

Redbelly Network (Layer 1)

SMART CONTRACT

Security Audit

Performed on Contracts:

ActivityMonitor.sol
Authorizable.sol
BootstrapContractRegistry.sol
GasFees.sol
IDPRegistry.sol
MockPriceFeed.sol
JailedGovernors.sol

MockRandomNumberGenerator.sol
NetworkConfiguration.sol
Permission.sol
RBAC.sol
Reconfiguration.sol
StakingDeposit.sol
Voting.sol

Platform

EVM

hashlock.com.au

JULY 2023

Table of Contents

Executive Summary	4
Project Context	4
Redbelly Network Project Leadership	7
Smart Contract Audit Scope	8
Penetration Test Scope	10
Security Rating	11
Standardised Checks	13
Intended Smart Contract Functions	15
Function List	18
Code Quality	24
Audit Resources	25
Dependencies	25
Severity Definitions	26
Smart Contract Audit Findings	26
Additional Contracts Smart Contract Audit Findings	60
Final Audit Round Findings	73
Penetration Test Summary	90
Penetration Test Findings	92
Centralisation	112
Conclusion	113
Our Methodology	114
Disclaimers	116
About Hashlock	117

CAUTION

THIS DOCUMENT IS A SECURITY AUDIT REPORT AND MAY CONTAIN CONFIDENTIAL INFORMATION. THIS INCLUDES IDENTIFIED VULNERABILITIES AND MALICIOUS CODE WHICH COULD BE USED TO COMPROMISE THE PROJECT. THIS DOCUMENT SHOULD ONLY BE FOR INTERNAL USE UNTIL ISSUES ARE RESOLVED. ONCE VULNERABILITIES ARE REMEDIATED, THIS REPORT CAN BE MADE PUBLIC. THE CONTENT OF THIS REPORT IS OWNED BY HASHLOCK PTY LTD FOR USE OF THE CLIENT.

Executive Summary

The Redbelly Network team partnered with Hashlock to conduct a penetration test of their layer 1 network and a smart contract audit of their embedded smart contracts. Hashlock manually and proactively reviewed the code in order to ensure the Redbelly Network team and community is ready for mainnet deployment.

Project Context

Redbelly Network is a soon to launch Layer 1 Network that focuses on real world assets, KYC, institutional use cases and compliance. The Redbelly Network was born out of the University of Sydney and the CSIRO, and has grown into a fully fledged team who has made innovations into the technology and verified them via formal verification and extensive auditing. The Redbelly Network team engaged Hashlock to ensure that their protocol is ready for launch.

Project Name: Redbelly Network

Properties: SEVM Compatible, Golang and Solidity Code

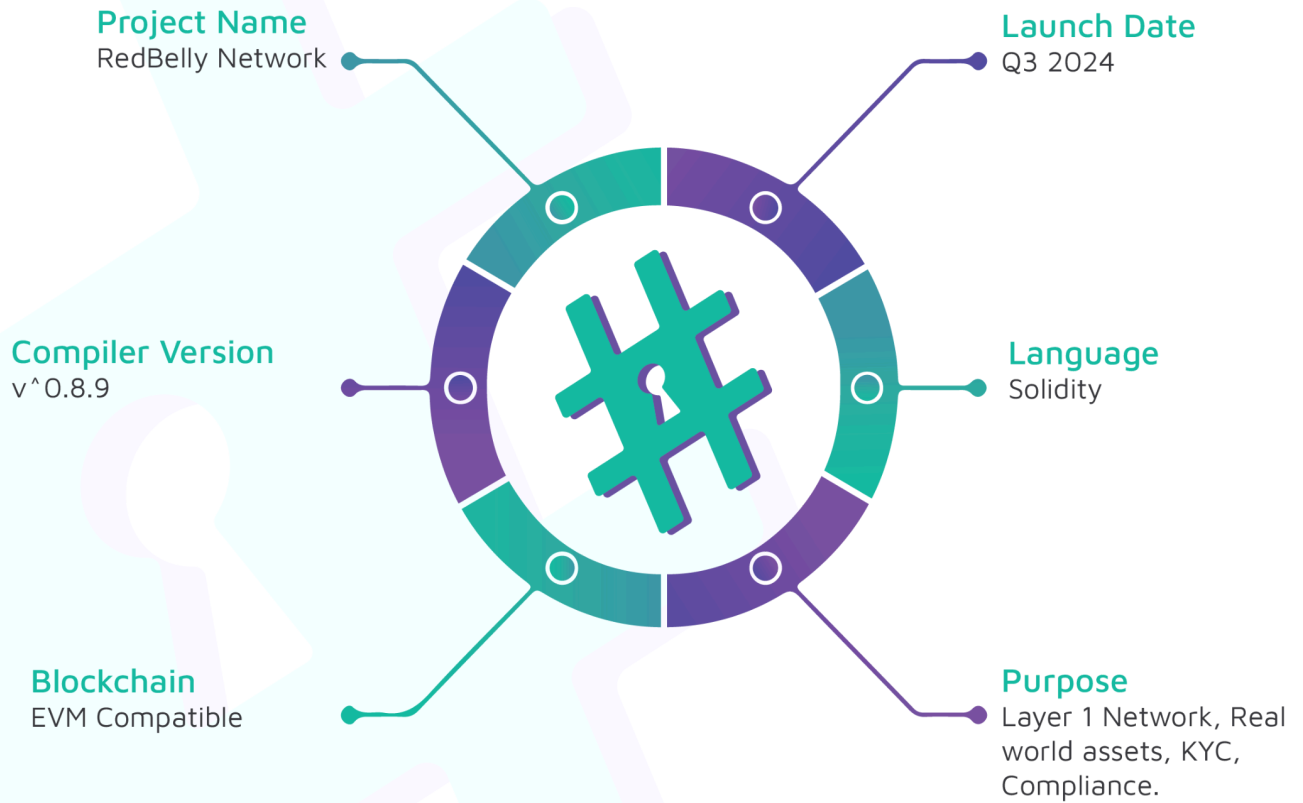
Compiler Version: ^0.8.9

Logo:



REDBELLY

Visualised Context:



Project Visuals:

REDBELLY

C.A.T. Solution Research Developers Blog Events About Programs Careers [Contact](#)

Compliant Asset Tokenisation (C.A.T.)

Regulated real world assets on the Blockchain?
Meet the world's **first C.A.T. platform.**

What it is How it works
Redbelly's Compliant Asset Tokenisation Solution
Vincent Gramoli
Redbelly Founder & CEO

Tokenisation and Compliance
Redbelly's Compliant Asset Tokenisation Solution
Warren Birington
Redbelly CEO

Scientific Research

Redbelly is unique in that the science came first. The conception, creation and development of the Redbelly Network followed rigorous scientific method, years of careful, methodical, peer-reviewed, published and proven research.

[The Redbelly Network: Whitepaper](#)
[DBFT: Efficient Leaderless Byzantine Consensus](#)
[Redbelly: A Secure, Fair and Scalable Open Blockchain](#)

[Read All](#)

REDBELLY

C.A.T. Solution Research Developers Blog Events About Programs Careers [Contact](#)




Our Research


Our Compliant Asset Tokenisation solution is built on rigorous scientific research.

The Redbelly Network results from peer-reviewed, proven results in the lab from Australia's most distinguished experts in the field of distributed systems.

[Our Whitepaper](#)

Redbelly Network Project Leadership

Title	Name	Photo	Links
CEO	Alan Burt		https://www.linkedin.com/in/alan-burt/
CTO & Founder	Vincent Gramoli		https://www.linkedin.com/in/vincent-gramoli-206148a/
COO	Kim Bartlett		https://www.linkedin.com/in/kim-bartlett-74a72117/

<p>CGO</p>	<p>Tim Bass</p>		<p>https://www.linkedin.com/in/tim-bass-96262198/</p>
------------	-----------------	--	--



Smart Contract Audit Scope

We at Hashlock audited the solidity code within the Redbelly Network Project, the scope of works included a comprehensive review of the smart contracts listed below. We tested the smart contracts to check for their security and efficiency. These tests were undertaken primarily through manual line by line analysis and were supported by software assisted testing.

Description	Project Review and Security Analysis Report for Redbelly Network Smart Contracts and other factors.
Language	Solidity
Audit Date	May - December, 2023
Contract 1	ActivityMonitor.sol
Contract 1 MD5 Hash	CADD74EB4390C14FAD71485D96015530
Contract 2	Authorizable.sol
Contract 2 MD5 Hash	3BA17E2A46B847BB831DB3F9C2F2A7C9
Contract 3	BootstrapContractRegistry.sol
Contract 3 MD5 Hash	8F184463D32C2A51E5E36BB4014FED4F
Contract 4	GasFees.sol
Contract 4 MD5 Hash	EA88443EA7B1EDEEB85AF3AFE6229513
Contract 5	IDPRegistry.sol
Contract 5 MD5 Hash	F80659599F1E2CFB7BB8F13622D2F16B
Contract 6	MockPriceFeed.sol
Contract 6 MD5 Hash	F8E454B599A88A974D952AE2684B09FC
Contract 7	JailedGovernors.sol
Contract 7 MD5 Hash	8011E8C1378207468D91588EE04865ED
Contract 8	MockRandomNumberGenerator.sol
Contract 8 MD5 Hash	3COB85DE97A5D4E1C340687E034ADFA7

Contract 9	NetworkConfiguration.sol
Contract 9 MD5 Hash	FF964C4895E96D5D8A938EBDB4518FFB
Contract 10	Permission.sol
Contract 10 MD5 Hash	655348C6FF54CCB5CFFB3C56CE15C9FD
Contract 11	RBAC.sol
Contract 11 MD5 Hash	CC1C1B569A3A39E4CD48F0DCAC1FC4A0
Contract 12	Reconfiguration.sol
Contract 12 MD5 Hash	1727098E8FOA38AF41D1FC212A537154
Contract 13	StakingDeposit.sol
Contract 13 MD5 Hash	A0767D73EB7A456C28DA4021AFAFF30B
Contract 14	Voting.sol
Contract 14 MD5 Hash	4FD99C2E4CA1CF9896A3B17EC20C47
Contract 15	TestPseudoRandomNumberGenerator.sol
Contract 15 MD5 Hash	4cabf4eac064a6d293b9f7180e22f72b

Penetration Test Scope

Description	Penetration Test for Layer 1 Core Network Infrastructure
Primary Language	GoLang
Audit Date	May - December, 2023

Repo Name	Repo Link	Repo Description
sevm	https://github.com/redbellynetw/ork/sevm	The SEVM (Ethereum Virtual Machine)
consensus	https://github.com/redbellynetw/ork/consensus	The consensus system to verify transactions
consensus-driver	https://github.com/redbellynetw/ork/consensus-driver	The middleware to allow the consensus system to talk to the SEVM
diablo-benchmark	https://github.com/redbellynetw/ork/diablo-benchmark	A benchmarking tool used to test the above systems

Security Rating

After our audit and analysis, we found the smart contracts to be **"Hashlocked"** and the layer 1 to be **"Hashlocked"**. The code follows simple logic, with correct and detailed ordering. They use a series of interfaces, and the contracts use a list of Open Zeppelin contracts.

All issues identified have since been resolved, actioned, or acknowledged and then re-reviewed.

Smart Contracts:



Hashlock found:

20 High severity vulnerabilities

16 Medium severity vulnerabilities

28 Low severity vulnerabilities

20 Gas Optimisations

Layer 1 Network:



The 'Hashlocked' rating is reserved for projects that ensure ongoing security via bug bounty programs or on chain monitoring technology.

Hashlock found:

5 High severity vulnerabilities

12 Medium severity vulnerabilities

45 Low severity vulnerabilities

All issues uncovered during automated and manual analysis were meticulously reviewed and fixed and applicable vulnerabilities are presented in the Audit overview section. General overview is presented in the Function list section and all identified issues can be found in the Audit overview section.

Caution: *Hashlock's audits do not guarantee a project's success or ethics, and are not liable or responsible for security. Always conduct independent research about any project before interacting.*



Standardised Checks

Main Category	Subcategory	Result
General Code Checks	Solidity/compiler version stated	Passed
	Consistent pragma version across each contract	Passed
	Outdated Solidity Version	Reviewed
	Overflow/underflow	Passed
	Correct checks, effects, interaction order	Reviewed
	Lack of check on input parameters	Reviewed
	Function input parameters check bypass	Reviewed
	Correct Access control	Reviewed
	Built in emergency features	Reviewed
	Correct event logs	Reviewed
	Human/contract checks bypass	Reviewed
	Random number generation/use vulnerability	Passed
	Fallback function misuse	Passed
	Race condition	Passed
	Logical vulnerability	Reviewed
	Features claimed	Passed
	delegatecall() vulnerabilities	Passed
	Other programming issues	Reviewed
Code Specification	Correctly declared function visibility	Passed
	Correctly declared variable storage location	Passed
	Use keywords/functions to be deprecated	Passed
	Unused code	Reviewed
Gas Optimization	"Out of Gas" Issue	Passed
	High consumption 'for/while' loop	Reviewed

	High consumption 'storage' storage	Passed
	Assert() misuse	Passed
Tokenomics Risk	The maximum limit for mintage not set	Passed
	"Short Address" Attack	Passed
	"Double Spend" Attack	Passed

Initial Audit Result: VULNERABLE

Revised Audit Result: PASSED

Intended Smart Contract Functions

Claimed Behaviour	Actual Behaviour
<p>File 1 ActivityMonitor.sol</p> <p><u>The authorised user has control over the following functions:</u></p> <ul style="list-style-type: none"> • Set to the current activity monitor address. • Sets the value of days to serve in jail for the governor. • Slashes the governor stake. • Remove the slashing of the governor stake. 	<p>Contract achieves this functionality.</p>
<p>File 2 Authorizable.sol</p> <ul style="list-style-type: none"> • Authorizable allows restricting actions to authorised users. • Maintains a list of authorised users and allows users to add / remove users to the list. 	<p>Contract achieves this functionality.</p>
<p>File 3 BootstrapContractRegistry.sol</p> <p><u>The authorised user has control over the following functions:</u></p> <ul style="list-style-type: none"> • Update/overwrite the bootstrap contract in the registry. 	<p>Contract achieves this functionality.</p>
<p>File 4 GasFees.sol</p> <p><u>The authorised has control over the following functions:</u></p> <ul style="list-style-type: none"> • Update the gas fee. • Update the currency value • Update value of decimals. • Add/remove authorised users. 	<p>Contract achieves this functionality.</p>
<p>File 5 IDPRegistry.sol</p> <p><u>The owner has control over the following functions:</u></p>	<p>Contract achieves this functionality.</p>

<ul style="list-style-type: none"> Update/remove user ID or issuer ID. 	
<p>File 6 MockPriceFeed.sol</p> <ul style="list-style-type: none"> Current price of 1 RBNT in USD. New price of 1 RBNT in USD. <p><u>The owner has control over the following functions:</u></p> <ul style="list-style-type: none"> Set a new price. 	<p>Contract achieves this functionality.</p>
<p>File 7 JailedGovernors.sol</p> <ul style="list-style-type: none"> This contract can expect to hold additional future logic of movement of governor states. 	<p>Contract achieves this functionality.</p>
<p>File 8 MockRandomNumberGenerator.sol</p> <ul style="list-style-type: none"> This contract is used for generating random numbers. 	<p>Contract achieves this functionality.</p>
<p>File 9 NetworkConfiguration.sol</p> <p><u>The owner has control over the following functions:</u></p> <ul style="list-style-type: none"> Update voting contract address. Update reconfiguration contract address. Add/remove the governor addresses. Add/remove the redbelly nodes addresses. 	<p>Contract achieves this functionality.</p>
<p>File 10 Permission.sol</p> <ul style="list-style-type: none"> Permission allows anybody to verify their identity. <p><u>The authorised user has control over the following functions:</u></p> <ul style="list-style-type: none"> Adds a user to the list of authorised users. 	<p>Contract achieves this functionality.</p>
<p>File 11 RBAC.sol</p> <ul style="list-style-type: none"> RBAC is used to initialise redbelly role, idp role, owner role. 	<p>Contract achieves this functionality.</p>
<p>File 12 Reconfiguration.sol</p>	<p>Contract achieves this</p>

<p><u>The authorised user has control over the following functions:</u></p> <ul style="list-style-type: none"> • Update value of tolerance factor. • Withdraw balance. • Set the unlock time interval reconfiguration interval. 	<p>functionality.</p>
<p>File 13 StakingDeposit.sol</p> <ul style="list-style-type: none"> • This contract can expect to hold additional future logic. 	<p>Contract achieves this functionality.</p>
<p>File 14 Voting.sol</p> <p><u>The authorised user has control over the following functions:</u></p> <ul style="list-style-type: none"> • Update the voting period, • Update the eligible voter details. • Setting vote status yes/no/abstain/closed. • Add/cancel proposals. 	<p>Contract achieves this functionality.</p>

Function List

Redbelly Network Contracts

- ActivityMonitor
- Authorizable
- BootstrapContractsRegistry
- GasFeesContract
- IDPRegistry
- JailedGovernors
- NetworkConfigurationContract
- Permission
- RBAC
- ReconfigurationContract
- StakingDeposit
- VotingContract

Functions

	Functions	Visibility	Observation	Conclusion
1	constructor - ActivityMonitor	Public		No Issue
2	updateJailOwner - ActivityMonitor	External	OnlyAuthUser	No Issue
4	setSlashPrcnt - ActivityMonitor	External	OnlyAuthUser	No Issue
7	setDaysToServe - ActivityMonitor	External	OnlyAuthUser	No Issue
8	slashStake - ActivityMonitor	External	OnlyAuthUser	No Issue
9	revertSlash - ActivityMonitor	External	OnlyAuthUser	No Issue
10	constructor - Authorizable	Public		No Issue
11	authorizeUser - Authorizable	External	onlyAuthorizedUser	No Issue
12	unauthorizeUser - Authorizable	External	onlyAuthorizedUser	No Issue

13	constructor - BootstrapContractsRegistry	Public		No Issue
14	Register - BootstrapContractsRegistry	External	onlyAuthorizedUser	No Issue
15	constructor - GasFeesContract	Public		No Issue
16	getBaseGasFee - GasFeesContract	Public		No Issue
17	updateGasFee - GasFeesContract	Public	OnlyAuthUser	No Issue
18	updateCurrency - GasFeesContract	Public	OnlyAuthUser	No Issue
19	updateDecimals - GasFeesContract	Public	OnlyAuthUser	No Issue
20	addAuthUser - GasFeesContract	Public	OnlyAuthUser	No Issue
21	removeAuthUser - GasFeesContract	Public	OnlyAuthUser	No Issue
22	Initialize - IDPRegistry	Public	initializer	No Issue
23	register - IDPRegistry	External	onlyRedbelly	No Issue
24	getAll - IDPRegistry	External		No Issue
25	getByUid - IDPRegistry	External		No Issue
26	getByIssuerDid - IDPRegistry	External		No Issue
27	updateByUid - IDPRegistry	Public	onlyIDP	No Issue
28	updateByIssuerDid - IDPRegistry	External	onlyIDP	No Issue
29	removeByUid - IDPRegistry	Public	onlyRedbelly idpExists(uid) notDeleted(uid)	No Issue

30	removeByIssuerDid - IDPRegistry	External	onlyRedbelly nonEmptyString(issue rDid,...) issuerDidExists(issuer Did)	No Issue
31	getDeletedCount - IDPRegistry	private		No Issue
32	constructor - JailedGovernors	Public		No Issue
33	jail - JailedGovernors	External	onlyOwner	No Issue
34	free - JailedGovernors	External	onlyOwner	No Issue
35	setSlashPrcnt - JailedGovernors	External	onlyOwner	No Issue
36	setOwner - JailedGovernors	External		No Issue
37	setDaysToServe - JailedGovernors	External	onlyOwner	No Issue
38	receive - JailedGovernors	External		No Issue
39	constructor - NetworkConfigurationContract	Public		No Issue
40	updateReconfigurationContract Address - NetworkConfigurationContract	Public	onlyGovernors	No Issue
41	updateVotingContractAddress - NetworkConfigurationContract	Public	onlyAdmin	No Issue
42	Register - NetworkConfigurationContract	Public		No Issue
43	addGovernor - NetworkConfigurationContract	Public	onlyGovernors	No Issue
44	removeGovernor - NetworkConfigurationContract	Public	onlyGovernors	No Issue
45	addRedbellyNodes - NetworkConfigurationContract	public	OnlyRedbellyNodes	No Issue

46	removeRedbellyNodes - NetworkConfigurationContract	public	OnlyRedbellyNodes	No Issue
47	isGovernor - NetworkConfigurationContract	public		No Issue
48	getCandidateIndex - NetworkConfigurationContract	public		No Issue
49	isBannedNode - NetworkConfigurationContract	public		No Issue
50	getGovernors - NetworkConfigurationContract	public		No Issue
51	getCandidates - NetworkConfigurationContract	public		No Issue
52	getNetworkSize - NetworkConfigurationContract	public		No Issue
53	getGovernorRedbellyNodeCount - NetworkConfigurationContract	private		No Issue
54	getNodeConfigurationsByAddresses - NetworkConfigurationContract	public		No Issue
55	setReconfigurationBiasness - NetworkConfigurationContract	public	OnlyRedbellyNodes	No Issue
56	swapNodesWithIndexes - NetworkConfigurationContract	private		No Issue
57	runBiasedReconfiguration - NetworkConfigurationContract	private		No Issue
58	reconfigureNetwork - NetworkConfigurationContract	public	OnlyReconfigurationContract	No Issue
59	getRandomIndex - NetworkConfigurationContract	public		No Issue
60	banGovernorNode - NetworkConfigurationContract	public	OnlyVotingContract	No Issue
61	Request Permission	External		No Issue
62	isAllowed Permission	External		No Issue

63	_authorizeUser Permission	private		No Issue
64	Initialize RBAC	public	onlyInitializing	No Issue
65	__AccessControl_init RBAC	internal	onlyInitializing	No Issue
66	__AccessControl_init_unchaine d RBAC	internal	onlyInitializing	No Issue
67	supportsInterface RBAC	public		No Issue
68	hasRole RBAC	public		No Issue
69	_checkRole RBAC	internal		No Issue
70	_checkRole RBAC	internal		No Issue
71	getRoleAdmin RBAC	public		No Issue
72	grantRole RBAC	public		No Issue
73	revokeRole RBAC	public		No Issue
74	renounceRole RBAC	public		No Issue
75	_setupRole RBAC	internal		No Issue
76	_setRoleAdmin RBAC	internal		No Issue
77	_grantRole RBAC	internal		No Issue
78	_revokeRole RBAC	internal		No Issue
79	constructor ReconfigurationContract	public		No Issue
80	setReconfigurationInterval ReconfigurationContract	public	onlyAuthorizedAccess	No Issue

81	Receive ReconfigurationContract	External	onlyAuthorizedAccess	No Issue
82	withdrawBalance ReconfigurationContract	public	onlyAuthorizedAccess	No Issue
83	updateToleranceFactor ReconfigurationContract	public	onlyAuthorizedAccess	No Issue
84	triggerReconfiguration ReconfigurationContract	public	onlyAfterReconfigInte rval	No Issue
85	getRandomIndexes ReconfigurationContract	internal		No Issue
86	Deposit StakingDeposit	External		No Issue
87	Redeposit StakingDeposit	External		No Issue
88	confiscateStake StakingDeposit	External		No Issue
89	releaseStake StakingDeposit	External		No Issue
90	stakeConfiscated StakingDeposit	External		No Issue
91	Receive StakingDeposit	External		No Issue
92	constructor VotingContract	public		No Issue
93	isAuthorisedUser VotingContract	private		No Issue
94	updateVotingPeriod VotingContract	public	onlyAuthorisedUser	No Issue
95	updateVotingDelay VotingContract	public	onlyAuthorisedUser	No Issue
96	isThresholdReached VotingContract	private		No Issue
97	isEligibleVoter VotingContract	public		No Issue
98	updateEligibleVoter VotingContract	public	OnlyNetworkConfigur ationContract	No Issue

99	addProposal VotingContract	public	onlyEligibleVoters	No Issue
100	voteYes VotingContract	public	onlyEligibleVoters	No Issue
101	voteNo VotingContract	public	onlyEligibleVoters	No Issue
102	voteAbstain VotingContract	public	onlyEligibleVoters	No Issue
103	endVoting VotingContract	public	onlyEligibleVoters votingPeriodEnded	No Issue
104	cancelProposal VotingContract	public	activeProposal.owner	No Issue
105	banGovernor VotingContract	public	proposalActivated onlyAuthorisedUser	No Issue
106	resetVoting VotingContract	private		No Issue

Code Quality

This audit scope involves the solidity smart contracts of the Redbelly Network project, as outlined in the Audit Scope section. All contracts, libraries and interfaces intend to follow standard best practices and to help avoid unnecessary complexity that increases the likelihood of exploitation, however some refactoring is required.

The code is extremely well commented and closely follows best practice nat-spec styling. All comments are correctly aligned with code functionality.

Audit Resources

We were given the Redbelly Network Protocol's smart contract code in the form of Github access.

As mentioned above, code parts are well commented. The logic is straightforward, and therefore it is easy to quickly comprehend the programming flow as well as the complex code logic. The comments are helpful in understanding the overall architecture of the protocol.

Dependencies

As per our observation, the libraries used in this smart contracts infrastructure are based on well known industry standard open source projects.

Apart from libraries, its functions are used in external smart contract calls.

Severity Definitions

Significance	Description
High	High severity vulnerabilities can result in loss of funds, asset loss, access denial, and other critical issues that will result in the direct loss of funds and control by the owners and community.
Medium	Medium level difficulties should be solved before deployment, but won't result in loss of funds.
Low	Low level vulnerabilities are areas that lack best practices that may cause small complications in the future.
Gas	Gas Optimisations, issues and inefficiencies

Smart Contract Audit Findings

High

[H-01] StakingDeposit#deposit - Arbitrary `amount` input parameter allows user to have any staked amount

Description

The `deposit` function creates a `Deposit` struct where the `depositedAmount` value is set to `amount`, which can be arbitrarily set in the function arguments.

```
function deposit(  
    address pubAddr,  
    address withdrawalAddr,  
    bool acceptDelegation,  
    uint256 amount  
) external payable {  
    //.....function body.....  
    depositors[msg.sender] = Depositors(  
        pubAddr,  
        address(0x00),  
        withdrawalAddr,  
        false,  
        acceptDelegation,  
        Deposit(amount, block.timestamp, 0)  
    );  
    ...  
}
```

Vulnerability Details

A malicious user can call `deposit` with an absurdly high `amount` such as `type(uint256).max`. The system will take that amount as their `depositedAmount`, regardless of the value of `msg.value`.

Impact

The malicious user can use their unnaturally large `depositedAmount` to drain the `StakingDeposit` contract by calling `confiscateStake` on themselves.

If the Redbelly Network relies on the amount of stake for governance or consensus, then the security of the network can be compromised as the malicious user would have an unnaturally large amount of stake.

Recommendation

Instead of taking in an arbitrary `amount` as an input parameter, set the depositor's `depositedAmount` as the `msg.value`

```
require(msg.value > 0, "Deposit should be greater than 0");
depositors[msg.sender] = Depositors(
    pubAddr,
    address(0x00),
    withdrawalAddr,
    false,
    acceptDelegation,
    Deposit(amount, block.timestamp, 0)
);
```

Status

Resolved

[H-02] StakingDeposit#redeposit - Function is not payable and allows arbitrary `amount` input parameter

Description

Similar to [C-01], the `redeposit` function allows an arbitrary `amount` to be added to the depositor's `depositedAmount`.

```
function redeposit(address pubAddr, uint256 amount) external {
    //.....function body.....
    //----increase the total deposit amount of a depositor----
    //.....function body.....
}
```

Vulnerability Details

Similar to [H-01], but in this case, there is no `msg.value` so the amount deposited into the contract will always be 0.

Impact

Refer to [H-01].

Recommendation

Instead of taking in an arbitrary `amount` as an input parameter, set the `redeposit` function as `payable` and use the following line:

```
depositors[msg.sender].deposit.depositedAmount += amount;
```

Status

Resolved

Redbelly

The redeposit function which is mentioned in this story has been removed in the latest code

[H-03] StakingDeposit - Missing access control in slashing functions can result in the contract being insolvent or drained

Description

`confiscateStake` and `releaseStake` do not have access control and can be called by anyone with arbitrary input.

Vulnerability Details

`confiscateStake`:

Arbitrary calls to `confiscateStake` allows any malicious user to slash and steal **all** the stake of **all** depositors. This can be done with multiple calls to `confiscateStake` where `_amountPrct = 100`.

The malicious user can also combine an exploit of **[H-01]** with **[H-03]** to drain the contract in a quicker manner.

`releaseStake`

`releaseStake` can be called to refund a slashed depositor their confiscated stake after `confiscateStake` has been called.

```
function releaseStake(address _depositor) external {
    require(
        depositors[_depositor].deposit.confiscatedAmount > 0,
        "No confiscation"
    );
    depositors[_depositor].deposit.depositedAmount =
        depositors[_depositor].deposit.depositedAmount +
        depositors[_depositor].deposit.confiscatedAmount;
    depositors[_depositor].deposit.confiscatedAmount = 0;
}
```

However, `confiscateStake` transfers the confiscated underlying asset to the caller, meaning that the underlying assets to back the deposit are now missing.

```
function confiscateStake(address _depositor, uint8 _amountPrct) external {
    ...
    payable(msg.sender).transfer(confiscationAmount);
}
```

```
}
```

The refund of underlying assets is handled in `JailedGovernors.free`, but the `releaseStake` function does not implement access control and so depositors can be refunded their balance without the underlying assets to back them.

Impact

Missing access control in `confiscateStake` will result in the contract being drained of all its native assets.

Missing access control in `releaseStake` will result in the contract being insolvent as deposits aren't sufficiently backed by the contract balance.

Recommendation

Add access control to `confiscateStake` and `releaseStake` so they can only be called by the `JailedGovernors` contract.

Status

Unresolved

[H-04] JailedGovernors#setOwner - A malicious user can front-run the transaction to become the owner

Description

The `setOwner` function initially allows anyone to call it since `ownerSet` is `false` by default. A malicious user can front-run the rightful transaction that calls `setOwner` to steal ownership of the contract.

Vulnerability Details

The `JailedGovernors` contract is deployed initially without an owner. An owner has to be set after deployment by calling `setOwner`, which allows anyone to set the owner.

```
function setOwner(address _owner) external {
    if (!ownerSet || msg.sender == owner) {
        owner = _owner;
        ownerSet = true;
    } else {
        revert("Unauthorised");
    }
}
```

A malicious user can front-run the rightful transaction that calls `setOwner`, setting themselves as the owner instead.

Impact

The malicious user will have control over the contract and its methods.

`ActivityMonitor` will not be able to be deployed as the `setOwner` call in the constructor will revert.

```

constructor(
    ...
    jailedGovernors.setOwner(address(this));
}

```

Recommendation

Assign a temporary owner to `JailedGovernors` inside the constructor. The temporary owner can then transfer ownership to `ActivityMonitor` afterwards.

Status

Resolved

Hashlock

`setOwner` function has been removed

[H-05] MockPriceFeed - Feed updates do not get time stamped and can report stale prices.

Description

The price feed design is flawed as it reports the price value without a timestamp or round ID. A contract that uses the price feed will not be able to determine the freshness of the price it's served.

Vulnerability Details

The setPrice function allows any authorised user to set the current price of the asset.

```

/// @notice Sets a new price
/// @dev Emits the PriceUpdated event
/// @param newPrice New price of 1 RBNT in USD
function setPrice(uint256 newPrice) external override onlyAuthorizedUser {
    price = newPrice;
    emit PriceUpdated(newPrice);
}

```


However, there is no state variable that keeps track of when the price was last updated. In the event that authorised users fail to update the price, the price feed will report a stale price.

Impact

Contracts that use the price feed will have no indication of knowing whether the price reported from the feed is up-to-date or not.

Recommendation

Keep track of the times when the price feed is updated by adding a new state variable `lastUpdated`. This state variable should be updated each time `setPrice` is called, and should also be returned when calling `getLatestPrice`.

```
contract MockPriceFeed is IPriceFeed, Authorizable {
    /// @notice Maintains price of RBNT in USD
    uint256 private price;
    uint256 private lastUpdated;

    constructor(uint256 _price, address[] memory authorizedUsers)
    Authorizable(authorizedUsers) {
        price = _price;
        lastUpdated = block.timestamp;
    }

    /// @notice Fetch the latest price of token
    /// @return Current price of 1 RBNT in USD
    function getLatestPrice() external view override returns (uint256, uint256) {
        return (price, lastUpdated);
    }

    /// @notice Sets a new price
    /// @dev Emits the PriceUpdated event
    /// @param newPrice New price of 1 RBNT in USD
    function setPrice(uint256 newPrice) external override onlyAuthorizedUser {
        price = newPrice;
        lastUpdated = block.timestamp;
        emit PriceUpdated(newPrice);
    }
}
```

Status

Unresolved

[H-06] VotingContract - Owner of an active proposal can DoS attack the contract

Description

If a proposal fails to reach the threshold of votes required to be activated before the voting period ends, the proposal owner can DoS attack the contract by not calling `cancelProposal`.

Vulnerability Details

There are two functions that call `resetVoting` to clear the current active proposal:

1. `cancelProposal`
2. `banGovernor`

In the case that an active proposal did not get enough votes, `cancelProposal` must be called to clear the proposal before a new one can take its place. However, the function can only be called by the owner of the proposal. If the proposal owner does not call `cancelProposal`, then there's no way for votes to be reset.

```
function cancelProposal() public {
    require(activeProposal.status, "No proposal is currently active");
    require(
        activeProposal.owner == msg.sender,
        "Not the owner of the proposal"
    );
    votingEnded = true;
    string memory proposalName = activeProposal.name;
    resetVoting();
    emit ProposalCancelled(proposalName);
}
```

Impact

The contract will be frozen; no one can start a new proposal.

Recommendation

Create an emergency function for authorised users that calls `resetVoting`.

Status

Resolved

Hashlock

`VotingContract` has been removed from the codebase.

[H-07] NetworkConfiguration#updateReconfigurationContractAddress - Incorrect access control

Description

The `updateReconfigurationContractAddress` function allows governors to call the function instead of admins.

Vulnerability Details

There is an `onlyGovernors` modifier on the function instead of `onlyAdmin`.

```
function updateReconfigurationContractAddress(
    address newAddress
) public onlyGovernors {
    reconfigurationContractAddress = newAddress;
    emit ReconfigurationContractAddressUpdated(newAddress);
}
```

Impact

Any governor can change the Reconfiguration contract address to any arbitrary address, allowing `reconfigureNetwork` to be called with arbitrary indexes as opposed to pseudo-random indexes. This can compromise the node distribution of the RedBelly network.

Recommendation

Change the modifier to `onlyAdmin`.

Status

Resolved

Hashlock

`updateReconfigurationContractAddress` function has been removed

[H-08] Contracts with `authorisable` access control patterns allow authorised users to add and remove other authorised users

Description

Authorizable.sol and other contracts that have similar access control patterns (Reconfiguration.sol, GasFees.sol, Voting.sol) allow its authorised users to add/remove other authorised users.

Vulnerability Details

The current access control pattern relies on an 'N of N' trust model, meaning that all authorised users have to act as expected for the system to be secure.

Any authorised user that is compromised/malicious can add/remove other authorised users as they wish, compromising the contract or security of the RedBelly network.

Impact

The security of the contract and the RedBelly network can be compromised.

Recommendation

Separate the role of adding/removing authorised users away from the authorised users themselves. This can be done in a secure manner by assigning the role to a multisig address that's controlled by a committee of authorised users.

Status

Resolved

Redbelly

We are moving away from AuthorisedUser functionality in all the smart contracts and replacing it with RBAC where only the owner of the smart contract has the RBAC admin rights to grant/revoke any permissions. So, only the owner will be granting/revoking access, and for the owner's wallet, we will most likely use a multi-sig account.

[H-09] Potential Denial of Service (DoS) Attack due to Linear Search in Mappings

Description

The contract uses a mapping to store candidates but performs a linear search to find a specific candidate in the `getCandidateIndex()` function. This can lead to inefficient gas usage, especially as the number of candidates increases.

Vulnerability Details

The `getCandidateIndex()` function loops over the candidates mapping to find a matching address, leading to $O(n)$ complexity. This means that as the number of candidates grows, the gas required for this function will increase proportionally. If there are a lot of candidates, this function could potentially run out of gas.

Furthermore, the `addGovernor()` function, which calls `getCandidateIndex()`, may also become gas-heavy and potentially run out of gas as the number of candidates increases. This could be exploited by an attacker adding many candidates, which could disrupt the operation of the contract.

Impact

This inefficiency could potentially allow an attacker to cause a Denial of Service (DoS) attack by continuously adding candidates and increasing the gas cost of the `addGovernor()` function, disrupting the operation of the contract.

Recommendation

Consider using a different approach that allows for more efficient lookups, such as an additional mapping that maps an address to its index, to avoid the need for a linear search. Be sure to update this mapping appropriately when adding or removing candidates.

Status

Resolved

Hashlock

Search now happens in constant time.

[H-10] Predictable Random Number Generation in `banGovernorNode()` Function **doublecheck**

Description

The `banGovernorNode()` function uses the `generateOne()` function to produce a pseudo-random index for selecting a new governor. This function uses the previous blockhash and a static seed as sources of randomness, making the outcome predictable once the block is mined.

Vulnerability Details

The `generateOne()` function uses the blockhash of the previous block and a static seed of 0 for a pseudo-random number generator. This makes the outcome predictable as the blockhash is a deterministic function of the contents of the previous block, and the seed does not introduce any additional randomness.

Impact

The predictability of this pseudo-random number generation could potentially be exploited if there's an advantage to knowing the outcome in advance. If the outcome determines the selection of a new governor, an adversary who can predict this could use this information to their advantage, leading to a manipulation of the governance process.

Recommendation

Consider using a more secure and unpredictable mechanism for generating random numbers. This might involve a commit-reveal scheme, an external randomness provider like Chainlink VRF, or other similar mechanisms.

Status

Unresolved

[H-11] StakingDeposit#deposit - Existing depositor can lose their pre-existing stake by calling `deposit` again

Description

`deposit` does not check whether the caller is an existing depositor or not. If an existing depositor calls `deposit`, their current deposit will be overwritten.

Vulnerability Details

`deposit` assigns a new `Depositors` struct to the caller.

```
function deposit(
    address pubAddr,
    address withdrawalAddr,
    bool acceptDelegation,
    uint256 amount
) external payable {
    //.....function body.....'
    depositors[msg.sender] = Depositors(
        pubAddr,
        address(0x00),
        withdrawalAddr,
        false,
        acceptDelegation,
        Deposit(amount, block.timestamp, 0)
    );
    ...
}
```

Since there is no check for whether a `Depositors` struct already exists for the caller, all the pre-existing details about their deposit including their `depositedAmount` and `confiscatedAmount` will be overwritten.

Impact

The depositor will lose their existing stake.

Recommendation

Add a check that ensures that the caller has not staked already.

```
require(depositors[msg.sender].deposit.depositTimestamp == 0, "Already
deposited");
```

Status

Resolved

Hashlock

There is now a check to ensure that an existing staker cannot call `deposit` again.

[H-12] Loss of funds: precision loss in getClaimableTokens function in vesting contract

Description

The contract performs division before multiplication in the `getClaimableTokens()` function which can lead to a loss of precision, especially when `_account.allocatedTokens` is not a multiple of `totalPeriodOfVesting`.

Vulnerability Details

The division (`_account.allocatedTokens / totalPeriodOfVesting`) is integer division, and if `allocatedTokens` is not a multiple of `totalPeriodOfVesting`, the division will lose precision as it rounds down to the nearest integer. This could lead to a lower `tokensToClaim` than expected.

Impact

This precision loss could result in users being able to claim fewer tokens than they are supposed to be able to, potentially leading to loss of funds.

Recommendation

Consider performing the multiplication operation before the division to minimise precision loss. The recommended code would look like this:

```
uint tokensToClaim = (currentTimestamp - _account.lastClaimedTimeStamp) *  
_account.allocatedTokens / totalPeriodOfVesting;
```

Additional Note:

While the recommended change minimises precision loss, it introduces a potential integer overflow issue if the product of `(currentTimestamp - _account.lastClaimedTimeStamp)` and `_account.allocatedTokens` exceeds the maximum value that can be stored in a `uint`. This risk can be mitigated by using the `SafeMath` library provided by `OpenZeppelin`, which includes safety checks to prevent overflow.

Status

Unresolved

[H-13] StakingDeposit#deposit - Malicious users can prevent other users from staking by depositing a zero amount on their behalf

Description

A malicious user can grief/cause DoS to other users by depositing a zero amount on their behalf. This is because the function `deposit` does not check if the `amount` is zero.

Vulnerability Details

The function `deposit` does not have access control. This means that another user can deposit on behalf of other users. However, the function does not check if the `amount` is zero. This means that a malicious user can deposit a zero amount on behalf of another user, preventing the other user from staking.

```
/**
 * @dev - main deposit function - can be called any entity - no validation for
msg.sender
 */
function deposit(address depositor) external payable {
    require(
        depositors[depositor].depositTimestamp == 0,
        "Existing deposit found"
    );
    depositors[depositor] = Deposit(msg.value, block.timestamp, 0);
    emit StakeDeposited(depositor, msg.value);
}
```

Impact

A malicious user can prevent other users from staking by depositing a zero amount on their behalf. Since there is no way of removing `Deposit` entries, the malicious user can prevent the other user from staking forever.

Recommendation

Add a `require` statement to check if the `amount` is zero.

```
/**
 * @dev - main deposit function - can be called any entity - no validation for
msg.sender
 */
function deposit(address depositor) external payable {
    require(
        depositors[depositor].depositTimestamp == 0,
        "Existing deposit found"
    );
    require(msg.value > 0, "Amount cannot be zero");
    depositors[depositor] = Deposit(msg.value, block.timestamp, 0);
    emit StakeDeposited(depositor, msg.value);
}
```

Adding access control to the function `deposit` such that only the `NetworkConfiguration` contract can call the function is also recommended.

Status

Unresolved

Hashlock

An attacker can cause a DoS on `NetworkConfiguration.register` by frontrunning the transaction with a zero deposit on behalf of `nodeAddress`.

Medium

[M-01] Using transfer in claimtokens function in vesting contract

Description

The claimTokens() function uses the transfer() method to send tokens to an address. While transfer() was previously recommended for sending Ether due to its automatic reentrancy protection, it is now considered less safe due to its gas limit of 2300 gas which can cause unexpected behaviour.

Vulnerability Details

The transfer() function in Solidity is considered less safe because it can cause contracts to break if they require more than 2300 gas.

Impact

If the receiving contract's fallback function consumes more than 2300 gas, the transfer() function will fail, potentially causing the claimTokens() function to behave unexpectedly.

Recommendation

Consider replacing transfer() with the call.

Status

Unresolved

[M-02] Contracts with `authorised` access control are frozen if no users are authorised during deployment

Description

Contracts that initialise an array of authorised users at deployment can be frozen if the `authorizedUsers` list is empty.

Vulnerability Details

The `Authorizable.sol` contract has the function `authorizeUser` which can only be called by an existing authorised user.

```

    /// @notice Authorizes a user
    /// @dev Adds a user to the list of authorized users
    /// @param _userAddress Address of the user who'll be authorized
    function authorizeUser(address _userAddress) external override
onlyAuthorizedUser {
    isAuthorizedUser[_userAddress] = true;
    emit UserAuthorized(_userAddress);
}

```

If there are no authorised users at deployment, then it's impossible to add them afterwards.

Impact

Any functions that can only be called by authorised users will never be able to be called, potentially bricking the contract after deployment.

Recommendation

Include the address of the contract deployer as an authorised user by default in the constructor.

```

    /// @notice Allows to check whether an address is authorized
    mapping(address => bool) public isAuthorizedUser;

    constructor(address[] memory authorizedUsers) {
        isAuthorizedUser[msg.sender] = true;

        for (uint i = 0; i < authorizedUsers.length; i++) {
            isAuthorizedUser[authorizedUsers[i]] = true;
        }
    }
}

```

If the contract deployer is another contract, make sure it can call `authorizeUser`.

Status

Resolved

Redbelly

We are moving away from AuthorisedUser functionality in all the smart contracts and replacing it with RBAC where the owner of the smart contract has the RBAC admin

rights to grant/revoke any permissions. So, even if no users are authorized during deployment the owner can easily grant access later in the life cycle.

[M-03] VotingContract - Voters can vote before a proposal is added

Description

Voters can vote before a proposal is added if the proposal hasn't been added before `startTime + votingDelay`.

Vulnerability Details

Voters can start voting after `startTime + votingDelay`, regardless of whether there is an active proposal or not.

```
function voteYes() public onlyEligibleVoters {
    require(
        block.timestamp >= (startTime + votingDelay),
        "Voting has not started yet"
    );
    require(
        block.timestamp < (startTime + votingPeriod + votingDelay),
        "Voting period has ended"
    );
    require(!hasVoted[msg.sender], "You have already voted");

    activeProposal.yesVotes++;
    hasVoted[msg.sender] = true;
    voters.push(msg.sender);
    emit VoteCasted("Yes");
}
```

If there is no active proposal before `startTime + votingDelay` then, a voter would waste their vote as it'll get cleared from the vote count when a new proposal is added, but their `hasVoted` status remains true.

Impact

The voter will not be able to vote for the upcoming proposal.

Recommendation

Inside the vote functions, check that there is an active proposal with the following line:

```
require(activeProposal.status, "No active proposal");
```

Status

Resolved

Hashlock

Contract has been removed.

[M-04] VotingContract - Governors can grief the proposal result by adding/removing governors

Description

Governors can grief the result of a proposal by adding/removing governors to change the networkSize, which is used to determine whether the proposal passes or not.

Vulnerability Details

A proposal can only be implemented by calling `banGovernor` if the following function returns true:

```
function isThresholdReached() private view returns (bool) {
    uint256 networkSize = networkConfigurationContract.getNetworkSize();
    if (activeProposal.yesVotes > (networkSize - 1) / 3) {
        return true;
    }
    return false;
}
```

Governors can change the network size by adding or removing other governors from the network by calling `addGovernor`/`removeGovernor` after voting has ended, changing the result of the vote.

Impact

A proposal that wouldn't have passed now can pass, and vice versa.

Recommendation

Pause functions that change the network configuration (`addGovernor`, `removeGovernor` inside NetworkConfigurationContract) while there is an active proposal being voted on.

Status

Resolved

Hashlock

Contract has been removed.

[M-05] TokenVestingUpgradeable - A zero vesting period will make the vested tokens unclaimable

Description

When tokens are allocated, it's possible to set a vesting period of 0. This results in the `getClaimableTokens` reverting due to a zero-division.

Vulnerability Details

`allocateTokens` allows vesting to occur where `_vestingStartTime == _vestingEndTime`.

```
require(
    _address != address(0) &&
    _vestingStartTime <= _vestingEndTime &&
    _amount > 0
);
```

This results in a `totalPeriodOfVesting = 0` inside `getClaimableTokens`, which is used as a divisor to calculate the amount of tokens to claim.

```
function getClaimableTokens(address _address) public view returns (uint) {
    AccountDetails memory _account = accountsList[_address];
    uint totalPeriodOfVesting = _account.vestingEndTime -
        _account.vestingStartTime;
    uint currentTimestamp = block.timestamp;
    if (block.timestamp > _account.vestingEndTime) {
        currentTimestamp = _account.vestingEndTime;
    }
    uint tokensToClaim = (currentTimestamp -
        _account.lastClaimedTimeStamp) *
        (_account.allocatedTokens / totalPeriodOfVesting);
    return tokensToClaim;
}
```

`getClaimableTokens` will revert due to a `division by zero`. Since the function is called inside `claimTokens`, the entities would not be able to claim their vested tokens.

Impact

Entities with vested tokens with a vested period of 0 will not be able to claim their tokens.

Recommendation

Change the require statement inside `allocateTokens` to not allow zero vesting periods by disallowing the case where `_vestingStartTime == vestingEndTime`.

```

require(
    _address != address(0) &&
    _vestingStartTime < _vestingEndTime &&
    _amount > 0
);

```

Status

Unresolved

[M-06] TokenVestingUpgradeable - Contract can run out of tokens to vest

Description

Users can call `allocateTokens` with any arbitrary `_amount`. This can lead to cases where vested tokens are unclaimable since the contract does not have enough tokens to transfer.

Vulnerability Details

When an entity tries to claim their tokens, the transaction will fail as the contract has sufficient funds to transfer.

Impact

Entities will not be able to claim their tokens if there aren't enough tokens in the contract.

Recommendation

Add a check to ensure that there are sufficient 'free' tokens left to be allocated inside `allocateTokens`. This can be done by keeping track of the total amount of allocated tokens.

```

require(_totalAllocatedTokens + _amount <= address(this).balance, "Not enough
tokens in contract");

```

Status

Unresolved

Low

[L-01] Reconfiguration - Address without authorized access can still forcibly send Ether to the contract

Description

Impact

"Reconfiguration` contract can still receive Ether when another contract calls `selfdestruct`. This does not use the `receive` function and therefore, the `onlyAuthorizedAccess` modifier is bypassed

NOTE: The SELFDESTRUCT opcode is being deprecated in Ethereum. As of July 2023, it still functions as normal and results in the unintended behaviour described above."

Recommendation

Status

Resolved

The RBN team has stated the following:

"[The contract] receives ethers/native assets because it was planned that the user who initiates the Reconfiguration transaction should be compensated for it and paid some reward for initiating it. But this requirement has changed since. The native coin receiving functionality has been removed or needs to be removed."

[L-02] Make sure vesting endtime is greater than block.timestamp in allocate tokens

Description

Impact

Recommendation

Status

Unresolved

[L-03] Default values being reinitialized

Description

Vulnerability Details

Impact

Recommendation

Status

Unresolved

[L-04] Missing sanity checks in admin setter functions

Description

Various admin functions that update state variables do not have sufficient input validation.

Vulnerability Details

In GasFees.sol, `updateCurrency` does not check for an empty input string.

In Voting.sol, `updateVotingPeriod` and `updateVotingDelay` do not check for zero input values.

In NetworkConfiguration.sol, `updateReconfigurationContractAddress` and `updateVotingContractAddress` do not check if the input address is a contract.

Impact

Without sanity checks, unintended state transitions can occur, resulting in unintended behaviour.

Recommendation

```
GasFees#updateCurrency
    require(bytes(_currency).length > 0, "Currency can't be empty");
```

```
VotingContract#updateVotingPeriod
    require(_newTimePeriod > 0, "Voting period cannot be 0");
```

```
VotingContract#updateVotingDelay
    require(_newTimePeriod > 0, "Voting delay cannot be 0");
```

```
NetworkConfigurationContract#updateReconfigurationContractAddress &
NetworkConfigurationContract#updateVotingContractAddress
    require(newAddress.code.length > 0, "Address must be a contract");
```


Status

Unresolved

[L-05] Pragma version old**Description****Impact****Recommendation****Status**

Unresolved

[L-06] VotingContract - `addProposal` does not check if `nodeAddress` is an existing governor**Description**

A governor can create a proposal with a `nodeAddress` that does not belong to an existing governor. This can lead to unintended behaviour if the vote passes.

Vulnerability Details

A successful proposal is executed by calling `banGovernor` which calls `NetworkConfigurationContract.banGovernorNode`

```
function banGovernorNode(address nodeAddress) public OnlyVotingContract {
    blockedGovernors.push(nodeAddress);
    uint256 candidateIndex = getRandomIndex(0, candidateCount.current());
    int256 governorIndex = isGovernor(nodeAddress);

    votingContract.updateEligibleVoter(
        candidates[candidateIndex],
        governors[uint256(governorIndex)]
    );
    governors[uint256(governorIndex)] = candidates[candidateIndex];

    candidates[uint256(candidateIndex)] = candidates[
        candidateCount.current() - 1
    ];
    candidateCount.decrement();
    address[] memory governorAddress = getGovernors();
    emit GovernorsUpdated(governorAddress);
}
```

`isGovernor(nodeAddress)` returns `-1`, since the governor does not exist. The random candidate gets added as an eligible voter, but does not become a governor, since they get assigned to the index `uint256(-1) = type(uint256).max` which is greater than the `governorsCount`.

Impact

If there is no governor in the max uint256 index value, then the randomly chosen candidate becomes a new voter and no one is removed from voting.

If there is a governor in the max uint256 index value due to a previous successful proposal with a non-existing governor `nodeAddress`, then that address is removed from voting and replaced with the candidate.

The list of governors does not change.

Recommendation

Add a check that requires the `nodeAddress` when creating a new proposal to be an existing governor

```
require(networkConfigurationAddress.isGovernor(nodeAddress) > 0, "Address is not a governor");
```

Status

Unresolved

[L-07] VotingContract#banGovenor - Function name has a typo

Description

`banGovenor` should be `banGovernor`

Vulnerability Details

N/A

Impact

N/A

Recommendation

Replace all mentions of `banGovenor` with `banGovernor`

Status

Unresolved

[L-08] Contracts should inherit from Authorizable instead of having its own access control logic

Description

The following contracts use an authorizable access control pattern but do not inherit from `Authorizable`.

1. GasFees
2. VotingContract
3. ReconfigurationContract

Vulnerability Details

N/A

Impact

N/A

Recommendation

The contracts listed above should inherit from `Authorizable` to improve the maintainability of the codebase.

Status

Unresolved

[L-09] Do not use floating pragma versions

Description

It is best practice to deploy all contracts with the same compiler version. Using a floating pragma version may not allow this to happen.

Vulnerability Details

Refer to [SWC-103](#) for more information.

Impact

Using different compiler versions may introduce bugs or unintended behaviours.

Recommendation

Lock the pragma version (remove `` from the Solidity version number)

Status

Unresolved

[L-10] JailedGovernors - Initialising slash percentage to be >100% will cause the `jail` method to always revert

Description

An arbitrary slash percentage can be set in the constructor and in `setSlashPrct`. If the value of `slashPrct` is greater than 100% (`slashPrct = 100`), the `JailedGovernors` contract will not be able to confiscate stake from governors.

Vulnerability Details

Inside `StakingDeposit.confiscateStake`, the slash percentage value is used to calculate the amount of stake to confiscate from the governor.

```
function confiscateStake(address _depositor, uint8 _amountPrct) external {
    ...
    uint confiscationAmount = (depositors[_depositor]
        .deposit
        .depositedAmount * (_amountPrct)) / 100;
    depositors[_depositor].deposit.confiscatedAmount = confiscationAmount;
    depositors[_depositor].deposit.depositedAmount =
        depositors[_depositor].deposit.depositedAmount -
        confiscationAmount;
    payable(msg.sender).transfer(confiscationAmount);
}
```

A slash percentage of over 100% will cause `confiscationAmount > depositedAmount`, which will cause the line below to revert due to underflow.

```
depositors[_depositor].deposit.depositedAmount =
    depositors[_depositor].deposit.depositedAmount -
    confiscationAmount;
```

Impact

The `JailedGovernors` contract will not be able to jail governors by calling `jail`.

Recommendation

Add input validation that only allows `slashPrct` to be set to values less than or equal to 100%.

```
require(_slashprct <= 100, "Invalid slash percentage");
```

Status

Unresolved

[L-11] ActivityMonitor - `updateJailOwner` function is unnecessary**Description**

The `updateJailOwner` function sets the owner of the `JailedGovernors` contract. This is unnecessary as the owner should always be the `ActivityMonitor` contract.

Vulnerability Details

N/A

Impact

N/A

Recommendation

Remove the `updateJailOwner` function.

Status

Unresolved

Gas**[G-01] Use `uint256` instead of `bool` for mappings****Description**

The SEVM stack works in 32-byte words. There's overhead costs with casting 256-bit numbers into 8-bit for `bool` variables. For boolean values that are regularly accessed, it is better to store them as `uint256` values to avoid the unnecessary overhead gas costs.

Recommendation

Use mappings that map to uint256 values instead of bool for mappings that are regularly accessed such as `Authorizable.isAuthorizedUser`.

Status

Unresolved

[G-02] Use `++i` instead of `i++`

Description

Pre-incrementing a value saves gas, as the SEVM does not need to return the non-incremented value.

Recommendation

For cases where the pre-incremented value is not needed for assignment to another variable, such as when through a for-loop, use `++i` instead of `i++`.

Status

Unresolved

[G-03] Use `unchecked` math for non-overflowing operations

Description

For Solidity pragma versions 0.8.0 and above, it is possible to add code inside an `unchecked` block to bypass overflow/underflow checks. These checks aren't necessary for operations that are guaranteed to not overflow/underflow, such as incrementing inside a for-loop.

Recommendation

Use unchecked blocks when incrementing the loop counter.

An example of a for-loop with an unchecked loop counter is below.

```
for (uint256 i; i < arrayLength;) {  
    // do something  
    unchecked {  
        ++i;  
    }  
}
```

Status

Unresolved

[G-04] RBAC - Use `constant` state variables for role types

Description

The `RBAC` abstract upgradeable contract initialises the hash digest of the roles in the `initialize` function and stores them in state variables. This costs more gas to access and store.

Recommendation

Declare the role state variables as `public constant` hash digests. This works with upgradeable contracts since `constant` state variables are stored inside the contract bytecode instead of storage.

Status

Unresolved

[G-05] Cache array length before iterating over it

Description

The length of an array is accessed directly before each iteration.

```
for (uint256 i = 0; i < addresses.length; i++) {
    nodeConfigs[i] = nodeAddressToConfigMap[addresses[i]];
}
```

There are many instances in the codebase of this occurring.

Recommendation

If the length of an array does not change while iterating over it, gas can be saved by caching the array length value to memory and accessing the value from memory.

```
uint256 addressesLength = addresses.length;
for (uint256 i = 0; i < addressesLength; i++) {
    nodeConfigs[i] = nodeAddressToConfigMap[addresses[i]];
}
```

Status

Unresolved

[G-06] Result of math operations can be cached into memory if reused

Description

Inside `Reconfiguration#getRandomIndexes`, the value `(networkSize - 1) / toleranceFactor` is calculated multiple times.

```

if (candidates.length >= (networkSize - 1) / toleranceFactor) {
    randomGovernorIndexes = PseudoRandomNumberGenerator
        .generateMultiple(
            0,
            networkSize,
            (networkSize - 1) / toleranceFactor,
            seed
        );
    randomCandidateIndexes = PseudoRandomNumberGenerator
        .generateMultiple(
            0,
            candidates.length,
            (networkSize - 1) / toleranceFactor,
            seed
        );
} else if (candidates.length < (networkSize - 1) / toleranceFactor) {
    randomGovernorIndexes = PseudoRandomNumberGenerator
        .generateMultiple(0, networkSize, candidates.length, seed);
    randomCandidateIndexes = PseudoRandomNumberGenerator
        .generateMultiple(
            0,
            candidates.length,
            candidates.length,
            seed
        );
}

```

Recommendation

Cache the result of `(networkSize - 1) / toleranceFactor` into memory and use the cached value to save gas on math operations.

Status

Unresolved

[G-07] VotingContract - Use a `mapping` for `authorisedUsers` instead of array

Description

Authorised users are stored in an array, which is iterated over to determine if a particular address is a user. Iterating over an array can get very costly if the array is large.

Recommendation

Use a mapping to store authorised user addresses.

Status

Unresolved

[G-08] VotingContract - `networkConfigurationAddress` state variable is redundant

Description

The address of the NetworkConfiguration contract and its interface are both stored inside state variables, taking up 2 storage slots.

Recommendation

Store the NetworkConfigurationInterface instance and when the address is required, cast to an address.

Status

Unresolved

[G-09] VotingContract - Assign `_authorisedUsers` array to `authorisedUsers` instead of iterating over array values

Description

Related to [G-07]

Inside the constructor, the `_authorisedUsers` array is iterated over and its elements are pushed to the `authorisedUsers` array individually.

```
for (uint256 i = 0; i < _authorisedUsers.length; i++) {
    authorisedUsers.push(_authorisedUsers[i]);
}
```

Recommendation

Assign the `_authorisedUsers` array to `authorisedUsers` to save gas on iterating through the array and pushing values individually.

```
authorisedUsers = _authorisedUsers;
```

Status

Unresolved

[G-10] VotingContract - Redundant updates to `votingEnded`

Description

Assigning `votingEnded = true` inside `cancelProposal` and `banGovernor` is not required, as both functions call `resetVoting` which resets the value back to false.

Furthermore, `banGovernor` can only be called if `votingEnded = true` due to the `proposalActivated` modifier`

```

modifier proposalActivated() {
    require(
        isThresholdReached() && votingEnded,
        "Proposal did not get enough yes votes"
    );
    _;
}

...

function banGovernor() public proposalActivated onlyAuthorisedUser {
    networkConfigurationContract.banGovernorNode(
        activeProposal.nodeAddress
    );
    votingEnded = true;
    resetVoting();
    emit VotingEnded("Voting has ended for the proposal");
}

function resetVoting() private {
    require(votingEnded, "Voting has not ended yet");

    activeProposal = Proposal(
        "",
        address(0),
        0,
        0,
        0,
        ""
    );
}

```

```

        address(0),
        false
    );
    votingEnded = false;
    for (uint256 i = 0; i < voters.length; i++) {
        delete hasVoted[voters[i]];
    }
    delete voters;
}

```

Recommendation

Remove the `votingEnded` check inside `resetVoting` and the `votingEnded = true` assignments inside `banGovernor` and `cancelProposal`.

Status

Unresolved

[G-11] VotingContract#resetVoting - Delete `activeProposal` instead of reassigning it to an empty Proposal object

Description

To reset the active proposal, unnecessary stack operations are performed to create a `Proposal` struct with default values.

```

    activeProposal = Proposal(
        "",
        address(0),
        0,
        0,
        0,
        "",
        address(0),
        false
    );

```

Recommendation

Save gas on unnecessary stack operations by using `delete activeProposal` to reset the state variable back to its default values.

```

    delete activeProposal;

```

Status

Unresolved

[G-12] ActivityMonitor - Unnecessary state variables**Description**

`slashPrctnt` and `daysToServInJail` are state variables that are initialised during deployment. However, they do not have any relation with `JailedGovernors.slashPrctnt` and `JailedGovernors.daysToServe`.

`inactivityThreshold` and `windowSize` state variables are not used inside the contract.

Recommendation

Remove the state variables stated above from the contract.

Status

Unresolved

Additional Contracts Smart Contract Audit Findings

High

[H-01] BusinessIdentifier - Authorised representatives can add and remove other authorised representatives

Description

The contract `BusinessIdentifier` allows authorised representatives to add and remove other authorised representatives. This may be a security risk as the authorised representatives may be able to add and remove other authorised representatives without the business owner's consent.

Vulnerability

The current access control pattern relies on an 'N of N' trust model, meaning that all authorised representatives have to act as expected for the contract to function as intended.

Any authorised representative that is compromised/malicious can add/remove other authorised representatives as they wish, compromising the security of the contract.

Recommendation

Change the access control such that only the admin can add and remove authorised representatives.

Status

Acknowledged

Redbelly

This is a business decision. We have decided to let this be the same.

[H-02] ReleaseAgreement#updateRetriever - Retriever can be changed at any time, even after the release has been approved and made effective

Description

The function `updateRetriever` can be called at any time, even when `userRecoveryDetails.release = true`.

Impact

A malicious approver can frontrun the legitimate retriever in calling `updateRelease` by calling `updateRetriever` then `updateRelease` to lock in a different retriever.

Recommendation

Add a require statement to check if `userRecoveryDetails.release` is false.

```
require(
  !userRecoveryDetails.release,
  "Release has already been approved"
);
```

Status

Resolved

Redbelly

We have added the check that if once the retriever is set by any one of the guardians, we won't allow it to be set again. Also another check has been applied that this will be allowed only if release is not true.

[H-03] ReleaseAgreementFactory#create - User can circumvent guardian permission checks by providing a user address as a guardian

Description

The same permission check is used for both guardians and users. This means that a user can circumvent the guardian permission check by providing a user address as a guardian.

Vulnerability Details

The following lines perform the checks for users and the guardian:

```
bool permissionCheckUser = RAFLibrary.checkPermissionOfAddress(
  permissionContractAddress, "credentialSubject.publicAddress",
  _backupCredentials
);
require(permissionCheckUser, "User do not have write permission");
bool permissionCheckGuardian = RAFLibrary.checkPermissionOfAddress(
  permissionContractAddress, "credentialSubject.guardianAddress",
  _backupCredentials
);
require(permissionCheckGuardian, "Guardian do not have write permission");
```

This accesses the `isAuthorizedUser` mapping to check if the user is authorised. However, the same mapping is used to check if the guardian is authorised.

```
function checkPermissionOfAddress(
```

```

    address _permissionContractAddress,
    string memory path,
    string[] memory _backupCredentials
) external returns (bool) {
    bool permissionCheck = true;
    Permission permission = Permission(_permissionContractAddress);
    for (uint256 i = 0; i < _backupCredentials.length; i++) {
        if (
            !(
                permission.isAllowed(
StringToAddress.stringToAddress(JsonParserAndVerify.parseJson(path,
_backupCredentials[i]))
                )
            ) {
                permissionCheck = false;
                break;
            }
        }
    }
    return permissionCheck;
}

```

Hence, providing a user address as a guardian will pass the permission check, as a user and guardian are both checked against the same mapping.

The steps to create an unauthorised release agreement are as follows:

1. As an approved user who is authorised in the Permission contract, create backup credentials with arbitrary data, making sure that the `credentialSubject.guardianAddress` is your address.
2. Generate proof of the backup credentials using your private key.
3. With the proof and backup credentials, you can create a release agreement with any user as the guardian. You will pass the signature check, since the proof matches with your public key.

Impact

A user can create an unauthorised release agreement with any list of approvers who are authorised in the Permission contract.

Recommendation

Add a separate mapping for guardians and users inside the Permission contract.

Status

Pending further review

Redbelly

We have added a check that the person who is trying to deploy the RA contract has not already deployed any contract before. We also have made the RA contract not upgradable so the user can make no changes to that. We also have added a check that

the guardians are both different. As we have decided to not have a GuardianRegistry, so these are the steps taken instead.

Hashlock

Checking for unique guardians, though recommended, still does not prevent this issue. The issue arises due to both user and guardian permission checks being functionally identical and interchangeable, which means that any allowed user can be a guardian/approver. Please refer back to the recommended fix in the report.

[H-04] ReleaseAgreementFactory#create - Function is vulnerable to replay attacks

Description

The `ReleaseAgreementFactory` does not make use of nonces to prevent replay attacks. This means that an attacker can replay a previous valid transaction to create the same release agreement multiple times.

Vulnerability Details

The create function uses `RAFLibrary.verifyBackupCredential1` which uses `JsonParserAndVerify.verifyED25519` to verify that the backup credentials provided are signed by the guardians with cryptographic signatures. However, no nonce is used and checked in the credentials payload, meaning that it is possible for a replay attack to occur.

Impact

A malicious user can create another identical release agreement contract, with the caveat that they're now the owner of the upgradeable contract. This allows them to perform the following attack:

1. Create an identical release agreement contract by replaying a past transaction, except now the owner of the contract is the malicious user.
2. Upgrade the logic contract to include a setter function that allows the owner of the contract to set `userRecoveryDetails.release` to true and `userRecoveryDetails.retriever` to the owner, without the need for approver checks.
3. Proceed with key recovery and obtain the key shards.

Recommendation

Include a nonce in the backup credentials payload, and track and check this nonce value inside `ReleaseAgreementFactory`.

Status

Resolved

Redbelly

We have added a condition to check that if a person's mapping is already present in RAF, i.e. if they have already deployed a RA contract so the contract won't allow them to deploy again. Hence no re-entrancy.

Hashlock

The proposed change does not address the issue, which is a replay attack (not reentrancy). However, the addition of a `verifyUser` check fixes the issue.

Medium

[M-01] ReleaseAgreement#updateIsValid - Missing access control allows any user to update the validity of the release agreement

Description

The function `updateIsValid` is missing access control and allows any user to update the validity of the release agreement.

Impact

Any user can update the validity of the release agreement.

Recommendation

Add access control to the function `updateIsValid`. We recommend limiting access to the admin as opposed to approved users.

Status

Resolved

Redbelly

`isValid` field has been removed.

[M-02] ReleaseAgreement#updateApproverChecks - Approved users can grief the release agreement by delaying the release process

Description

When the minimum approval threshold is reached and the retriever is waiting for the cooldown time to pass to initiate the release, any approved user can call the function `updateApproverChecks` to reset the cooldown time, delaying the release.

Vulnerability Details

Impact

The release can be delayed for many days or weeks, depending on the number of approvals after the threshold has already been reached.

Recommendation

Allow approvers to update their approval status only if the minimum approval threshold has not been reached.

```
function updateApproversChecks() public onlyApprovers {
    require(
        userRecoveryDetails.approversChecks.length <
        userRecoveryDetails.minApprover,
        "Minimum approval threshold reached"
    );
    ...
}
```

Status

Resolved

[Redbelly](#)

We have added the check that if minimum approvers have already been approved, then there is no need to change the timestamp again.

Low

[L-01] BootstrapContractRegistry#getContractAddress - Function should validate against the zero address

Description

The function `getContractAddress` gets the address of a contract by its name. However, if the `contractName` is not inside the registry, it'll return the zero address.

Impact

An external function that calls `getContractAddress` and doesn't validate the returned address against the zero address can be vulnerable to a DoS attack.

Recommendation

Add a `require` statement to validate the to-be-returned address against the zero address.

```
function getContractAddress(string memory contractName) public view returns
(address) {
    address _contractAddress = registry[contractName];
    require(_contractAddress != address(0), "Contract address not found");
    return _contractAddress;
}
```

Status

Unresolved

[L-02] BootstrapContractRegistry - Do not hard-code addresses

Description

The contract `BootstrapContractRegistry` hard-codes the addresses of the contracts inside the registry. This is not a good practice because the stored addresses of the contracts may be incorrect.

Impact

The registry may return incorrect contract addresses.

Recommendation

Allow the constructor of the contract to take in another two arrays:

1. An array of contract names
2. An array of contract addresses

The constructor will then store the contract names and addresses into the registry.

This way, the registry deployer can initialise the registry with the correct contract names and addresses.

Status

Unresolved

[L-03] BusinessIdentifier#onlyAuthorisedDelegte - Typo in modifier name

Description

The `onlyAuthorisedDelegte` function is spelt incorrectly.

Recommendation

Correct the spelling to `onlyAuthorisedDelegate`.

Status

Unresolved

[L-04] BusinessIdentifierFactory#changeBusinessPublicAddress - Typo in function name

Description

The `changeBusinessPublicAddress` function is spelt incorrectly.

Recommendation

Correct the spelling to `changeBusinessPublicAddress`.

Status

Unresolved

[L-05] BusinessIdentifierFactory#deployContract - BusinessIdentifier logic contract should be initialised after deployed

Description

It is best practice to initialise the logic contract after it has been deployed. This is because the logic contract in some cases can have logic that changes the behaviour of the contract after it has been deployed.

Recommendation

Initialise the logic contract after it has been deployed.

```
function deployContract(
    address _businessPublicAddress,
    string memory _companyName,
    string memory _incorporatedName,
    string memory _identifierType,
    string memory _identifier,
    string memory _businessAddress,
    bool _isBeneficialOwner
) external onlyOwner {
    ...
    address _businessIdentifier = address(new BusinessIdentifier());

    ERC1967Proxy proxy = new ERC1967Proxy(
        address(_businessIdentifier),
        initData
    );

    BusinessIdentifier(_businessIdentifier).initialize(
        msg.sender,
        _companyName,
        _incorporatedName,
        _identifierType,
        _identifier,
        _businessAddress,
        _isBeneficialOwner
    );
}
```

```
);
...
}
```

Status

Unresolved

[L-06] Only use one version of Solidity

Description

`ReleaseAgreement` and `ReleaseAgreementFactory` have the following `pragma` lines

```
pragma solidity >=0.8.2 <0.9.0;
```

It is best practice to set the `pragma` to one version of Solidity to prevent any unexpected behaviour that would result from contracts being compiled with different compiler versions.

Recommendation

Use one `pragma` version for the entire project. We recommend a more up-to-date version such as `v0.8.19`.

Status

Unresolved

[L-07] JsonParserAndVerify - Use `staticcall` instead of `call` for non-state changing calls

Description

The contract `JsonParserAndVerify` uses `call` for non-state changing calls to pre-compiles. For safety reasons, it is best practice to use `staticcall` instead.

Recommendation

Replace `call` with `staticcall` for non-state changing calls to pre-compiles.

Status

Unresolved

Gas

[G-01] IDPRegistry - Contract should store IDP Proofs in arrays instead of a mapping to save gas

Description

The contract `IDPRegistry` stores IDP proofs in nested mappings. This provides no benefit over an array as the proof count is still stored in another mapping and is still retrieved by iterating through the mapping as if it were an array.

Recommendation

For simplicity, the contract should store IDP proofs in arrays. This will actually save gas as the contract will not have to store the proof and proof count in two mappings.

```
mapping(uint256 => Proof[]) private idpProofs;
```

Status

Unresolved

[G-02] BusinessIdentifier - Switch order of checks in require statements to save gas

Description

The access control modifiers check for the default admin role and then the appropriate authorised roles. This is not gas efficient in most cases where the functions will be called by the authorised roles as opposed to the admin.

Recommendation

Swap the order of the checks inside the require statements such that it checks for the admin role last.

```
modifier onlyAuthorisedRepresentative() {
    require(
        hasRole(DEFAULT_ADMIN_ROLE, _msgSender()) ||
        hasRole(AUTHORISED_REPRESENTATIVE_ROLE, _msgSender()),
        "Caller must have IDP or Authorised Representative"
    );
    -
}
```

becomes

```

modifier onlyAuthorisedRepresentative() {
    require(
        hasRole(AUTHORISED_REPRESENTATIVE_ROLE, _msgSender()) ||
        hasRole(DEFAULT_ADMIN_ROLE, _msgSender()),
        "Caller must have IDP or Authorised Representative"
    );
}

```

Status

Unresolved

[G-03] BusinessIdentifierFactory#changeBusinessPublicAddress - Redundant check for invalid business address

Description

The `changeBusinessPublicAddress` function checks if the input `_contractAddress` matches with the current business address. This is redundant as the function can just access the current business address directly to store it in the new public address key-value pair.

Recommendation

Cache the current business address directly from the mapping and store it in the new public address key-value pair.

```

function changeBusinessPublicAddress(
    address _oldAddress,
    address _newAddress,
) external onlyOwner {
    address _contractAddress = businessContracts[_oldAddress];

    delete businessContracts[_oldAddress];
    businessContracts[_newAddress] = _contractAddress;

    emit BusinessPublicAddressUpdated(
        _oldAddress,
        _newAddress,
        _contractAddress
    );
}

```

Status

Unresolved

[G-04] BusinessIdentifier and ReleaseAgreement details for every business or release agreement can be stored inside the same contract

Description

Business identifiers and release agreements are deployed using factory contracts. The data of every business or release agreement are stored in each contract which needs to be deployed separately. This is very gas inefficient.

Recommendation

Instead of having one contract per business or release agreement and factory contracts to deploy them, the data for every business or release agreement can be stored inside the same contract using a mapping to access the data for each business or release agreement.

This will simplify the codebase and reduce the gas costs of deploying and interacting with the contracts.

Status

Unresolved

[G-05] JsonParserAndVerify#verifyED25519 - Redundant check on callresult

Description

The function `verifyED25519` has a `require` statement to check that `callresult` is true. However, this has already been checked by the assembly code.

```
// Check the result of the call
switch callresult
case 0 {
    invalid()
}
```

Recommendation

Remove the following `require` statement:

```
require(callresult, "fail-hash");
```

Status

Unresolved

[G-06] JsonParserAndVerify#verifyED25519 - Use revert instead of invalid

Description

The function `verifyED25519` uses the `invalid` Yul function to stop execution if the result of the precompile call is `false`. This is wasteful as the `invalid` function does not refund the remaining gas to the transaction originator.

Recommendation

Use `revert` instead of `invalid`.

Status

Unresolved

[G-07] JailedGovernors#jail - External call to TombstonedGovernors.tombstoneThreshold uses more gas

Description

The tombstone threshold variable used to determine whether a governor is eligible for being tombstoned is stored in the `TombstonedGovernors` contract as a state variable when it's only referenced by `JailedGovernors`. It makes more sense to store it in `JailedGovernors` instead, so that the external call to `TombstonedGovernors.tombstoneThreshold` can be avoided.

Recommendation

Move the `tombstoneThreshold` variable from `TombstonedGovernors` to `JailedGovernors`.

Status

Unresolved

Final Audit Round Findings

High

[H-01] PriceFeedContract#getLatestPrice - Function should not return price if stale

Description

The `getLatestPrice` function returns the latest price as well as the timestamp of when the price was last updated. However, this is insufficient as external contracts may accidentally use the stale price.

Vulnerability Details

Currently, `getLatestPrice` returns `(price, lastUpdated)`. It is up to contracts that rely on the price feed to implement staleness checks. In this case, staleness is subjective to external contracts and the threshold that they decide on. This can lead to integration issues if there are interoperable contracts that rely on the same price feed, but implement different staleness checks. `PriceFeedContract` should determine the threshold for price staleness.

Impact

External contracts may not implement staleness checks, or implement them incorrectly, potentially leading to loss of funds.

Recommendation

Include a state variable `updateThreshold` that determines the threshold for price staleness. This can be set in the constructor. `updateThreshold` can be used inside `getLatestPrice` to change the behavior of the function depending on whether the price is stale. There are two different possibilities:

1. If the price is determined to be stale, return `-1` in place of `price` and include in documentation and Natspec comments for contract developers to include a check for staleness

```

```diff
 /// @notice Fetches the token price along with the last updated timestamp
- /// @return Current price of 1 RBNT in USD and the last updated timestamp
+ /// @dev Include check for price staleness
+ /// @return Current price of 1 RBNT in USD and the last updated timestamp. Returns `price = -1` when price is stale
 function getLatestPrice() external view override returns(int256, uint256) {
+ if (block.timestamp > lastUpdated + updateThreshold) {
+ return (-1, lastUpdated);
+ }
 return (price, lastUpdated);
 }
...

```

2. If the price is stale, throw an error:

```

```diff
  /// @notice Fetches the token price along with the last updated timestamp
  /// @return Current price of 1 RBNT in USD and the last updated timestamp
  function getLatestPrice() external view override returns (uint256, uint256) {
+   require(block.timestamp <= lastUpdated + updateThreshold, "Price is stale");
    return (price, lastUpdated);
  }
...

```

Status

Resolved

[H-02] StakingDepositUpgradeable#unstake - Function is vulnerable to re-entrancy attacks

Description

The `unstake` function does not follow the Checks-Effects-Interaction (CEI) pattern and is vulnerable to a re-entrancy attack that can drain the `StakingDepositUpgradeable` contract.

Vulnerability Details

The `unstake` function transfers the depositor's `depositedAmount` before resetting their balance to 0. This is an 'interaction' before 'effects' and hence is vulnerable to re-entrancy.

```

```solidity
function unstake() external payable canUnstake {
 uint256 unstakeValue = deposits[_msgSender()].depositedAmount;
 (bool sent,) = payable(_msgSender()).call{value: unstakeValue}(""); // interactions
 require(sent, "Failed to unstake");
 delete coolOffStartTimestamp[_msgSender()]; // effects
 deposits[_msgSender()].depositedAmount = 0; // effects
 emit StakeWithdrawn(_msgSender(), unstakeValue);
}
...

```

An attacker can call `unstake` again as a callback after they receive the native tokens, allowing the same amount of tokens to be withdrawn recursively until the `StakingDepositUpgradeable` contract is drained.

## Impact

The `StakingDepositUpgradeable` contract will be drained completely of all of its native tokens.

## Recommendation

Follow the CEI pattern by resetting the depositor's `depositedAmount` to 0 before sending the native token as shown:

```

```solidity
function unstake() external payable canUnstake {
    uint256 unstakeValue = deposits[_msgSender()].depositedAmount;

    delete coolOffStartTimestamp[_msgSender()]; // effects
    deposits[_msgSender()].depositedAmount = 0; // effects

    (bool sent, ) = payable(_msgSender()).call{value: unstakeValue}(""); // interactions
    require(sent, "Failed to unstake");

    emit StakeWithdrawn(_msgSender(), unstakeValue);
}
```

```

## Status

Resolved

### [H-03] PriceFeedContract - Contract does not specify decimals of `price`

**Description** The price feed does not indicate the number of decimals of `price`. This can result in incorrect integrations.

**Vulnerability Details** It is good practice for price feeds to also indicate the number of decimals of `price`. This makes sure that contracts that read from the price feed know the correct number of decimals to use. This is especially the case if the unit of denomination is an asset that does not have a specified number of decimals.

**Impact** Contracts that read from the price feed may assume the wrong number of decimals, resulting in incorrect prices.

**Recommendation** The contract can specify a standard number of decimals and make it clear in documentations and comments. A good number of decimals to use is 18. Another option is to add a new state variable for the number of decimals which is set in the constructor and return that information inside `getLatestPrice`.

**Status** Resolved

**Medium**

## [M-01] StakingDepositUpgradeable#canUnstake - Cool-off period uses `depositTimestamp` as the start time

### Description

The `canUnstake` modifier checks if the cool-off period is complete by using the `depositTimestamp` as the start time. This is incorrect

### Vulnerability Details

The `canUnstake` modifier checks if the cool-off period is complete by using the `depositTimestamp` as the start time instead of the `coolOffStartTimestamp` mapping. ``solidity require( block.timestamp >= deposits[\_msgSender()].depositTimestamp + coolOffPeriod, "Cool-off Period not complete" ); `` This allows nodes to withdraw their stake before their actual cooloff period has finished.

```
``solidity
require(
 block.timestamp >=
 deposits[_msgSender()].depositTimestamp + coolOffPeriod,
 "Cool-off Period not complete"
);
````
```

Impact

A node operator can unstake before the cool-off period is complete.

Recommendation

Create a new `public view` function to determine if the node has completed the cool-off period. An example of one is shown below:

```
``solidity
function isCoolOffFinished(address _nodeAddress) public view returns (bool) {
    uint256 coolOffEndTimestamp = coolOffStartTimestamp[_nodeAddress] + coolOffPeriod;
    return coolOffStartTimestamp[_nodeAddress] > 0 && block.timestamp >= coolOffEndTimestamp;
}
````
```

### Status

Unresolved

## [M-02] BusinessIdentifierFactory#deployContract - Businesses can be grieved by IDPs overwriting their `businessContracts` mapping value

### Description

The `deployContract` function does not contain any verification checks to ensure that the user calling the function represents the business. Another user can grief the business by overwriting their `BusinessIdentifier` contract with another one.

### Vulnerability Details

The `deployContract` function uses the `onlyIDP` modifier for access control. However, there is no checks to ensure that the `msg.sender` is associated to the given `_businessPublicAddress`, or if the `_businessPublicAddress` is already associated with a `BusinessIdentifier` contract. This makes it possible for another IDP to grief the business by overwriting their `BusinessIdentifier` contract with another one by calling `deployContract` with the same `_businessPublicAddress`. The business cannot revert the changes and will have to deploy a new `BusinessIdentifier` contract to replace it.

### Impact

The business will have their `BusinessIdentifier` contract overwritten, and the parameters of the new contract can be arbitrarily set by the griever.

### Recommendation

Two checks should be performed in the `deployContract` function to restrict its use: 1. Require that the `_businessPublicAddress` inputted does not already have a `BusinessIdentifier` contract associated to it. 2. Require proof that the `_businessPublicAddress` intended to deploy the contract with the given arguments. This can be done via EIP-712 signatures, or by limiting calls to the `deployContract` function to the `_businessPublicAddress` itself.

### Status

Acknowledged

### [M-03] `PermissionUpgradeable#enablePermissionedAccess` - Function is missing access control

#### Description

The `enablePermissionedAccess` function does not have access control checks, allowing anyone to enable permissioned access.

#### Vulnerability Details

The `isPermissionedAccessEnabled` boolean state variable determines if access control is enabled for contracts that inherit from the `PermissionUpgradeable` contract. Permissioned access control is enabled by calling the `enablePermissionedAccess` function, which does not have any access control itself. This means that any user can enable permissioned access control, even when it's not intended by the contract owner.

#### Impact

A malicious user can grief and cause DoS for other users if they call `enablePermissionedAccess`.

## Recommendation

Add the `onlyOwner` modifier from `RBACUpgradeable` to `enablePermissionedAccess`, so that the function can only be called by the contract owner.

## Status

Resolved

## [M-04] NetworkConfigurationUpgradeable#initialize - Candidate nodes have incorrect config

### Description

The `initialize` function uses bootstrap nodes' `signingAddress` instead of candidate nodes' `signingAddress` inside the `NodeConfig`.

### Vulnerability Details

Inside `initialize`, the signing address of bootstrap nodes is assigned inside the `NodeConfig` of candidate nodes

```

```solidity
for (uint256 i = 0; i < _candidateNodes.length; i++) {
    NodeConfig memory newConfig = NodeConfig(
        _candidateNodes[i].id,
        _candidateNodes[i].jsonRpcPort,
        _candidateNodes[i].consensusPort,
        _candidateNodes[i].grpcPort,
        _candidateNodes[i].nodeAddress,
        _bootstrapNodes[i].signingAddress,
        _candidateNodes[i].hostname
    );
    ...
}
```

```

This results in candidate nodes having the same signing address as bootstrap nodes. If there are more candidate nodes than bootstrap nodes, the `initialize` call will fail and the contract will not be able to be deployed. If the `initialize` call does not revert, then the `removeNodeDetailsFromNetwork` function will delete the incorrect `signingAddressToNodeAddress` mapping value.

```

```solidity
function removeNodeDetailsFromNetwork(
    address nodeAddress
) external onlyJailedGovernorsContract {
    ...
    delete signingAddressToNodeAddress[nodeConfig.signingAddress];
    delete isHostnameInUse[nodeConfig.hostname];
    delete nodeAddressToConfigMap[nodeConfig.nodeAddress];
}
```

```

The `signingAddressToNodeAddress` mapping is used to check if a node address is a governor inside the `ActivityMonitorUpgradeable.isGovernor` function, which is used by the `ActivityMonitorUpgradeable.onlyGovernor` modifier. This modifier is used as access control in `ActivityMonitorUpgradeable.addProposal` and `ActivityMonitorUpgradeable.vote`. The `removeNodeDetailsFromNetwork` function is called when a governor is to be jailed after a vote. This means that whenever a candidate inserted at initialization is jailed, a bootstrap node is no longer able to call `ActivityMonitorUpgradeable.addProposal` or `ActivityMonitorUpgradeable.vote`.

### Impact

A candidate being jailed results in another bootstrap node being unable to add new proposals or vote in the Activity Monitor.

### Recommendation

Use the candidate node's `signingAddress` instead of the bootstrap node's `signingAddress` inside the `NodeConfig`.



```

```diff
for (uint256 i = 0; i < _candidateNodes.length; i++) {
    NodeConfig memory newConfig = NodeConfig(
        _candidateNodes[i].id,
        _candidateNodes[i].jsonRpcPort,
        _candidateNodes[i].consensusPort,
        _candidateNodes[i].grpcPort,
        _candidateNodes[i].nodeAddress,
-       _bootStrapNodes[i].signingAddress,
+       _candidateNodes[i].signingAddress,
        _candidateNodes[i].hostname
    );
    ...
}
```

```

## Status

Resolved

## [M-05] NodeRecordStorage#insert/remove - No checks to make sure that `nodeAddr` exists/doesn't exist

### Description

There is no check in `insert` to make sure that `nodeAddr` doesn't already exist inside the `NodeRecord` struct, and no check in `remove` to make sure that `nodeAddr` exists. This results in the node address and index mapping being out of sync and having duplicate values.

### Vulnerability Details

When trying to insert a `nodeAddr` that already exists, its `nodeAddressToIdx` value gets replaced with the latest index:

```

```solidity
function insert(NodeRecord storage self, address nodeAddr) internal {
    // @audit `nodeAddr` key already has a value which is replaced by current `nodeCounter`
    self.nodeAddressToIdx[nodeAddr] = self.nodeCounter.current();
    self.nodeIdxToAddress[self.nodeCounter.current()] = nodeAddr;
    self.nodeCounter.increment();
}
```

```

When trying to remove a `nodeAddr` that doesn't exist, the node at index 0 gets removed from the `nodeIdxToAddress` mapping as described below:

```

```solidity
function remove(NodeRecord storage self, address nodeAddr) internal {
    // @audit `nodeIndex = 0` if `nodeAddr` doesn't exist
    uint256 nodeIndex = self.nodeAddressToIdx[nodeAddr];
    // @audit Node at index 0 gets replaced
    self.nodeIdxToAddress[nodeIndex] = self.nodeIdxToAddress[
        self.nodeCounter.current() - 1
    ];
    // @audit Latest added node points to index 0
    self.nodeAddressToIdx[self.nodeIdxToAddress[nodeIndex]] = nodeAddr;

    delete self.nodeIdxToAddress[self.nodeCounter.current() - 1];
    // @audit this deletes nothing!
    delete self.nodeAddressToIdx[nodeAddr];

    self.nodeCounter.decrement();
}
```

```

Both of these situations lead to the two mappings being out of sync, pointing to incorrect values and also having duplicate values. However, the node originally at index 0 does not get deleted from the `nodeAddressToIdx` mapping, resulting in it being out of sync with `nodeIdxToAddress`

### Impact

For `insert`: 1. The original index value of `nodeAddr` is replaced with `nodeCounter.current()`. 2. Because the original node index is not removed from `nodeIdxToAddress`, there are now two indexes (original index and current index) which now point to `nodeAddress`. For `remove`: 1. The node at index 0 is unintentionally removed from the `nodeAddressToIdx` mapping. 2. Because the node at index 0 is not removed from `nodeAddressToIdx`, this results in the two mappings being out of sync. This now means that there are now two addresses (original index 0 node and node at index `nodeCounter.current()`) that point to index 0. This finding is related to [M-09]

### Recommendation

1. Include a check at the beginning of `insert` to make sure that `nodeAddr` doesn't exist:

```
```diff
function insert(NodeRecord storage self, address nodeAddr) internal {
+   require(!nodeExists(self, nodeAddr), "nodeAddr already exists");
   self.nodeAddressToIdx[nodeAddr] = self.nodeCounter.current();
   self.nodeIdxToAddress[self.nodeCounter.current()] = nodeAddr;
   self.nodeCounter.increment();
}
```
```

2. Include a check at the beginning of `remove` to make sure that `nodeAddr` exists:

```
```diff
function remove(NodeRecord storage self, address nodeAddr) internal {
+   require(nodeExists(self, nodeAddr), "nodeAddr doesn't exist");
   uint256 nodeIndex = self.nodeAddressToIdx[nodeAddr];
   self.nodeIdxToAddress[nodeIndex] = self.nodeIdxToAddress[
       self.nodeCounter.current() - 1
   ];
   self.nodeAddressToIdx[self.nodeIdxToAddress[nodeIndex]] = nodeIndex;
   delete self.nodeIdxToAddress[self.nodeCounter.current() - 1];
   delete self.nodeAddressToIdx[nodeAddr];
   self.nodeCounter.decrement();
}
```
```

## Status

Resolved

## [M-06] JailedGovernorsUpgradeable#free - `jailTenures` only stores latest jail time

### Description

The `jailTenures` mapping is supposed to store the aggregate duration of time that a governor has been jailed for. However, the mapping only stores the duration of the latest jail.

### Vulnerability Details

The `jailTenures` mapping is supposed to store the aggregate duration of time that a governor has been jailed for. However, the mapping only stores the duration of the latest jail as indicated in the `free` function: ``solidity function free( address \_governor ) external onlyActivityMonitorContract notTombstoned(\_governor) returns (bool) { ... jailTenures[\_governor] = block.timestamp - jailStartTimestamps[\_governor]; ... } `` Since `jailTenures` is used inside `StakingDepositUpgradeable.canUnstake` modifier to determine if a node can initiate their cool-off or unstake, this means that the node can perform these actions earlier than intended.

### Impact

A node that has been jailed can initiate their cool-off and unstake earlier than intended.

### Recommendation

Fix the calculation of `jailTenures[\_governor]` to be `+=` instead of `=`: ```diff function free( address \_governor ) external onlyActivityMonitorContract notTombstoned(\_governor) returns (bool) { ... - jailTenures[\_governor] = + jailTenures[\_governor] += block.timestamp - jailStartTimestamps[\_governor]; ... } ```

### Status

Unresolved

## [M-07] StakingDepositUpgradeable#unstake - Function is marked as `payable`

### Description

The `unstake` function is marked as `payable`. Any tokens sent will be lost.

### Vulnerability Details

The `unstake` function allows a node to unstake their tokens from the network and receive back their staked tokens. It should not be accepting native tokens via the `payable` keyword. Since there is no logic in the function to account for `msg.value`, any native tokens sent to the contract via the `unstake` function will be lost.

### Impact

Users who send tokens via `unstake` will lose them.

### Recommendation

Remove the `payable` keyword.

### Status

Acknowledged

## [M-08] NetworkConfiguration#initialize - Candidate nodes do not get added to `signingAddressToNodeAddress` mapping

### Description

Initialized candidate nodes do not get added to the `signingAddressToNodeAddress` mapping. This means that once these candidate nodes become governors, they cannot vote in or create new proposals.

### Vulnerability Details

The `initialize` function adds initial candidate nodes into the `candidates` `NodeRecord`. However, it does not store the signing address of the candidate nodes

```

```solidity
for (uint256 i = 0; i < _candidateNodes.length; i++) {
    NodeConfig memory newConfig = NodeConfig(
        _candidateNodes[i].id,
        _candidateNodes[i].jsonRpcPort,
        _candidateNodes[i].consensusPort,
        _candidateNodes[i].grpcPort,
        _candidateNodes[i].nodeAddress,
        _bootStrapNodes[i].signingAddress,
        _candidateNodes[i].hostname
    );
    isHostnameInUse[_candidateNodes[i].hostname] = true;
    nodeAddressToConfigMap[_candidateNodes[i].nodeAddress] = newConfig;
    // @audit does not write to `signingAddressToNodeAddress` mapping
    candidates.insert(_candidateNodes[i].nodeAddress);
    idCounter.increment();
}
```

```

The `signingAddressToNodeAddress` mapping is accessed by `ActivityMonitor.getNodeAddressUsingSignerAddress` for access control. Hence, an initial candidate that is now a governor will not be able to vote or add new proposals.

## Impact

Initialized candidates who are now governors will not be able to vote or add new proposals in `ActivityMonitor`.

## Recommendation

Add the candidate's signing address to the `signingAddressToNodeAddress` mapping.

## Status

Resolved

## Low

### [L-01] RBACUpgradeable & ContractRoleAuthUpgradeable - Role identifiers can be made `constant`

## Description

The role identifiers inside the `RBACUpgradeable` and `ContractRoleAuthUpgradeable` contract can be made `constant`, instead of being initialized in the `initialize` function. This is safe to do for upgradeable contracts since constants are stored in the contract's bytecode instead of storage. This is best practice as recommended by the OpenZeppelin Natspec documentation. ``solidity /\*\* \* ... \* Roles are referred to by their `bytes32` identifier. These should be exposed \* in the external API and be unique. The best way to

achieve this is by \* using `public constant` hash digests: \* \* ``solidity \* bytes32 public constant MY\_ROLE = keccak256("MY\_ROLE"); \* `` \* ... \*/ ``

## Recommendation

Make the role identifiers `constant` instead of initializing them in the `initialize` function.

```
In `RBACUpgradeable`: ``solidity bytes32 public constant OWNER_ROLE =
keccak256("OWNER_ROLE"); bytes32 public constant REDBELLY_ROLE =
keccak256("REDBELLY_ROLE"); bytes32 public constant IDP_ROLE =
keccak256("IDP_ROLE"); bytes32 public constant REDBELLY_NODE_OPERATOR_ROLE =
keccak256("REDBELLY_NODE_OPERATOR_ROLE"); bytes32 public constant
GOVERNOR_ROLE = keccak256("GOVERNOR_ROLE"); `` In
`ContractRoleAuthUpgradeable`: ``solidity bytes32 public constant
RECONFIGURATION_CONTRACT_ROLE =
keccak256("RECONFIGURATION_CONTRACT_ROLE"); bytes32 public constant
ACTIVITY_MONITOR_CONTRACT_ROLE =
keccak256("ACTIVITY_MONITOR_CONTRACT_ROLE"); bytes32 public constant
JAILED_GOVERNORS_CONTRACT_ROLE =
keccak256("JAILED_GOVERNORS_CONTRACT_ROLE"); bytes32 public constant
NETWORK_CONFIG_CONTRACT_ROLE =
keccak256("NETWORK_CONFIG_CONTRACT_ROLE"); `` Make sure the `initialize` function
does not initialize the role identifiers again since they're now `constant`.
```

## Status

Acknowledged

### [L-02] ActivityMonitorUpgradeable#setDaysToServe - Can only set up to 255 days to serve in jail

## Description

The `\_daysToServe` input parameter is `uint8` which means that the maximum number of days that can be set is 255.

## Recommendation

Change the `\_daysToServe` input parameter to `uint256`.

## Status

Acknowledged

### [L-03] JailedGovernorsUpgradeable#initialize - No input validation on `\_slashPrcnt`

## Description

The `\_slashPrcnt` input parameter is not validated to be between 0 and 100. The contract deployer can set it to a value more than 100%.

## Recommendation

```

Add input validation to `_slashPrct` to ensure it is between 0 and 100.
```solidity
function initialize( uint _daysToServe, uint8 _slashPrct, address
_bootstrapContractsRegistry ) public initializer {
ContractRoleAuthUpgradeable.initialize(); bootstrapContractsRegistry =
BootstrapContractsRegistry( _bootstrapContractsRegistry ); require(_slashPrct <= 100,
"Slash percentage cannot be more than 100") daysToServe = _daysToServe; slashPrct
= _slashPrct; address activityMonitor =
bootstrapContractsRegistry.getContractAddress( "activitymonitor" );
_grantRole(ACTIVITY_MONITOR_CONTRACT_ROLE, activityMonitor); }
```

```

**Status** Acknowledged

### [L-04] JailedGovernorsUpgradeable - `tombstonedContracts` and `stakingDepository` state variables are being incorrectly used

#### Description

The `tombstonedContracts` and `stakingDepository` state variables are being used as if they are `memory` variables. They are set each time either `jail` or `free` is called.
 

```

```solidity (tombstonedContract, stakingDepository) = getTombstonedAndStaking();
```

```

 Instead, they should be set once in the `initialize` function.

#### Recommendation

Set the `tombstonedContracts` and `stakingDepository` state variables in the `initialize` function and remove the lines in `jail` and `free` where they're set again. If these addresses are frequently changing, then the `tombstonedContract` and `stakingDepository` state variables aren't needed and the `jail` and `free` functions can call `BootstrapContractsRegistry.getContractAddress` can get their addresses each time they're required.

#### Status

Acknowledged

### [L-05] Contracts make use of the deprecated `Counters` library

**Description** Many of Redbelly's smart contracts make use of OpenZeppelin's `Counters` library. This library has been deprecated and removed from v5.0 of OpenZeppelin's contracts, and may cause issues in the future when the contract's dependencies are updated.

#### Recommendation

Instead of using the `Counters` library, make use of normal `uint256` variables as counters and increment/decrement them using native Solidity.

**Status** Acknowledged

## [L-06] RedbellyContractRegistry#register - Function does not emit event

**Description** The `register` function does not emit the `RegistryUpdated` event to indicate that a new contract has been registered.

### Recommendation

Emit the event at the end.

### Status

Acknowledged ### [L-07] TokenVestingUpgradeable#haveEnoughTokens - Contract needs to be overfunded to pass modifier check

### Description

The `haveEnoughTokens` contains the following `require` statement to check if the contract has enough tokens to allocate ```solidity require( totalAllocatedTokens + _amount <= address(this).balance, "Not enough tokens to allocate" ); ``` `totalAllocatedTokens` is the total amount of tokens that have been allocated throughout the life of the contract. Its value is non-decreasing. This does not take into account tokens that have already been vested, meaning that the contract needs to be overfunded for allocations to be made.

### Recommendation

Instead of using `totalAllocatedTokens`, keep track of the total number of tokens that are still vesting `totalVestingTokens`. This value decreases each time `claimTokens` is called. The new `require` statement becomes: ```solidity require( totalVestingTokens + _amount <= address(this).balance, "Not enough tokens to allocate" ); ```

### Status

Resolved

## [L-08] ActivityMonitorUpgradeable#jail - Governor can be jailed again even though they're currently jailed

### Description

There is no check to ensure that if a governor is currently jailed, then they can not be jailed again. This can lead to the governor being unfairly jailed again.

### Recommendation

Include a check inside `JailedGovernorsUpgradeable.jail` to make sure that a governor currently being jailed cannot be jailed again. ```diff function jail(address _governor) external onlyActivityMonitorContract { ... require( !tombstonedContract.isTombstoned(_governor), "node address is tombstoned" ); + require( + !isJailedGovernor(_governor), + "node address is still jailed" + ); ... } ```



## Status

Acknowledged ### [L-09] NetworkConfigurationUpgradeable#initialize - Potential incorrect candidate node configuration

## Description

The `initialize` function allows candidate node IDs to be arbitrarily set. They could potentially be accidentally set incorrectly which could lead to duplicate IDs or IDs that are out of range.

## Recommendation

Use `idCounter.current()` and `idCounter.increment()` for candidate node IDs.

## Status

Resolved

## [L-10] ActivityMonitorUpgradeable#endVoting - Delayed jailing due to access control

## Description

The `endVoting` function can only be called by the Redbelly team. This can result in significant delays and damage to the network, since jailing a malicious node requires input from the Redbelly team.

## Recommendation

Remove the `onlyRedbelly` modifier from `endVoting` and replace it with the `onlyGovernor` modifier. Since `endVoting` calls `jail` which also uses the `onlyRedbelly` modifier, the logic inside `jail` will need to be moved to a private function `\_jail` which does not contain the modifier. The `jail` public function can do an internal call to `\_jail`.

## Status

Unresolved

## Gas

## [G-01] StakingDepositUpgradeable#initialize - `stakeDuration` is set twice

## Description

The following line appears twice inside the `initialize` function. ``solidity stakeDuration = 30 days; ``

## Recommendation

Remove one of the lines.

**Status**

Acknowledged

**[G-02] NetworkConfigurationUpgradeable#removeRedbellyNodes - Loop continues after deleting node**

**Status**

Acknowledged



# Penetration Test Summary

## Goals

The primary goal of this audit was to identify and assess potential security risks within the Redbelly Network, a layer 1 blockchain written in Golang. These security risks could range from coding vulnerabilities to potential flaws within the network's underlying consensus protocol. Our comprehensive audit and infrastructure hardening aimed to ensure the Redbelly Network's integrity, confidentiality, and availability, as well as the robustness of its consensus mechanism.

## What We Looked For

During the audit, we focused our attention on the following key aspects:

**Code vulnerabilities and optimisations:** Our aim was to identify weak coding practices, lack of error handling, and other potential flaws in the programming.

**Blockchain-specific vulnerabilities:** Our focus was on smart contract vulnerabilities such as reentrancy attacks, underflows and overflows, gas limit issues, access control bugs and front running just to name a few.

**Consensus protocol vulnerabilities:** We looked for potential race conditions, Sybil attacks, and other possible flaws within the consensus mechanism that could lead to unauthorised changes within the Redbelly Network's blockchain.

## Tools We Used

For the code audit, we employed a combination of static and dynamic analysis tools, including:

**Gosec (Golang Security Checker):** This tool was used to inspect the Golang source code for security issues within the Redbelly Network.

**Golint:** for top level analysis and code formatting.

For the penetration testing, we utilised:

**Gosec;** Security auditing tool

**Ganache:** A personal sandboxed blockchain used for development testing purposes.

## Manual Analysis & Black Box Testing

In addition to the use of automated tools, we conducted manual analysis for a thorough audit. We reviewed the Redbelly Network's codebase for secure coding practices, error handling, data validation, and more. Furthermore, we examined the design and implementation of the consensus protocol for any potential flaws.

We also conducted thorough manual dependency auditing.

In our black box testing, we treated the Redbelly Network as an opaque box with zero code visibility until a bug was found, focusing on inputs and outputs without the need to understand the system's internal workings. This approach helped us understand how the network behaved under various input conditions, and if there were any unexpected behaviours that could indicate a security risk which includes assessing actions such as day to day smart contract deployment and edge cases.

### **Intrusion into Consensus Protocol**

The consensus protocol is one of the most critical aspects of a blockchain, ensuring all nodes agree on the state of the distributed ledger. A successful intrusion into the consensus protocol could allow an attacker to manipulate the state of the blockchain, leading to disastrous consequences such as a full network takeover allowing attackers to propose malicious blocks. Within the Redbelly Network, we scrutinised the consensus protocol against known vulnerabilities such as Sybil attacks, race conditions, and 51% attacks. Although our analysis did not reveal any immediate threats, it is vital to continually update and check this protocol against evolving threat landscapes.

Hashlock also recommends that the Redbelly protocol is mindful of how native tokens are distributed. A user with an overwhelming amount of native tokens usually has the authority to stake and control most of the network through malicious proposals of new blocks through a group of rogue nodes. It is best practice to spread native tokens amongst the user population as much as possible or create some incentive or activity to reward users with native tokens as opposed to making them readily available until the blockchain is fully established with a vast user base.

### **Conclusion**

The audit has provided invaluable insights into the potential security risks associated with the Redbelly Network, an SEVM blockchain written in Golang. While no major vulnerabilities were found, it's essential to continue regular audits and testing as new vulnerabilities and attack vectors can emerge over time. Upholding a commitment to security will help ensure the resilience and success of the Redbelly Network.

# Penetration Test Findings

## Penetration Test Overview

| Repo Name        | Repo Link                                                                                                             | Repo Description                                                 | Audit Branch | Last commit ID                            |
|------------------|-----------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------|--------------|-------------------------------------------|
| sevm             | <a href="https://github.com/redbellynetwork/sevm">https://github.com/redbellynetwork/sevm</a>                         | The SEVM                                                         | dev          | f4719da5ae0c0mebbfd088a706bd06fd71705369b |
| consensus        | <a href="https://github.com/redbellynetwork/consensus">https://github.com/redbellynetwork/consensus</a>               | The consensus system to verify transactions                      |              |                                           |
| consensus-driver | <a href="https://github.com/redbellynetwork/consensus-driver">https://github.com/redbellynetwork/consensus-driver</a> | The middleware to allow the consensus system to talk to the SEVM |              |                                           |
| diablo-benchmark | <a href="https://github.com/redbellynetwork/diablo-benchmark">https://github.com/redbellynetwork/diablo-benchmark</a> | A benchmarking tool used to test the above systems               |              |                                           |

## Methodology

The Penetration test is broken down into 4 main areas:

1. Static Analysis
2. Dynamic Analysis
3. Automated Analysis
4. Black box testing

## SEVM

## Static analysis;

| Item                                                              | Outcome         | Comment                                                                                                                                                                           | Threat Level |
|-------------------------------------------------------------------|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| GoLang Version                                                    | 1.18            | latest version 1.20                                                                                                                                                               | medium       |
| Number of GoLang modules 1 or more major versions behind          | 10              | Ideally modules should never be a major version behind                                                                                                                            | high         |
| Number of GoLang modules 1 or more minor versions behind          | 36              | Minor versions often aren't too critical, more nice to have                                                                                                                       | medium       |
| Number of GoLang modules 1 or more security patch versions behind | 13              | Patches are generally bug or security fixes, on need by need basis                                                                                                                | medium       |
| Number of GoLang latest releases older than 1 year                | 33              | Modules that have the latest release over a year old can be considered abandoned, which is a high security concern as no bug or security patches will be continued to be released | high         |
| Use of unsecure web protocol when requesting external resources   | File: client.sh | Try to always use secure protocols where possible                                                                                                                                 | medium       |
| Use of unsecure web protocol when requesting external resources   | File: Makefile  | Try to always use secure protocols where possible                                                                                                                                 | medium       |

## Module analysis

| Module                                      | Version  | Latest Version | Current Is Latest | Version Status                   | Last Release Date | Days since latest release |
|---------------------------------------------|----------|----------------|-------------------|----------------------------------|-------------------|---------------------------|
| github.com/ethereum/go-ethereum             | v1.10.18 | v1.12.0        | FALSE             | 2x Minor Version Behind          | 2023/05/25        | 47                        |
| github.com/go-kit/kit                       | v0.9.0   | v0.12.0        | FALSE             | 3x Minor Version Behind          | 2021/09/23        | 656                       |
| github.com/mitchellh/go-homedir             | v1.1.0   | v1.1.0         | TRUE              | Latest                           | 2019/01/27        | 1626                      |
| github.com/montanaflynn/stats               | v0.7.0   | v0.7.1         | FALSE             | 1x Security Patch Version Behind | 2023/05/11        | 61                        |
| github.com/prometheus/client_golang         | v1.11.1  | v1.15.1        | FALSE             | 4x Minor Version Behind          | 2023/05/03        | 69                        |
| github.com/rebillynet/work/consensus-driver | v0.17.0  | v0.17.0        | TRUE              | Latest                           | 2023/05/05        | 67                        |
| github.com/rebillynet/work/logger           | v1.0.0   | -              | FALSE             | -                                | -                 | N/A                       |
| github.com/spf13/cobra                      | v0.0.5   | v1.7.0         | FALSE             | 1x Major Version Behind          | 2023/04/04        | 98                        |

|                                              |                                    |         |       |                                  |            |      |
|----------------------------------------------|------------------------------------|---------|-------|----------------------------------|------------|------|
| github.com/<br>stretchr/testify              | v1.8.0                             | v1.8.4  | FALSE | 4x Security Patch Version Behind | 2023/05/30 | 42   |
| go.uber.org/<br>zap                          | v1.24.0                            | v1.24.0 | FALSE | Latest                           | 2022/12/30 | 193  |
| google.golang.org/<br>grpc                   | v1.48.0                            | v1.55.0 | FALSE | 7x Minor Version Behind          | 2023/05/04 | 68   |
| google.golang.org/<br>protobuf               | v1.28.0                            | v1.30.0 | FALSE | 2x Minor Version Behind          | 2023/03/16 | 117  |
| gopkg.in/urfave/<br>cli.v1                   | v1.20.0                            | v2.25.5 | FALSE | 1x Major Version Behind          | 2023/05/30 | 42   |
| gopkg.in/yaml.v2                             | v2.4.0                             | v3.0.1  | FALSE | 1x Major Version Behind          | 2022/05/27 | 410  |
| github.com/<br>StackExchange/<br>wmi         | v0.0.0-20180116203802-5d049714c4a6 | v1.1.0  | FALSE | 1x Major Version Behind          | 2019/12/28 | 1291 |
| github.com/<br>VictoriaMetrics/<br>fastcache | v1.6.0                             | v1.12.1 | FALSE | 6x Minor Version Behind          | 2023/02/22 | 139  |
| github.com/<br>VividCortex/<br>gohistogram   | v1.0.0                             | v1.0.0  | TRUE  | Latest                           | 2017/07/15 | 2187 |
| github.com/<br>beorn7/perks                  | v1.0.1                             | v1.0.1  | TRUE  | Latest                           | 2019/07/31 | 1441 |



|                                                       |                                                    |                                                |       |                                        |            |      |
|-------------------------------------------------------|----------------------------------------------------|------------------------------------------------|-------|----------------------------------------|------------|------|
| github.com/<br>btcsuite/btc<br>d/btcec/v2             | v2.2.0                                             | v0.23.3                                        | FALSE | 21x Minor<br>Version Behind            | 2022/11/01 | 252  |
| github.com/<br>cespare/xxh<br>ash/v2                  | v2.1.1                                             | v2.2.0                                         | FALSE | 1x Minor<br>Version Behind             | 2022/12/04 | 219  |
| github.com/<br>davecgh/go<br>-spew                    | v1.1.1                                             | v1.1.1                                         | TRUE  | Latest                                 | 2018/08/17 | 1789 |
| github.com/<br>deckarep/go<br>lang-set                | v1.8.0                                             | v2.3.0                                         | FALSE | 1x Major<br>Version Behind             | 2023/03/14 | 119  |
| github.com/<br>decred/dcrd<br>/dcrec/secp<br>256k1/v4 | v4.0.1                                             | release-v1.<br>7.7                             | FALSE | 7x Minor<br>Version Behind             | 2023/04/07 | 95   |
| github.com/<br>deepmap/o<br>api-codegen               | v1.8.2                                             | v1.13.0                                        | FALSE | 5x Minor<br>Version Behind             | 2023/06/02 | 39   |
| github.com/<br>edsrjf/mma<br>p-go                     | v1.0.0                                             | v1.1.0                                         | FALSE | 1x Minor<br>Version Behind             | 2021/12/17 | 571  |
| github.com/<br>fjl/memsize                            | v0.0.0-<br>2019071<br>0130421<br>-bcb579<br>9ab5e5 | v0.0.1                                         | FALSE | 1x Security<br>Patch Version<br>Behind | 2021/07/29 | 712  |
| github.com/<br>gballet/go-li<br>bpcsclite             | v0.0.0-<br>201906<br>070651<br>34-2772<br>fd86a8ff | v0.0.0-201<br>90607065<br>134-2772f<br>d86a8ff | TRUE  | Latest                                 | 2019/11/08 | 1341 |

|                                         |                              |         |       |                                        |            |      |
|-----------------------------------------|------------------------------|---------|-------|----------------------------------------|------------|------|
| github.com/<br>go-ole/go-ole            | v1.2.1                       | v1.2.1  | TRUE  | Latest                                 | 2015/11/05 | 2805 |
| github.com/<br>go-stack/stack           | v1.8.0                       | v1.8.1  | FALSE | 1x Security<br>Patch Version<br>Behind | 2021/08/18 | 692  |
| github.com/<br>golang-jwt/jwt/v4        | v4.3.0                       | v5.0.0  | FALSE | 1x Major<br>Version Behind             | 2023/04/17 | 85   |
| github.com/<br>golang/protobuf          | v1.5.2                       | v1.5.3  | FALSE | 1x Security<br>Patch Version<br>Behind | 2023/03/08 | 125  |
| github.com/<br>golang/snappy            | v0.0.4                       | v0.0.4  | TRUE  | Latest                                 | 2021/06/08 | 763  |
| github.com/<br>google/uuid              | v1.2.0                       | v1.3.0  | FALSE | 1x Minor<br>Version Behind             | 2021/07/12 | 729  |
| github.com/<br>gorilla/websocket        | v1.4.2                       | v1.5.0  | FALSE | 1x Minor<br>Version Behind             | 2022/02/15 | 511  |
| github.com/<br>graph-gophers/graphql-go | v1.3.0                       | v1.5.0  | FALSE | 2x Minor<br>Version Behind             | 2022/12/19 | 204  |
| github.com/<br>hashicorp/gobexpr        | v0.1.10                      | v0.1.12 | FALSE | 2x Security<br>Patch Version<br>Behind | 2023/04/26 | 76   |
| github.com/<br>hashicorp/golang-lru     | v0.5.5-0.20210104140557-80c9 | v2.0.3  | FALSE | 2x Major<br>Version Behind             | 2023/06/06 | 35   |

|                                             |                                    |         |       |                                  |            |      |
|---------------------------------------------|------------------------------------|---------|-------|----------------------------------|------------|------|
|                                             | 8217689d                           |         |       |                                  |            |      |
| github.com/holiman/blobfilter/v2            | v2.0.3                             | v2.0.3  | TRUE  | Latest                           | 2020/12/20 | 933  |
| github.com/holiman/uint256                  | v1.2.0                             | v1.2.2  | FALSE | 2x Security Patch Version Behind | 2023/03/22 | 111  |
| github.com/huin/goupnp                      | v1.0.3                             | v1.2.0  | FALSE | 2x Minor Version Behind          | 2023/05/10 | 62   |
| github.com/inconshreveable/mousetrap        | v1.0.0                             | v1.0.0  | TRUE  | Latest                           | 2017/07/29 | 2173 |
| github.com/influxdata/influxdb              | v1.8.3                             | v2.7.1  | FALSE | 1x Major Version Behind          | 2023/04/28 | 74   |
| github.com/influxdata/influxdb-client-go/v2 | v2.4.0                             | v2.12.3 | FALSE | 8x Minor Version Behind          | 2023/03/29 | 104  |
| github.com/influxdata/line-protocol         | v0.0.0-20210311194329-9aa0e372d097 | v2.2.1  | FALSE | 2x Major Version Behind          | 2021/11/24 | 594  |
| github.com/jackpal/golang-pmp               | v1.0.2                             | v1.0.2  | TRUE  | Latest                           | 2019/11/16 | 1333 |

|                                                                  |         |         |       |                                        |            |      |
|------------------------------------------------------------------|---------|---------|-------|----------------------------------------|------------|------|
| github.com/<br>klauspost/c<br>puid/v2                            | v2.0.9  | v2.2.5  | FALSE | 2x Minor<br>Version Behind             | 2023/06/02 | 39   |
| github.com/<br>mattn/go-c<br>olorable                            | v0.1.8  | v0.1.13 | FALSE | 5x Security<br>Patch Version<br>Behind | 2022/08/15 | 330  |
| github.com/<br>mattn/go-is<br>atty                               | v0.0.12 | v0.0.19 | FALSE | 7x Security<br>Patch Version<br>Behind | 2023/03/23 | 110  |
| github.com/<br>mattn/go-ru<br>newidth                            | v0.0.9  | v0.0.14 | FALSE | 5x Security<br>Patch Version<br>Behind | 2022/09/20 | 294  |
| github.com/<br>matttproud/<br>golang_prot<br>obuf_extens<br>ions | v1.0.1  | v2.0.0  | FALSE | 1x Major<br>Version Behind             | 2022/10/26 | 258  |
| github.com/<br>mitchellh/m<br>apstructure                        | v1.4.1  | v1.5.0  | FALSE | 1x Minor<br>Version Behind             | 2022/04/21 | 446  |
| github.com/<br>mitchellh/po<br>interstructur<br>e                | v1.2.0  | v1.2.1  | FALSE | 1x Security<br>Patch Version<br>Behind | 2021/11/03 | 615  |
| github.com/<br>olekukonko/<br>tablewriter                        | v0.0.5  | v0.0.5  | TRUE  | Latest                                 | 2021/02/11 | 880  |
| github.com/<br>opentracing<br>/opentracin<br>g-go                | v1.1.0  | v1.2.0  | FALSE | 1x Minor<br>Version Behind             | 2020/07/01 | 1105 |

|                                            |                                                          |         |       |                                        |            |      |
|--------------------------------------------|----------------------------------------------------------|---------|-------|----------------------------------------|------------|------|
| github.com/<br>peterh/liner                | v1.1.1-0.<br>2019012<br>317454<br>0-a2c9a<br>5303de<br>7 | v1.2.2  | FALSE | 1x Minor<br>Version Behind             | 2022/01/15 | 542  |
| github.com/<br>pkg/errors                  | v0.9.1                                                   | v0.9.1  | TRUE  | Latest                                 | 2020/01/14 | 1274 |
| github.com/<br>pmezard/go-<br>difflib      | v1.0.0                                                   | v1.0.0  | TRUE  | Latest                                 | 2016/08/08 | 2528 |
| github.com/<br>prometheus/<br>client_model | v0.2.0                                                   | v0.4.0  | FALSE | 2x Minor<br>Version Behind             | 2023/05/03 | 69   |
| github.com/<br>prometheus/<br>common       | v0.26.0                                                  | v0.44.0 | FALSE | 18x Minor<br>Version Behind            | 2023/05/22 | 50   |
| github.com/<br>prometheus/<br>procfs       | v0.6.0                                                   | v0.10.1 | FALSE | 4x Minor<br>Version Behind             | 2023/05/28 | 44   |
| github.com/<br>prometheus/<br>tsdb         | v0.7.1                                                   | v0.10.0 | FALSE | 3x Minor<br>Version Behind             | 2019/07/24 | 1448 |
| github.com/<br>rjeczalik/not<br>ify        | v0.9.1                                                   | v0.9.3  | FALSE | 2x Security<br>Patch Version<br>Behind | 2023/01/13 | 179  |
| github.com/<br>rs/cors                     | v1.7.0                                                   | v1.7.0  | TRUE  | Latest                                 | 2019/08/08 | 1433 |

|                                         |                                                                            |         |       |                                        |            |      |
|-----------------------------------------|----------------------------------------------------------------------------|---------|-------|----------------------------------------|------------|------|
| github.com/<br>shirou/gops<br>util      | v3.21.4-<br>0.20210<br>419000<br>835-c7a<br>38de76e<br>e5+inco<br>mpatible | v3.23.5 | FALSE | 2x Minor<br>Version Behind             | 2023/06/02 | 39   |
| github.com/<br>spf13/pflag              | v1.0.5                                                                     | v1.0.5  | TRUE  | Latest                                 | 2019/09/18 | 1392 |
| github.com/<br>status-im/k<br>eycard-go | v0.0.0-<br>2019031<br>609033<br>5-8537d<br>3370df4                         | v0.2.0  | FALSE | 2x Minor<br>Version Behind             | 2022/11/08 | 245  |
| github.com/<br>syndtr/golev<br>elldb    | v1.0.1-0<br>.202108<br>190228<br>25-2ae1<br>ddf74ef<br>7                   | v1.0.1  | FALSE | Latest                                 | 2019/02/22 | 1600 |
| github.com/<br>tklauser/go-<br>sysconf  | v0.3.5                                                                     | v0.3.11 | FALSE | 6x Security<br>Patch Version<br>Behind | 2022/11/09 | 244  |
| github.com/<br>tklauser/nu<br>mcpus     | v0.2.2                                                                     | v0.6.1  | FALSE | 4x Minor<br>Version Behind             | 2023/06/02 | 39   |
| github.com/<br>tyler-smith/<br>go-bip39 | v1.0.1-0<br>.201810<br>170606<br>43-dbb3<br>b84ba2e<br>f                   | v1.1.0  | FALSE | 1x Minor<br>Version Behind             | 2020/10/27 | 987  |
| go.uber.org/<br>atomic                  | v1.9.0                                                                     | v1.11.0 | FALSE | 2x Minor<br>Version Behind             | 2023/05/03 | 69   |

|                                     |                                                        |                                                |       |                            |            |      |
|-------------------------------------|--------------------------------------------------------|------------------------------------------------|-------|----------------------------|------------|------|
| go.uber.org/<br>multierr            | v1.7.0                                                 | v1.11.0                                        | FALSE | 4x Minor<br>Version Behind | 2023/03/29 | 104  |
| golang.org/<br>x/crypto             | v0.1.0                                                 | v0.9.0                                         | FALSE | 8x Minor<br>Version Behind | 2023/05/08 | 64   |
| golang.org/<br>x/net                | v0.7.0                                                 | v0.10.0                                        | FALSE | 3x Minor<br>Version Behind | 2023/05/04 | 68   |
| golang.org/<br>x/sync               | v0.0.0-<br>202207<br>2215525<br>5-886fb<br>9371eb4     | v0.2.0                                         | FALSE | 2x Minor<br>Version Behind | 2023/04/19 | 83   |
| golang.org/<br>x/sys                | v0.5.0                                                 | v0.8.0                                         | FALSE | 3x Minor<br>Version Behind | 2023/05/03 | 69   |
| golang.org/<br>x/text               | v0.7.0                                                 | v0.9.0                                         | FALSE | 2x Minor<br>Version Behind | 2023/04/04 | 98   |
| golang.org/<br>x/time               | v0.0.0-<br>202102<br>200331<br>41-f8bd<br>a1e9f3b<br>a | v0.3.0                                         | FALSE | 3x Minor<br>Version Behind | 2022/11/16 | 237  |
| google.gola<br>ng.org/genp<br>roto  | v0.0.0-<br>202005<br>2621185<br>5-cb27e<br>3aa2013     | v0.0.0-20<br>230530153<br>820-e85fd<br>2cbaebc | FALSE | Latest                     | 2023/05/30 | 42   |
| gopkg.in/nat<br>efinch/npipe<br>.v2 | v2.0.0-<br>201606<br>210349<br>01-c1b8<br>fa8bdcc<br>e | v2.0.0-201<br>60621034<br>901-c1b8fa<br>8bdcce | TRUE  | Latest                     | 2016/06/21 | 2576 |

|                         |        |        |       |                         |            |     |
|-------------------------|--------|--------|-------|-------------------------|------------|-----|
| gopkg.in/yaml.v3        | v3.0.1 | v3.0.1 | TRUE  | Latest                  | 2022/05/27 | 410 |
| lukechampine.com/blake3 | v1.1.7 | v1.2.1 | FALSE | 1x Minor Version Behind | 2023/05/15 | 57  |

### Summary;

|                                   |    |
|-----------------------------------|----|
| Major Versions Behind             | 10 |
| Minor Versions Behind             | 36 |
| Security Patch Versions Behind    | 13 |
|                                   |    |
| Latest Releases older than 1 year | 33 |

### SEVM Code Analysis;

| File                             | Line | Pos | Message                                                                         | Risk Level | Comment                       |
|----------------------------------|------|-----|---------------------------------------------------------------------------------|------------|-------------------------------|
| database/tracer.go               | 50   | 4   | t.Logger undefined (type EthDBTracer has no field or method Logger) (typecheck) | low        | Update type to include Logger |
| database/tracer.go               | 54   | 5   | t.Logger undefined (type EthDBTracer has no field or method Logger) (typecheck) | low        | Update type to include Logger |
| database/tracer.go               | 64   | 5   | t.Logger undefined (type EthDBTracer has no field or method Logger) (typecheck) | low        | Update type to include Logger |
| redbellyclient/redbellyclient.go | 19   | 36  | undefined: ethereum (typecheck)                                                 | low        | can be ignored                |



|                                              |    |    |                                                                                                        |     |                                                                                                                                                       |
|----------------------------------------------|----|----|--------------------------------------------------------------------------------------------------------|-----|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| redbelly<br>client/re<br>dbellycli<br>ent.go | 69 | 18 | undefined: ethereum<br>(typecheck)                                                                     | low | can be ignored                                                                                                                                        |
| fs/io.go                                     | 15 | 3  | use of `fmt.Println` forbidden<br>by pattern<br>`^(fmt\.Print( f ln) print prin<br>tln)\$` (forbidigo) | low | %w generally is used for<br>logging errors, if possible<br>update to using a dedicate<br>logging module                                               |
| network<br>/networ<br>k.go                   | 48 |    | File is not `gofumpt`-ed<br>(gofumpt)                                                                  | low | check the file formatting for<br>best practises (gofumpt)                                                                                             |
| go.mod                                       | 5  | 1  | replacement are not<br>allowed:<br>github.com/ethereum/go-et<br>hereum (gomoddirectives)               | low | generally not best practise<br>as it should only be used for<br>overriding unsecure<br>module versions, however<br>should be fine in this<br>instance |
| network<br>/networ<br>k.go                   | 33 |    | line is 568 characters (lll)                                                                           | low | if possible, move string into<br>its own file, however this<br>only affects code<br>readability                                                       |
| fs/dir.go                                    | 18 | 2  | assignments should only be<br>cuddled with other<br>assignments (wsl)                                  | low | styling code format, can be<br>ignored                                                                                                                |
| fs/dir.go                                    | 23 | 3  | return statements should not<br>be cuddled if block has more<br>than two lines (wsl)                   | low | styling code format, can be<br>ignored                                                                                                                |
| fs/dir.go                                    | 19 | 2  | only one cuddle assignment<br>allowed before if statement<br>(wsl)                                     | low | styling code format, can be<br>ignored                                                                                                                |
| fs/dir.go                                    | 25 | 2  | only one cuddle assignment<br>allowed before defer<br>statement (wsl)                                  | low | styling code format, can be<br>ignored                                                                                                                |
| fs/dir.go                                    | 42 | 2  | return statements should not<br>be cuddled if block has more<br>than two lines (wsl)                   | low | styling code format, can be<br>ignored                                                                                                                |
| network<br>/networ<br>k.go                   | 37 | 2  | missing cases in switch of<br>type network.Network:<br>network.OtherNetwork<br>(exhaustive)            | low | ideally all network modes<br>should have an explicit<br>case, however as the<br>default only logs an error, it                                        |

|                                           |    |    |                                                                                                                                       |        |                                                                                                                                                                        |
|-------------------------------------------|----|----|---------------------------------------------------------------------------------------------------------------------------------------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                           |    |    |                                                                                                                                       |        | should be safe to leave as is                                                                                                                                          |
| network/<br>network.go                    | 46 | 2  | missing cases in switch of type network.Network: network.OtherNetwork (exhaustive)                                                    | low    | ideally all network modes should have an explicit case, however as the default only logs an error, it should be safe to leave as is                                    |
| types/safetypes/<br>safemap_string_int.go | 13 | 10 | mu is missing in SafeMapStringInt (exhaustivestruct)                                                                                  | low    | false positive - ignore                                                                                                                                                |
| types/safetypes/<br>safemap_uint64.go     | 11 | 10 | mutex is missing in SafeUint64 (exhaustivestruct)                                                                                     | low    | false positive - ignore                                                                                                                                                |
| network/<br>network.go                    | 32 | 5  | Keystore is a global variable (gochecknoglobals)                                                                                      | medium | depending on the intended use of the variable, it is usually always best practice to limit the scope of any variable as much as possible, and prevent global variables |
| fs/dir.go                                 | 15 | 10 | err113: do not define dynamic errors, use wrapped static errors instead: "fmt.Errorf(\"invalid directory path %v\", name)" (goerr113) | low    | where possible, use static error message, can be ignored                                                                                                               |
| network/<br>network.go                    | 41 | 20 | err113: do not define dynamic errors, use wrapped static errors instead: "fmt.Errorf(\"invalid network selected %s\", n)" (goerr113)  | low    | where possible, use static error message, can be ignored                                                                                                               |
| network/<br>network.go                    | 55 | 31 | err113: do not define dynamic errors, use wrapped static errors instead: "fmt.Errorf(\"invalid network selected %s\", n)" (goerr113)  | low    | where possible, use static error message, can be ignored                                                                                                               |

|           |    |   |                                                                                                         |     |                                                                                                           |
|-----------|----|---|---------------------------------------------------------------------------------------------------------|-----|-----------------------------------------------------------------------------------------------------------|
| fs/dir.go | 38 | 2 | variable 'err' is only used in the if-statement (fs/dir.go:39:2); consider using short syntax (ifshort) | low | personal preference, having the variable defined on a new line is better for readability - can be ignored |
| fs/dir.go | 16 | 3 | return with no blank line before (nlreturn)                                                             | low | personal preference - can be ignored                                                                      |

### SEVM Security Analysis;

| File                           | Line | Message                                                                | Risk Level | Comment                                                                                                                                                                                                                                                                                       |                                                                                                         |
|--------------------------------|------|------------------------------------------------------------------------|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------|
| utils/utlils.go                | 157  | Use of weak random number generator (math/rand instead of crypto/rand) | High       | As the method "RandomIntegers" is used twice in the "blockdownloader/blockdownloader.go" package. It should be reviewed and determined if it is a security risk in the context it is being used for                                                                                           | <a href="https://redbelly.atlassian.net/browse/GRT-96">https://redbelly.atlassian.net/browse/GRT-96</a> |
| blockdownloader/grpc_client.go | 37   | TLS InsecureSkipVerify set true.                                       | High       | As the current implementation "NewGrpcClient" offers a bypass self-signed certificate check which only appears to be use for local development, this can be ignored. However ensure that this is not enabled when in production as it poses large security risk for man-in-the-middle attacks | <a href="https://redbelly.atlassian.net/browse/GRT-97">https://redbelly.atlassian.net/browse/GRT-97</a> |
| tracer/tracer.go               | 29   | Potential file inclusion via variable                                  | Medium     | Best practice is to use hardcoded file names where possible. However as this relates to a log file, it should                                                                                                                                                                                 | <a href="https://redbelly.atlassian.net/browse">https://redbelly.atlassian.net/browse</a>               |

|  |  |  |  |                                                                            |               |
|--|--|--|--|----------------------------------------------------------------------------|---------------|
|  |  |  |  | be fine to leave as is. If possible, add some type of file name validation | <u>GRT-98</u> |
|--|--|--|--|----------------------------------------------------------------------------|---------------|

## Consensus

### Static Analysis;

| Item                                                              | Outcome | Comment                                                                                                                                                                           | Threat Level |
|-------------------------------------------------------------------|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| GoLang Version                                                    | 1.19    | latest version 1.20                                                                                                                                                               | low          |
| Number of GoLang modules 1 or more major versions behind          | 0       | Ideally modules should never be a major version behind                                                                                                                            | low          |
| Number of GoLang modules 1 or more minor versions behind          | 2       | Minor versions often aren't too critical, more nice to have                                                                                                                       | low          |
| Number of GoLang modules 1 or more security patch versions behind | 1       | Patches are generally bug or security fixes, on need by need basis                                                                                                                | low          |
| Number of GoLang latest releases older than 1 year                | 3       | Modules that have the latest release over a year old can be considered abandoned, which is a high security concern as no bug or security patches will be continued to be released | high         |

### Module Analysis;

| Module      | Version | Latest Version | Current Is Latest | Version Status | Last Release Date | Days since latest release |
|-------------|---------|----------------|-------------------|----------------|-------------------|---------------------------|
| github.com/ | v0.18.0 | v0.18.0        | TRUE              | Latest         | 2023/06/13        | 29                        |

|                                           |         |         |       |                                        |            |      |
|-------------------------------------------|---------|---------|-------|----------------------------------------|------------|------|
| redbellynet<br>work/consensus-driver      |         |         |       |                                        |            |      |
| github.com/<br>redbellynet<br>work/logger | v1.0.0  | v1.0.0  | TRUE  | Latest                                 | 2023/04/12 | 91   |
| github.com/<br>stretchr/testify           | v1.8.0  | v1.8.4  | FALSE | 4x<br>Security Patch<br>Version Behind | 2023/05/30 | 43   |
| github.com/<br>davecgh/go-spew            | v1.1.1  | v1.1.1  | TRUE  | Latest                                 | 2016/08/08 | 2529 |
| github.com/<br>pmezard/go-difflib         | v1.0.0  | v1.0.0  | TRUE  | Latest                                 | 2018/08/17 | 1790 |
| go.uber.org/<br>atomic                    | v1.7.0  | v1.11.0 | FALSE | 4x<br>Minor<br>Version Behind          | 2023/05/03 | 70   |
| go.uber.org/<br>multierr                  | v1.6.0  | v1.11.0 | FALSE | 5x<br>Minor<br>Version Behind          | 2023/03/29 | 105  |
| go.uber.org/<br>zap                       | v1.24.0 | v1.24.0 | TRUE  | Latest                                 | 2022/11/30 | 224  |
| gopkg.in/yaml.v3                          | v3.0.1  | v3.0.1  | TRUE  | Latest                                 | 2022/05/27 | 411  |

**Summary;**

|                                   |   |
|-----------------------------------|---|
| Major Versions Behind             | 0 |
| Minor Versions Behind             | 2 |
| Security Patch Versions Behind    | 1 |
|                                   |   |
| Latest Releases older than 1 year | 3 |

**Consensus code analysis;**

| File                   | Line | Position | Message                                                                                                                                           | Risk Level | Comment                                                                     |
|------------------------|------|----------|---------------------------------------------------------------------------------------------------------------------------------------------------|------------|-----------------------------------------------------------------------------|
| configurations/pool.go | 20   | 23       | var-declaration: should omit type ConnPoolConfig from declaration of var DefaultPoolConfig; it will be inferred from the right-hand side (revive) | low        |                                                                             |
| parser/parser.go       | 12   | 10       | PeerNodeIds is missing in NodeConfig (exhaustivestruct)                                                                                           | low        | make sure to include all necessary struct values if required                |
| parser/parser.go       | 122  | 10       | err113: do not define dynamic errors, use wrapped static errors instead:<br>"fmt.Errorf(\"error: IP isn't provided in URL string\")" (goerr113)   | low        | where possible, use static error message, can be ignored                    |
| configurations/pool    | 21   | 45       | mnd: Magic number: 100, in <assign> detected (gomnd)                                                                                              | low        | can be ignored                                                              |
| parser/parser.go       | 81   | 41       | mnd: Magic number: 2, in <argument> detected (gomnd)                                                                                              | low        | can be ignored                                                              |
| parser/parser.go       | 17   | 16       | mnd: Magic number: 1888, in <assign> detected (gomnd)                                                                                             | low        | can be ignored                                                              |
| parser/parser.go       | 25   | 18       | mnd: Magic number: 30, in <assign> detected (gomnd)                                                                                               | low        | can be ignored                                                              |
| parser/parser.go       | 69   | 16       | error returned from external package is unwrapped: sig: func strconv.Atoi(s string) (int, error) (wrapcheck)                                      | low        | where possible try to wrap errors in fmt.Errorf() for better error handling |

### Consensus Security Analysis;

| File                 | Line | Position | Message                 | Risk Level | Comment                                   |
|----------------------|------|----------|-------------------------|------------|-------------------------------------------|
| networks/dbft_rpc.go | 35   |          | TLS MinVersion too low. | low        | False positive - minVersion not defined   |
| pool/pool_rpc.go     | 212  |          | Errors unhandled.       | medium     | Make sure to handle errors where possible |

|                  |     |  |                   |        |                                           |
|------------------|-----|--|-------------------|--------|-------------------------------------------|
| pool/pool_rpc.go | 174 |  | Errors unhandled. | medium | Make sure to handle errors where possible |
| pool/pool_rpc.go | 20  |  | Errors unhandled. | medium | Make sure to handle errors where possible |

## Consensus Driver

### Static analysis;

| Item                                                              | Outcome | Comment                                                                                                                                                                           | Threat Level |
|-------------------------------------------------------------------|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| GoLang Version                                                    | 1.19    | latest version 1.20                                                                                                                                                               | low          |
| Number of GoLang modules 1 or more major versions behind          | 0       | Ideally modules should never be a major version behind                                                                                                                            | low          |
| Number of GoLang modules 1 or more minor versions behind          | 0       | Minor versions often aren't too critical, more nice to have                                                                                                                       | low          |
| Number of GoLang modules 1 or more security patch versions behind | 0       | Patches are generally bug or security fixes, on need by need basis                                                                                                                | low          |
| Number of GoLang latest releases older than 1 year                | 0       | Modules that have the latest release over a year old can be considered abandoned, which is a high security concern as no bug or security patches will be continued to be released | low          |

### Consensus Driver Code analysis;

| File                   | Line | Position | Message                                             | Risk Level | Comment                                                               |
|------------------------|------|----------|-----------------------------------------------------|------------|-----------------------------------------------------------------------|
| consensus/consensus.go | 11   | 2        | protocolsMu is a global variable (gochecknoglobals) | Medium     | where possible limit the use of global variables. Ideally try to keep |

|                        |    |    |                                                                                                                                                            |        |                                                                                                    |
|------------------------|----|----|------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|----------------------------------------------------------------------------------------------------|
|                        |    |    |                                                                                                                                                            |        | variables to a narrow scope                                                                        |
| consensus/consensus.go | 12 | 2  | protocols is a global variable (gochecknoglobals)                                                                                                          | Medium | where possible limit the use of global variables. Ideally try to keep variables to a narrow scope  |
| consensus/consensus.go | 33 | 15 | do not define dynamic errors, use wrapped static errors instead:<br>"fmt.Errorf(\"consensus: unknown consensus %q (forgotten import?)\", name)" (goerr113) | low    | where possible, use static error message, can be ignored                                           |
| consensus/consensus.go | 46 | 10 | do not define dynamic errors, use wrapped static errors instead:<br>"fmt.Errorf(\"consensus: Protocol is not registered\")" (goerr113)                     | low    | where possible, use static error message, can be ignored                                           |
| consensus/consensus.go | 27 | 1  | Open returns interface (github.com/rebillynetwork/consensus-driver/consensus/protocol.Engine) (ireturn)                                                    | low    | where possible return interface types instead of concrete types. helps to promote low coupled code |

### Consensus Driver Security Analysis;

| File               | Line | Position | Message | Risk Level | Comment |
|--------------------|------|----------|---------|------------|---------|
| No issues reported |      |          |         |            |         |



## Conclusion

After Hashlocks analysis, the Redbelly Network project seems to have a sound and well tested code base after the resolution of our findings. Overall, most of the code is correctly ordered and follows industry best practices. The code is extremely well commented and has great supporting documentation.



# Our Methodology

Hashlock strives to maintain a transparent working process and to make our audits a collaborative effort. The objective of our security audits are to improve the quality of systems and upcoming projects we review and to aim for sufficient remediation to help protect users and project leaders. Below is the methodology we use in our security audit process.

## **Manual Code Review:**

In manually analysing all of the code, we seek to find any potential issues with code logic, error handling, protocol and header parsing, cryptographic errors, and random number generators. We also watch for areas where more defensive programming could reduce the risk of future mistakes and speed up future audits. Although our primary focus is on the in-scope code, we examine dependency code and behaviour when it is relevant to a particular line of investigation.

## **Vulnerability Analysis:**

Our methodologies include manual code analysis, user interface interaction, and whitebox penetration testing. We consider the project's website, specifications, and whitepaper (if available) to attain a high level understanding of what functionality the smart contract under review contains. We then communicate with the developers and founders to gain insight into their vision for the project. We install and deploy the relevant software, exploring the user interactions and roles. While we do this, we brainstorm threat models and attack surfaces. We read design documentation, review other audit results, search for similar projects, examine source code dependencies, skim open issue tickets, and generally investigate details other than the implementation.

**Documenting Results:**

We undergo a robust, transparent process for analysing potential security vulnerabilities and seeing them through to successful remediation. When a potential issue is discovered, we immediately create an issue entry for it in this document, even though we have not yet verified the feasibility and impact of the issue. This process is vast because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities. We generally follow a process of first documenting the suspicion with unresolved questions, then confirming the issue through code analysis, live experimentation, or automated tests. Code analysis is the most tentative, and we strive to provide test code, log captures, or screenshots demonstrating our confirmation. After this we analyse the feasibility of an attack in a live system.

**Suggested Solutions:**

We search for immediate mitigations that live deployments can take and finally we suggest the requirements for remediation engineering for future releases. The mitigation and remediation recommendations should be scrutinised by the developers and deployment engineers, and successful mitigation and remediation is an ongoing collaborative process after we deliver our report, and before the contracts details are made public.

# Disclaimers

## Hashlock's Disclaimer

Hashlock's team has analysed these smart contracts in accordance with the best industry practices at the date of this report, in relation to: cybersecurity vulnerabilities and issues in the smart contract source code, the details of which are disclosed in this report, (Source Code); the Source Code compilation, deployment and functionality (performing the intended functions).

Due to the fact that the total number of test cases are unlimited, the audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only. We also suggest conducting a bug bounty program to confirm the high level of security of this smart contract.

Hashlock is not responsible for the safety of any funds, and is not in any way liable for the security of the project.

## Technical Disclaimer

Smart contracts are deployed and executed on a blockchain platform. The platform, its programming language, and other software related to the smart contract can have their own vulnerabilities that can lead to attacks. Thus, the audit can't guarantee explicit security of the audited smart contracts.

## About Hashlock

Hashlock is an Australian based company aiming to help facilitate the successful widespread adoption of distributed ledger technology. Our key services all have a focus on security, as well as projects that focus on streamlined adoption in the business sector.

Hashlock is excited to continue to grow its partnerships with developers and other web3 oriented companies to collaborate on secure innovation, helping businesses and decentralised entities alike.

**Website:** [hashlock.com.au](https://hashlock.com.au)

**Contact:** [info@hashlock.com.au](mailto:info@hashlock.com.au)



**#Hashlock.**