ChaosFinance

Audit Report

Thu Mar 20 2025

contact@bitslab.xyz https://twitter.com/scalebit_

ScaleBit

ChaosFinance Audit Report

1 Executive Summary

1.1 Project Information

Description	A staking project on sonic.
Type	Staking
Auditors	ScaleBit
Timeline	Mon Mar 17 2025 - Thu Mar 20 2025
Languages	Rust
Platform	Solana
Methods	Architecture Review, Unit Testing, Manual Review
Source Code	https://github.com/chaosfinance/sonic-lsd-contracts
Commits	45e9426eecf16f455727d7c08c95b241db9296d1 e70b73ae05c6e9e3b5be5d18860accea590cbd34

1.2 Files in Scope

The following are the SHA1 hashes of the original reviewed files.

ID	File	SHA-1 Hash
ERR	programs/lsd-program/src/errors.rs	ddb06c7df6401b3c0712d7d5440fd8e9bc7f3c95
LIB	programs/lsd-program/src/lib.rs	4f88d577acf4a64b8a5756015276e022b8947dfe
SST	programs/lsd-program/src/staker_stake.rs	b25c82fc88e5816e6281d14fecd41ad6a58e10cc
STA	programs/lsd-program/src/states.rs	c179a71f0e0f57656bd5f2195c5ea44adc6b4c30
EAC	programs/lsd-program/src/era_active.rs	e28efc1cb35990a2ee7f555e4023923953171a5a
ADM	programs/lsd-program/src/admin.rs	0bf1e54c16cb100b0be80c2e41257bb28f9dbcc3
ISM	programs/lsd-program/src/initialize_stake_manager.rs	366b69d02029c96bf14f9966297c9c8e8f610113
ENE	programs/lsd-program/src/era_new.rs	cfa3bae889202a340d51342c797b5155adfa8775
EBO	programs/lsd-program/src/era_bond.rs	321570b1c8d2c8b17596dbf5092173e63c1346c9
SUN	programs/lsd-program/src/staker_unstake.rs	9e0b41cb205646a2b3a103be3d641eced7b18ce2
EWI	programs/lsd-program/src/era_withdraw.rs	b1191cb4aaf1ce2623fde248b486060636b44088

SWI	programs/lsd-program/src/staker_withdraw.rs	92889d8b4b2618021ac46aa88673cbe4f8491bdf
HEL	programs/lsd-program/src/helper.rs	50870efcfe89a27f00f54887b27fdecd2ef86f97

1.3 Issue Statistic

Item	Count	Fixed	Partially Fixed	Acknowledged
Total	7	2	1	4
Informational	3	1	0	2
Minor	3	1	1	1
Medium	1	0	0	1
Major	0	0	0	0
Critical	0	0	0	0

1.4 ScaleBit Audit Breakdown

ScaleBit aims to assess repositories for security-related issues, code quality, and compliance with specifications and best practices. Possible issues our team looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Integer overflow/underflow
- Number of rounding errors
- Unchecked External Call
- Unchecked CALL Return Values
- Functionality Checks
- Reentrancy
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic issues
- Gas usage
- Fallback function usage
- tx.origin authentication
- Replay attacks
- Coding style issues

1.5 Methodology

The security team adopted the **"Testing and Automated Analysis"**, **"Code Review"** and **"Formal Verification"** strategy to perform a complete security test on the code in a way that is closest to the real attack. The main entrance and scope of security testing are stated in the conventions in the "Audit Objective", which can expand to contexts beyond the scope according to the actual testing needs. The main types of this security audit include:

(1) Testing and Automated Analysis

Items to check: state consistency / failure rollback / unit testing / value overflows / parameter verification / unhandled errors / boundary checking / coding specifications.

(2) Code Review

The code scope is illustrated in section 1.2.

(3) Audit Process

- Carry out relevant security tests on the testnet or the mainnet;
- If there are any questions during the audit process, communicate with the code owner in time. The code owners should actively cooperate (this might include providing the latest stable source code, relevant deployment scripts or methods, transaction signature scripts, exchange docking schemes, etc.);
- The necessary information during the audit process will be well documented for both the audit team and the code owner in a timely manner.

2 Summary

This report has been commissioned by ChaosFinance to identify any potential issues and vulnerabilities in the source code of the ChaosFinance smart contract, as well as any contract dependencies that were not part of an officially recognized library. In this audit, we have utilized various techniques, including manual code review and static analysis, to identify potential vulnerabilities and security issues.

During the audit, we identified 7 issues of varying severity, listed below.

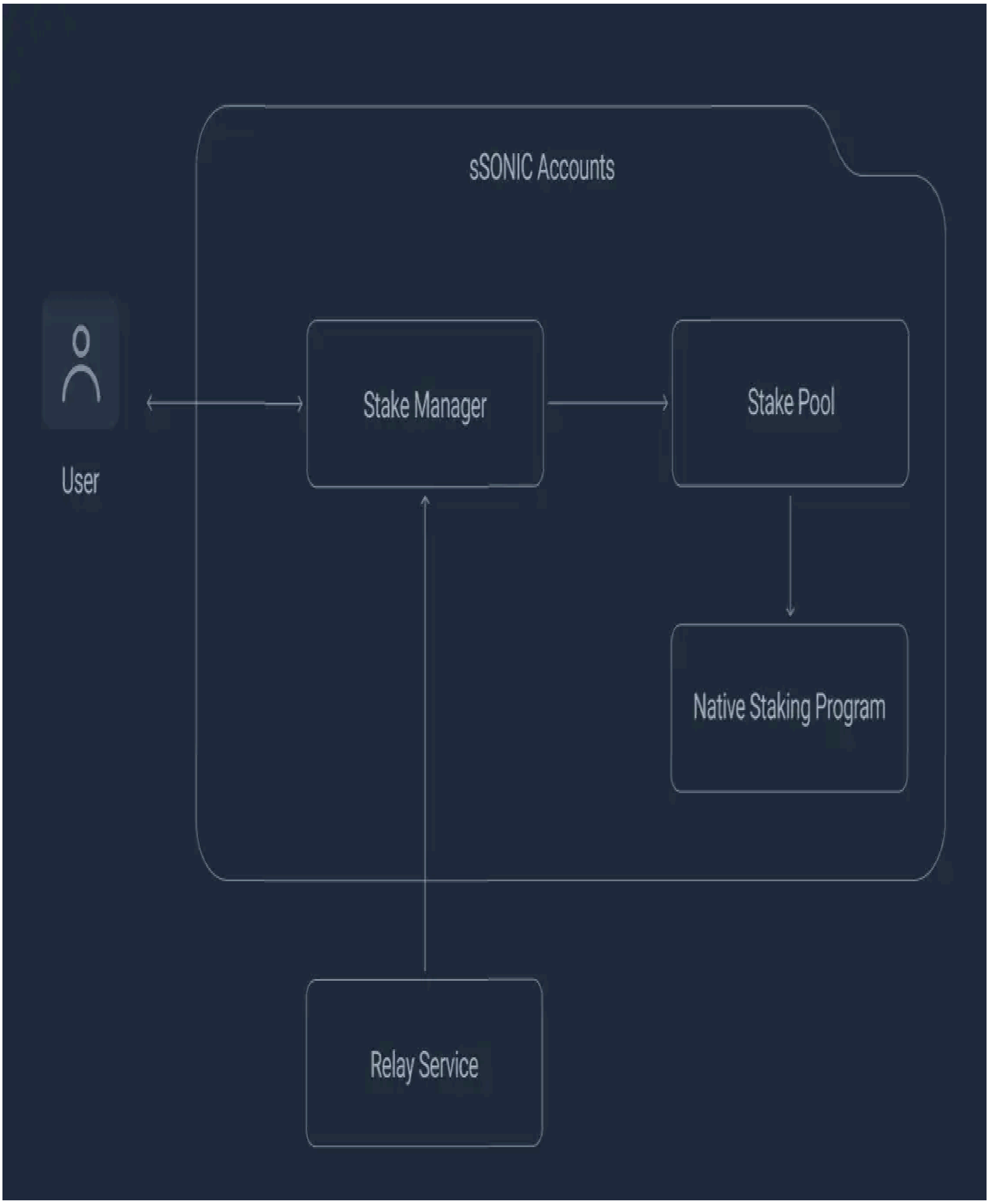
ID	Title	Severity	Status
ADM-1	Lack of Two-Step Admin Transfer	Minor	Fixed
EAC-1	Incorrect Fee Calculation	Medium	Acknowledged
EBO-1	Better Way to Get `Sysvar<Rent>`	Informational	Acknowledged
ENE-1	Error Updating `stake_manager.latest_era`	Minor	Acknowledged
SST-1	Possibly confusing logs	Minor	Partially Fixed
STA-1	Precision Loss of `total_bond_and_reward` Caculation	Informational	Acknowledged
SWI-1	Duplicate Checks on `unstake_account.stake_manager`	Informational	Fixed

3 Participant Process

Here are the relevant actors with their respective abilities within the ChaosFinance Smart Contract :

Chaos Architecture

Chaos Finance is built on top of the Solana LSD Stack from StaFi's AI-powered LSaaS.



In the Solana ecosystem, "smart contracts" are called programs. Each program is an on-chain account that stores executable logic, organized into specific functions referred to as instructions.

Core Accounts:

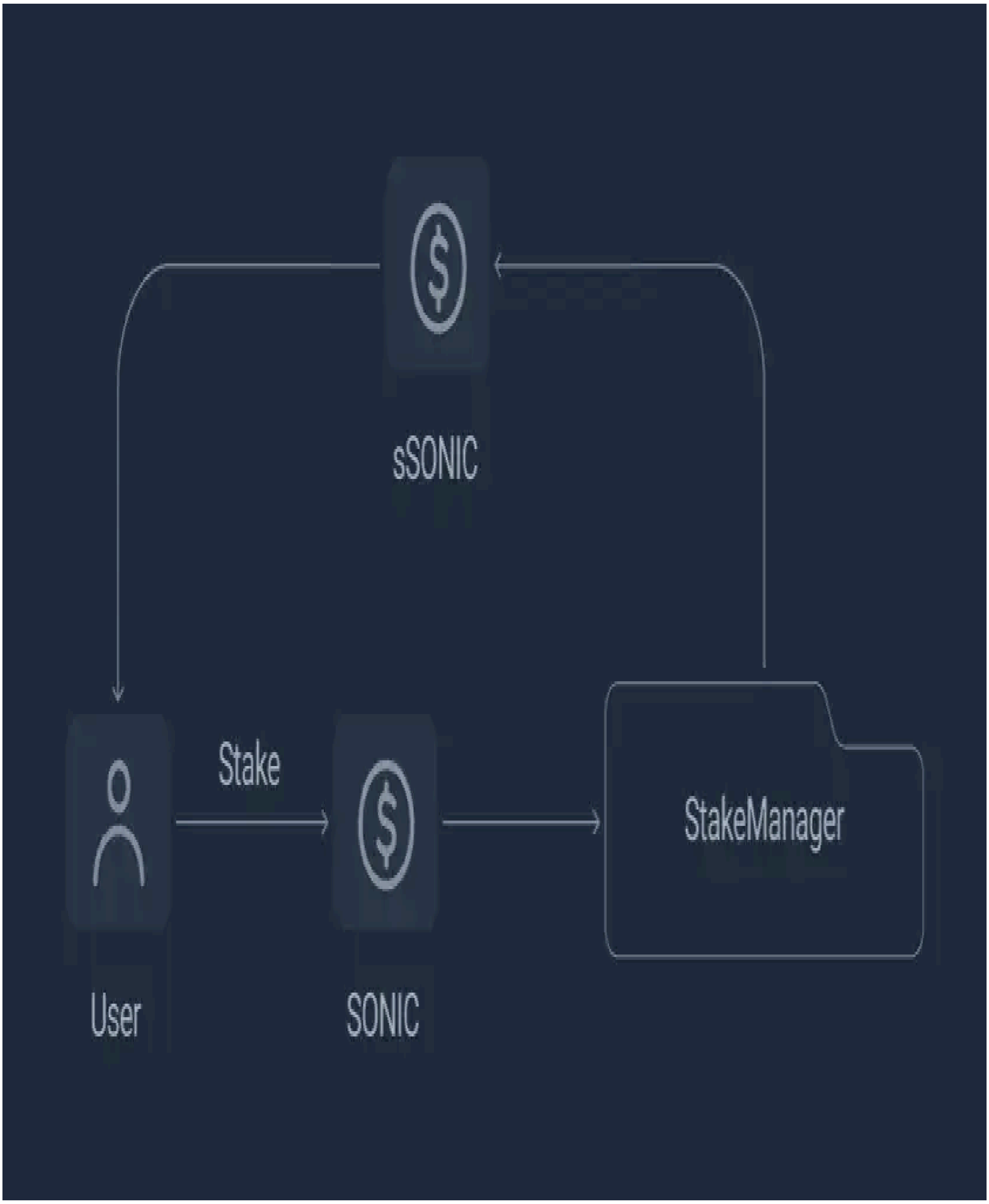
- StakeManager: the main account of sSONIC that stores all the states such as: rate, stake info list and commission fee
- StakePool: an escrow account manages funds between stakers and sonic staking program
- UnstakeAccount: an account stores unstake info such as: recipient, amount and withdrawal index

Core roles:

- Admin: manages parameters of sSONIC network

Stake Flow

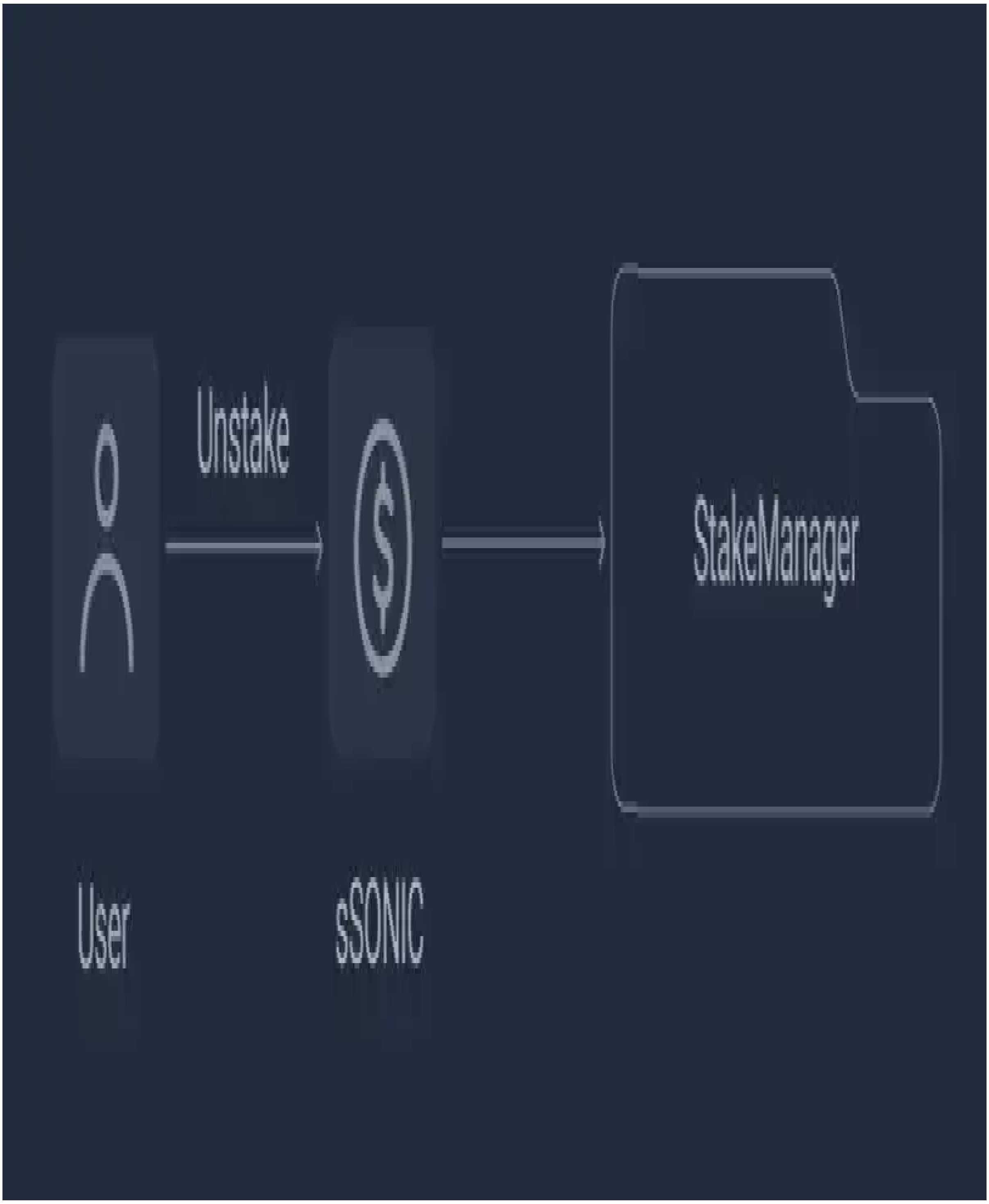
Users can stake SONIC to the network via stake method, and users will receive equivalent amount of sSONIC. `Amount of sSONIC = stakingAmount * Total sSONIC Supply / Total SONIC`



Amount

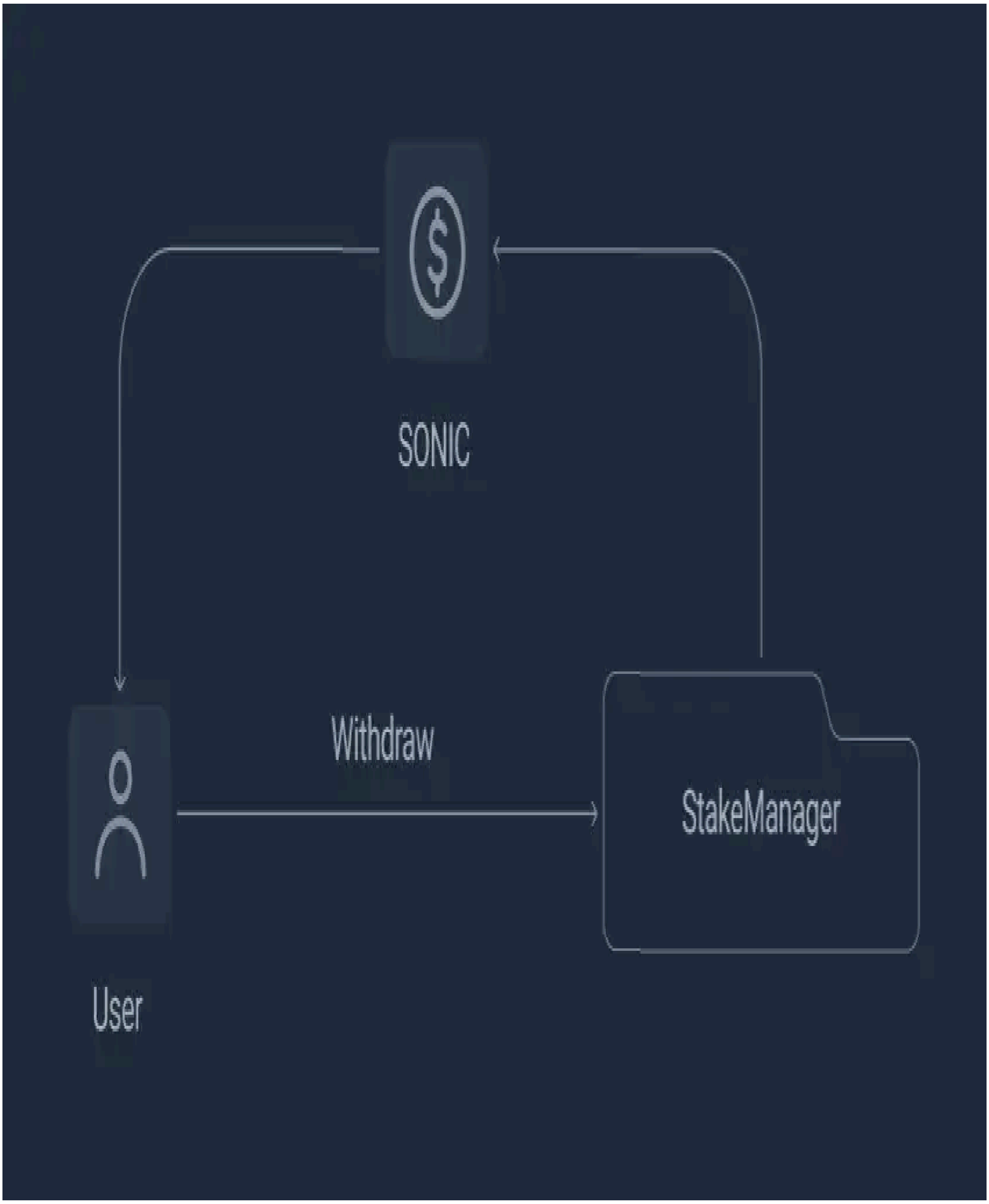
Unstake Flow

Any sSONIC holders are valid users, and can call unstake method to exchange SONIC with sSONIC.



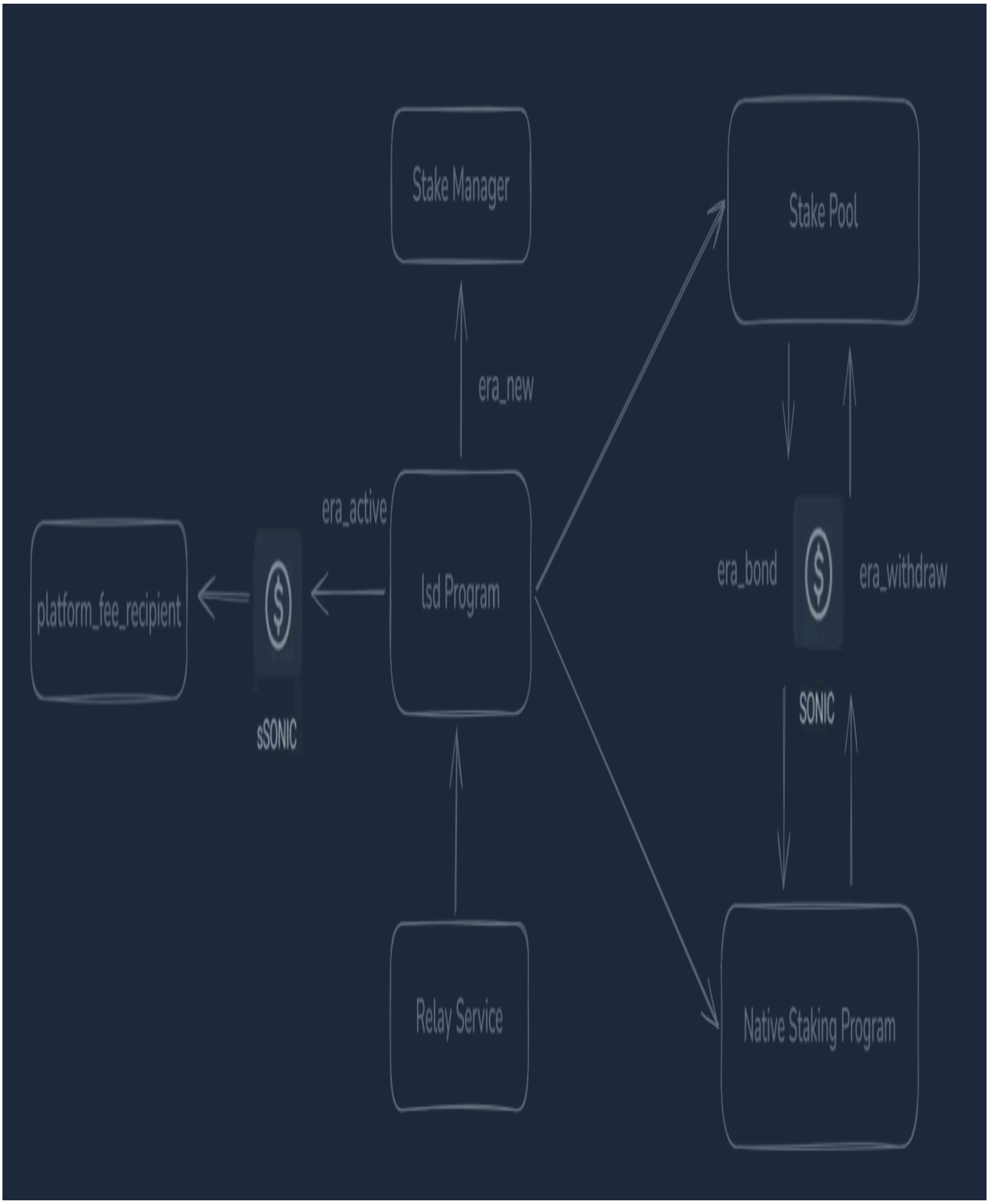
Withdraw Flow

Users can get their principals and rewards by calling withdraw method when there are enough amount of matured stakes.



Relay Service

Due to the limitation of smart contract, it could not launch an execution. So Chaos Finance introduces Relay service. It will trigger StakeManager, at a certain interval, to collect and calculate users' reward, distribute it to the platform and users.



4 Findings

ADM-1 Lack of Two-Step Admin Transfer

Severity: Minor

Status: Fixed

Code Location:

programs/lsd-program/src/admin.rs#15

Descriptions:

The function `process()` allows immediate transfer of admin privileges to a new address. This is risky because if an admin mistakenly enters the wrong address, there is no way to revert the change.

```
pub fn process(&mut self, new_admin: Pubkey) -> Result<()> {
        self.stake_manager.admin = new_admin;
        msg!("NewAdmin: {}", new_admin);
        Ok(())
    }
```

Suggestion:

A safe admin transfer should involve two steps:

Step 1: The current admin nominates a new admin. Step 2 : The new admin accepts the role.

Resolution:

This issue has been fixed. The client has adopted our suggestions.

EAC-1 Incorrect Fee Calculation

Severity: Medium

Status: Acknowledged

Code Location:

programs/lsd-program/src/era_active.rs#71-94

Descriptions:

The function responsible for calculating and distributing platform fees introduces an inconsistency due to the order in which `lsd_token_mint.supply` is updated. When determining the amount of `lsd_token` to mint as a fee, the function uses the supply value before the minting occurs. However, once the fee tokens are minted and added to the total supply, the staking rate increases. As a result, users who later unstake will be charged a higher fee than initially intended.

This issue can lead to the following problems:

1. Inaccurate Fee Deductions: Users may end up paying more than the stated or expected fee percentage.
2. User Confusion and Loss of Trust: A mismatch between the documented fee structure and actual deductions could lead to reputational damage.
3. Incorrect Data for Monitoring and Integrations: On-chain analytics tools, dashboards, and third-party services relying on expected staking mechanics may display incorrect information, leading to potential misinterpretations.

Suggestion:

Calculate the handling fee in the correct way or collect the handling fee in other ways, and test the handling fee part so as to collect the correct amount of handling fee from the project party. This is the correct calculation before minting the `platform_fee`

```
platform_fee = reward*(new_active/(lsd_token_mint.supply+platform_fee))*0.1
```

Resolution:

The proposed change would indeed be more accurate, but considering the simplicity of the current implementation logic and the fact that the FEE is only 10% from the reward portion, the difference in the real calculation is very small, and the impact on the user is so small that it's basically negligible, we're going to leave this place unchanged for now.

EBO-1 Better Way to Get `Sysvar<Rent>`

Severity: Informational

Status: Acknowledged

Code Location:

programs/lsd-program/src/era_bond.rs#62;

programs/lsd-program/src/era_withdraw.rs#62;

programs/lsd-program/src/initialize_stake_manager.rs#35;

programs/lsd-program/src/staker_unstake.rs#43

Descriptions:

According to Anchor docs:

"If possible, sysvars should not be used via accounts but by using the get function on the desired sysvar. This is because using get does not run the risk of Anchor having a bug in its Sysvar type and using get also decreases tx size, making space for other accounts that cannot be requested via syscall."

And the `rent` account doesn't seem to be used to.

Suggestion:

Suggest using a better way to get `Sysvar<Rent>`

Resolution:

This optimization has been confirmed, but considering that we have already deployed to the Mainnet beta environment, modifying it would require both front-end and back-end to be upgraded and deployed, and not modifying it wouldn't cause any problems, so we plan to do it as a whole in a future version iteration.

ENE-1 Error Updating `stake_manager.latest_era`

Severity: Minor

Status: Acknowledged

Code Location:

programs/lsd-program/src/era_new.rs#47,51

Descriptions:

In the `EraNew` instruction, `stake_manager.latest_era` should be updated to the latest time `current_era` instead of `new_era`. Otherwise, if the `EraNew` instruction is not called for a while, the `latest_era` will be smaller than the real one, which will prevent the withdraw operation from completing in time.

Suggestion:

Suggest updating `stake_manager.latest_era` to `current_era`.

Resolution:

stake_manager.latest_era, our design is to keep the continuous growth, when there is an abnormal situation when latest_era is smaller than current_era, sonic-lsd-relay service will call EraNew several times until latest_era and current_era are the same, at the same time, the exchange rate corresponding to each latest_era will be calculated according to the reward at that time. until the latest_era and current_era are the same. Meanwhile, the exchange rate corresponding to each latest_era will be calculated according to the reward at that time, so as to ensure that the exchange rate corresponding to each latest_era is also accurate. This ensures the accuracy and consistency of the data of the front-end and back-end services as well as the subsequent statistical services.

SST-1 Possibly confusing logs

Severity: Minor

Status: Partially Fixed

Code Location:

programs/lsd-program/src/staker_stake.rs#119;

programs/lsd-program/src/staker_unstake.rs#92;

programs/lsd-program/src/staker_withdraw.rs#117

Descriptions:

The current implementation allows a user to create and initialize a `stake_manager` of their own and invoke the contract. However, the `stake_manager` field is not recorded in the event, which can lead to event confusion.

Suggestion:

Suggest recording `stake_manager` in the event.

STA-1 Precision Loss of `total_bond_and_reward` Caculation

Severity: Informational

Status: Acknowledged

Code Location:

programs/lsd-program/src/states.rs#109

Descriptions:

In the `calc_total_bond_and_reward` function, the `s.amount + (s.reward as u128 * elapsed as u128 / duration as u128) as u64` may result in a loss of precision, causing `total_bond_and_reward` to be smaller than it actually is.

Suggestion:

It is recommended to apply a precision offset to limit the loss, ensuring that the losses due to rounding are minimized.

Resolution:

The problem has been confirmed, and it is not easy to ensure that there is no Precision Loss here, but considering that the numerical effect is negligible, it can be ignored, so we will not modify it here for now.

SWI-1 Duplicate Checks on `unstake_account.stake_manager`

Severity: Informational

Status: Fixed

Code Location:

programs/lsd-program/src/staker_withdraw.rs#34;

programs/lsd-program/src/staker_withdraw.rs#77-81

Descriptions:

Duplicate checks on `unstake_account.stake_manager`.

Suggestion:

Suggest removing duplicate checks.

Resolution:

This issue has been fixed. The client has adopted our suggestions.

Appendix 1

Issue Level

- **Informational** issues are often recommendations to improve the style of the code or to optimize code that does not affect the overall functionality.
- **Minor** issues are general suggestions relevant to best practices and readability. They don't post any direct risk. Developers are encouraged to fix them.
- **Medium** issues are non-exploitable problems and not security vulnerabilities. They should be fixed unless there is a specific reason not to.
- **Major** issues are security vulnerabilities. They put a portion of users' sensitive information at risk, and often are not directly exploitable. All major issues should be fixed.
- **Critical** issues are directly exploitable security vulnerabilities. They put users' sensitive information at risk. All critical issues should be fixed.

Issue Status

- **Fixed:** The issue has been resolved.
- **Partially Fixed:** The issue has been partially resolved.
- **Acknowledged:** The issue has been acknowledged by the code owner, and the code owner confirms it's as designed, and decides to keep it.

Appendix 2

Disclaimer

This report is based on the scope of materials and documents provided, with a limited review at the time provided. Results may not be complete and do not include all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your own risk. A report does not imply an endorsement of any particular project or team, nor does it guarantee its security. These reports should not be relied upon in any way by any third party, including for the purpose of making any decision to buy or sell products, services, or any other assets. TO THE FULLEST EXTENT PERMITTED BY LAW, WE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, IN CONNECTION WITH THIS REPORT, ITS CONTENT, RELATED SERVICES AND PRODUCTS, AND YOUR USE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NOT INFRINGEMENT.

SCALEBIT VERIFIED

SCALEBIT VERIFIED