CODEZEN

Banksy Chain

Date: 11/01/2024

Rev 1.0

Visit: https://codezen.tech

Summary

Disclaimer

This report is provided "as is" by Codezen SRLS. The findings, recommendations, and information contained in this report are based on the security review conducted at the time of assessment.

Codezen SRLS assumes no responsibility or liability for the accuracy, completeness, or effectiveness of the report or any actions taken based on its contents.

The report is intended for informational purposes only and should not be considered a guarantee of the absence of all security risks. The findings and recommendations presented in this report are specific to the reviewed system and may not apply universally.

Furthermore, Codezen SRLS disclaims any responsibility for any damages, losses, or negative consequences arising from the use or reliance upon this report, including but not limited to financial, reputational, or operational damages. It is the responsibility of the recipient of this report to independently evaluate and assess the findings, recommendations, and their potential impact on their specific environment.

By accepting and utilizing this report, the recipient acknowledges and agrees that Codezen SRLS shall not be held liable for any claims, damages, or losses arising from the use or reliance upon this report.

While we will conduct solo audits with professional diligence and following industry standards, we recommend you consider engaging additional auditors or auditing firms for further code reviews to minimize risks and obtain a broader range of perspectives.

Scope

Codezen SRLS has been engaged to perform an 8 days timeframe code review starting on 06/05/2023 of the following scope:

- Repository: https://github.com/notional-labs/composable-centauri
- Commit Hash: `cd98916087ba1b327b28c4e83db7752e35ec804a`

```
├─ app
└─ x
   ├─ mint
   └─ transfermiddleware
```

Methodology

The security review was conducted using a combination of manual code review, automated vulnerability scanning, and fuzzing techniques.

The assessment followed industry best practices, including the use of standard security frameworks and guidelines.

This report represents a summary of the findings.

As with any code audit, there is a limit to which vulnerabilities can be found, and unexpected execution paths may still be possible. The authors of this report do not guarantee complete coverage (see disclaimer).

The following team has been involved in the review:

Christian Vari	**Security Researcher**
Maria Iacobelli	**Technical Writer**

Severity Levels

Severity Level	Description
HIGH	Indicates serious and exploitable security issues that pose a significant risk. Broken main features are also reported with this severity level.
MEDIUM	Indicates security weaknesses or bugs that should be addressed to mitigate potential risks.
LOW	Indicates security concerns or bugs that may not have an immediate impact but should be resolved. It is also used to classify vulnerabilities that can only be exploited by the governance or accounts with administrator permissions.
INFO	Indicates informational findings and recommendations for best practices and future improvements.

Findings

1. [HIGH] The `mint` module address is blacklisted and it is not possible to replenish incentives

Location	`x/mint/abci.go` `x/mint/types/genesis.go`
Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/169

The `mint` module is specifically designed to mint a predetermined `IncentivesSupply` during the `InitGenesis` phase and subsequently distribute these minted coins to validators via the `BeginBlocker` function.

However, the current design raises some concerns. Firstly, the `IncentivesSupply` is fixed and cannot be replenished because the `mint` module account is blacklisted in the `Bank` module. Consequently, once the incentives are depleted, it will not be possible to replenish them.

Moreover, the default `IncentiveSupply` token amount is set at `100,000,000,000`, which may not be sufficient in comparison to the `MinTokenPerYear` value of `800,000,000,000,000`. This discrepancy suggests that there may not be enough tokens available for the first year of incentives.

Consequently, no additional incentives will be available for validators to continue running the chain, apart from transaction fees, once the provided coins are exhausted. This situation could lead to a decline in validator participation and compromise the network's overall stability.

Recommendation

It is recommended to consider the following actions:

- Evaluate the feasibility of allowing the mint module account to be replenished, either by removing its blacklist status in the `Bank` module or by implementing an alternative mechanism.
- Review the `IncentiveSupply` token amount and ensure that it adequately fulfills the `MinTokenPerYear` requirement. Adjustments may be necessary to ensure a sufficient supply of tokens to meet the demands over the course of the year.
- Explore alternative options to supply ongoing incentives even after the initially provided coins are depleted. This approach will incentivize validators to continue their active participation in the chain, safeguarding its stability.

2. [HIGH] Panics in `PrepareProposal` when sending a transaction

Location	`app/app.go`
Status	Resolved in https://github.com/notional-labs/composable-centauri/commit/7c3b4390aeba942430c86f8f65631184df635c26

The execution panics after submitting transactions because of an "invalid memory address or nil pointer dereference" error.

This error is directly associated with the `PrepareProposal` handler, indicating a potential problem with the handling of transaction data.

The issue stems from incorrectly encoded/decoded transaction data, leading to the occurrence of an "invalid memory address or nil pointer dereference" error.

The error log is provided in the Appendix.

Recommendation

It is recommended to verify the codec used in the transaction for decoding/encoding and enforcing a supported one.

3. [HIGH] The `mint` module panics if there are no staked coins

Location	`x/mint/types/minter.go:44-65`
Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/170

The division operations in lines 63 and 64 within the `NextInflationRate` function involve dividing `MaxTokenPerYear` and `MinTokenPerYear` by `totalStakingSupplyDec`, respectively.

However, it is essential to be aware that this division operation could potentially result in a panic situation, particularly when the initial staking supply is zero. In such cases, a division by zero error may occur, leading to an unexpected and problematic situation in the code execution.

A test case showcasing the issue is provided in the Appendix.

Recommendation

It is recommended to implement a conditional check before performing the division operation to ensure that the `totalStakingSupplyDec` is not zero, preventing any potential division by zero errors.

4. [HIGH] Parachains do not use the Cosmos denom convention leading to the impossibility of the transfer middleware to recognize coins

Location	`x/transfermiddleware`
Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/159

The `transfermiddleware` module relies on the Cosmos denom convention to identify specific coins defined with `AddParachainIBCTokenInfo`.

However, the parachain `Currency` struct does not follow the Cosmos convention and defines a `u128 AssetID` as specified here.

Consequently, the middleware will not be able to recognize the coins defined in `AddParachainIBCTokenInfo` and it will execute the specific logic to mint native tokens.

Recommendation

It is recommended to rework the `AddParachainIBCTokenInfo` function to accept an `AssetID` instead of the `IBCDenom` to identify coins from parachains.

5. [MEDIUM] `RemoveParachainIBCTokenInfo` could permanently block IBC tokens in the escrow address

Location	`x/transfermiddleware/keeper/keeper.go:46`
Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/201

Upon executing the `RemoveParachainIBCTokenInfo` function, the information facilitating the redemption of IBC coins with native ones will be eliminated.

Consequently, all the IBC coins in the escrow address would not be redeemable, as the backing for native tokens will no longer be available.

Recommendation

It is recommended to enable users to return their tokens to the origin chain, even after executing `RemoveParachainIBCTokenInfo`.

6. [MEDIUM] It is not possible to export the app genesis for zero height

Location	`app/export.go:176-178`
Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/162

In the specified code location, there is an issue with the iterator generation for the validator reversed prefix, which is intended to enable iteration through validators in descending order. However, the retrieved key is not being utilized correctly.

The problem arises in the subsequent handling of the key. During execution, the first byte is removed from the key, and an attempt is made to cast it to `ValAddress`.

Unfortunately, the key points to the `Validator` struct, rather than directly representing the address. As a result, the cast operation generates an invalid address.

As a consequence, when attempting to retrieve the validator using `GetValidator`, the execution encounters an error and terminates, rendering it impossible to export the app genesis.

Recommendation

It is recommended to substitute line 177 with the following code:

```
addr := sdk.ValAddress(stakingtypes.AddressFromValidatorsKey(iter.Key()))
```

7. [MEDIUM] It is not possible to export the app genesis for zero height if there is a validator with zero commission to withdraw

Location	`app/export.go:77-85`
Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/162

In the indicated code location, the current implementation iterates through all validators to initiate the withdrawal of their commissions.

The problem arises from the fact that the `WithdrawValidatorCommission` function returns an error when there are no commissions available to withdraw.

Several potential scenarios need to be taken into consideration. Firstly, there could be validators with zero commissions available, resulting in no actual withdrawal. Additionally, an attacker could register a validator within the staking module, intentionally introducing a zero commission element to the list.

As a consequence, in line 81, an execution panic occurs when encountering the `WithdrawValidatorCommission` error. This situation presents a significant challenge as it prevents the successful exportation of the app genesis.

Recommendation

It is recommended to skip validators with no available commissions to withdraw.

8. [LOW] Missing `transfermiddleware` module genesis validation

Location	`x/transfermiddleware/types/genesis.go:10-12`
Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/162

The `ValidateGenesis` function in its current implementation lacks validation for the `GenesisState`, as it consistently returns nil.

To solve this, it is crucial to introduce validation checks, specifically for the `TokenInfos` parameter, to prevent the presence of duplicate entries.

Additionally, it is essential to perform thorough validation for each element within `TokenInfos` to ensure their integrity and conformity.

Recommendation

It is recommended to implement the `ValidateGenesis` function.

9. [LOW] The `transfermiddleware` module does not export the genesis

Location	`x/transfermiddleware/types/genesis.go:16-19`
Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/162

The `ExportGenesis` function fails to export the genesis with the actual data, as it consistently returns an empty `GenesisState` struct.

Specifically, it should retrieve the `TokenInfos` slice and incorporate it into the exported genesis state. This ensures that the function accurately reflects the current state by returning the relevant data along with the exported genesis.

Recommendation

It is recommended to populate the `GenesisState` struct before exporting it.

10. [LOW] Missing `MsgAddParachainIBCTokenInfo` message validation

Location	`x/transfermiddleware/types/msg.go:23-35`
Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/163

The `ValidateBasic` handler of `MsgAddParachainIBCTokenInfo` message lacks validation to ensure that the provided `IbcDenom` is a valid IBC trace denom.

Recommendation

It is recommended to ensure the integrity and reliability of the system by enhancing the `ValidateBasic` function to incorporate a validation step for the `IbcDenom` parameter. This validation should verify that the provided value conforms to the required format of a valid IBC trace denom.

11. [LOW] It is possible to map an `ibcDenom` to multiple `nativeDenom` and overwrite the legit one

Location	`x/transfermiddleware/keeper/keeper.go:46-67`
Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/164

The current implementation of the `MsgAddParachainIBCTokenInfo` message permits the mapping of an `ibcDenom` to multiple `nativeDenom` values.

However, this behavior leads to the overwriting of the `IBCDenomAndNativeIndex` with the new `nativeDenom`, potentially resulting in undesired consequences.

Recommendation

It is recommended to return an error whenever an attempt is made to map an `ibcDenom` to multiple `nativeDenom` values to maintain data consistency and prevent unexpected behavior.

12. [INFO] Missing simulation features in the `mint` module

Location	`x/mint/module.go`
Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/174

The Cosmos SDK provides a comprehensive simulation framework that enables thorough fuzz-testing of every message defined by a module.

This blockchain simulator evaluates the behavior of blockchain applications, simulating real-life circumstances by generating and sending randomized messages.

However, the mint module does not implement simulation features.

Recommendation

It is recommended to implement simulation features in the mint module.

13. [INFO] `MsgAddParachainIBCTokenInfo` and `MsgRemoveParachainIBCTokenInfo` message execution do not emit events

Location	`x/transfermiddleware/keeper/msg_server.go:27-39` `x/transfermiddleware/keeper/msg_server.go:41-53`
Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/165

The current implementation of the `MsgAddParachainIBCTokenInfo` and `MsgRemoveParachainIBCTokenInfo` message handlers lack event emission, which hinders external systems from effectively monitoring and responding to changes and actions related to the addition or removal of `ParachainIBCTokenInfo`.

Recommendation

It is recommended to enhance these message handlers by emitting relevant events.

14. [INFO] Magic numbers

Location	`x/mint/types/minter.go:31`
	`x/mint/types/params.go:45-54`
	`x/mint/types/genesis.go:35`

Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/161

Within the codebase, it is important to note the presence of hard-coded number literals that lack appropriate context or descriptions.

It adversely affects code readability and maintainability, as it becomes challenging for developers to understand the purpose and significance of these numbers.

Recommendation

It is recommended to define constants for number literals.

15. [INFO] Misleading error message in `RemoveParachainIBCInfo`

Location	`x/transfermiddleware/keeper/keeper.go:72-73`
Status	Resolved in https://github.com/notional-labs/composable-centauri/pull/161

The `RemoveParachainIBCInfo` function in line 72 validates whether the requested deletion of the `nativeDenom` corresponds to an existing `ParachainIBCTokenInfo`. If the `nativeDenom` is not registered as a `ParachainIBCTokenInfo`, an error is returned.

However, the error message returned in this specific scenario is misleading and does not accurately describe the underlying issue. The current error message states "duplicate ParachainIBC Token Info", which is incorrect and may cause confusion for developers attempting to diagnose the problem.

Recommendation

It is recommended to revise the error message to accurately reflect the situation and provide developers with more accurate information about the root cause of the issue, returning a meaningful error message.

Appendix

1. **Test case for "The mint module panics if there are no staked coins"**

```
func TestSimulateMint(t *testing.T) {
	minter := DefaultInitialMinter()
	params := DefaultParams()
	totalSupply := sdk.NewInt(1_000_000_000_000_000_000_000)
	totalStaked := sdk.NewInt(0)
	tokenMinted := sdk.NewCoin("stake", sdk.NewInt(0))

	for i := 1; i <= int(params.BlocksPerYear)*2; i++ {

		stakingDiff := sdk.NewDec(int64(rand.Intn(10))).QuoInt(sdk.NewInt(1_000_000)).MulInt(totalSupply)
		if (rand.Float32() > 0.5 || totalStaked.Add(stakingDiff.RoundInt()).GT(totalSupply)) && !totalStaked.Sub(stakingDiff.RoundInt()).IsNegative() {
			stakingDiff = stakingDiff.Neg()
		}
		totalStaked = totalStaked.Add(stakingDiff.RoundInt())
		bondedRatio := sdk.NewDecFromInt(totalStaked).Quo(sdk.NewDecFromInt(totalSupply))
		minter.Inflation = minter.NextInflationRate(params, bondedRatio, totalStaked)
		minter.AnnualProvisions = minter.NextAnnualProvisions(params, totalStaked)

		// mint coins, update supply
		mintedCoin := minter.BlockProvision(params)
		tokenMinted = tokenMinted.Add(mintedCoin)
		// if i%100000 == 0 {
		//    fmt.Println(i, bondedRatio, tokenMinted, mintedCoin, minter.Inflation, minter.AnnualProvisions)
		// }
	}
	require.True(t, params.MaxTokenPerYear.MulRaw(2).GTE(tokenMinted.Amount))
	require.True(t, params.MinTokenPerYear.MulRaw(2).LTE(tokenMinted.Amount))
}
```

2. **Panic log for "Panics in PrepareProposal when sending a transaction"**

```
module=server height=106 time="2023-06-14 10:15:50.376337976 +0000 UTC"
panic="runtime error: invalid memory address or nil pointer dereference"
```

About

Codezen SRLS is a cybersecurity company focused on blockchain and fintech technologies.

Established with a vision to create a safer digital environment, Codezen offers consultancy services, security audits, and code reviews.

Thank you for choosing our company and helping make the blockchain ecosystem a safer place.