SOLIDProof
Bring trust into your projects

Blockchain Security | Smart Contract Audits | KYC
Development | Marketing

MADE IN GERMANY

Vibranium Fi

AUDIT

SECURITY ASSESSMENT

31. July, 2023

FOR

VIBRANIUM

SolidProof_io @solidproof_io

Introduction

SolidProof.io is a brand of the officially registered company MAKE Network GmbH, based in Germany. We're mainly focused on Blockchain Security such as Smart Contract Audits and KYC verification for project teams.
Solidproof.io assess potential security issues in the smart contracts implementations, review for potential inconsistencies between the code base and the whitepaper/documentation, and provide suggestions for improvement.

Disclaimer

SolidProof.io reports are not, nor should be considered, an "endorsement"or "disapproval" of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team. SolidProof.io do not cover testing or auditing the integration with external contract or services (such as Unicrypt, Uniswap, PancakeSwap etc'...)

SolidProof.io Audits do not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technology proprietors. SolidProof Audits should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.

SolidProof.io Reports represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology. Blockchain technology and cryptographic assets present a high level of ongoing risk. SolidProof's position is that each company and individual are responsible for their own due diligence and continuous security. SolidProof in no way claims any guarantee of the security or functionality of the technology we agree to analyze.

Project Overview

Summary

Project Name	Vibranium Fi
Website	https://vibranium.finance/
About the project	The Vibranium Finance is an innovative fork of Lybra Finance with the advanced multichain development to optimize users' experiences and bring stability to the volatile world of cryptocurrency. The protocol's foundation lies in LSD (Liquid Staking Derivatives), initially capitalizing on Lido Finance-issued ETH proof-of-stake and stETH. Future plans include expanding support for additional LSD assets.
Chain	TBA
Language	Solidity
Codebase	https://github.com/vibraniumfi/contract/tree/main/contracts
Commit	N/A
Unit Tests	Not Provided

Social Medias

Telegram	N/A
Twitter	https://twitter.com/VibraniumLSD
Facebook	N/A
Instagram	N/A
GitHub	N/A
Reddit	N/A
Medium	N/A
Discord	https://discord.com/invite/4jGNCWcVMh
YouTube	N/A
TikTok	N/A
LinkedIn	N/A

Audit Summary

Version	Delivery Date	Change Log
v1.0	29. July 2023	· Layout Project · Automated/ Manual-Security Testing · Summary
v1.1	31. July 2023	· Reaudit

Note - The following audit report presents a comprehensive security analysis of the smart contract utilized in the project. This analysis did not include functional testing (or unit testing) of the contract's logic. We cannot guarantee 100% logical correctness of the contract as it was not functionally tested by us.

File Overview

The Team provided us with the files that should be tested in the security assessment. This audit covered the following files listed below with an SHA-1 Hash.

File Name	SHA-1 Hash
contracts/interfaces/IStargatePool.sol	c142c424d5594763dac4cb6d5e7d5c7be278c714
contracts/interfaces/IStargateFactory.sol	17b21854cc3ae9f4489a83ad359de0cc9b6a5946
contracts/interfaces/ILayerZeroUserApplicationConfig.sol	619d629e55c336e6297b51444c9751b25d699b7c
contracts/interfaces/ILayerZeroReceiver.sol	40fdfafbbd43411c04fab817e023e5162ab86386
contracts/interfaces/IStargateRouterETH.sol	d8d0519e640edc392abc63bad0e92d99354da38f
contracts/interfaces/IStargateWidget.sol	45612c2e0e71a0e2792ce83274a409a107f406d0
contracts/interfaces/ILayerZeroEndpoint.sol	89e89b610c294d760037ba188a7a470a890a64fa
contracts/interfaces/IStargateReceiver.sol	8aca46f58c5ee6c7872144173bd8195b42271777
contracts/interfaces/IStargateRouter.sol	4ed246cb421a53dc93517160760a5e363314a65f
contracts/CollateralRatioGuardian.sol	aa82501ccc0fca7a5c92caeee7538c5896775fe2
contracts/OFTCore.sol	14c51e43e9a7c3bda8ad1f8edfa52704f71d9e58
contracts/VIB.sol	77f14d31b2eb71ac17d672448ede727b78ff8772
contracts/esVIBMinter.sol	19cb00eec72c8875dbfd6901f7375672ca7e5c2a
contracts/Context.sol	6a0b5b8e1b849d1ea73eabcfb1c9cd7e0cdbc91b
contracts/StakerewardV2pool.sol	19220b596c34dce695fc7c272f5882e204dc3468
contracts/IOFTCore.sol	91aae58196d8a9a15c4d7053e6b67142868fca64
contracts/Ilido.sol	065e65c367f2c3619e2dfed306df3ba1039e7ca0
contracts/VibStakingPool.sol	873c55a8b6272f42ebcb5975ae9041ef9c3e2547

File Name	SHA-1 Hash
contracts/esVIBMinerV2.sol	5f4dfebe5929f02bee002f4ea2407eee1d92076c
contracts/lzApp/NonblockingLzApp.sol	57afe74643d79539dd4cf82d464d08eb4b9f2a8e
contracts/lzApp/LzApp.sol	dc15cbb7541df6c566c8c7f2fb3bab1e66a0172c
contracts/VibraniumHelper.sol	de61ba1f4314989ba3307f006a36881f85dc0255
contracts/IPriceFeed.sol	fdbc1451ee1624802afcaaaa22b7688ae38ab8e0
contracts/IOFT.sol	32b24b838e1a2a8e97bbef3febdb0c965e90640e
contracts/Ireth.sol	041a7b5879d63d7a5593332c197d83c1ee2e4e5c
contracts/Governable.sol	6c1a913adb1380256040db2392c46325244b88db
contracts/RVUSD.sol	66d7b619ab855bd390e93e8f93b68ce947b86ae6
contracts/Vibranium.sol	440d1be5d71b600c493e7de6d8f34ff6cceed1f2
contracts/AggregatorV3Interface.sol	33756d949e5dd4b734ad5a37a3cec1378e65006e
contracts/ISwap.sol	20744d1a6096d043092b0b4b93e7175963fec93a
contracts/esVIB.sol	10429ff417233cfaf89295dfab8a0cae720a0298
contracts/SafeMath.sol	0b7ce4a22381d1ebc23904d311ab54a3c357119f
contracts/esVIBBoost.sol	d61a4afeacb9e25c3324b08ccba1bc6ba1d068fa
contracts/VibraniumFund.sol	45ade6e4f27dec2e560fefe996e8a297a8c28844
contracts/IesVIB.sol	b4f3041a304ee20a661e1ab01b87ef6e95161524
contracts/Ownable.sol	b081753867999e7701baea067992ed330b36ff72
contracts/VibraniumrvUSD.sol	06f7e4ec2efccc13ee814d37c14016b0cf299a48
contracts/VUSD.sol	6a5fcbabbebc89239dec0de0797088a0dc9dd5a1
contracts/PublicSale.sol	01a278c7064696771c90083f109215c523610387
contracts/util/BytesLib.sol	90329d1b8d0302c4824da5a42d311e03703a3094
contracts/util/ExcessivelySafeCall.sol	27563f19821da6d9e59e31996e33cd0af4e8c2cc
contracts/TokenUnLocking.sol	f3955445ef4d1b61c238fdef687a82069ca82f66

File Name	SHA-1 Hash
contracts/ERC20.sol	9e1d0c7c2e84df39c7688aadbf1499b7ec106009
contracts/OFT.sol	e16cbe7c1ed1079c76c594facef49351ea1ca074
contracts/IERC20.sol	ae3129214fc49e54b40a7265b9b0fd05677d381b
contracts/IVibranium.sol	973203b8a2e5b4602eb533ff43abf5df6eebc6ab

Please note: Files with a different hash value than in this table have been modified after the security check, either intentionally or unintentionally. A different hash value may (but need not) be an indication of a changed state or potential vulnerability that was not the subject of this scan.

Imported packages

Used code from other Frameworks/Smart Contracts.

Dependency / Import Path	Count
@chainlink/contracts/src/v0.8/interfaces/AggregatorV3Interface.sol	2
@openzeppelin/contracts/access/Ownable.sol	2
@openzeppelin/contracts/token/ERC20/ERC20.sol	1
@openzeppelin/contracts/token/ERC20/IERC20.sol	2
@openzeppelin/contracts/token/ERC20/extensions/ERC20Votes.sol	1
@openzeppelin/contracts/utils/introspection/ERC165.sol	1
@openzeppelin/contracts/utils/introspection/IERC165.sol	4

Note for Investors: We advise the highest caution on any presale hosted on any non-reputable launchpad website. This smart contract audit only included the mentioned file and does not support any kind of presale.

External/Public functions

External/public functions are functions that can be called from outside of a contract, i.e., they can be accessed by other contracts or external accounts on the blockchain. These functions are specified using the function declaration's external or public visibility modifier.

State variables

State variables are variables that are stored on the blockchain as part of the contract'sstate. They are declared at the contract level and can be accessed and modified by any function within the contract. State variables can be needed within visibility modifier, such as public, private or internal, which determines the access level of the variable.

Components

Contracts	Libraries	Interfaces	Abstract
15	3	29	7

Exposed Functions

This section lists functions that are explicitly declared public or payable. Please note that getter methods for public stateVars are not included.

Public	Payable
288	15

External	Internal	Private	Pure	View
197	312	2	35	138

StateVariables

Total	Public
132	104

Capabilities

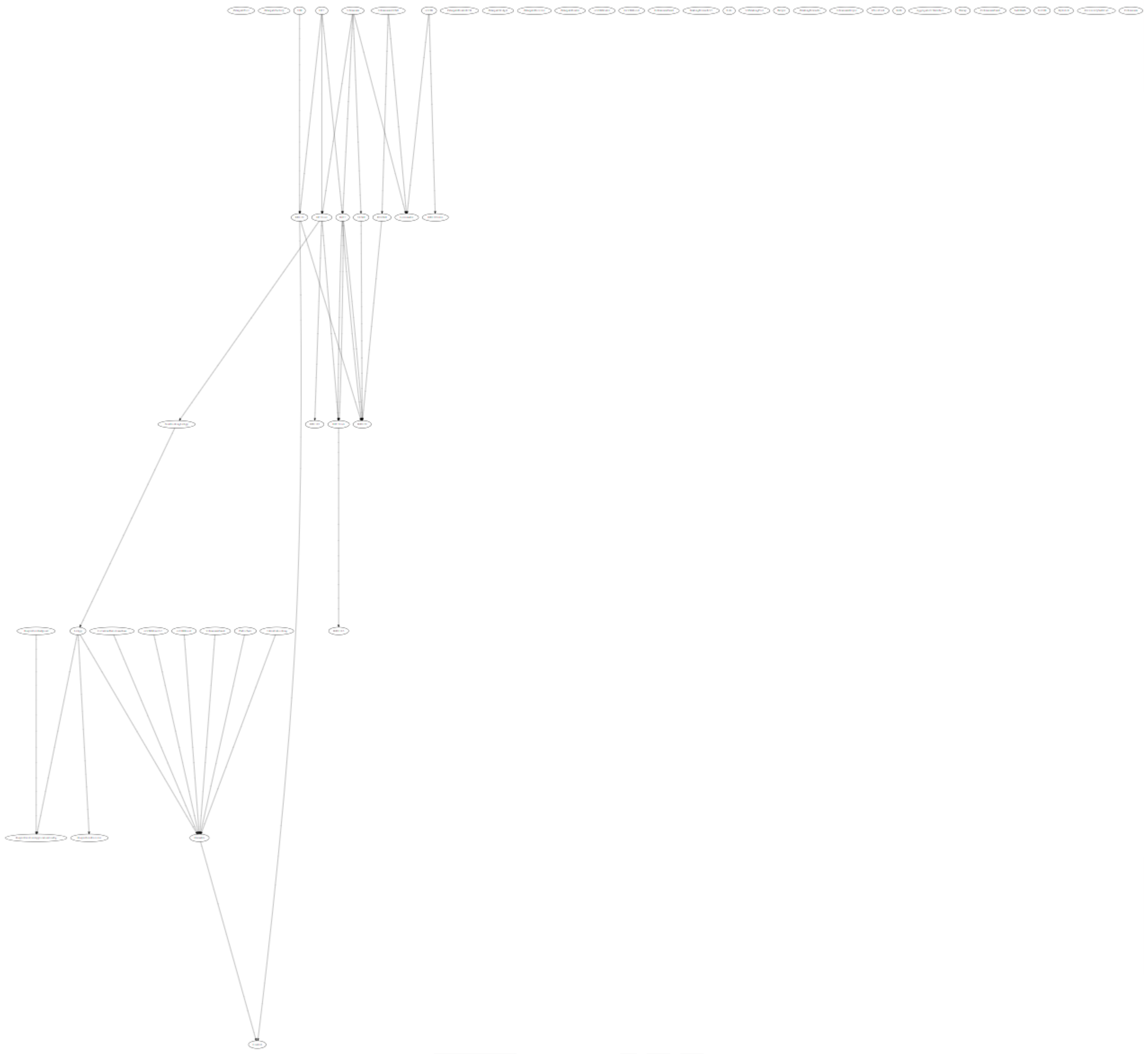
Solidity Versions observed	🧪 Experimental Features	💰 Can Receive Funds	🖥 Uses Assembly	💣 Has Destroyable Contracts
`^0.8.4` `>=0.5.0` `0.8.17` `^0.8.0` `^0.8` `>=0.8.0` `<0.9.0` `>=0.7.6`	`----------`	`yes`	`yes` (19 asm blocks)	`----------`

📤 Transfers ETH	⚡ Low-Level Calls	👥 DelegateCall	🧮 Uses Hash Functions	🔖 ECRecover	🌀 New/Create/Create2
`yes`			`yes`		

♻️ TryCatch	Σ Unchecked
	`yes`

Inheritance Graph

An inheritance graph is a graphical representation of the inheritance hierarchy among contracts. In object-oriented programming, inheritance is a mechanism that allows one class (or contract, in the case of Solidity) to inherit properties and methodsfrom another class. It shows the relationships between different contracts and how they are related to each other through inheritance.



Audit Information

Vulnerability & Risk Level

Risk represents the probability that a certain source threat will exploit the vulnerability and the impact of that event on the organization or system.The risk level is computed based on CVSS version 3.0.

Level	Value	Vulnerability	Risk (Required Action)
Critical	9 - 10	A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken.	Immediate action to reduce risk level.
High	7 – 8.9	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.	Implementation of corrective actions as soon aspossible.
Medium	4 – 6.9	A vulnerability that could affect the desired outcome of executingthe contract in a specific scenario.	Implementation of corrective actions in a certain period.
Low	2 – 3.9	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.	Implementation of certain corrective actions or accepting the risk.
Informational	0 – 1.9	A vulnerability that have informational character but is not effecting any of the code.	An observation that does not determine a level of risk

Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to check the repository for security-related issues, code quality, and compliance with specifications and best practices. To this end, our team of experienced pen-testers andsmart contract developers reviewed the code line by line and documented any issues discovered.

We check every file manually. We use automated tools only so that they help us achieve faster and better results.

Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:
   a. Reviewing the specifications, sources, and instructions provided to
   SolidProof to ensure we understand the size, scope, and functionality of the
   smart contract.
   b. Manual review of the code, i.e., reading the source code line by line to identify potential vulnerabilities.
   c. Comparison to the specification, i.e., verifying that the code does what is described in the specifications, sources, and instructions provided to SolidProof.

2. Testing and automated analysis that includes the following:
   a. Test coverage analysis determines whether test cases cover codeand how much code is executed when those test cases are executed.
   b. Symbolic execution, which is analysing a program to determine what inputs cause each part of a program to execute.

3. Review best practices, i.e., review smart contracts to improve efficiency, effectiveness, clarity, maintainability, security, and control based on best practices, recommendations, and research from industry and academia.

4. Concrete, itemized and actionable recommendations to help you secure your smart contracts.

Overall Security

Upgradeability

Contract is not an upgradable	✅ Deployer cannot update the contract with new functionalities.
Description	The contract is not an upgradeable contract. The Deployer is not able to change or add any functionalities to the contract after deploying.
Comment	N/A

Ownership

The ownership is not renounced	❌ The ownership is not renounced
Description	The owner has not renounced the ownership that means that the owner retains control over the contract's operations, including the ability to execute functions that may impact the contract's users or stakeholders. This can lead to several potential issues, including: • Centralizations • The owner has significant control over contract's operations.
Example	N/A
Comment	N/A

Note – *The contract cannot be considered as renounced till it is not deployed or having some functionality that can change the state of the contract.*

Ownership Privileges

These functions can be dangerous. Please note that abuse can lead to financial loss. We have a guide where you can learn more about these Functions.

Minting tokens

Minting tokens refer to the process of creating new tokens in a cryptocurrency or blockchain network. This process is typically performed by the project's owner or designated authority, who has the ability to add new tokens to the network's total supply.

Contract owner can mint new tokens	❌ **The owner can mint new tokens**
Description	Owners who have the ability to mint new tokens can reward themselves or other stakeholders, who can then sell the newly minted tokens on a cryptocurrency exchange to raise funds. However, there is a risk that the owner may abuse this power, leading to a decrease in trust and credibility in the project or platform. If stakeholders perceive that the owner is using their power to mint new tokens unfairly or without transparency, it can result in decreased demand for the token and a reduction in its value.
Comment	This project contains multiple functions where the minting of tokens is possible without any limit. Also, there are few functions exist where any one can mint tokens. Kindly check the audit result section for all the minting issues.

Burning tokens

Burning tokens is the process of permanently destroying a certain number of tokens, reducing the total supply of a cryptocurrency or token. This is usually done to increase the value of the remaining tokens, as the reduced supply can create scarcity and potentially drive up demand.

Contract owner can burn tokens	❌ **The owner can burn tokens**
Description	In some cases, burning tokens can be used as a tactic by the owner or developers to manipulate the token's value. If the owner or developers burn a significant number of tokens without transparency or justification, it can cause stakeholders to lose trust in the project and lead to a decrease in demand for the token.
Comment	This project contains multiple functions where the burning of tokens is possible without any limit. Also, there are few functions that exist where anyone can burn tokens. Kindly check the audit result section for all the burning issues.

Blacklist addresses

Blacklisting addresses in smart contracts is the process of adding a certain address to a blacklist, effectively preventing them from accessing or participating in certain functionalities or transactions within the contract. This can be useful in preventing fraudulent or malicious activities, such as hacking attempts or money laundering.

Contract owner cannot blacklist addresses	✅ **The owner cannot blacklist addresses**
Description	The owner is not able blacklist addresses to lock funds.
Comment	N/A

Fees and Tax

In some smart contracts, the owner or creator of the contract can set fees for certain actions or operations within the contract. These fees can be used to cover the cost of running the contract, such as paying for gas fees or compensating the contract's owner for their time and effort indeveloping and maintaining the contract.

Contract owner cannot set fees more than 25%.	✅ **The owner cannot set fees more than 25%**
Description	The owner is not able to set the fees above 25%.
Comment	N/A

Lock User Funds

In a smart contract, locking refers to the process of restricting access to certain tokens or assets for a specified period of time. When token or assets are locked in a smart contract, they cannot be transferred or used until the lock-up period has expired or certain conditions have been met.

Contract owner cannot lock user funds.	✅ **The owner cannot lock user funds**
Description	The owner is not able to lock the contract by any functions or updating any variables.
Comment	N/A

Centralization Privileges

Centralization can arise when one or more parties have privileged access or control over the contract's functionality, data, or decision-making. This can occur, for example, if the contract is controlled by a single entity or if certain participants have special permissions or abilities that others do not.

In the project, there are authorities that have access to the following functions:

File	Privileges
CollateralRatioGuardian.sol	➢ The owner can set fees between 100 to 500 vUSD.
VibraniumvUSD.sol	➢ The governor can set mint fee apy not greater than 1.5%. ➢ The governor can set safe collateral rate not less than 160 * 1e18. ➢ The governor can set keepers rate not more than 5%. ➢ The governor can set redemption fee not more than 5%. ➢ The governor can set service fee pool address. ➢ The governor can set esvib minter address.
VibraniumFund.sol	➢ The owner can set vibranium token address. ➢ The owner can set esVIB and VIB token address. ➢ The owner can set any number to claimable Time.
Vibranium.sol	➢ The governor can set mint fee apy not greater than 1.5%. ➢ The governor can set safe collateral rate not less than 160 * 1e18. ➢ The governor can set keepers rate not more than 5%. ➢ The governor can set redemption fee not more than 5%. ➢ The governor can set service fee pool address. ➢ The governor can set esvib minter address.
VIB.sol	➢ The owner can set a vibranium fund address.
TokenUnlocking.sol	➢ The owner can set unlock rule.
StakerewardV2pool.sol	➢ The owner can set esVIB boost address.

	➢ The owner can set the mining rewards.
PublicSale.sol	➢ The owner can set price before the IDO is started. ➢ The owner can set the start time and end time. ➢ The owner can withdraw ether only after the IDO is started and cannot withdraw if the softcapmet is set to false. ➢ The owner can send the stuck tokens.
OFTCore.sol	➢ The owner can set true/false to useCustomAdapterParams.
LzApp.sol	➢ The owner can set config to lzEndpoint. ➢ The owner can set send version,receive version, forceResumereceive, in lzEndpoint. ➢ The owner can set trusted remote chainId and path. ➢ The owner can set trusted remote address. ➢ The owner can set Precrime address. ➢ The owner can set minimum distribution gas. ➢ The owner can set payload size limit.
esVIBMinerV2.sol	➢ The owner can set esVIB address. ➢ The owner can set any number to extra rate. ➢ The owner can set any time period in the setLockdown Period. ➢ The owner can set esVIBBoost address. ➢ The owner can set vibranium fund address. ➢ The owner can set the reward duration period.
esVIBBoost.sol	➢ The owner can set lock settings.
esVIB.sol	➢ The governor can set minter role to multiple addresses.

Recommendations

To avoid potential hacking risks, it is advisable for the client to manage the private key of the privileged account with care. Additionally, we recommend enhancing the security practices of centralized privileges or roles in the protocol through a decentralized mechanism or smart-contract-based accounts, such as multi-signature wallets.

Here are some suggestions of what the client can do:

- Consider using multi-signature wallets: Multi-signature wallets require multiple parties to sign off on a transaction before it can be executed, providing an extra layer of security e.g. Gnosis Safe
- Use of a timelock at least with a latency of e.g. 48-72 hours for awareness of privileged operations
- Introduce a DAO/Governance/Voting module to increase transparency and user involvement
- Consider Renouncing the ownership so that the owner cannot modify any state variables of the contract anymore. Make sure to set up everything before renouncing.

Audit Result

#1 | Authorized addresses can burn tokens.

File	Severity	Location	Status
esVIB.sol	High	L55-62	ACK

Description

- Governor Wallet can whitelist addresses and can burn tokens without any allowance from any wallet address.

#2 | Authorized addresses can mint unlimited tokens.

File	Severity	Location	Status
esVIB.sol	Medium	L41-53	ACK

Description

- Governor can whitelist addresses to mint tokens without any limit.

#3 | Missing Events Arithmetic

File	Severity	Location	Status
esVIB.sol	Low	L27,28,37	ACK

Description

- Emit an event for critical parameter changes.

#4 | Missing require check

File	Severity	Location	Status
esVIBBoost.sol	Low	L39-46	ACK

Description

- Id is less than esVIBSettings require check is missing.

#5 | Missing require check

File	Severity	Location	Status
esVIBBoost.sol	Low	L39-46	ACK

Description

- userUpdatedAt must be greater than time otherwise the userBoost will fail. Please add a require check that it must be greater than time.

#6 | Owner can manipulate reward amount

File	Severity	Location	Status
esVIBMinerV2.sol	Medium	L209	ACK

Description

- owner can set any number in the duration which will manipulate the reward amount.

#7 | Missing require check

File	Severity	Location	Status
esVIBMinerV2.sol	Low	L132	ACK

Description

- Add a require check that the lastTimeRewardApplicable() must be greater than updatedAt.

#8 | Missing Events Arithmetic

File	Severity	Location	Status
Governable.sol	Low	L12-14	ACK

Description

- Emit an event for critical parameter changes.

#9 | Missing zero address validation

File	Severity	Location	Status
Governable.sol	Low	L12-14	ACK

Description

- Check that the address is not zero.

#10 | Missing Renounce Function

File	Severity	Location	Status
Governable.sol	Low	--	ACK

Description

- Add a renounce function to renounce governor access.

#11 | Missing Immutable

File	Severity	Location	Status
PublicSale.sol	Low	L9	ACK

Description

- Add immutable on the variables that will not going to be changed after deployment.

#12 | Missing require check

File	Severity	Location	Status
PublicSale.sol	Low	L26	ACK

Description

- Add a require check that the startTime must be less than endTime.

#13 | Missing Error message

File	Severity	Location	Status
PublicSale.sol	Low	L39,112	ACK

Description

- Add the error message on the require check.

#14 | Missing events Arithmetic

File	Severity	Location	Status
PublicSale.sol	Low	L42,113	ACK

Description

- Emit an event for critical parameter changes.

#15 | Add different Name and symbol

File	Severity	Location	Status
RVUSD.sol	Informational	L91	ACK

Description

- Add a different name and symbol.

#16 | Prevent Approve Frontrunning

File	Severity	Location	Status
RVUSD.sol	Informational	L91	ACK

Description

- Always try to prevent the contract from frontrunning. Here, Is the article to go through it. (https://github.com/OpenZeppelin/openzeppelin-contracts/issues/599)

#17 | Missing Timelock in the Withdraw Function

File	Severity	Location	Status
StakerewardV2pool.sol	Medium	L110	ACK

Description

The contract misses a timelock in the stake/withdraw function. This means that the withdraw function can be called recursively. We recommend putting a timelock so that the deposit/withdraw function cannot be called by an external contract recursively.

#18 | Missing zero address validation

File	Severity	Location	Status
StakerewardV2pool.sol	Low	L54,55,56,57	ACK

Description

- Check that the address is not zero.

#19 | Missing require check

File	Severity	Location	Status
StakerewardV2pool.sol	Low	L97	ACK

Description

- Add a require check that the lastTimeRewardApplicable() must be greater than updatedAt.

#20 | Missing events Arithmetic

File	Severity	Location	Status
StakerewardV2pool.sol	Low	L114	ACK

Description

- Emit an event for critical parameter changes.

#21 | Anyone can withdraw tokens for other user.

File	Severity	Location	Status
TokenUnlocking.sol	Medium	L101-132	ACK

Description

- Add a ‘require’ check that no one apart from the user can withdraw their tokens.

#22 | Missing zero check

File	Severity	Location	Status
TokenUnlocking.sol	Low	L114	ACK

Description

- Check that the value is not zero.

#23 | Vibranium Fund address can burn tokens.

File	Severity	Location	Status
VIB.sol	High	L39-43	ACK

Description

- The vibranium fund address can burn tokens without any allowance from any wallet address.

#24 | Vibranium fund address can mint unlimited tokens.

File	Severity	Location	Status
VIB.sol	Medium	L32-37	ACK

Description

- The Vibranium fund address can mint tokens without any limit.

#25 | Missing events Arithmetic

File	Severity	Location	Status
VIB.sol	Low	L29,46	ACK

Description

- Emit an event for critical parameter changes.

#26 | Vibranium Fund address can burn tokens.

File	Severity	Location	Status
Vibranium.sol	High	L297-298	ACK

Description

- Addresses can burn tokens without any allowance from any wallet address.

#27 | Anyone can withdraw tokens for other user.

File	Severity	Location	Status
Vibranium.sol	Medium	L260-270	ACK

Description

- Add a 'require' check that no one apart from the user can withdraw their tokens.

#28 | Addresses can mint unlimited tokens.

File	Severity	Location	Status
Vibranium.sol	Medium	L280-287	ACK

Description

- Addresses can mint tokens without any limit.

#29 | Missing Visibility

File	Severity	Location	Status
Vibranium.sol	Low	L38	ACK

Description

- Add 'public' or 'private' during the state variable initialization.

#30 | Missing zero address validation

File	Severity	Location	Status
Vibranium.sol	Low	L105	ACK

Description

- Check that the address is not zero.

#31 | Missing events Arithmetic

File	Severity	Location	Status
Vibranium.sol	Low	L147,174, 178, 182, 552	ACK

Description

- Emit an event for critical parameter changes.

#32 | Missing zero check

File	Severity	Location	Status
Vibranium.sol	Low	L319,368,471,570, 556	ACK

Description

- Add a 'require' check that borrowed cannot be zero.

#33 | Missing require check

File	Severity	Location	Status
Vibranium .sol	Low	L558	ACK

Description

- Add a ‘require’ check that the minting is not paused otherwise it will revert.

#34 | Addresses can mint unlimited tokens.

File	Severity	Location	Status
VibraniumFund.sol	Medium	L77-83	ACK

Description

- Addresses can mint tokens without any limit.

#35 | Missing events Arithmetic

File	Severity	Location	Status
VibraniumFund.sol	Low	L38,43,47	ACK

Description

- Emit an event for critical parameter changes.

#36 | Missing zero check

File	Severity	Location	Status
VibraniumFund.sol	Low	L38, 42	ACK

Description

- Add a ‘require’ check that value cannot be zero.

#37 | Missing Error message

File	Severity	Location	Status
VibraniumFund.sol	Low	L139	ACK

Description

- Add the error message on the require check.

#38 | Missing zero check

File	Severity	Location	Status
VibraniumHelper.sol	Low	L20	ACK

Description

- Add a ‘require’ check that address cannot be set to zero.

#39 | Missing zero check

File	Severity	Location	Status
VibraniumHelper.sol	Low	L140	ACK

Description

- Add a ‘require’ check that value cannot be zero.

#40 | High Gas Fees ignore looping.

File	Severity	Location	Status
VibraniumHelper.sol	Informational	--	ACK

Description

- Always try to ignore the looping as it can affect a very high gas fees and will throw an error of insufficient gas and the transaction can get failed.

#41 | Addresses can mint tokens.

File	Severity	Location	Status
VibraniumvUSD.sol	Medium	L209-216	ACK

Description

- Any address can mint tokens of 10% of total circulation.

#42 | Missing Visibility

File	Severity	Location	Status
VibraniumvUSD.sol	Low	L27,28,33	ACK

Description

- Add ‘public’ or ‘private’ during the state variable initialization.

#43 | Missing zero check

File	Severity	Location	Status
VibraniumvUSD.sol	Low	L95,101, 149,248, 260, 324,428, 451, 505	ACK

Description

- Add a ‘require’ check that value cannot be zero.

#44 | Missing Error message

File	Severity	Location	Status
VibraniumvUSD.sol	Low	L100	ACK

Description

- Add the error message on the require check.

#45 | Add different Name and symbol

File	Severity	Location	Status
VUSD.sol	Informational	L91	ACK

Description

- Add a different name and symbol.

#46 | NatSpec Documentation missing

File	Severity	Location	Status
ALL	Informational	--	ACK

Description

- If you started to comment on your code, also comment on all other functions, variables, etc.

Legend for the Issue Status

Attribute or Symbol	Meaning
Open	The issue is not fixed by the project team.
Fixed	The issue is fixed by the project team.
Acknowledged(ACK)	The issue has been acknowledged or declared as part of business logic.

Solid Proofed

Blockchain Security | Smart Contract Audits | KYC
Development | Marketing

MADE IN GERMANY