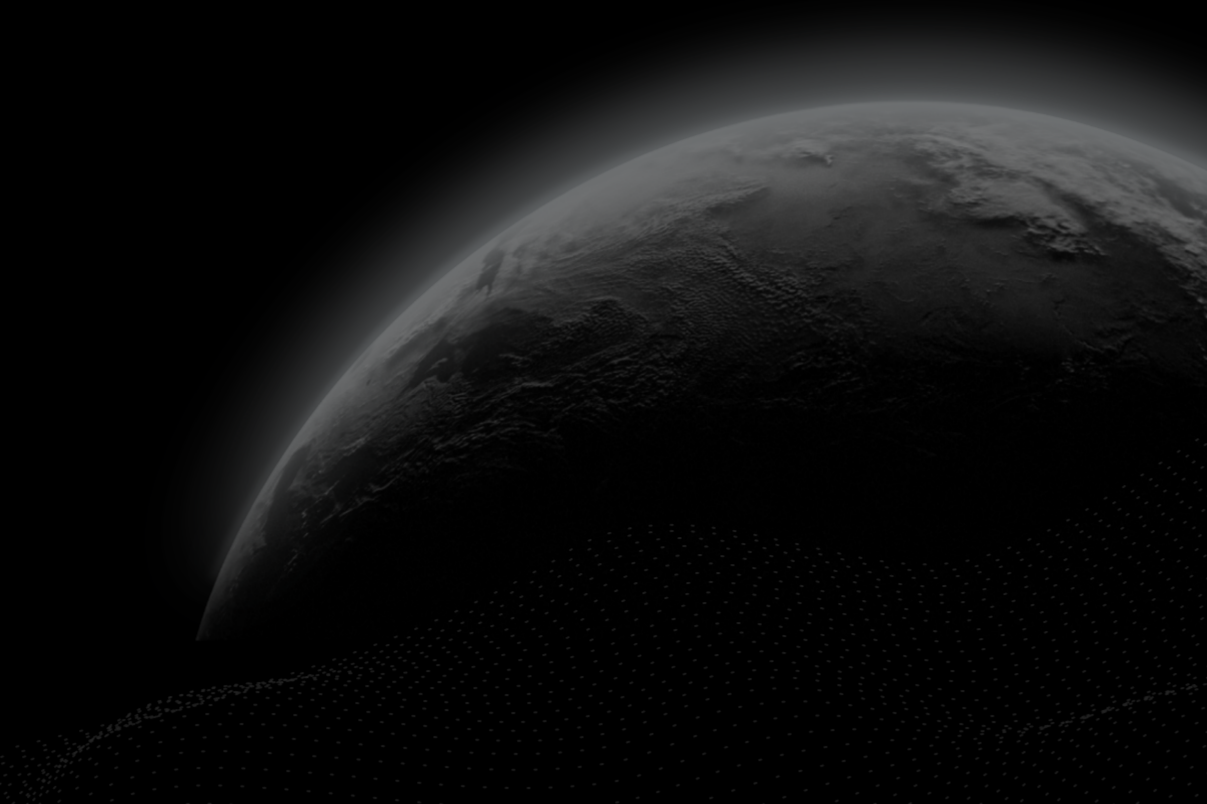
CERTIK

Preliminary Comments

NodeReal - BAS

CertiK Verified on Sept 8th, 2022



CertiK Verified on Sept 8th, 2022

NodeReal - BAS

The security assessment was prepared by CertiK, the leader in Web3.0 security.

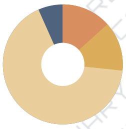
Executive Summary

TYPES	ECOSYSTEM	METHODS
Chain, Chain-Consensus, Other-Contract	Binance Smart Chain (BSC)	Manual Review, Static Analysis

LANGUAGE	TIMELINE	KEY COMPONENTS
Golang, Solidity	Delivered on 09/08/2022	N/A

CODE BASE	CODE BASE
github.com/node-real/semita-bas-genesis-config/tree/652…	N/A

Vulnerability Summary



15	0	0	0	0	0	15
Total Findings	Resolved	Mitigated	Partially Resolved	Acknowledged	Declined	Unresolved

🟥	0	Critical		Critical risks are those that impact the safe functioning of a platform and must be addressed before launch. Users should not invest in any project with outstanding critical risks.
🟧	2	Major	2 Unresolved	Major risks can include centralization issues and logical errors. Under specific circumstances, these major risks can lead to loss of funds and/or control of the project.
🟨	2	Medium	2 Unresolved	Medium risks may not pose a direct risk to users' funds, but they can affect the overall functioning of a platform.
🟨	10	Minor	10 Unresolved	Minor risks can be any of the above, but on a smaller scale. They generally do not compromise the overall integrity of the project, but they may be less efficient than other solutions.
🟦	1	Informational	1 Unresolved	Informational errors are often recommendations to improve the style of the code or certain operations to fall within industry best practices. They usually do not affect the overall functioning of the code.
⬜	0	Discussion		The impact of the issue is yet to be determined, hence requires further clarifications from the project team.

TABLE OF CONTENTS | NODEREAL - BAS

CODE BASE | NODEREAL - BAS

Repository

https://github.com/node-real/semita-bas-genesis-config/tree/652deaeea4b4197e3d02d163d6e76b58569ce000

https://github.com/node-real/semita-bas-template-bsc/tree/4ef88cb69162d1b28a502118ddd418176cb47fab

AUDIT SCOPE | NODEREAL - BAS

17 files audited ● 9 files with Unresolved findings ○ 8 files without findings

ID	File	SHA256 Checksum
● TXP	core/tx_pool.go	a515eaab4a08eb42aaa6ef8c8e2f062b2bf3adbfa825fbfee087b40a360f6911
● BAC	eth/backend.go	e52785358d0236a28cd94a4ec70e48eeed1d7712fa7b0a144d6553d93aa1e34a
● CRE	create-genesis.go	93e3d6a19fad39e3396e8c48573609b387918b4d6c830d7a1a0308b26f3b5890
● CCB	contracts/ChainConfig.sol	c3be0557947f8e8af0840d62a9f1491158d919ac3fbc821429ac5a71786a42eb
● RES	contracts/Reserve.sol	bb487f38737d07a04e87264b5899c94b651a93cc702b0d79f2acbd932aab52a2
● REW	contracts/Reward.sol	1d3f541196183f6ba3a73ec49afd70d0fa981465e1a2f1a23208938883e4cbab
● SPB	contracts/StakingPool.sol	7cd12d26b5f2fa7d913dfd7ff84c371b8b401f08e2782be78fddc67eb5027a0f
● SRB	contracts/SystemReward.sol	5bd0c2bac7b2402171b073ed937d78b6dcb6cef25ca41ec58c368367a557e8de
● TLB	contracts/TimeLock.sol	3b49fd1d9880b4b9e3a9154f7530fc1a1cd400af29684595f594ea078bc868a4
○ CON	common/systemcontract/const.go	1d5904e1a07eb9527a99c7ce23011babf9457eafd008c7a595fc2923c915152c
○ PAR	consensus/parlia/parlia.go	78a3ea99c2bfd278dd95e915b685a92d929b50668a8822e4b3e7dcba510d8432
○ DPB	contracts/DeployerProxy.sol	ce4331ec3d14841d5075a753e8518a29fde7635fd40fb71966bdd9ec4d63ef75
○ GOV	contracts/Governance.sol	776505f816a9c4cf74075b1ea088db39caa61b5c50cd12b7afabfab8f9722440
○ ICH	contracts/InjectorContextHolder.sol	59cbc8adae75619c477c6249142c5771a928e1301b1fe9103c716c15653c9928

ID	File	SHA256 Checksum
● RUB	contracts/RuntimeUpgrade.sol	407c9b4c24573cfe284a57194cede8e0b1b164a9f292ad606a99159586f0d862
● SIB	contracts/SlashingIndicator.sol	4572e0dfb21c03bd7153f3ee787fef46e15edec38f680cb7f24bdb1b1210dd39
● STA	contracts/Staking.sol	b9f35efd61b4502075a6b3dd21729fc1c4c22eec21d0eeb31632a4054d68f59f

APPROACH & METHODS | NODEREAL - BAS

This report has been prepared for NodeReal to discover issues and vulnerabilities in the source code of the NodeReal - BAS project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Manual Review and Static Analysis techniques.

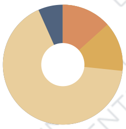
The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Testing the smart contracts against both common and uncommon attack vectors;
- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

FINDINGS | NODEREAL - BAS



15	0	2	2	10	1	0
Total Findings	Critical	Major	Medium	Minor	Informational	Discussion

This report has been prepared to discover issues and vulnerabilities for NodeReal - BAS. Through this audit, we have uncovered 15 issues ranging from different severity levels. Utilizing Static Analysis techniques to complement rigorous manual code reviews, we discovered the following findings:

ID	Title	Category	Severity	Status
BAC-01	Returned Error Not Checked	Control Flow	Minor	● Unresolved
CCB-01	**Centralization Risks In ChainConfig.Sol**	**Centralization / Privilege**	**Major**	● **Unresolved**
CCB-02	Potential DOS Attack	Logical Issue	Informational	● Unresolved
CON-01	Unprotected Upgradeable Contract	Language Specific	Minor	● Unresolved
CON-02	Missing Zero Address Validation	Volatile Code	Minor	● Unresolved
CON-03	Usage Of `transfer` / `send` For Sending Ether	Volatile Code	Minor	● Unresolved
CRE-01	Incorrect Comment About Token Amount	Inconsistency	Minor	● Unresolved
REW-01	`burnAndRelease()` Algorithm Is Not Deterministic	Logical Issue	Medium	● Unresolved
REW-02	Unnecessary Use Of `return`	Language Specific	Minor	● Unresolved
TLB-01	**Centralization Risks In TimeLock.Sol**	**Centralization / Privilege**	**Major**	● **Unresolved**
TLB-02	Lack Of Storage Gap In Upgradeable Contracts	Language Specific	Medium	● Unresolved

ID	Title	Category	Severity	Status
TLB-03	Unused Import Library `Strings.sol`	Volatile Code	Minor	● Unresolved
TLB-04	Unused State Variable	Volatile Code	Minor	● Unresolved
TLB-05	Missing Zero Address Validation On `admin`	Logical Issue	Minor	● Unresolved
TXP-01	Local Accounts Can Not Be Added/Removed To/From Gas Free Account Set	Logical Issue	Minor	● Unresolved

BAC-01 | FINDING DETAILS

Finding Title

Returned Error Not Checked

Category	Severity	Location	Status
Control Flow	● Minor	eth/backend.go (template-bsc-v1): 615~616	● Pending

Description

The error returned by `abi.JSON()` is not checked. If `abi.JSON()` returns error, the current function should return with the error immediately. Otherwise, invalid `chainConfig` will be used by subsequent code and may cause confusion since different error may be returned.

Recommendation

We recommend handling returned error properly.

CCB-01 | FINDING DETAILS

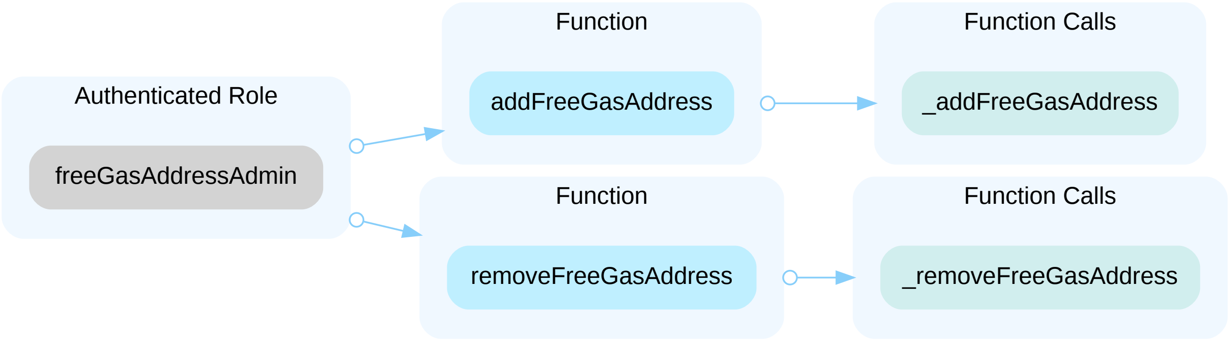
Finding Title

Centralization Risks In ChainConfig.Sol

Category	Severity	Location	Status
Centralization / Privilege	● **Major**	**contracts/ChainConfig.sol (genesis-config-v1): 203, 215**	● **Pending**

Description

In the contract `ChainConfig` the role `freeGasAddressAdmin` has authority over the functions shown in the diagram below. Any compromise to the `freeGasAddressAdmin` account may allow the hacker to take advantage of this authority and add/remove gas free addresses at will.



Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets. Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND

- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

CCB-02 | FINDING DETAILS

Finding Title

Potential DOS Attack

Category	Severity	Location	Status
Logical Issue	● Informational	contracts/ChainConfig.sol (genesis-config-v1): 203	● Pending

Description

The owners of gas free addresses can send transactions without any cost. So if they become malicious or their private keys are stolen, the hackers can shut down the blockchain network by flooding the network with numerous gas free transactions.

Recommendation

We recommend carefully protecting private keys of gas free addresses.

CON-01 | FINDING DETAILS

Finding Title

Unprotected Upgradeable Contract

Category	Severity	Location	Status
Language Specific	● Minor	contracts/ChainConfig.sol (genesis-config-v1): 67; contracts/Reserve.sol (genesis-config-v1): 35; contracts/Reward.sol (genesis-config-v1): 49	● Pending

Description

"Do not leave an implementation contract uninitialized. An uninitialized implementation contract can be taken over by an attacker, which may impact the proxy." See https://docs.openzeppelin.com/upgrades-plugins/1.x/writing-upgradeable#initializing_the_implementation_contract

Recommendation

We recommend invoking the `_disableInitializers()` function in the constructor to automatically lock it when it is deployed.

CON-02 | FINDING DETAILS

Finding Title

Missing Zero Address Validation

Category	Severity	Location	Status
Volatile Code	● Minor	contracts/ChainConfig.sol (genesis-config-v1): 96, 188, 207; contracts/Reward.sol (genesis-config-v1): 50, 69	● Pending

Description

Addresses should be checked before assignment or external call to make sure they are not zero addresses.

```
96          freeGasAddressAdmin = _freeGasAddressAdmin;
```

- `_freeGasAddressAdmin` is not zero-checked before being used.

```
50          foundationAddress = _foundationAddress;
```

- `_foundationAddress` is not zero-checked before being used.

```
69          foundationAddress = _foundationAddress;
```

- `_foundationAddress` is not zero-checked before being used.

Recommendation

We advise adding a zero-check for the passed-in address value to prevent unexpected errors.

CON-03 | FINDING DETAILS

Finding Title

Usage Of `transfer` / `send` For Sending Ether

Category	Severity	Location	Status
Volatile Code	● Minor	contracts/Reserve.sol (genesis-config-v1): 39; contracts/Reward.sol (genesis-config-v1): 138; contracts/StakingPool.sol (genesis-config-v1): 188; contracts/SystemReward.sol (genesis-config-v1): 126	● Pending

Description

It is not recommended to use Solidity's `transfer()` and `send()` functions for transferring Ether, since some contracts may not be able to receive the funds. Those functions forward only a fixed amount of gas (2300 specifically) and the receiving contracts may run out of gas before finishing the transfer. Also, EVM instructions' gas costs may increase in the future. Thus, some contracts that can receive now may stop working in the future due to the gas limitation.

```
39          payable(address(addr)).transfer(amount);
```

- `Reserve.release` uses `transfer()` .

```
138             payable(deadAddress).transfer(burned);
```

- `Reward.burnAndRelease` uses `transfer()` .

```
188         payable(address(msg.sender)).transfer(amount);
```

- `StakingPool.claim` uses `transfer()` .

```
126             payableTreasury.transfer(amountToPay);
```

- `SystemReward._claimSystemFee` uses `transfer()` .

Recommendation

We recommend using the `Address.sendValue()` function from OpenZeppelin.

Since `Address.sendValue()` may allow reentrancy, we also recommend guarding against reentrancy attacks by utilizing the Checks-Effects-Interactions Pattern or applying OpenZeppelin ReentrancyGuard.

CRE-01 | FINDING DETAILS

Finding Title

Incorrect Comment About Token Amount

Category	Severity	Location	Status
Inconsistency	● Minor	create-genesis.go (genesis-config-v1): 491, 492	● Pending

Description

The relevant token amount number in code is actually in wei, NOT in ether.

Recommendation

We recommend changing the identified comments to `(in wei)`.

REW-01 | FINDING DETAILS

Finding Title

`burnAndRelease()` Algorithm Is Not Deterministic

Category	Severity	Location	Status
Logical Issue	● Medium	contracts/Reward.sol (genesis-config-v1): 132~147	● Pending

Description

The result depends on timing of transactions. It is possible that the same queue of transactions can lead to different results due to different timing of transactions. For example, if the timing of calls to `burnAndRelease()` is carefully designed, it is possible that no token will be burned. This behavior will make the promise of deflationary token economy broken.

Recommendation

We recommend reviewing the `burnAndRelease()` function to make sure it works as intended.

REW-02 | FINDING DETAILS

Finding Title

Unnecessary Use Of `return`

Category	Severity	Location	Status
Language Specific	● Minor	contracts/Reward.sol (genesis-config-v1): 78, 101, 125	● Pending

Description

The function `cancelTransaction()` returns nothing. Thus the use of `return` is unnecessary.

Recommendation

We recommend removing the use of `return`.

TLB-01 | FINDING DETAILS

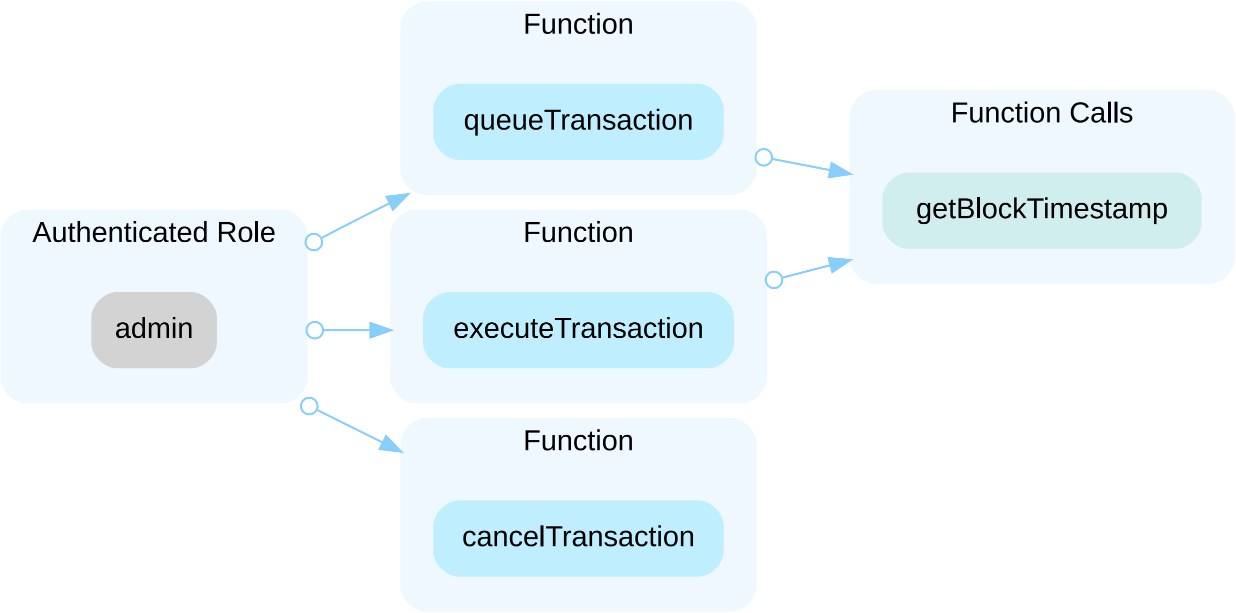
Finding Title

Centralization Risks In TimeLock.Sol

Category	Severity	Location	Status
Centralization / Privilege	● **Major**	**contracts/TimeLock.sol (genesis-config-v1): 124, 144, 159**	● **Pending**

Description

In the contract `TimeLock` the role `admin` has authority over the functions shown in the diagram below. Any compromise to the `admin` account may allow the hacker to take advantage of this authority and change admin and delay at will.



Recommendation

The risk describes the current project design and potentially makes iterations to improve in the security operation and level of decentralization, which in most cases cannot be resolved entirely at the present stage. We advise the client to carefully manage the privileged account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., multisignature wallets. Indicatively, here are some feasible suggestions that would also mitigate the potential risk at a different level in terms of short-term, long-term and permanent:

Short Term:

Timelock and Multi sign (⅔, ⅗) combination *mitigate* by delaying the sensitive operation and avoiding a single point of key management failure.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Assignment of privileged roles to multi-signature wallets to prevent a single point of failure due to the private key compromised;
  AND
- A medium/blog link for sharing the timelock contract and multi-signers addresses information with the public audience.

Long Term:

Timelock and DAO, the combination, *mitigate* by applying decentralization and transparency.

- Time-lock with reasonable latency, e.g., 48 hours, for awareness on privileged operations;
  AND
- Introduction of a DAO/governance/voting module to increase transparency and user involvement.
  AND
- A medium/blog link for sharing the timelock contract, multi-signers addresses, and DAO information with the public audience.

Permanent:

Renouncing the ownership or removing the function can be considered *fully resolved*.

- Renounce the ownership and never claim back the privileged roles.
  OR
- Remove the risky functionality.

TLB-02 | FINDING DETAILS

Finding Title

Lack Of Storage Gap In Upgradeable Contracts

Category	Severity	Location	Status
Language Specific	● Medium	contracts/TimeLock.sol (genesis-config-v1): 196	● Pending

Description

For upgradeable contracts, there must be storage gap to "allow developers to freely add new state variables in the future without compromising the storage compatibility with existing deployments". Otherwise it may be very difficult to write new implementation code. See https://docs.openzeppelin.com/contracts/4.x/upgradeable#storage_gaps

Recommendation

We recommend adding storage gap at the end of upgradeable contracts.

TLB-03 | FINDING DETAILS

Finding Title

Unused Import Library `Strings.sol`

Category	Severity	Location	Status
Volatile Code	● Minor	contracts/TimeLock.sol (genesis-config-v1): 15	● Pending

Description

The imported library `Strings.sol` is not used in the contract.

Recommendation

We recommend removing the unused import.

TLB-04 | FINDING DETAILS

Finding Title

Unused State Variable

Category	Severity	Location	Status
Volatile Code	● Minor	contracts/TimeLock.sol (genesis-config-v1): 53	● Pending

Description

The state variable `_admin_initialized` is not used in the contract.

Recommendation

We recommend removing the unused variable.

TLB-05 | FINDING DETAILS

Finding Title

Missing Zero Address Validation On `admin`

Category	Severity	Location	Status
Logical Issue	● Minor	contracts/TimeLock.sol (genesis-config-v1): 65	● Pending

Description

`admin` can not be invalid zero address. Otherwise, the modifier `onlyAdmin` and functions `queueTransaction()/cancelTransaction()/executeTransaction()` will not work.

Recommendation

We recommend adding a check to make sure `admin` is not zero address.

TXP-01 | FINDING DETAILS

Finding Title

Local Accounts Can Not Be Added/Removed To/From Gas Free Account Set

Category	Severity	Location	Status
Logical Issue	● Minor	core/tx_pool.go (template-bsc-v1): 723~726, 1282	● Pending

Description

If a local account becomes a gas free account, it can not be added/removed to/from gas free account set. And local account set can not shrink. This may deviate from intended design.

Recommendation

We recommend reviewing the mentioned situation and make sure it is intended.

APPENDIX | NODEREAL - BAS

Finding Categories

Categories	Description
Centralization / Privilege	Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.
Logical Issue	Logical Issue findings detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.
Control Flow	Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.
Volatile Code	Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in a vulnerability.
Language Specific	Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete.
Inconsistency	Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing different require statements on the input variables than a setter function.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

DISCLAIMER | CERTIK

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without CertiK's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing 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. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

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

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER'S OR ANY OTHER PERSON'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY

KIND THAT THE SERVICE WILL MEET CUSTOMER'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK'S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER'S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED "AS IS" AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK'S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

CertiK | Securing the Web3 World

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.