



Fragmetric Restaking Program

Executive Summary

This audit report was prepared by Quantstamp, the leader in blockchain security.

Type	Solana Restaking Protocol
Timeline	2024-09-23 through 2024-10-24
Language	Rust
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	Documentation
Source Code	<ul style="list-style-type: none"> https://github.com/fragmetric-labs/fragmetric-contracts #0b3eeff
Auditors	<ul style="list-style-type: none"> Mostafa Yassin Auditing Engineer Michael Boyle Auditing Engineer Valerian Callens Senior Auditing Engineer

Documentation quality	High 
Test quality	Medium 
Total Findings	17 Fixed: 4 Acknowledged: 12 Mitigated: 1
High severity findings ⓘ	2 Fixed: 2
Medium severity findings ⓘ	3 Fixed: 1 Acknowledged: 1 Mitigated: 1
Low severity findings ⓘ	2 Acknowledged: 2
Undetermined severity findings ⓘ	1 Acknowledged: 1
Informational findings ⓘ	9 Fixed: 1 Acknowledged: 8

Summary of Findings

Fix Review

The client fixed/mitigated all the high and medium-severity issues.

Engagement Overview

The codebase is well-engineered with a modular approach, and the test suite covers several happy and unhappy paths. The quality of the documentation is good with both textual descriptions and visual flows.

We highly appreciate that the Fragmetric team was highly engaged and responsive throughout the audit, promptly addressing our questions and participating in productive discussions.

The audit identified 2 high severity issues, both of which were also identified by the development team during the audit.

Protocol Overview

Fragmetric is a re-staking protocol, that allows users to stake SOL, or different liquid staking derivatives supported by Fragmetric, to mint fragSol. fragSol represents a user share in the total SOL equivalent amount held by the protocol. Currently, the protocol supports two LSTs, namely jitoSol and mSol (Marinade LST).

The protocol relies on the spot price of SPL Stake Pools or Marinade Stake Pools to update prices. Price updates happen whenever the user interacts with the protocol, during deposits and withdrawals, and protocol operators can perform arbitrary price updates as well.

As an incentive, the protocol offers different types of rewards, such as points or SPL tokens. The fund manager adds rewards that are stored in the system in the form of reward blocks, which can be claimed by users on demand, or when they deposit or request withdrawal. Currently, `fragSol` is not transferrable, but the protocol implements custom transfer hooks in order to track `fragSol` transfers across the Solana network once transfers are enabled.

The reward calculation mechanism in the protocol performs reward allocation updates during deposits, withdrawals, and transfers (when they are enabled). Rewards are mainly a function of the `contribution accrual rate` which itself is just the amount deposited by a user multiplied by the time elapsed. There are two levels for tracking contributions:

- Tracking the global contribution of all users across the global `RewardPool`.
- Tracking the contribution of a specific user across their `UserRewardPool`.

In almost all cases, the `RewardPool` and `UserRewardPool` are updated together in the same instruction. The only exception is the instruction `user_update_reward_pools` which performs an update for the `UserRewardPool`.

It is worth noting that reward claiming is currently not implemented in the protocol.

The default value for a contribution rate is `100`, meaning that no multiplier is used. However, the admin can allow deposits performed through a specific wallet to earn a higher contribution rate. For this, the admin will sign a `metadata` object that contains information about the allowed wallet. Then, users can use that signature with the deposit instructions to earn a higher contribution rate. It is worth mentioning that if a user requests a withdrawal, and then cancels the withdrawal request, the contribution rate resets to its default value of `100`.

For withdrawals, users need to submit a withdrawal request, that is then added to a batch. An operator then needs to invoke the `process()` method after a pre-determined amount of time passed or if the batch threshold amount of `fragSol` is reached. After a batch is processed, users can call the `withdraw()` function to get their withdrawal in `SOL`. This means that even if the user deposited in `jitoSol` or `mSol`, when they withdraw, they will get the equivalent amount of `SOL` at the time of withdrawal. However, in the first phase, are not supported and will not be enabled.

ID	DESCRIPTION	SEVERITY	STATUS
FRA-1	Updating Price Before Withdrawal Can Cause Inaccuracies for the Withdrawal Amount	• High ⓘ	Fixed
FRA-2	Closing a Rewardpool Will Block Deposits, Withdrawals, Updates, and Transfers	• High ⓘ	Fixed
FRA-3	Signature Replay Attack Allows for Increased Rewards	• Medium ⓘ	Mitigated
FRA-4	Denial of Service when Updating Pool Parameters	• Medium ⓘ	Fixed
FRA-5	Users may Not Be Able to Withdraw	• Medium ⓘ	Acknowledged
FRA-6	Admin Can Set Arbitrary Withdrawal Fee Rate	• Low ⓘ	Acknowledged
FRA-7	Missing input validations	• Low ⓘ	Acknowledged
FRA-8	Risk of out-of-bounds error in function <code>init_without_load()</code>	• Informational ⓘ	Acknowledged
FRA-9	Impact of using <code>from_utf8_trim_null()</code> for objects <code>RewardPool</code> , <code>Holder</code> and <code>Reward</code> .	• Informational ⓘ	Fixed
FRA-10	Underflow Error Can Happen without Explicit Error if a User Executes a Transaction with <code>user_deposit_sol()</code> or <code>user_deposit_supported_token()</code> as the First Instruction with <code>metadata</code>	• Informational ⓘ	Acknowledged
FRA-11	The Closure in <code>check_valid_addition()</code> Checks Instead if the Addition Is Invalid	• Informational ⓘ	Acknowledged

ID	DESCRIPTION	SEVERITY	STATUS
FRA-12	It Is Possible to Update <code>UserRewardPool</code> without Updating <code>RewardPool</code>	• Informational ⓘ	Acknowledged
FRA-13	Immediate Withdrawals Are Possible	• Informational ⓘ	Acknowledged
FRA-14	Token Allocation Are not Updated During Transfers	• Informational ⓘ	Acknowledged
FRA-15	No Functionality To Withdraw Deposited Assets	• Informational ⓘ	Acknowledged
FRA-16	Data Used for Events Should Be Clarified	• Informational ⓘ	Acknowledged
FRA-17	If All Lamports Are Withdrawn From <code>fund_account</code> , the Account Data Could Be Deleted	• Undetermined ⓘ	Acknowledged

Assessment Breakdown

Quantstamp's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.

i Disclaimer

Only features that are contained within the repositories at the commit hashes specified on the front page of the report are within the scope of the audit and fix review. All features added in future revisions of the code are excluded from consideration in this report.

Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Integer overflow / underflow
- Number rounding errors
- Reentrancy and cross-function vulnerabilities
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting

Methodology

1. Code review that includes the following
 1. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the smart contract.
 2. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
 3. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Quantstamp describe.
2. Testing and automated analysis that includes the following:
 1. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
 2. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarity, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, and actionable recommendations to help you take steps to secure your smart contracts.

Scope

Files Included

- `programs/restaking/src/*`

Files Excluded

- `tools/*`
- `.bak/*`

Key Actors And Their Capabilities

There are 4 different actors in the protocol

The Admin can execute the instructions:

- `admin_initialize_receipt_token_lock_authority()` - initializes the authority for the lock account.
- `admin_initialize_receipt_token_lock_account()` - initializes the lock account.
- `admin_initialize_fund_account()` - initializes the fund account.
- `admin_initialize_receipt_token_mint_authority()` - initializes the mint authority for the receipt token mint.
- `admin_initialize_receipt_token_mint_extra_account_meta_list()` - initializes the extra accounts for the receipt token mint.
- `admin_update_receipt_token_mint_extra_account_meta_list()` - updates the extra accounts for the receipt token mint.
- `admin_initialize_reward_account()` - initialize the reward account.
- `admin_update_reward_accounts_if_needed()` - re-allocate the size of the reward account.
- `operator_process_fund_withdrawal_job()` - starts processing pending withdrawals even if none of the applicable thresholds is reached.

The Fund Manager can execute the instructions:

- `fund_manager_update_sol_capacity_amount()` - Updates the capacity variable for SOL deposits.
- `fund_manager_update_supported_token_capacity_amount()` - Updates the capacity variable for supported tokens deposits.
- `fund_manager_update_withdrawal_enabled_flag()` - Updates the withdrawal boolean flag.
- `fund_manager_update_sol_withdrawal_fee_rate()` - Updates the withdrawal fee applied to SOL amounts.
- `fund_manager_update_batch_processing_threshold()` - Updates the amount threshold for batch processing.
- `fund_manager_initialize_supported_token_authority()` - initializes the authority for the support token account.
- `fund_manager_initialize_supported_token_account()` - initializes the support token account.
- `fund_manager_add_supported_token()` - adds a supported token to the protocol.
- `fund_manager_add_reward_pool_holder()` - adds a reward pool holder to the protocol.
- `fund_manager_add_reward_pool()` - adds a new reward pool in a reward account.
- `fund_manager_close_reward_pool()` - closes a reward pool in a reward account.
- `fund_manager_add_reward()` - adds a new reward object.
- `fund_manager_settle_reward()` - adds a new reward settlement.

The Operator can execute the instructions:

Note: Any signer can invoke operator-specific functions

The operator can invoke the following main functions:

- `operator_process_fund_withdrawal_job()` - starts processing pending withdrawals if any of the applicable thresholds are reached.
- `operator_update_prices()` - performs a token price update for supported tokens.
- `operator_update_reward_pools()` - calls the `update()` function on the global reward pool and on user reward pools.

The User can execute the instructions:

The user can invoke the following main functions:

- `user_initialize_receipt_token_account()` - initializes the `fragSol` account for the user.
- `user_initialize_fund_account()` - initializes the fund account of the user.
- `user_update_fund_account_if_needed()` - initializes the user fund account if needed.
- `user_deposit_sol()` - deposits SOL in the protocol.
- `user_request_withdrawal()` - request a withdrawal request.
- `user_cancel_withdrawal_request()` - cancels a previously created withdrawal request.
- `user_withdraw()` - withdraw processed funds in a withdrawal request.
- `user_deposit_supported_token()` - deposits a supported token in the protocol.
- `user_initialize_reward_account()` - initializes the reward account for the user.

- `user_update_reward_accounts_if_needed()` - re-allocate the size of user reward accounts.
- `user_update_reward_pools()` - performs an update for user reward pools.
- `user_claim_rewards()` - claim user rewards (**not implemented yet**).

Findings

FRA-1

Updating Price Before Withdrawal Can Cause Inaccuracies for the Withdrawal Amount

• High ⓘ

Fixed

✓ Update

Marked as "Fixed" by the client.

Addressed in: `65f4a86e6ec470bab893340784b6f1df3cdaa1e2`.

File(s) affected: `user_fund_context.rs`

Description: The protocol updates `fragSol` price with respect to `SOL` in order to maintain an up-to-date price. The price update is triggered through the following statement:

```
fund.update_token_prices(ctx.remaining_accounts)?;
```

The `update_token_prices()` first verifies that the `ctx.remaining_accounts` contain the address of a pricing source that is specified by the admin when they add a `supportedToken`. Then, it proceeds to get the current price of the asset by getting the spot price from either `Marinade pool` or `SPL Stake pool`.

When a user requests withdrawal, by calling the `request_withdrawal()` function, a withdrawal request is created and added to a batch. Then an operator needs to call the `process()` method to start processing the batch. During processing, the `fragSol` amount is converted into `SOL` and the total `SOL` amount to be withdrawn is reserved in the `fundAccount.withdrawalStatus.reserved_fund`.

Then, the user can call the `withdraw()` function to finally receive their `SOL` amount.

The issue is, the token price can still be updated between the user calling `request_withdrawal()`. This can happen if another user calls the `deposit_sol()` because it will trigger a price update. Or if an operator calls the `update_prices()` function. Currently, even the `withdraw()` function performs a price update. and `withdraw()`.

- Some users may not be able to withdraw because the subtraction from `sol_remaining` in `reserved_fund` could fail due to underflow.
- The calculation for the value of `SOL` per `fragSol` would be inaccurate since the amount of `SOL` could be greater than the amount of `SOL` subtracted from `sol_operation_reserved_amount` which is used in `assets_total_sol_value()`. This would result in an artificially inflated token value. The inflated token price would allow more `SOL` to be withdrawn per token than expected. If `SOL` is withdrawn more than what is subtracted from `sol_operation_reserved_amount`, then the `SOL` stored for the rent-exemption could be transferred out.

Recommendation: A possible solution is to keep track of the amount owed to the user when batch processing is done. And then during the `withdraw()` function, send that amount to the user instead of recalculating the `sol_amount`.

FRA-2

Closing a Rewardpool Will Block Deposits, Withdrawals, Updates, and Transfers

• High ⓘ

Fixed

✓ Update

Marked as "Fixed" by the client.

Addressed in: `d71f29ee6150349331749c6602e7ebc648467097`.

File(s) affected: `modules/reward/update.rs`

Description: A given `RewardAccount` can contain up to 4 pools of the type `RewardPool`. When `fragSol` moves from one user to another, or during minting and burning, the function `update_reward_pools_token_allocation` is invoked to update the reward pools in the given `rewardAccount`.

This happens in the following loop:

```

for reward_pool in self.get_related_pools(&from.user, receipt_token_mint)? {
  ...
  reward_pool.update(effective_deltas, current_slot)?;
  ...
}

```

However, if a `RewardPool` is closed, the `update()` function will throw an error:

```

fn update(
  &mut self,
  deltas: Vec<TokenAllocatedAmountDelta>,
  current_slot: u64,
) -> Result<Vec<TokenAllocatedAmountDelta>> {
  if self.is_closed() {
    err!(ErrorCode::RewardPoolClosedError)?
  }
  ...
}

```

Thus, closing a single `RewardPool` in any `RewardAccount` will prevent updating all the reward pools. And also all user reward pools.

Exploit Scenario:

The `RewardAccount` in this example has two `RewardPool`, and it closes the second one with ID `1`.

The `await restaking.runUserDepositSOL(user1, amount, null);` will fail, because it will attempt to update all the reward pools, and since one is closed, it will throw an error and will not continue the loop.

```

step("PoC for DoS", async function () {
  await new Promise(resolve => setTimeout(resolve, 10000));
  await Promise.all([
    restaking.tryAirdrop(user1.publicKey, 100),
    restaking.tryAirdrop(user2.publicKey, 100),
  ]);
  await restaking.sleep(1);
  const amount = new BN(10 * anchor.web3.LAMPORTS_PER_SOL);
  let ix = await restaking.methods.fundManagerCloseRewardPool(1).accounts({
    fundManager: user1.publicKey,
  }).signers([user1]).rpc();
  let tx = new anchor.web3.Transaction();
  await restaking.runUserDepositSOL(user1, amount, null); // <- fails
});

```

Recommendation: Consider returning from the reward pool's `update()` function if it is closed instead of throwing an error.

FRA-3

Signature Replay Attack Allows for Increased Rewards

• Medium ⓘ

Mitigated

✓ Update

Marked as "Fixed" by the client.

Addressed in: `65f4a86e6ec470bab893340784b6f1df3cdaa1e2`.

File(s) affected: `user_fund_context.rs`

Description: The protocol allows users to benefit from an accrual rate based on the wallet they use to deposit. The protocol relies on the on-chain `ed25519` program to verify signatures related to metadata that the user supplies. The metadata needs to be signed by the admin, and the user passes that signature to the `ed25519` program by placing the instruction before the `deposit`:

```

let current_ix_index: usize =
  instructions::load_current_index_checked(instructions_sysvar)?.into();

```

```
let ix = instructions::load_instruction_at_checked(current_ix_index - 1, instructions_sysvar)?;
require_eq!(ix.program_id, ed25519_program::ID);
```

The `ed25519 program` will verify that the admin private key signed the metadata, indicating that the metadata is safe to use.

However, there is no way to cancel a valid signature, this is because the signed data only includes the wallet name and the accrual rate related to it, but no information about an expiry time:

```
pub struct DepositMetadata {
pub wallet_provider: String,
pub contribution_accrual_rate: u8, // 100 is 1.0
}
```

There is also no way to verify that the user used the same wallet to deposit as the one specified in the `wallet_provider` unless the verification is done through the front end. However, users can still interact with the program directly, use whatever wallet they want with valid metadata, and get the accrual rate.

Recommendation: Add an expiry timestamp to the signed metadata, and perform a time-based check against it `verify_preceding_ed25519_instruction`. It is also recommended to add the address of the user getting the accrual rate to the signature and verifying it on-chain to prevent re-using of the signature in a short timeframe.

FRA-4 Denial of Service when Updating Pool Parameters

• Medium ⓘ Fixed

✓ Update

Marked as "Fixed" by the client.

Addressed in: `51b9c90e5c8a2a39862457636779153df2f70783`.

File(s) affected: `fund/update.rs`

Description: The function `set_capacity_amount()` allows the admin to set a new `capacity_amount` for the `SupportedTokenInfo` type. However, it currently compares between two state variables instead of comparing against the input provided:

```
if self.capacity_amount < self.accumulated_deposit_amount {
err!(ErrorCode::FundInvalidUpdateError)?
}
```

The same issue also exists in the function `set_sol_capacity_amount()`:

```
if self.sol_capacity_amount < self.sol_accumulated_deposit_amount {
err!(ErrorCode::FundInvalidUpdateError)?
}
```

It is also worth noting that the `sol_accumulated_deposit_amount` and `accumulated_deposit_amount` do not seem to be decreasing at any point. So it will always be needed to update the capacity; and if the admin mistakenly sets the capacity to a value lower than `accumulated_deposit_amount`, the admin will not be able to update the capacity again.

Recommendation: Compare against the input `capacity_amount` instead of `self.capacity_amount`:

```
if capacity_amount < self.accumulated_deposit_amount {
err!(ErrorCode::FundInvalidUpdateError)?
}
```

FRA-5 Users may Not Be Able to Withdraw

• Medium ⓘ Acknowledged

i Update

Marked as "Acknowledged" by the client.

The client provided the following explanation:

Restaking ecosystem is building at a rapid pace. We are developing a procedure to integrate with the Jito restaking protocol and staking pool programs to circulate all assets in a fund through SOL, LST and VRT to fulfill investment and withdrawal obligations. This feature will be included in the next release and will enable withdrawals and transfers.

File(s) affected: `modules/fund/withdraw.rs`

Description: The function `check_withdrawal_enabled()` checks for the boolean flag `withdrawal_enabled_flag` that can be set by the admin. The flag determines if users are allowed to call the `request_withdrawal()` function.

However, the flag is also checked during the final call to `withdraw()`, when users can withdraw the amount they requested after the batch is processed. This means that if the flag is set to true, users will not be able to call the function `withdraw()` even if they have an already processed amount they can claim.

Recommendation: Consider using a different flag for the `withdraw()` function, or remove this check if it is not needed.

FRA-6 Admin Can Set Arbitrary Withdrawal Fee Rate

• Low ⓘ

Acknowledged

i Update

Marked as "Acknowledged" by the client.

The client provided the following explanation:

We will mitigate this issue in next update by setting a hard limit (possibly like 5%).

File(s) affected: `/programs/restaking/src/modules/fund/update.rs`

Description: The admin has the ability to set a withdrawal fee rate. This value is stored in basis points, which means that a rate of `1` is equal to a 0.01% fee. There are no restrictions on the value the admin can set. This includes the ability to set withdrawal rates above 100% which would block withdrawals. In order to make users feel more secure and prevent the admin from disabling withdrawals via the rate, it is recommended to enforce a maximum allowed rate.

Recommendation: Consider creating a constant to enforce as the maximum allowed withdrawal fee rate. This value should be at most `10000`.

FRA-7 Missing input validations

• Low ⓘ

Acknowledged

i Update

Marked as "Acknowledged" by the client.

The client provided the following explanation:

1. will be elaborated when we substantially implements DeFi integrations. We don't have a finalized design for this yet.
2. acknowledged, but still following 'update_price' would prevent invalid configuration in this case.
3. wallet-provider is a kind of informational field only for off-chain usage. and custom contribution rate is basically sanitized in TokenAllocationRecord related codes.

File(s) affected: `modules/reward/update.rs`, `fund_manager_fund_supported_token_context.rs`, `user_fund_supported_token_context.rs`, `fund_manager_fund_context.rs`, `lib.rs`

Description: Description It is important to validate inputs, even if they only come from trusted addresses, to avoid human error:

1. In `update.rs`, it is possible to call `add_holder()` with: a) an empty vector `pubKeys`; b) a vector `pubKeys` containing duplicates;
2. In `fund_manager_fund_supported_token_context.rs`, we can have in `FundManagerFundSupportedTokenContext` the following equality: `receipt_token_mint == supported_token_mint`.
3. In `user_fund_supported_token_context.rs`, function `deposit_supported_token()` and in `user_fund_context.rs`, function `deposit_sol()`, the values of metadata (`wallet_provider` and `contribution_accrual_rate`) are not sanitized.
4. In `fund_manager_fund_context.rs`, we can use: a) in `update_sol_withdrawal_fee_rate()` any value for `sol_withdrawal_fee_rate`; b) in `update_batch_processing_threshold()` any value for `amount` and `duration`;
5. It is possible to execute the instruction `fund_manager_settle_reward()` for an `amount` of `0`, which can create a `RewardSettlementBlock` of `amount == 0`.

Recommendation: Consider adding the relevant checks.

FRA-8

Risk of out-of-bounds error in function

• Informational ⓘ

Acknowledged

`init_without_load()`

i Update

Marked as "Acknowledged" by the client.

The client provided the following explanation:

Acknowledged, and it is an intended panic in this case.

Also hope this issue to be considered as informational one, because this is the instruction only for the initialization/migration phase which only can be called by admin. And we believe such exception might be better to be implicit to hide unnecessary information.

File(s) affected: `zero_copy.rs`

Description: The functions `init_without_load()` and `bump()` access the value of `data` at a given `offset`. However, no check makes sure that `offset` is not an out-of-bound ID.

Recommendation: Consider checking the length of `data` before trying to access the value at id `offset`.

FRA-9

Impact of using `from_utf8_trim_null()` for objects `RewardPool`, `Holder` and `Reward`.

• Informational ⓘ

Fixed

i Update

Marked as "Acknowledged" by the client.

File(s) affected: `utils.rs`

Description: There are three issues related to using the function `from_utf8_trim_null()` for the objects `RewardPool`, `Holder` and `Reward`.

1. There is a mismatch between the code and the spec for the function `from_utf8_trim_null()`, where the comment says:

```
/// Truncates null (0x0000) at the end.
```

But the function removes all `null` in `v`, not only at the end.

2. When objects mentioned above are initialized, the value of `name` or `description` is saved without being trimmed. However, what is returned by the functions `name()` and `description()` is trimmed. As a result, trimming is done `n` times instead of only once at initialization. Also, when calling `add_reward_pool()`, `add_holder()`, and `add_reward()`, the check of an existing item with the same name can be bypassed by using a name with added `null` chars. The values will differ, but once the object is added, the function `name()` will return the same value (trimmed) for both objects.
3. These objects can be created with an empty name or description.

Recommendation: Consider:

1. Aligning the code and the documentation with what is expected for the function `from_utf8_trim_null()`;
2. Trimming the name and description before checking for similar existing objects, and initializing these objects with trimmed strings.
3. Clarifying if it is allowed to create these objects with an empty name or description.

FRA-10

Underflow Error Can Happen without Explicit Error if a User Executes a Transaction with `user_deposit_sol()` or `user_deposit_supported_token()` as the First Instruction with `metadata`

• Informational ⓘ

Acknowledged

i Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
Acknowledged, and intended panic in this case. But we will elaborate the UX by adding an explicit exception in the future release.
```

File(s) affected: `user_fund_context.rs`

Description: An underflow error can happen without explicit error if a user executes a transaction with `user_deposit_sol()` or `user_deposit_supported_token()` as the first instruction with `metadata`. In this case, the variable `current_ix_index` in the function `verify_preceding_instruction()` will take the value `0`, and underflow will happen at the line:

```
let ix = instructions::load_instruction_at_checked(current_ix_index - 1, instructions_sysvar)?;
```

Recommendation: Consider returning an explicit error when `current_ix_index == 0` in the function `verify_preceding_instruction()`.

FRA-11

The Closure in `check_valid_addition()` Checks Instead if the Addition Is Invalid

• Informational ⓘ

Acknowledged

i Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
Fixed in a branch for next phase.
```

File(s) affected: `update.rs`

Description: The closure `is_contribution_accrual_rate_valid` currently checks if the rate is invalid (i.e., not in the range `100..200`), so the closure name is misleading.

Recommendation: Consider either:

1. renaming the closure to `is_contribution_accrual_rate_invalid` if you want to preserve the current logic;
2. changing the logic to check if the rate is valid by removing the `!`;

FRA-12

It Is Possible to Update `UserRewardPool` without Updating `RewardPool`

• Informational ⓘ

Acknowledged

i Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
It is intended behavior, just for the synchronization of user pool contribution amount based on last contribution accrual rate of the user without touching global pool just for off-chain use cases. Activities like user's balance update should be followed with global/user reward accounts, but just for the synchronization global pool doesn't need to be updated.
```

File(s) affected: `user_reward_context.rs`

Description: In all but one case, the global `RewardPool` is always updated alongside the `UserRewardPool`. During deposits and withdrawals for instance.

However, there is one case where a `UserRewardPool` will be updated without updating the `RewardPool` it is part of, namely this is the `update_user_reward_pools()` function that is callable by any user.

There are no specific issues that are related to this case, but it might cause issues down the line if some future functionality expects the `UserRewardPool` and `RewardPool` to be updated together.

Recommendation: Consider removing this instruction altogether. Users will be able to update their reward pools during reward claiming, or through any interaction with the pool.

FRA-13 Immediate Withdrawals Are Possible

• Informational ⓘ

Acknowledged

i Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
Acknowledged, and we believe such fast withdrawal still demands withdrawal fee. And there is no reason to prevent that in current implementation. But this procedure will be refined naturally in next release with the feature managing full circulation of fund assets.
```

Description: The protocol implements a withdrawal flow, where a user has to first request a withdrawal, through the `request_withdrawal()`, and then it has to be processed along with other withdrawal requests, if any.

To process withdrawal requests, the operator needs to either wait for a pre-determined amount of time between each batch (`batch_processing_threshold_duration`), or a threshold amount (`batch_processing_threshold_amount`).

This means that a user with a high stake, that is greater than `batch_processing_threshold_amount` can deposit and withdraw instantly.

Recommendation: Consider having withdrawal conditions that are not related to the amount to be withdrawn in a batch.

FRA-14

Token Allocation Are not Updated During Transfers

• Informational ⓘ

Acknowledged

i Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

```
It is implemented and tested but commented out in this release. Why transfer is disabled is like below.  
At the beginning of the launch, we believed that fragSOL's lack of liquidity in De-Fi could lead to de-pegging issues, regardless of the actual fund's underlying asset reserve. After securing sufficient liquidity with the Phase1 launch, we are planning to integrate the reward system into De-Fi, including DEX and lending protocols, and enable token transfers in the next release.
```

File(s) affected: `user_receipt_token_transfer_context.rs`

Description: The protocol tracks `fragSol` transfers using a custom transfer hook. Currently, the feature is disabled, and transfers are not allowed. However, the transfer hook does not perform an allocation update for the source and the destination.

This means if transfers were to be enabled, specifically in the function `UserReceiptTokenTransferContext.handle_transfer()` the balances and rewards of the source and destination will remain the same.

Recommendation: Before enabling transfers, it is critical to implement update logic in the `handle_transfer()`

FRA-15

No Functionality To Withdraw Deposited Assets

• Informational ⓘ

Acknowledged

i Update

Marked as "Acknowledged" by the client.

Description: The protocol allows the deposit of a set of supported assets, or simply SOL, in exchange for fragSol. When users want to withdraw, they will get an amount of SOL that is equivalent to the fragSol value at the time of withdrawal.

However, **there is no functionality for the admin to withdraw the supported tokens deposited by the users.** The protocol will not allow withdrawal at its first stage, which mitigates the risk that users might withdraw all the SOL deposited and leave the other supported tokens.

Recommendation: Consider adding a privileged instruction that allows the admin to withdraw supported tokens/SOL.

FRA-16 Data Used for Events Should Be Clarified

• Informational ⓘ

Acknowledged

i Update

Marked as "Acknowledged" by the client.

The client provided the following explanation:

```
Fixed in a branch for next phase. And the fix is not included in current release as it is just for off-chain usage.
```

File(s) affected: `user_fund_supported_token_context.rs`, `user_fund_context.rs`, `user_receipt_token_transfer_context.rs`, `operator_fund_context.rs`

Description: 1. In `user_fund_supported_token_context.rs`, function `deposit_supported_token()`, when emitting the event `UserDepositedSupportedTokenToFund`, it should be clarified if `fund_account` should be created with the values of `receipt_token_price` and `receipt_token_total_supply` before or after the deposit is recorded.

2. In `user_fund_context.rs`, function `deposit_sol()`, when emitting the event `UserDepositedSOLToFund`, it should be clarified if `fund_account` should be created with the values of `receipt_token_price` and `receipt_token_total_supply` before or after the deposit is recorded.

3. In `user_receipt_token_transfer_context.rs`, function `handle_transfer()`, when emitting the event `UserTransferredReceiptToken`, it should be clarified if `source_fund_account` and `destination_fund_account` should use placeholders created with the address of the funds, or with the address of the underlying users.

4. In `operator_fund_context.rs`, function `process_fund_withdrawal_job()`, when emitting the event `OperatorProcessedJob`, it should be clarified if `fund_account` should be created with the values of `receipt_token_total_supply` before or after the withdrawal is recorded.

Recommendation: Consider clarifying what data should be used and, if required, updating the code accordingly.

FRA-17

If All Lamports Are Withdrawn From `fund_account`, the Account Data Could Be Deleted

• Undetermined ⓘ

Acknowledged

i Update

Marked as "Acknowledged" by the client.

The client provided the following explanation:

```
We will add strict invariants to related codes in next update to fundamentally prevent possible human fault or bugs causing misaccount.
```

File(s) affected: `user_fund_context.rs`

Description: In `user_fund_context.rs` and the function `withdraw()`, if accounting discrepancies occur, no check prevents withdrawing all lamports from `fund_account`, which would result in the deletion of the account's data.

Recommendation: Consider making sure that it is not possible to withdraw all lamports from `fund_account`.

Auditor Suggestions

S1 Improve Error Handling

Acknowledged

i Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

Will be applied in a branch for next phase.

File(s) affected: `/programs/restaking/src/utils.rs` , `/programs/restaking/src/modules/reward/update.rs`

Description: Avoid the use of `unwrap()` and use custom errors or `?` to propagate errors. There are four uses of `unwrap()` in the codebase but only one has comments documenting its safety.

Functions using `unwrap()` :

1. `utils.rs::timestamp_now()`
2. `RewardAccount::get_related_pools()`
3. `TokenAllocatedAmount::add()`
4. `TokenAllocatedAmount::subtract()`

File(s)

- `/programs/restaking/src/utils.rs`
- `/programs/restaking/src/modules/reward/update.rs`

Recommendation: Consider throwing more descriptive errors where applicable.

S2 Incorrect Variable Naming

Acknowledged

i Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

Will be applied in a branch for next phase.

File(s) affected: `/programs/restaking/src/instructions/user_fund_supported_token_context.rs`

Description: In `UserFundSupportedTokenContext` , the name for the variable that holds the `instructions_sysvar` account is `instruction_sysvar` instead of `instructions_sysvar` .

Recommendation: Consider changing the name of the variable to match the spelling in `UserFundContext` .

S3 Documentation Issues

Acknowledged

i Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

Will be applied in a branch for next phase.

Description: 1. In `update.rs` , function `update_reward_pools_token_allocation()` , the intent of the following `if` condition could be clarified by adding parenthesis:

```
if from.is_none() && to.is_none() || to.is_none() && contribution_accrual_rate.is_some() {}
```

Recommendation: Consider addressing the items listed above.

S4 Enforce the Least Privilege Principle in `operator_fund_context.rs`

Acknowledged

i Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

Applied in a branch for next phase.

Description: In `operator_fund_context.rs`, the context `OperatorFundContext` involves three mutable accounts: `receipt_token_mint`, `receipt_token_lock_account`, and `fund_account`. However, the mutability of these three accounts is not required in the implemented functions `process_fund_withdrawal_job` and `update_prices`.

Recommendation: Consider using two different contexts, each being a more restricted and adapted context for each implemented function.

S5 Code Conciseness

Acknowledged

i Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

This instruction is for the purpose of tracing from off-chain.

Description: Description

1. The purpose of keeping the instruction `operator_log_message()` is unclear. Should this function be used in production?
2. In `operator_reward_context.rs`, struct `OperatorRewardContext`, using `#[account(mut)]` for `operator: Signer<'info>` and `pub system_program: Program<'info, System>` seems not necessary, since no lamport transfer is expected for `operator`, except when paying for the instruction.

Recommendation: Consider addressing the items listed above.

S6 Suggestion to Add Invariants to the System

Acknowledged

i Update

Marked as "Acknowledged" by the client.
The client provided the following explanation:

Will be applied in a branch for next phase.

File(s) affected: `user_reward_settlement.rs`, `user_reward_account.rs`, `reward_account.rs`, `reward_settlement.rs`, `spl_stake_pools.rs`, `fund_withdrawal_job.rs`

Description: 1. In `user_reward_settlement.rs`, function `UserRewardSettlement.update_settled_slot()`, a check could make sure that `self.settled_slot < settled_slot` if that storage value is expected to monotonically increase.

2. In `user_reward_account.rs`, function `UserRewardPool.add_contribution()`, a check could make sure that `self.updated_slot < current_slot` if that storage value is expected to monotonically increase.
3. In `reward_account.rs`, function `RewardPool.add_contribution()`, a check could make sure that `self.updated_slot <= current_slot` if that storage value is expected to monotonically increase.
4. In `reward_settlement.rs`, function `RewardSettlement.add_contribution()`, a check could make sure that `self.settlement_blocks_last_slot <= current_slot` if that storage value is expected to monotonically increase.
5. In `spl_stake_pools.rs`, function `calculate_lamports_from_pool_tokens()`, a check could make sure that `pool_tokens` is not greater than `self.pool_token_supply`.
6. In `fund_withdrawal_job.rs`, function `process()`, a check could make sure that `total_sol_value_in_fund` is not `0` to prevent burning receipt tokens for no underlying tokens.

S7 Side Effect Due to Fixing FRA-1

Acknowledged

i Update

The team is aware of this issue and will separate the reserved funds using PDAs in the next iteration.

File(s) affected: `user_fund_context.rs`

Description: Due to the fix implemented to `FRA-1`, which allows users to get a portion of `SOL` from the reserved fund based on the amount of `fragSol` they want to withdraw, it is now possible for users to get a more favorable rate if they wait for other users to perform a withdrawal. That is because, when new users perform a withdrawal, they might do it at a higher price for `fragSol`, so the old users would get the average price between the old and new price.

Definitions

- **High severity** – High-severity issues usually put a large number of users' sensitive information at risk, or are reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
- **Medium severity** – Medium-severity issues tend to put a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or are reasonably likely to lead to moderate financial impact.
- **Low severity** – The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.
- **Informational** – The issue does not post an immediate risk, but is relevant to security best practices or Defence in Depth.
- **Undetermined** – The impact of the issue is uncertain.
- **Fixed** – Adjusted program implementation, requirements or constraints to eliminate the risk.
- **Mitigated** – Implemented actions to minimize the impact or likelihood of the risk.
- **Acknowledged** – The issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).

Appendix

File Signatures

The following are the SHA-256 hashes of the reviewed files. A file with a different SHA-256 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different SHA-256 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review.

Files

- `a7f...248 ./instructions/admin_fund_context.rs`
- `fb9...948 ./instructions/admin_receipt_token_mint_context.rs`
- `08a...2cf ./instructions/admin_reward_context.rs`
- `72d...bc7 ./instructions/fund_manager_fund_context.rs`
- `4f8...bf4 ./instructions/fund_manager_fund_supported_token_context.rs`
- `0c7...980 ./instructions/fund_manager_reward_context.rs`
- `1ae...db1 ./instructions/mod.rs`
- `60e...eb9 ./instructions/operator_empty_context.rs`
- `71c...324 ./instructions/operator_fund_context.rs`
- `381...dab ./instructions/operator_reward_context.rs`
- `b6b...a99 ./instructions/user_fund_context.rs`
- `441...b84 ./instructions/user_fund_supported_token_context.rs`
- `001...d33 ./instructions/user_receipt_token_transfer_context.rs`
- `4e7...f0d ./instructions/user_reward_context.rs`

Tests

- `61d...1ba ./restaking/1_initialize.ts`
- `871...b5f ./restaking/2_deposit_sol.ts`
- `c10...5d8 ./restaking/3_deposit_token.ts`
- `5c6...343 ./restaking/4_withdraw.ts`
- `3fc...659 ./restaking/5_transfer_hook.ts`
- `a28...0cb ./restaking/6_reward.ts`

Automated Analysis

N/A

Test Suite Results

To run tests, the following command is used:

```
anchor test --detach -p restaking
```

```
[9:55:49 PM] [keychain] loaded local wallet
[9:55:49 PM] [keychain] WALLET                               GiDkDCZjVC8Nk1Fd457qGSV2g3MQX62n7cV5CvgFyGfF
[9:55:49 PM] [keychain] loading restaking program keypairs
[9:55:49 PM] [keychain] ledger keypairs (0):
[9:55:49 PM] [keychain] local keypairs (15):      PROGRAM, FRAGSOL_MINT, ADMIN, FUND_MANAGER,
MOCK_ALL_MINT_AUTHORITY, MOCK_USER1, MOCK_USER2, MOCK_USER3, MOCK_USER4, MOCK_USER5, MOCK_USER6,
MOCK_USER7, MOCK_USER8, MOCK_USER9, MOCK_USER10
[9:55:49 PM] [keychain] applying keypairs to restaking program workspace:
[9:55:49 PM] [keychain] checking /home/mostafa/fragmetric-
contracts/programs/restaking/src/constants/local.rs
[9:55:49 PM] [keychain] checking /home/mostafa/fragmetric-contracts/target/deploy/restaking-
keypair.json
[9:55:49 PM] [keychain] loaded restaking program keypairs' pubkey:
[9:55:49 PM] [keychain] PROGRAM                       4qEHCzsLFUnw8jmhmRSmAK5VhZVoSD1iVqukAf92yHi5
[9:55:49 PM] [keychain] FRAGSOL_MINT                   Cs29UiPhAkM2v8fZW7qCJ1UjhF1UAhgrsKj61yGGYizD
[9:55:49 PM] [keychain] ADMIN                           9b2RSMdYskVvjVbwF4cVwEhZUaaaUgyYSxvESmnoS4LL
[9:55:49 PM] [keychain] FUND_MANAGER                   5FjrErTQ9P1ThYVdY9RamrPUCQGTMCcczUjH21iKzbx
[9:55:49 PM] [keychain] MOCK_ALL_MINT_AUTHORITY       24z2hejEqmQGpPKU3q2xZe1ZuAzPsNeEU55KT3k629e6
[9:55:49 PM] [keychain] MOCK_USER1                    24z2hejEqmQGpPKU3q2xZe1ZuAzPsNeEU55KT3k629e6
[9:55:49 PM] [keychain] MOCK_USER2                    3VPkgde6n22TAD5w69yZbqGJ8ELGdSt7K2kSUjvGYWnR
[9:55:49 PM] [keychain] MOCK_USER3                    E48eqXgsHCSF9MkNvXZ3krHbcBjtsfw95a91hbzenUzv
[9:55:49 PM] [keychain] MOCK_USER4                    4zFAD5DEJtteKEeHpRYghwopiS4cJuC2wxA998nLaxgN
[9:55:49 PM] [keychain] MOCK_USER5                    HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:55:49 PM] [keychain] MOCK_USER6                    71TKdMbS3vwQH8WxVmrpZ1JZSXdixyScdSwwawDCAj9C
[9:55:49 PM] [keychain] MOCK_USER7                    A5jsUAujiuoW8Lc5pb6R7XYrD5HH2gTBpMZDCpDMqxcS
[9:55:49 PM] [keychain] MOCK_USER8                    6tqUdVfNE9SUuipcFiKQBZjyqNa99gc4KSC4CLEZShcQ
[9:55:49 PM] [keychain] MOCK_USER9                    2UhB1hD8ihBaxAQRf48rWXPx6g3EFvEnmeMiour6777
[9:55:49 PM] [keychain] MOCK_USER10                   2PLskyDxJ4ZpPccrjHQFh3V9aPpu5JtvjvJwdobqr8Zh
[9:55:49 PM] [anchor] initializing restaking playground
[9:55:49 PM] [anchor] connected to:
http://0.0.0.0:8899
[9:55:49 PM] [anchor] loaded program restaking:
4qEHCzsLFUnw8jmhmRSmAK5VhZVoSD1iVqukAf92yHi5

[9:55:49 PM] [restaking] fragSOL metadata file:
> https://quicknode.quicknode-ipfs.com/ipfs/QmcueajXkNzoYRhcCv323PMC8VVGiDvXaaVXkMyYcyUSRw
> {"name":"Fragmetric Restaked SOL","symbol":"fragSOL","description":"fragSOL is Solana's first
native LRT that provides optimized restaking rewards.","image":"https://fragmetric-assets.s3.ap-
northeast-2.amazonaws.com/fragsol.png"}
[9:55:49 PM] [anchor] ADMIN (signer)
9b2RSMdYskVvjVbwF4cVwEhZUaaaUgyYSxvESmnoS4LL
[9:55:49 PM] [anchor] FRAGSOL_MINT (signer)
Cs29UiPhAkM2v8fZW7qCJ1UjhF1UAhgrsKj61yGGYizD
[9:55:49 PM] [anchor] transaction confirmed:
2twxKrjeZqbFjVsrS4WKgCoS37dERKdNpX7fgCkC ...
[9:55:49 PM] [restaking] fragSOL token mint created with extensions
Cs29UiPhAkM2v8fZW7qCJ1UjhF1UAhgrsKj61yGGYizD
  ✓ create fragSOL token mint with extensions (101ms)
  ✓ mock supported token mints
[9:55:49 PM] [anchor] ADMIN (signer)
9b2RSMdYskVvjVbwF4cVwEhZUaaaUgyYSxvESmnoS4LL
[9:55:50 PM] [anchor] transaction confirmed:
2SHQyEiizcQ57E6y9X1yzRb6Wird7bBx8cBYTSxC ...
[9:55:50 PM] [restaking] fragSOL fund account created
```


7xraTDZ4QWgvgJ5SCZp4hyJN2XEFyGRySQjdG49iZfU8
✓ initialize fund accounts (398ms)
[9:55:50 PM] [restaking] running batched instructions 35/35
[9:55:50 PM] [anchor] ADMIN (signer)
9b2RSMdySkVvjVbwF4cVwEhZUaaaUgyYSxvESmnoS4LL
[9:55:50 PM] [anchor] transaction confirmed:
5dtousWj86PYeJsdSbW4d6Jg3Cs4Jxm283AiB7fh ...
[9:55:50 PM] [restaking] updated reward account version from=0, to=34, target=34
EujaAdDdHVBbSYdyX85TjRzdkxdEvoJYcd1s2bNNB6Xs
✓ initialize reward accounts (422ms)
[9:55:50 PM] [anchor] ADMIN (signer)
9b2RSMdySkVvjVbwF4cVwEhZUaaaUgyYSxvESmnoS4LL
[9:55:50 PM] [anchor] transaction confirmed:
2fSAb7AsYJ35rRHHvubq9Sq35sVMB7bkXaaa1T9dq ...
[9:55:50 PM] [restaking] transferred fragSOL mint authority to the PDA
At8Lu3kHBPxHCHYxmcG8r9X34ehHsYVRph6iLjNSMLEE
✓ transfer token mint authority to PDA (386ms)
[9:55:50 PM] [anchor] FUND_MANAGER (signer)
5FjrErTQ9P1ThYVdY9RamrPUCQGTMCcczUjH21iKzbxw
[9:55:51 PM] [anchor] transaction confirmed:
3deaTtfec2XZXuLgU27Ra1NS7tJmLf2AUdECoeue ...
[9:55:51 PM] [restaking] configured fragSOL supported tokens
7xraTDZ4QWgvgJ5SCZp4hyJN2XEFyGRySQjdG49iZfU8
✓ initialize fund and supported tokens configuration (412ms)
[9:55:51 PM] [anchor] FUND_MANAGER (signer)
5FjrErTQ9P1ThYVdY9RamrPUCQGTMCcczUjH21iKzbxw
[9:55:51 PM] [anchor] transaction confirmed:
2WwY9iL7ZQgKxooJiiNJGgkfQtexPycoyXPjD8ZV ...
[9:55:51 PM] [restaking] configured fragSOL reward pools and reward
EujaAdDdHVBbSYdyX85TjRzdkxdEvoJYcd1s2bNNB6Xs
✓ initialize reward pools and rewards (412ms)
[9:55:52 PM] [anchor] FUND_MANAGER (signer)
5FjrErTQ9P1ThYVdY9RamrPUCQGTMCcczUjH21iKzbxw
[9:55:52 PM] [anchor] transaction confirmed:
2t2vUK5ovdL99YgNw4Ey7GMX9LzPfkXnLcLkH9eQ ...
[9:55:52 PM] [restaking] settled fragSOL reward to pool=1/bonus, rewardId=0/fPoint, amount=0
(decimals=4)
✓ settle fPoint reward (zeroing) (1206ms)
[9:55:53 PM] [anchor] SOL airdropped (+100): 100.000000000 SOL
3VPkgde6n22TAD5w69yZbqGJ8ELGdSt7K2kSUjvGYWnR
[9:55:53 PM] [anchor] SOL airdropped (+100): 100.000000000 SOL
24z2hejEqmQGpPKU3q2xZe1ZuAzPsNeEU55KT3k629e6
[9:55:53 PM] [anchor] slept for 1 slots, started=12, ended=13, requested=13
✓ try airdrop SOL to mock accounts (995ms)
[9:55:54 PM] [anchor] transaction confirmed:
4gs4WefeHgTdtQ6FPZZjXxa9BrKu3SGmUtCi8UAP ...
[9:55:54 PM] [restaking] operator updated prices: 1.000000000 SOL/fragSOL
[9:55:54 PM] [anchor] transaction confirmed:
56Vm4tXAdDiaKcxhf4vysib9owrtLnCyp3RcyCb3 ...
[9:55:54 PM] [restaking] user deposited: 10.000000000 SOL
24z2hejEqmQGpPKU3q2xZe1ZuAzPsNeEU55KT3k629e6
[9:55:54 PM] [restaking] user fragSOL balance: 10.000000000 fragSOL
24z2hejEqmQGpPKU3q2xZe1ZuAzPsNeEU55KT3k629e6
[9:55:54 PM] [anchor] transaction confirmed:
58fhWmZnB3Z3xSw1iyn622G2sD6zbhcfabTKzSSZ ...
[9:55:54 PM] [restaking] operator updated prices: 1.000000000 SOL/fragSOL
✓ user1 deposits SOL without metadata to mint fragSOL (1024ms)
[9:55:55 PM] [anchor] transaction confirmed:
4bfmkWSg3sqYYtPoxiZPoSZ1vqfLGUweyTkPP8GW ...
[9:55:55 PM] [restaking] operator updated prices: 1.000000000 SOL/fragSOL
[9:55:55 PM] [anchor] transaction confirmed:
3Ko46PeY7cKzeYfDTshNW7BtN3gfaP6uFpxvs2oU ...
[9:55:55 PM] [restaking] user deposited: 6.000000000 SOL
3VPkgde6n22TAD5w69yZbqGJ8ELGdSt7K2kSUjvGYWnR
[9:55:55 PM] [restaking] user fragSOL balance: 6.000000000 fragSOL
3VPkgde6n22TAD5w69yZbqGJ8ELGdSt7K2kSUjvGYWnR
[9:55:56 PM] [anchor] transaction confirmed:
9QRqAyUo2UMERqhpcyjotRUMPrunx6wog7PDXPxd ...

```
[9:55:56 PM] [restaking] user deposited: 4.000000000 SOL
3VPkgde6n22TAD5w69yZbqGJ8ELGdSt7K2kSUjvGYWnR
[9:55:56 PM] [restaking] user fragSOL balance: 10.000000000 fragSOL
3VPkgde6n22TAD5w69yZbqGJ8ELGdSt7K2kSUjvGYWnR
  ✓ user2 deposits SOL with metadata to mint fragSOL (1228ms)
[9:55:56 PM] [anchor] transaction failed
5y3iHb9JZpD3kxkUfJ1k87Zx4rQYiYqJXH5opXoR ...
  ✓ user2 cannot cheat metadata
[9:55:56 PM] [anchor] SOL airdropped (+100): 100.000000000 SOL
E48eqXgsHCSF9MkNvXZ3krHbcBjtsfw95a91hbzenUzv
[9:55:56 PM] [anchor] SOL airdropped (+100): 100.000000000 SOL
4zFAD5DEJtteKEEhPryghwopiS4cJuC2wxA998nLaxgN
[9:55:56 PM] [anchor] slept for 1 slots, started=19, ended=20, requested=20
[9:55:57 PM] [anchor] MOCK_ALL_MINT_AUTHORITY (signer)
24z2hejEqmQGpPKU3q2xZe1ZuAzPsNeEU55KT3k629e6
[9:55:57 PM] [anchor] MOCK_ALL_MINT_AUTHORITY (signer)
24z2hejEqmQGpPKU3q2xZe1ZuAzPsNeEU55KT3k629e6
[9:55:57 PM] [anchor] transaction confirmed:
3sixFuNipKefgRtuncNQbaKMyqtvUVCawPpdSSmR ...
[9:55:57 PM] [anchor] transaction confirmed:
WlrayUBFxZ6RfjueE24WHw2o42YySHrLoCRBF2rK ...
[9:55:57 PM] [restaking] bSOL airdropped (+100000): 100000.000000000 bSOL
BNUfZiKtDjoV7g1HQpwo46FDnao5qVAJkpAuwUqJTy5W
[9:55:57 PM] [restaking] bSOL airdropped (+100000): 100000.000000000 bSOL
AyejRb7s9rVFGYx6E7jTHFZyywXMDakodCR5X7uSuGBN
[9:55:57 PM] [restaking] jitoSOL airdropped (+100000): 100000.000000000 jitoSOL
A4dK4AatPEtgKgg61jpeSsbQ6vzcPxiCoDLMZ5h6MGR9
[9:55:57 PM] [restaking] jitoSOL airdropped (+100000): 100000.000000000 jitoSOL
8npF2w7J2nu6jAP8ajxi1jDt7yjdWfdQi9SBtzQJq1uq
[9:55:57 PM] [restaking] mSOL airdropped (+100000): 100000.000000000 mSOL
8JqX8YRrnnMxoQo1QefDQYeQWkFpiQDbSAW2nYLREwjT
[9:55:57 PM] [restaking] mSOL airdropped (+100000): 100000.000000000 mSOL
DYmY6Shdin4UERpfdkzSoFcJaTp8aYeQdeYqztzLYXuR
[9:55:57 PM] [anchor] slept for 1 slots, started=22, ended=23, requested=23
  ✓ try airdrop SOL and supported tokens to mock accounts (1604ms)
[9:55:58 PM] [anchor] transaction confirmed:
4vFKWqSXajbYesCkSX8ZQHhtGisjxyakjwjrHAWG ...
[9:55:58 PM] [restaking] operator updated prices: 1.000000000 SOL/fragSOL
[9:55:58 PM] [anchor] transaction confirmed:
2t9YsiaCgdTXKgbwvTEYEHpMTnu87C7Ff55kbehh ...
[9:55:58 PM] [restaking] user deposited: 10.000000000 bSOL
AyejRb7s9rVFGYx6E7jTHFZyywXMDakodCR5X7uSuGBN
[9:55:58 PM] [restaking] user fragSOL balance: 11.643906210 fragSOL
E48eqXgsHCSF9MkNvXZ3krHbcBjtsfw95a91hbzenUzv
  ✓ user3 deposits supported token without metadata to mint fragSOL (823ms)
[9:55:58 PM] [anchor] transaction failed
rP3DATqXxfVKxN26skEokPZiXcdKX5sLNLkaYjv2 ...
  ✓ user3 fails to deposit too many tokens
[9:55:59 PM] [anchor] transaction confirmed:
38cRTvQiTiVjvDmzLcj8p9Nub7LiZADX2y53RrQH ...
[9:55:59 PM] [restaking] operator updated prices: 1.000000000 SOL/fragSOL
[9:55:59 PM] [anchor] transaction confirmed:
uJYfBe1izWt4iCPVyrLSffQV6AZwESj6N2Bd3cXu ...
[9:55:59 PM] [restaking] user deposited: 6.000000000 bSOL
BNUfZiKtDjoV7g1HQpwo46FDnao5qVAJkpAuwUqJTy5W
[9:55:59 PM] [restaking] user fragSOL balance: 6.986343726 fragSOL
4zFAD5DEJtteKEEhPryghwopiS4cJuC2wxA998nLaxgN
[9:55:59 PM] [anchor] transaction confirmed:
4a2GJPuKwWbW74bVmFbKJDkmau3q1HuUjqdF3mgz ...
[9:55:59 PM] [restaking] user deposited: 4.000000000 bSOL
BNUfZiKtDjoV7g1HQpwo46FDnao5qVAJkpAuwUqJTy5W
[9:55:59 PM] [restaking] user fragSOL balance: 11.643906210 fragSOL
4zFAD5DEJtteKEEhPryghwopiS4cJuC2wxA998nLaxgN
  ✓ user4 deposits supported token with metadata to mint fragSOL (1216ms)
[9:56:00 PM] [anchor] SOL airdropped (+100): 100.000000000 SOL
71TKdMbS3vwQH8WxVmrpZ1JZSXdixyScdSwwawDCAj9C
[9:56:00 PM] [anchor] SOL airdropped (+100): 100.000000000 SOL
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
```

```
[9:56:00 PM] [anchor] slept for 1 slots, started=29, ended=30, requested=30
  ✓ try airdrop SOL to mock accounts (977ms)
[9:56:01 PM] [anchor] transaction confirmed:
5BC1oJxFgcrDwbznKqLaoD5DJkzKcwEkX2mxmBE3 ...
[9:56:01 PM] [restaking] operator updated prices: 1.000000000 SOL/fragSOL
[9:56:01 PM] [anchor] transaction confirmed:
31uaNSomQCSpXQkL3FvXrr2NASCrenkwtVBggWNz ...
[9:56:01 PM] [restaking] user deposited: 20.000000000 SOL
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:01 PM] [restaking] user fragSOL balance: 20.000000000 fragSOL
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:01 PM] [anchor] slept for 0 slots, started=32, ended=32, requested=32
[9:56:01 PM] [anchor] transaction confirmed:
4wrf1uVkkfv6hGR5Ghm5VA1JQTGtsA75oixmgayd ...
[9:56:01 PM] [restaking] user requested withdrawal: 4.000000000 fragSOL #1/1
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:01 PM] [restaking] user fragSOL balance: 16.000000000 fragSOL
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:01 PM] [anchor] slept for 1 slots, started=32, ended=33, requested=33
[9:56:02 PM] [anchor] transaction confirmed:
2QKQTkLkRecjyPHdcccqTGBYDXZqZMhPzChveEvv ...
[9:56:02 PM] [restaking] user requested withdrawal: 4.000000000 fragSOL #2/1
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:02 PM] [restaking] user fragSOL balance: 12.000000000 fragSOL
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:02 PM] [anchor] slept for 2 slots, started=32, ended=34, requested=34
[9:56:02 PM] [anchor] transaction confirmed:
4WDkfn7YGeNcSRR3zjvfANPTRGMGm11wQE2fwn6T ...
[9:56:02 PM] [restaking] user requested withdrawal: 4.000000000 fragSOL #3/1
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:02 PM] [restaking] user fragSOL balance: 8.000000000 fragSOL
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:02 PM] [anchor] slept for 3 slots, started=32, ended=35, requested=35
[9:56:03 PM] [anchor] transaction confirmed:
2uxkTAueschLzh1LVvvRgACpUZTQUdAf8HMKghm ...
[9:56:03 PM] [restaking] user requested withdrawal: 4.000000000 fragSOL #4/1
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:03 PM] [restaking] user fragSOL balance: 4.000000000 fragSOL
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:03 PM] [anchor] transaction confirmed:
21S3WHU6m84q9Jw27UQgxhch6fxVARv45kK2kXmF ...
[9:56:03 PM] [restaking] operator updated prices: 1.000000000 SOL/fragSOL
  ✓ user5 deposits and withdraws (2648ms)
[9:56:03 PM] [anchor] transaction confirmed:
4MAc59CK9fhN7ag4958rxLupbiYKv9DkgpkBEgB3 ...
[9:56:03 PM] [restaking] operator updated prices: 1.000000000 SOL/fragSOL
[9:56:03 PM] [anchor] transaction failed
57EBMUAJ5Wgn1o6Y7QAe6FFZNdRKP1pt2RXZjmFW ...
[9:56:04 PM] [anchor] transaction confirmed:
2pgGHBp3FHHwpxVdfs6oD57jmYyyKEXRSE9uo6NV ...
[9:56:04 PM] [restaking] user canceled withdrawal request: #1
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:04 PM] [restaking] user fragSOL balance: 8.000000000 fragSOL
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:04 PM] [anchor] transaction confirmed:
2N2peUnGfA3VvgveSKLzmXicviiThTBfhr6wDn22 ...
[9:56:04 PM] [restaking] user canceled withdrawal request: #3
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:04 PM] [restaking] user fragSOL balance: 12.000000000 fragSOL
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:04 PM] [anchor] transaction failed
4zsBZqnEDFmYXyoaoFKY4CQ7C6Z1ghZProunJYJu ...
  ✓ user5 cancels withdrawal request (1239ms)
[9:56:05 PM] [anchor] transaction confirmed:
6ZDYLeUsDcbahfv5bKq3mZy7KrMTMT2LJdKARZz ...
[9:56:05 PM] [restaking] operator processed withdrawal job: #1
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
[9:56:05 PM] [anchor] slept for 1 slots, started=41, ended=42, requested=42
```

```
[9:56:05 PM] [anchor] transaction failed
2kT53zjST3SaGirSxQtB6bHneTWSonkUGspDwwpL ...
[9:56:05 PM] [anchor] slept for 1 slots, started=42, ended=43, requested=43
[9:56:05 PM] [anchor] transaction failed
4L4ZJHRBC1ZiY7f66W9y8nmZXNN5gR8FgH6ScByv ...
[9:56:06 PM] [anchor] slept for 1 slots, started=43, ended=44, requested=44
[9:56:06 PM] [anchor] transaction confirmed:
5RTRo6qWCoZTfVjYZQNA2Zpseie2ELGnFEDPQf1h ...
[9:56:06 PM] [restaking] operator processed withdrawal job: #2
9b2RSM DYskVvjVbwF4cVwEhZUaaaUgyYSxvESmnoS4LL
  ✓ user5 (operator) processes queued withdrawals (2017ms)
[9:56:07 PM] [anchor] transaction confirmed:
5JB477k7PvSRghfUJm2Qb71k6xML51SwNoK4ic9n ...
[9:56:07 PM] [restaking] user withdrew: 3.996000000 SOL #2
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
  ✓ user5 can withdraw SOL (402ms)
[9:56:07 PM] [anchor] FUND_MANAGER (signer)
5FjrErTQ9P1ThYVdY9RamrPUCQGTMcczUjH21iKzbwx
[9:56:07 PM] [anchor] transaction confirmed:
5XDgZ4q9PmtmTtaw6RRXCpALAJRdNRMU6WjBKHYt ...
[9:56:07 PM] [anchor] transaction failed
4fyej3nvCeoBRKxZkd94meiE2WNb3tF5198r1F4V ...
[9:56:07 PM] [anchor] FUND_MANAGER (signer)
5FjrErTQ9P1ThYVdY9RamrPUCQGTMcczUjH21iKzbwx
[9:56:07 PM] [anchor] transaction confirmed:
67H9V6zLMmGKcik9zxx7BJK4S58MwacPnmYwtS8 ...
[9:56:08 PM] [anchor] transaction confirmed:
LR9D7gZZAvgM6vL2xcPsKVYKBNbf9aUS41NUXJyq ...
[9:56:08 PM] [restaking] user withdrew: 3.996000000 SOL #4
HpbPhk7yNyLWRcoutGhnp8bXwzkiAf4iM5DTj48QMQtG
  ✓ user5 cannot withdraw when withdrawal is disabled (1215ms)
[9:56:08 PM] [anchor] SOL airdropped (+100): 100.000000000 SOL
A5jsUAujiuoW8Lc5pb6R7XYrD5HH2gTBpMZDCpDMqxcS
[9:56:08 PM] [anchor] SOL airdropped (+100): 100.000000000 SOL
6tqUdVfNE9SUuipcFikQBZjyqNa99gc4KSC4CLEZShcQ
[9:56:09 PM] [anchor] slept for 1 slots, started=50, ended=51, requested=51
  ✓ try airdrop SOL to mock accounts (992ms)
[9:56:09 PM] [anchor] transaction confirmed:
38dGKD76tUcKDi5H9ZS4WNRZGGK2zQPpXRSEdYc ...
[9:56:09 PM] [restaking] user deposited: 10.000000000 SOL
A5jsUAujiuoW8Lc5pb6R7XYrD5HH2gTBpMZDCpDMqxcS
[9:56:09 PM] [restaking] user fragSOL balance: 10.000000000 fragSOL
A5jsUAujiuoW8Lc5pb6R7XYrD5HH2gTBpMZDCpDMqxcS
  ✓ user7 deposit SOL to mint fragSOL and create accounts (227ms)
[9:56:09 PM] [anchor] transaction failed null
  ✓ transfer fails from client-side SDK when dest PDA is not created yet
[9:56:09 PM] [anchor] transaction confirmed:
2kC1pDeqe76vRr4Jdag5Ni1qFapRnrB6BkpKGabx ...
[9:56:09 PM] [restaking] user deposited: 10.000000000 SOL
6tqUdVfNE9SUuipcFikQBZjyqNa99gc4KSC4CLEZShcQ
[9:56:09 PM] [restaking] user fragSOL balance: 10.000000000 fragSOL
6tqUdVfNE9SUuipcFikQBZjyqNa99gc4KSC4CLEZShcQ
  ✓ user8 deposit SOL to mint fragSOL and create accounts (400ms)
[9:56:10 PM] [anchor] transaction failed
25KixESNqYBwSJmfh4MrGpKv9JVSQheNsb4PRDRX ...
  ✓ transfer blocked from onchain-side for now
[9:56:10 PM] [anchor] SOL airdropped (+1000): 1000.000000000 SOL
2PLskyDxJ4ZpPccrjHQFh3V9aPpu5JtvjvJwdobqr8Zh
[9:56:10 PM] [anchor] SOL airdropped (+1000): 1000.000000000 SOL
2UhB1hD8ihBaxAQB RF48rWXPx6g3EFvEnmeMiour6777
[9:56:10 PM] [anchor] slept for 1 slots, started=53, ended=54, requested=54
  ✓ try airdrop SOL to mock accounts (573ms)
[9:56:10 PM] [anchor] transaction confirmed:
2Eaz7oh2jsAUiKkyHFqrA1SpBAaatXoT8jr24soC ...
[9:56:10 PM] [restaking] user deposited: 100.000000000 SOL
2UhB1hD8ihBaxAQB RF48rWXPx6g3EFvEnmeMiour6777
[9:56:10 PM] [restaking] user fragSOL balance: 100.000000000 fragSOL
2UhB1hD8ihBaxAQB RF48rWXPx6g3EFvEnmeMiour6777
```

```
[9:56:10 PM] [reward] [slot=55] A-pool#0: allocated=100,000,000,000, contribution=0
[9:56:10 PM] [reward] [slot=55] A-pool#1: allocated=100,000,000,000, contribution=0
[9:56:10 PM] [reward] > A-pool#1-reward#0: settled-slot=11, settled-amount=0, settled-
contribution=0
[9:56:11 PM] [anchor] slept for 1 slots, started=55, ended=56, requested=56
[9:56:11 PM] [anchor] transaction confirmed:
2ni83HsCd5H7fPGArNDm9hnKk4qteutrY1hdqJPy ...
[9:56:11 PM] [restaking] user deposited: 200.000000000 SOL
2PLskyDxJ4ZpPccrjHQFh3V9aPpu5JtvjvJwdobqr8Zh
[9:56:11 PM] [restaking] user fragSOL balance: 200.000000000 fragSOL
2PLskyDxJ4ZpPccrjHQFh3V9aPpu5JtvjvJwdobqr8Zh
[9:56:11 PM] [reward] [slot=57] B-pool#0: allocated=200,000,000,000, contribution=0
[9:56:11 PM] [reward] [slot=57] B-pool#1: allocated=200,000,000,000, contribution=0
[9:56:11 PM] [reward] > B-pool#1-reward#0: settled-slot=11, settled-amount=0, settled-
contribution=0
[9:56:12 PM] [anchor] slept for 1 slots, started=57, ended=58, requested=58
[9:56:12 PM] [anchor] transaction confirmed:
3KAvJ6jnda7HpibMhhe4mHRQCPkrhdewSJyN8k5W ...
[9:56:12 PM] [restaking] user manually updated user reward pool:
2PLskyDxJ4ZpPccrjHQFh3V9aPpu5JtvjvJwdobqr8Zh
[9:56:12 PM] [anchor] transaction confirmed:
Sz9cYtKEJzH61bUgtsDbkxg1bsEkC6CiiG363J7J ...
[9:56:12 PM] [restaking] user deposited: 300.000000000 SOL
2UhB1hD8ihBaxAQbRF48rWXPx6g3EFvEnmeMiour6777
[9:56:12 PM] [restaking] user fragSOL balance: 400.000000000 fragSOL
2UhB1hD8ihBaxAQbRF48rWXPx6g3EFvEnmeMiour6777
[9:56:12 PM] [reward] [slot=59] A-pool#0: allocated=400,000,000,000,
contribution=40,000,000,000,000
[9:56:12 PM] [reward] [slot=59] A-pool#1: allocated=400,000,000,000,
contribution=40,000,000,000,000
[9:56:12 PM] [reward] > A-pool#1-reward#0: settled-slot=11, settled-amount=0, settled-
contribution=0
[9:56:12 PM] [reward] [slot=59] B-pool#0: allocated=200,000,000,000,
contribution=40,000,000,000,000
[9:56:12 PM] [reward] [slot=59] B-pool#1: allocated=200,000,000,000,
contribution=40,000,000,000,000
[9:56:12 PM] [reward] > B-pool#1-reward#0: settled-slot=11, settled-amount=0, settled-
contribution=0
[9:56:12 PM] [anchor] slept for 1 slots, started=59, ended=60, requested=60
[9:56:13 PM] [anchor] transaction confirmed:
5sgH1vSokqTCpuenM1MeUtYKdrK6zwRodAvkfmYB ...
[9:56:13 PM] [restaking] user manually updated user reward pool:
2UhB1hD8ihBaxAQbRF48rWXPx6g3EFvEnmeMiour6777
[9:56:13 PM] [anchor] transaction confirmed:
5dprdnx5NtbJGKdoZyjsLqrbwfZr8hGhgnyFSBRD ...
[9:56:13 PM] [restaking] user manually updated user reward pool:
2PLskyDxJ4ZpPccrjHQFh3V9aPpu5JtvjvJwdobqr8Zh
[9:56:13 PM] [reward] [slot=61] A-pool#0: allocated=400,000,000,000,
contribution=120,000,000,000,000
[9:56:13 PM] [reward] [slot=61] A-pool#1: allocated=400,000,000,000,
contribution=120,000,000,000,000
[9:56:13 PM] [reward] > A-pool#1-reward#0: settled-slot=11, settled-amount=0, settled-
contribution=0
[9:56:13 PM] [reward] [slot=61] B-pool#0: allocated=200,000,000,000,
contribution=80,000,000,000,000
[9:56:13 PM] [reward] [slot=61] B-pool#1: allocated=200,000,000,000,
contribution=80,000,000,000,000
[9:56:13 PM] [reward] > B-pool#1-reward#0: settled-slot=11, settled-amount=0, settled-
contribution=0
[9:56:13 PM] [anchor] transaction confirmed:
38UukBbmQimNrWeJFtG1AgQEG8WkLsEj1S6Y5BXT ...
[9:56:13 PM] [restaking] operator manually updated global reward pool:
GiDkDCZjVC8Nk1Fd457qGSV2g3MQX62n7cV5CvgFyGfF
[9:56:13 PM] [anchor] FUND_MANAGER (signer)
5FjrErTQ9P1ThYVdY9RamrPUCQGTMCcczUjH21iKzbx
[9:56:14 PM] [anchor] transaction confirmed:
4NQBr8kApdephsEBRSCBDzjexrmHfUcwjcS9JgTk ...
[9:56:14 PM] [restaking] settled fragSOL reward to pool=1/bonus, rewardId=0/fPoint,
```

```
amount=50552960 (decimals=4)
[9:56:14 PM] [anchor] slept for 1 slots, started=63, ended=64, requested=64
[9:56:14 PM] [anchor] transaction confirmed:
52HCdhoK9GocoExmz8T8Uh68xvtUVtcMnyCNLiBT ...
[9:56:14 PM] [restaking] user manually updated user reward pool:
2PLskyDxJ4ZpPccrjHQFh3V9aPpu5JtvjvJwdoqr8Zh
[9:56:14 PM] [anchor] transaction confirmed:
3pbcQ29zYj5DQgrVyFBmWijBK11274x8BD7KNpkL ...
[9:56:14 PM] [restaking] user manually updated user reward pool:
2UhB1hD8ihBaxAQbRF48rWXPx6g3EFvEnmeMiour6777
[9:56:14 PM] [reward] [slot=65] A-pool#0: allocated=400,000,000,000,
contribution=280,000,000,000,000
[9:56:14 PM] [reward] [slot=65] A-pool#1: allocated=400,000,000,000,
contribution=280,000,000,000,000
[9:56:14 PM] [reward] > A-pool#1-reward#0: settled-slot=63, settled-amount=17,628,566, settled-
contribution=200,000,000,000,000
[9:56:14 PM] [reward] [slot=65] B-pool#0: allocated=200,000,000,000,
contribution=160,000,000,000,000
[9:56:14 PM] [reward] [slot=65] B-pool#1: allocated=200,000,000,000,
contribution=160,000,000,000,000
[9:56:14 PM] [reward] > B-pool#1-reward#0: settled-slot=63, settled-amount=10,577,139, settled-
contribution=120,000,000,000,000
  ✓ rewards are settled based on the contribution proportion (4263ms)
[9:56:15 PM] [anchor] transaction confirmed:
2jS4B9hvMyJEiENq8NXShHZLEn3vpfU9Rxf3VCz9 ...
[9:56:15 PM] [restaking] user deposited: 200.0000000000 SOL
2PLskyDxJ4ZpPccrjHQFh3V9aPpu5JtvjvJwdoqr8Zh
[9:56:15 PM] [restaking] user fragSOL balance: 400.0000000000 fragSOL
2PLskyDxJ4ZpPccrjHQFh3V9aPpu5JtvjvJwdoqr8Zh
[9:56:15 PM] [anchor] FUND_MANAGER (signer)
5FjrErTQ9P1ThYVdY9RamrPUCQGTMCcczUjH21iKzbxw
[9:56:15 PM] [anchor] FUND_MANAGER (signer)
5FjrErTQ9P1ThYVdY9RamrPUCQGTMCcczUjH21iKzbxw
[9:56:15 PM] [anchor] transaction confirmed:
43nGij4G3Db37hpTCgNZpcsp4AWp97tF4GbYzAff ...
[9:56:15 PM] [restaking] settled fragSOL reward to pool=0/base, rewardId=0/fPoint, amount=0
(decimals=4)
[9:56:15 PM] [anchor] transaction confirmed:
4oLUzfYnZK5VJ7s7EVd4uo828HBdEgBbr9wdiWHU ...
[9:56:15 PM] [restaking] settled fragSOL reward to pool=1/bonus, rewardId=0/fPoint, amount=0
(decimals=4)
[9:56:16 PM] [anchor] slept for 1 slots, started=67, ended=68, requested=68
[9:56:16 PM] [anchor] transaction confirmed:
VpnBfMa55P5BpEtsLB8ghhoLt51yqHxrUhyAjJd7 ...
[9:56:16 PM] [restaking] user manually updated user reward pool:
2UhB1hD8ihBaxAQbRF48rWXPx6g3EFvEnmeMiour6777
[9:56:16 PM] [anchor] transaction confirmed:
wDBus5y8mHzfoLFx9bL9RTNhKFME9iYbNmXcFaQ1 ...
[9:56:16 PM] [restaking] user manually updated user reward pool:
2PLskyDxJ4ZpPccrjHQFh3V9aPpu5JtvjvJwdoqr8Zh
[9:56:16 PM] [reward] [slot=69] A-pool#0: allocated=400,000,000,000,
contribution=440,000,000,000,000
[9:56:16 PM] [reward] > A-pool#0-reward#0: settled-slot=67, settled-amount=0, settled-
contribution=360,000,000,000,000
[9:56:16 PM] [reward] [slot=69] A-pool#1: allocated=400,000,000,000,
contribution=440,000,000,000,000
[9:56:16 PM] [reward] > A-pool#1-reward#0: settled-slot=67, settled-amount=17,628,566, settled-
contribution=360,000,000,000,000
[9:56:16 PM] [reward] [slot=69] B-pool#0: allocated=400,000,000,000,
contribution=300,000,000,000,000
[9:56:16 PM] [reward] > B-pool#0-reward#0: settled-slot=67, settled-amount=0, settled-
contribution=220,000,000,000,000
[9:56:16 PM] [reward] [slot=69] B-pool#1: allocated=400,000,000,000,
contribution=330,000,000,000,000
[9:56:16 PM] [reward] > B-pool#1-reward#0: settled-slot=67, settled-amount=10,577,139, settled-
contribution=230,000,000,000,000
[9:56:16 PM] [anchor] transaction confirmed:
51jZDtjidzHyG1uUAP5FziADoQ3Lcn6nCvf9RbbM ...
```

```
[9:56:16 PM] [restaking] operator manually updated global reward pool:
GiDkDCZjVC8Nk1Fd457qGSV2g3MQX62n7cV5CvgFyGfF
[9:56:16 PM] [anchor] FUND_MANAGER (signer)
5FjrErTQ9P1ThYVdY9RamrPUCQGTMcczUjH21iKzbx
[9:56:16 PM] [anchor] FUND_MANAGER (signer)
5FjrErTQ9P1ThYVdY9RamrPUCQGTMcczUjH21iKzbx
[9:56:17 PM] [anchor] transaction confirmed:
5qafZmuWhzbZQDWv7U9DfdLHhjbfnzcHYgyfGSP ...
[9:56:17 PM] [restaking] settled fragSOL reward to pool=0/base, rewardId=0/fPoint,
amount=233913835 (decimals=4)
[9:56:17 PM] [anchor] transaction confirmed:
5YPy8KyJ3Y1YrhC51uX6WjmorVjejN3XdZU4BanC ...
[9:56:17 PM] [restaking] settled fragSOL reward to pool=1/bonus, rewardId=0/fPoint,
amount=233913835 (decimals=4)
[9:56:17 PM] [anchor] transaction confirmed:
xC1bwDosqTgtc1f7aRiou5SnHbmhTHpFPWTCd8Hh ...
[9:56:17 PM] [restaking] operator manually updated global reward pool:
GiDkDCZjVC8Nk1Fd457qGSV2g3MQX62n7cV5CvgFyGfF
[9:56:18 PM] [anchor] transaction confirmed:
Zw989MRPRKfdw3XTUMmeSv5J4aPcnZacghUMbFxi ...
[9:56:18 PM] [restaking] user manually updated user reward pool:
2PLskyDxJ4ZpPccrjHQFh3V9aPpu5JtvjvJwdoqr8Zh
[9:56:18 PM] [anchor] transaction confirmed:
xW9bKppNDzrRhAUqQ8y3fEPcuieBQbXQ8752GqC9 ...
[9:56:18 PM] [restaking] user manually updated user reward pool:
2UhB1hD8ihBaxAQbRF48rWXPx6g3EFvEnmeMiour6777
  ✓ rewards can be settled with custom contribution accrual rate enabled (3242ms)

30 passing (28s)
```

Changelog

- 2024-10-24 - Initial report
- 2024-11-01 - Fix review

About Quantstamp

Quantstamp is a global leader in blockchain security. Founded in 2017, Quantstamp's mission is to securely onboard the next billion users to Web3 through its best-in-class Web3 security products and services.

Quantstamp's team consists of cybersecurity experts hailing from globally recognized organizations including Microsoft, AWS, BMW, Meta, and the Ethereum Foundation. Quantstamp engineers hold PhDs or advanced computer science degrees, with decades of combined experience in formal verification, static analysis, blockchain audits, penetration testing, and original leading-edge research.

To date, Quantstamp has performed more than 500 audits and secured over \$200 billion in digital asset risk from hackers. Quantstamp has worked with a diverse range of customers, including startups, category leaders and financial institutions. Brands that Quantstamp has worked with include Ethereum 2.0, Binance, Visa, PayPal, Polygon, Avalanche, Curve, Solana, Compound, Lido, MakerDAO, Arbitrum, OpenSea and the World Economic Forum.

Quantstamp's collaborations and partnerships showcase our commitment to world-class research, development and security. We're honored to work with some of the top names in the industry and proud to secure the future of web3.

Notable Collaborations & Customers:

- Blockchains: Ethereum 2.0, Near, Flow, Avalanche, Solana, Cardano, Binance Smart Chain, Hedera Hashgraph, Tezos
- DeFi: Curve, Compound, Maker, Lido, Polygon, Arbitrum, SushiSwap
- NFT: OpenSea, Parallel, Dapper Labs, Decentraland, Sandbox, Axie Infinity, Illuvium, NBA Top Shot, Zora
- Academic institutions: National University of Singapore, MIT

Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by Quantstamp; however, Quantstamp does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication or other making available of the report to you by Quantstamp.

Notice of confidentiality

This report, including the content, data, and underlying methodologies, are subject to the confidentiality and feedback provisions in your agreement with Quantstamp. These materials are not to be disclosed, extracted, copied, or distributed except to the extent expressly authorized by Quantstamp.

Links to other websites

You may, through hypertext or other computer links, gain access to web sites operated by persons other than Quantstamp. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites' owners. You agree that Quantstamp are not responsible for the content or operation of such web sites, and that Quantstamp shall have no liability to you or any other person or entity for the use of third-party web sites. Except as described below, a hyperlink from this web site to another web site does not imply or mean that Quantstamp endorses the content on that web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the report. Quantstamp assumes no responsibility for the use of third-party software on any website and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any output generated by such software.

Disclaimer

The review and this report are provided on an as-is, where-is, and as-available basis. To the fullest extent permitted by law, Quantstamp disclaims all warranties, expressed implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. You agree that access and/or use of the report and other results of the review, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE. This report is based on the scope of materials and documentation provided for a limited review at the time provided. You acknowledge that Blockchain technology remains under development and is subject to unknown risks and flaws and, as such, the report may not be complete or inclusive of all vulnerabilities. The review is limited to the materials identified in the report and does not extend to the compiler layer, or any other areas beyond the programming language, or programming aspects that could present security risks. The report does not indicate the endorsement by Quantstamp of any particular project or team, nor guarantee its security, and may not be represented as such. No third party is entitled to rely on the report in any any way, including for the purpose of making any decisions to buy or sell a product, product, service or any other asset. Quantstamp does not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party, or any open source or third-party software, code, libraries, materials, or information to, to, called by, referenced by or accessible through the report, its content, or any related related services and products, any hyperlinked websites, or any other websites or mobile applications, and we will not be a party to or in any way be responsible for monitoring any any transaction between you and any third party. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate.

