



TECHNICAL SECURITY ASSESSMENT

Tamarind Staking

Smart Contract Audit · Staking

REPORT ID QX-TSA-2026-006

QUANTIX RESEARCH ASSESSMENT

Aptos Move modules for staking, reward-index accounting, the unstake cooldown queue and the admin capability.

Document status
Published

Published
12 May 2026

Review period
20 April 2026 – 29 April 2026

Ecosystems
Aptos, Sui

Languages
Move (Aptos), Move (Sui)

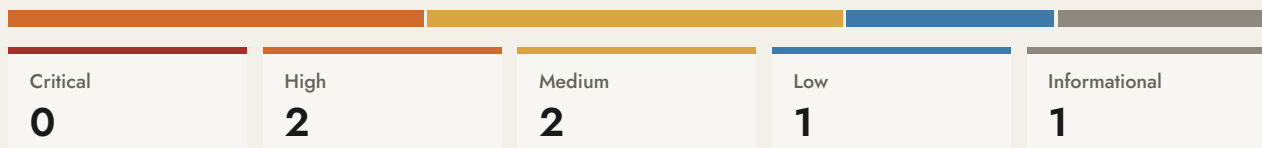
Review effort
Two reviewers, eight working days

Reviewed commit
00af372ccd088a27187d390b402ef9d23d19824b

Remediation commit
8cd82e7b6c829eee4d5ea44bc796ddcc9a8aa238

FINDINGS BY SEVERITY

6 findings · 75% remediated



This document is a Quantix research assessment of Tamarind Staking, a protocol design from Quantix Protocol Research. It is not a client engagement, it does not describe a deployed system, and it is not a certification.

DOCUMENT CONTROL

Document ID	QX-TSA-2026-006
Title	Technical Security Assessment: Tamarind Staking
Version	1.0
Status	Published
Assessment type	Smart Contract Audit
Review period	20 April 2026 – 29 April 2026
Published	12 May 2026
Reviewed commit	00af372ccd088a27187d390b402ef9d23d19824b

CONTENTS

1	Executive summary	3
2	Scope	4
3	Protocol architecture	4
4	Threat model	5
5	Methodology and severity classification	6
6	Findings summary	6
7	Detailed findings	8
	TS-01 Reward index is not settled before stake changes	8
	TS-02 Cooldown can be bypassed by splitting a queued position	9
	TS-03 Batch claim iterates over an unbounded list	10
	TS-04 Rounding leaves undistributed dust in the reward pool	10
	TS-05 Parameter updates emit no events	11
	TS-06 AdminCap can change the cooldown with immediate effect	12
8	Severity and risk analysis	13
9	Remediation status	13
10	Formal verification and invariants	14
11	Recommendations	15
12	Limitations	15
A	Technical appendix	16

1

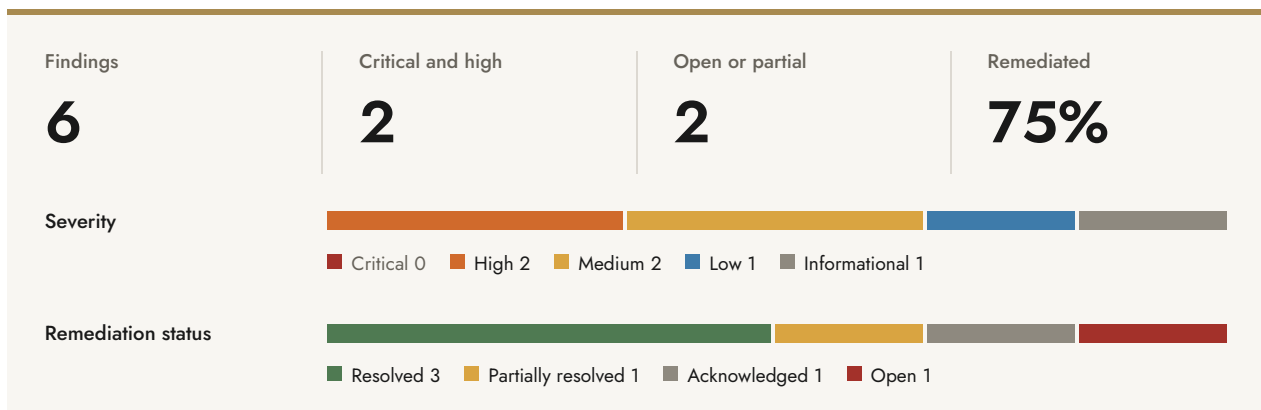
Executive summary

Tamarind lets users stake the native asset for a receipt token, accrue rewards through a reward-per-share index, and withdraw through a cooldown queue. In this design the critical invariant is that every user's accrued rewards are settled at the index value in force when their stake changed.

Six findings were reported. Both high-severity issues were accounting bugs: stake changes did not settle the reward index first, so new stake earned rewards accrued before it arrived, and the cooldown could be bypassed by splitting a position. Both were fixed.

Unbounded iteration in batch claims was only partially addressed, rounding dust was acknowledged, and an admin capability that can change the cooldown with immediate effect remains open. Those open items are reflected in the protocol's governance and code-security components.

AT A GLANCE



SEVERITY SUMMARY

Severity	Definition	Count	Open
■ Critical	Direct loss of funds or permanent loss of control is achievable with little or no precondition.	0	0
■ High	Material loss or protocol disruption is achievable under realistic conditions.	2	0
■ Medium	Limited loss, degraded functionality, or impact that depends on specific conditions.	2	2
■ Low	Minor impact, defence-in-depth gaps, or issues with unlikely preconditions.	1	0
■ Informational	Code quality, documentation or best-practice observations without direct security impact.	1	0

2

Scope

Aptos Move modules for staking, reward-index accounting, the unstake cooldown queue and the admin capability.

Repository	Private repository
Reviewed commit	00af372ccd088a27187d390b402ef9d23d19824b
Remediation commit	8cd82e7b6c829eee4d5ea44bc796ddcc9a8aa238
Review period	20 April 2026 – 29 April 2026
Review effort	Two reviewers, eight working days

Path	Description	nSLOC
sources/staking_pool.move	Stake, unstake and receipt-token accounting	412
sources/rewards.move	Reward-per-share index, accrual and claims	286
sources/unstake_queue.move	Cooldown queue and withdrawal processing	238
sources/admin.move	AdminCap capability and parameter updates	96
Total		1,032

Out of scope

- The Sui port (compared for parity of the TS-01 and TS-02 fixes only)
- Validator selection and delegation off-chain
- Aptos framework modules (assumed to behave as documented)

3

Protocol architecture

Users call `stake` to deposit the native coin into the pool and receive a receipt token at the current exchange rate. Validator rewards arrive through `distribute`, which increases a global `reward_per_share` index.

Each position stores `reward_debt`, the index value at which it was last settled. Pending rewards are $\text{shares} \times (\text{index} - \text{reward_debt})$. Claims pay pending rewards and reset the debt.

`request_unstake` moves shares into the unstake queue with an unlock timestamp; `withdraw` pays out once the cooldown has passed. Parameters, including cooldown length and reward rate, are changed by the holder of `AdminCap`.

Component	Role	Trust level
staking_pool	Stake, unstake and receipt-token accounting	Trusted
reward_index	Reward-per-share accrual and claims	Trusted
unstake_queue	Cooldown and withdrawal processing	Trusted

Component	Role	Trust level
admin_cap	Capability object for parameter updates	Trusted
Stakers	Stake, split, merge, claim and unstake	Untrusted
Reward distributor	Calls distribute with validator rewards	Semi-trusted

Trust assumptions

- The AdminCap is held by a multisig account controlled by the protocol team.
- The reward distributor reports rewards honestly; incorrect reports affect the rate, not individual balances.
- Aptos framework coin and timestamp modules behave as documented.

4

Threat model

Assets

- Staked native coin
- Undistributed and accrued rewards
- Cooldown guarantees relied on by integrators

Actors

Actor	Capability	Trust
Staker	Stakes, splits positions, claims and unstakes	Untrusted
Admin	Holds AdminCap; updates parameters	Trusted
Distributor	Calls distribute	Semi-trusted

Threats

Threat	Actor and vector	Mitigation	Findings
Rewards captured by late stakers	Staker: Staking just before distribute is settled	Settle index before stake changes	TS-01
Cooldown bypass	Staker: Splitting a queued position	Carry unlock time with shares	TS-02
Claim function unusable	Staker: Many small positions in a batch	Bounded batch size	TS-03
Cooldown changed under users	Admin: Immediate parameter update	Delayed parameter changes	TS-06

5

Methodology and severity classification

1. Manual review of every public entry function and resource transfer
2. Reward accounting review against the reward-per-share model, including stake changes between distributions
3. Move Prover specifications for the core accounting invariants
4. Unit and scenario tests of split, merge and unstake sequences
5. Remediation review against the fix commit, with parity checks of the TS-01 and TS-02 fixes in the Sui port

Severity classification

Each finding is rated on the impact it would have if exploited and the likelihood that its preconditions occur. The rating reflects the code at the reviewed commit.

Severity	Meaning
■ Critical	Direct loss of funds or permanent loss of control is achievable with little or no precondition.
■ High	Material loss or protocol disruption is achievable under realistic conditions.
■ Medium	Limited loss, degraded functionality, or impact that depends on specific conditions.
■ Low	Minor impact, defence-in-depth gaps, or issues with unlikely preconditions.
■ Informational	Code quality, documentation or best-practice observations without direct security impact.

Remediation status

Status	Meaning
Resolved	Fixed and verified in the remediation review.
Partially resolved	Mitigated; the residual risk is documented.
Acknowledged	Accepted by the protocol team with a stated rationale.
Open	Not yet addressed at the reviewed remediation commit.

6

Findings summary

ID	Title	Severity	Category	Status
TS-01	Reward index is not settled before stake changes	■ High	Business logic	Resolved
TS-02	Cooldown can be bypassed by splitting a queued position	■ High	Business logic	Resolved
TS-03	Batch claim iterates over an unbounded list	■ Medium	Denial of service	Partially resolved
TS-04	Rounding leaves undistributed dust in the reward pool	■ Low	Precision and rounding	Acknowledged

ID	Title	Severity	Category	Status
TS-05	Parameter updates emit no events	Informational	Code quality	Resolved
TS-06	AdminCap can change the cooldown with immediate effect	Medium	Governance manipulation	Open

7

Detailed findings

TS-01 ■ High Resolved

Reward index is not settled before stake changes

sources/staking_pool.move:L118-L156

Category	CWE	Likelihood	Impact
Business logic	CWE-682	High	Medium

DESCRIPTION

`stake` increases a position's shares without first paying out pending rewards or moving `reward_debt` to the current index. Pending rewards are later computed as $\text{shares} \times (\text{index} - \text{reward_debt})$, so the newly added shares earn every increment of the index since the position was last settled.

The same defect exists in the reverse direction in `request_unstake`, where shares removed before settlement forfeit their accrued rewards.

PRECONDITIONS

- An existing position with an old `reward_debt`
- Rewards distributed since that position was last settled

EXPLOIT SCENARIO

1. Attacker opens a tiny position early, so its `reward_debt` is very low.
2. Months later, just after a large distribution, they stake a large amount into the same position.
3. They claim; the large stake is paid as if it had been staked since the beginning.

IMPACT

Rewards owed to long-term stakers are captured by late stakers; the pool becomes unable to pay all claims.

RECOMMENDATION

Call an internal `settle(position)` at the start of every function that changes shares: pay or credit pending rewards, then set `reward_debt = index`.

FIX

sources/staking_pool.move	DIFF
1 public entry fun stake(user: &signer, amount: u64) acquires Pool, Position {	
2 let pos = borrow_global_mut<Position>(signer::address_of(user));	
3 + rewards::settle(pos);	
4 let shares = shares_for(amount);	
5 pos.shares = pos.shares + shares;	

RESOLUTION

`settle()` is called first in `stake`, `request_unstake`, `split` and `merge`. MP-02 proved.

REFERENCES

- Aptos Move book
<https://aptos.dev/build/smart-contracts/book>
- CWE-682: Incorrect Calculation
<https://cwe.mitre.org/data/definitions/682.html>

TS-02 ■ High Resolved

Cooldown can be bypassed by splitting a queued position

sources/unstake_queue.move:L64-L102

Category	CWE	Likelihood	Impact
Business logic	CWE-841	High	Medium

DESCRIPTION

`split` creates a new position from part of an existing one. When the source position has queued shares, the new position receives them with `unlock_at` set to zero, because the field is initialised by default rather than copied.

A user can request unstake, immediately split the queued position, and withdraw from the new half without waiting for the cooldown.

PRECONDITIONS

- A position with queued shares

EXPLOIT SCENARIO

1. User requests unstake of all shares.
2. User splits the position in the same block.
3. User withdraws from the new position immediately.

IMPACT

The cooldown no longer protects the pool's liquidity planning or integrators that rely on it.

RECOMMENDATION

Copy `unlock_at` with queued shares in `split`, and in `merge` take the later of the two unlock times.

FIX

sources/unstake_queue.move	DIFF
1 <code>let new_entry = QueueEntry {</code>	
2 <code>shares: amount,</code>	
3 - <code>unlock_at: 0,</code>	
4 + <code>unlock_at: entry.unlock_at,</code>	
5 <code>};</code>	

RESOLUTION

`unlock_at` copied on `split` and maximised on `merge`. MP-03 proved; the Sui port carries the same fix.

REFERENCES

- CWE-841: Improper Enforcement of Behavioral Workflow
<https://cwe.mitre.org/data/definitions/841.html>
- Aptos timestamp module
<https://aptos.dev/build/smart-contracts/book>

TS-03 ■ Medium Partially resolved

Batch claim iterates over an unbounded list

sources/rewards.move:L201

Category	CWE	Likelihood	Impact
Denial of service	CWE-400	Medium	Medium

DESCRIPTION

`batch_claim` iterates over every queue entry owned by the caller. Users who unstake in many small steps, or integrators who manage many positions under one account, accumulate entries until the function exceeds the transaction gas limit.

Because entries are only removed by a successful claim, an account that crosses the limit cannot reduce its list through this function.

PRECONDITIONS

- An account with many queue entries

EXPLOIT SCENARIO

1. An integrator account accumulates thousands of entries.
2. `batch_claim` always runs out of gas; rewards are stuck until a manual path exists.

IMPACT

Rewards and withdrawals for affected accounts become unclaimable through the batch path.

RECOMMENDATION

Add a `max_entries` argument and process the list in pages, or cap the number of entries per account.

RESOLUTION

Partially resolved. A `max_entries` argument was added, but entries are still stored in a vector that is fully loaded on each call; a table-based store is planned.

REFERENCES

- Aptos gas and storage fees
<https://aptos.dev/network/blockchain/gas-txn-fee>
- CWE-400: Uncontrolled Resource Consumption
<https://cwe.mitre.org/data/definitions/400.html>

TS-04 ■ Low Acknowledged

Rounding leaves undistributed dust in the reward pool

sources/rewards.move:L88

Category	CWE	Likelihood	Impact
Precision and rounding	CWE-1339	Medium	Low

DESCRIPTION

`distribute` computes `amount × PRECISION / total_shares` with truncation. The remainder is never carried forward, so a small amount of each distribution can never be claimed.

PRECONDITIONS

- Any distribution not divisible by total shares

EXPLOIT SCENARIO

1. Over many distributions, unclaimable dust accumulates in the pool.

IMPACT

Small, permanent loss of rewards to stakers; no exploitation path.

RECOMMENDATION

Carry the remainder forward to the next distribution.

RESOLUTION

Acknowledged. The team will sweep dust to the treasury periodically and document it.

REFERENCES

- CWE-1339: Insufficient Precision or Accuracy of a Real Number
<https://cwe.mitre.org/data/definitions/1339.html>

TS-05 ■ Informational Resolved

Parameter updates emit no events

`sources/admin.move:L40`

Category	CWE	Likelihood	Impact
Code quality	—	Low	Low

DESCRIPTION

Changes to cooldown length, reward rate and fee are applied without events, so indexers and monitoring cannot observe them.

PRECONDITIONS

- None

EXPLOIT SCENARIO

1. A parameter change goes unnoticed by users and integrators.

IMPACT

Reduced transparency of administrative actions.

RECOMMENDATION

Emit a module event for every parameter change, including old and new values.

RESOLUTION

Events added for all parameter changes.

REFERENCES

- Aptos events
<https://aptos.dev/network/blockchain/events>

TS-06 ■ Medium Open

AdminCap can change the cooldown with immediate effect

`sources/admin.move:L58`

Category	CWE	Likelihood	Impact
Governance manipulation	—	Low	High

DESCRIPTION

`set_cooldown(cap, seconds)` updates the cooldown immediately and applies to entries already in the queue, because `withdraw` compares against `requested_at + cooldown` using the current value.

The holder of the capability, or anyone who compromises it, can lengthen the cooldown to trap queued withdrawals, or shorten it to zero.

PRECONDITIONS

- Control of AdminCap

EXPLOIT SCENARIO

1. The capability holder raises the cooldown to a year.
2. All queued withdrawals are frozen without notice.

IMPACT

Users cannot rely on the cooldown they accepted when they requested unstake.

RECOMMENDATION

Record the unlock time at request time so later changes apply only to new requests, and queue parameter changes behind a delay.

RESOLUTION

Open. The team intends to store `unlock_at` at request time in the next release.

REFERENCES

- Aptos Move book: resources and capabilities
<https://aptos.dev/build/smart-contracts/book>

8

Severity and risk analysis

The matrix places each finding by the likelihood of its preconditions and the impact of exploitation. Identifiers in each cell refer to the detailed findings in section 7.

Impact

	Low	Medium	High
High	1 TS-06	0	0
Medium	0	1 TS-03	2 TS-01 TS-02
Low	1 TS-05	1 TS-04	0
	Low	Medium	High
	Likelihood		

Distribution by vulnerability class

Business logic	2
Code quality	1
Denial of service	1
Governance manipulation	1
Precision and rounding	1

Severity distribution



9

Remediation status

3 of 6 findings are resolved, 1 acknowledged, 1 partially resolved and 1 open, as reviewed at remediation commit 8cd82e7b6c82.

Severity	Resolved	Partially resolved	Acknowledged	Open	Total
■ Critical	0	0	0	0	0
■ High	2	0	0	0	2

Severity	Resolved	Partially resolved	Acknowledged	Open	Total
■ Medium	0	1	0	1	2
■ Low	0	0	1	0	1
■ Informational	1	0	0	0	1

Remediation progress

75%



Remediation history



Residual items

ID	Status	Notes
TS-03	Partially resolved	Partially resolved. A max_entries argument was added, but entries are still stored in a vector that is fully loaded on each call; a table-based store is planned.
TS-04	Acknowledged	Acknowledged. The team will sweep dust to the treasury periodically and document it.
TS-06	Open	Open. The team intends to store unlock_at at request time in the next release.

10

Formal verification and invariants

Core accounting properties were specified for the Move Prover on the remediation commit. Properties involving the unstake queue's iteration were not proved because the prover times out on the unbounded loop.

Tool Move Prover

ID	Property	Result	Notes
MP-01	Total receipt shares equal the sum of position shares plus queued shares	■ Proved	—
MP-02	A position's reward_debt equals the current index after any stake, unstake or claim	■ Proved	Violated on the original commit (TS-01).
MP-03	Shares can leave the queue only after their own unlock timestamp	■ Proved	Violated on the original commit (TS-02).
MP-04	batch_claim terminates within the gas limit for any input	■ Inconclusive	Prover timeout; see TS-03.

11

Recommendations

Fix TS-06 before integrations rely on the cooldown

Store unlock times at request and delay parameter changes so integrators can quote withdrawal times reliably.

Move the queue to a table

Replace the vector-backed queue with a table keyed by entry ID so batch operations are bounded and MP-04 can be proved.

Keep the two ports in step

Maintain a parity checklist between the Aptos and Sui modules and run the same scenario tests against both.

12

Limitations

- Only the Aptos modules at the listed commit were reviewed in full; the Sui port was compared for parity of two fixes.
- Off-chain reward reporting and validator delegation were out of scope.
- MP-04 could not be proved because of prover timeouts on unbounded iteration.

No assessment can show that a system is free of vulnerabilities. Conclusions apply to the code at the commits listed and the conditions stated in this document.

Appendix

Technical appendix

A.1 Tools

Tool	Purpose
Move Prover	Formal specifications of accounting invariants
Aptos CLI test framework	Unit and scenario tests
Manual review	Entry-function and resource-flow analysis

A.2 Reviewed files (SHA-256)

File	SHA-256
sources/staking_pool.move	c6ced7bd910f9f4970192e1fabe718dc40eebbf3d81662e9028aaec1bf2d32b4
sources/rewards.move	a1369cb508a2fe4f6017fb704811621a50fd01f2da2cdac1fdadcda192a6c820
sources/unstake_queue.move	b43a8d6cdb4f77bbb4b7e52ded70607916432f377a3ada993dcd7d72c5c4f9eb
sources/admin.move	9b629bd43493f102e5864d262f83605ee5a8d858065044c126e70a6b5c13600b

A.3 Notes

- The source repository is private; commit hashes identify the reviewed snapshot.

A.4 Methodology and technical references

Standards, specifications and weakness classifications cited in the findings. Each entry links to the official source.

1. Aptos Move book
<https://aptos.dev/build/smart-contracts/book>
2. CWE-682: Incorrect Calculation
<https://cwe.mitre.org/data/definitions/682.html>
3. CWE-841: Improper Enforcement of Behavioral Workflow
<https://cwe.mitre.org/data/definitions/841.html>
4. Aptos gas and storage fees
<https://aptos.dev/network/blockchain/gas-txn-fee>
5. CWE-400: Uncontrolled Resource Consumption
<https://cwe.mitre.org/data/definitions/400.html>
6. CWE-1339: Insufficient Precision or Accuracy of a Real Number
<https://cwe.mitre.org/data/definitions/1339.html>
7. Aptos events
<https://aptos.dev/network/blockchain/events>

A.5 Document information

Document	Quantix Technical Security Assessment QX-TSA-2026-006, version 1.0
Subject	Tamarind Staking, Quantix Protocol Research
Status	Published 12 May 2026

Disclosure

This document is a Quantix research assessment of Tamarind Staking, a protocol design from Quantix Protocol Research. It is not a client engagement, it does not describe a deployed system, and it is not a certification.
