet_creation_params(token)
    user = params[0]
    wallets = params[1]


    tx = manager.proposeWalletCreation(
        get_wallet_creation_params(token),
        {"from": user}
```

```
    )
    request_id = tx.events['WalletCreationProposed'][0]['requestId']
    manager.signWalletCreation(request_id, {"from": wallets[1]})
    manager.signWalletCreation(request_id, {"from": wallets[2]})
    tx = manager.finalizeWalletCreation(request_id, {"from": account, "value":
INITIALISATION_FEE})
    wallet = tx.events['WalletCreated'][0]['wallet']

    return Contract.from_abi("RollbackWallet", wallet, RollbackWallet.abi), params


def get_wallet_params(token):
    '''
    struct RollbackWalletParams {
        address agentWalletAddress;
        bool isRandomized;
        address fallbackWalletAddress;
        address[] walletList;
        uint256 timeThreshold;
        address[] tokensToMonitor;
        TOKEN_TYPE[] tokenTypes;
        address treasuryAddress;
    }
    '''
    # Mock addr
    agent_wallet_addr = "0x9e9A4dead36999144e836136F9B8b167f6F3036B"
    is_randomized = False
    fallback = "0xBF224703cAAf2B9219023edF6bdEc5Ebf282Ed68"
    wallet_lst = [
```

```
        agent_wallet_addr,
        "0x58DF75fAFc603B69c2e05A2bBDa22Fa5Cf396CDb",
        "0x03a9f18887dBB54A8684E2478e2Ce67d38eCB51d"
    ]
    time_threshold = 864000
    tokens_to_monitor = [token]
    token_types = [0]
    treasury = "0x44C5A8483c9D8cc3e317E897dA1306586F34a9A6"

    wallet_params = [
        agent_wallet_addr, is_randomized, fallback,
        wallet_lst, time_threshold, tokens_to_monitor,
        token_types, treasury
    ]
    return wallet_params

def get_wallet_creation_params(token):
    '''
    struct CreateWalletParams {
        address userAddress;
        address[] walletList;
        uint256 timeThreshold;
        address[] tokensToMonitor;
        RollbackWallet.TOKEN_TYPE[] tokenTypes;
        bool isRandomized;
        address fallbackWallet;
        address agentWallet;
    }
```

```
    '''
    # Mock addr
    user_addr = "0x9e9A4dead36999144e836136F9B8b167f6F3036B"
    wallet_lst = [
        user_addr,
        "0x58DF75fAFc603B69c2e05A2bBDa22Fa5Cf396CDb",
        "0x03a9f18887dBB54A8684E2478e2Ce67d38eCB51d"
    ]
    time_threshold = 864000
    tokens_to_monitor = [token]
    token_types = [0]
    is_randomized = False
    fallback = "0xBF224703cAAf2B9219023edF6bdEc5Ebf282Ed68"
    agent = "0x44C5A8483c9D8cc3e317E897dA1306586F34a9A6"

    wallet_params = [
        user_addr, wallet_lst, time_threshold,
        tokens_to_monitor, token_types,
        is_randomized, fallback, agent
    ]
    return wallet_params
```

Technical Findings Summary

Findings

Vulnerability Level		Total	Pending	Not Apply	Acknowledged	Partially Fixed	Fixed
	HIGH	8					8
	MEDIUM	5				1	4
	LOW	6					6
	INFORMATIONAL	3					3

Assessment Results

Score Results

Review	Score
Global Score	**90/100**
Assure KYC	https://projects.assuredefi.com/project/rollback
Audit Score	85/100

The Following Score System Has been Added to this page to help understand the value of the audit, the maximum score is 100, however to attain that value the project must pass and provide all the data needed for the assessment. Our Passing Score has been changed to 84 Points for a higher standard, if a project does not attain 85% is an automatic failure. Read our notes and final assessment below. The Global Score is a combination of the evaluations obtained between having or not having KYC and the type of contract audited together with its manual audit.

Audit PASS

Following our comprehensive security audit of the token contract for the RollBackWallet project, we inform you that the project has met the necessary security standards.

Disclaimer

Assure Defi has conducted an independent security assessment to verify the integrity of and highlight any vulnerabilities or errors, intentional or unintentional, that may be present in the reviewed code for the scope of this assessment. This report does not constitute agreement, acceptance, or advocating for the Project, and users relying on this report should not consider this as having any merit for financial RollBackWallet in any shape, form, or nature. The contracts audited do not account for any economic developments that the Project in question may pursue, and the veracity of the findings thus presented in this report relate solely to the proficiency, competence, aptitude, and discretion of our independent auditors, who make no guarantees nor assurance that the contracts are entirely free of exploits, bugs, vulnerabilities or deprecation of technologies.

All information provided in this report does not constitute financial or investment RollBackWallet, nor should it be used to signal that any person reading this report should invest their funds without sufficient individual due diligence, regardless of the findings presented. Information is provided ‘as is, and Assure Defi is under no covenant to audit completeness, accuracy, or solidity of the contracts. In no event will Assure Defi or its partners, employees, agents, or parties related to the provision of this audit report be liable to any parties for, or lack thereof, decisions or actions with regards to the information provided in this audit report.

The assessment RollBackWallets provided by Assure Defi are subject to dependencies and are under continuing development. You agree that your access or use, including but not limited to any RollBackWallet reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies with high levels of technical risk and uncertainty. The assessment reports could include false positives, negatives, and unpredictable results. The RollBackWallet may access, and depend upon, multiple layers of third parties.