Audit Report

Dymension Phase 1 Stream 2: Virtual Frontier Contract

v1.0

April 30, 2024

Table of Contents

License

Disclaimer

THE CONTENT OF THIS AUDIT REPORT IS PROVIDED “AS IS”, WITHOUT REPRESENTATIONS AND WARRANTIES OF ANY KIND.

THE AUTHOR AND HIS EMPLOYER DISCLAIM ANY LIABILITY FOR DAMAGE ARISING OUT OF, OR IN CONNECTION WITH, THIS AUDIT REPORT.

THIS AUDIT REPORT WAS PREPARED EXCLUSIVELY FOR AND IN THE INTEREST OF THE CLIENT AND SHALL NOT CONSTRUE ANY LEGAL RELATIONSHIP TOWARDS THIRD PARTIES. IN PARTICULAR, THE AUTHOR AND HIS EMPLOYER UNDERTAKE NO LIABILITY OR RESPONSIBILITY TOWARDS THIRD PARTIES AND PROVIDE NO WARRANTIES REGARDING THE FACTUAL ACCURACY OR COMPLETENESS OF THE AUDIT REPORT.

FOR THE AVOIDANCE OF DOUBT, NOTHING CONTAINED IN THIS AUDIT REPORT SHALL BE CONSTRUED TO IMPOSE ADDITIONAL OBLIGATIONS ON COMPANY, INCLUDING WITHOUT LIMITATION WARRANTIES OR LIABILITIES.

This audit has been performed by

Oak Security GmbH

https://oaksecurity.io/
info@oaksecurity.io

Introduction

Purpose of This Report

Oak Security has been engaged by Dymension Technologies Ltd to perform a security audit of the Virtual Frontier Contract.

The objectives of the audit are as follows:

1. Determine the correct functioning of the protocol, in accordance with the project specification.
2. Determine possible vulnerabilities, which could be exploited by an attacker.
3. Determine smart contract bugs, which might lead to unexpected behavior.
4. Analyze whether best practices have been applied during development.
5. Make recommendations to improve code safety and readability.

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 author of this report does not guarantee complete coverage (see disclaimer).

Codebase Submitted for the Audit

The audit has been performed on the following targets:

Repository	https://github.com/dymensionxyz/ethermint
Commit	`8cdb58f3d43b0504cad3ef8b9fba059f7d10f67c`
Scope	The changes introduced in pull request #11 into commit `9940df7eaafad2ddddcb704366329b34d4970b16` were in scope.
Identifier	In this report, all paths pointing to this repository are prefixed with `ethermint:`

Repository	https://github.com/dymensionxyz/dymension
Commit	`edd0adb3d7119550c0bbed7e3e6020878bddb7be`

Scope	The changes introduced by pull request #668 into commit `927bdd7b72dfdd09563d02adacee7266ed2a1373` were in scope.
Identifier	In this report, all paths pointing to this repository are prefixed with `dymension:`

Methodology

The audit has been performed in the following steps:

1. Gaining an understanding of the code base's intended purpose by reading the available documentation.
2. Automated source code and dependency analysis.
3. Manual line-by-line analysis of the source code for security vulnerabilities and use of best practice guidelines, including but not limited to:
    a. Race condition analysis
    b. Under-/overflow issues
    c. Key management vulnerabilities
4. Report preparation

Functionality Overview

Dymension's Virtual Frontier Contract (VFC) is a special type of virtual smart contract that stands in front of the EVM and can be directly accessed through Ethereum wallets such as Metamask, enabling the execution of Cosmos SDK business logic. In particular, the Virtual Frontier Bank Contract mimics an ERC-20 token contract, allowing users to import and transfer assets from the Cosmos bank module using Ethereum wallets.

How to Read This Report

This report classifies the issues found into the following severity categories:

Severity	Description
Critical	A serious and exploitable vulnerability that can lead to loss of funds, unrecoverable locked funds, or catastrophic denial of service.
Major	A vulnerability or bug that can affect the correct functioning of the system, lead to incorrect states or denial of service.
Minor	A violation of common best practices or incorrect usage of primitives, which may not currently have a major impact on security, but may do so in the future or introduce inefficiencies.
Informational	Comments and recommendations of design decisions or potential optimizations, that are not relevant to security. Their application may improve aspects, such as user experience or readability, but is not strictly necessary. This category may also include opinionated recommendations that the project team might not share.

The status of an issue can be one of the following: **Pending, Acknowledged**, or **Resolved**.

Note that audits are an important step to improving the security of smart contracts and can find many issues. However, auditing complex codebases has its limits and a remaining risk is present (see disclaimer).

Users of the system should exercise caution. In order to help with the evaluation of the remaining risk, we provide a measure of the following key indicators: **code complexity**, **code readability**, **level of documentation**, and **test coverage**. We include a table with these criteria below.

Note that high complexity or low test coverage does not necessarily equate to a higher risk, although certain bugs are more easily detected in unit testing than in a security audit and vice versa.

Code Quality Criteria

The auditor team assesses the codebase's code quality criteria as follows:

Criteria	Status	Comment
Code complexity	**Low-Medium**	-
Code readability and clarity	**High**	-
Level of documentation	**High**	-
Test coverage	**Medium-High**	`go test` reports test coverage for the `x/evm` module in the `ethermint` repository of 80%. `go-acc` reports test coverage for the `dymension` repository of 63%.

Summary of Findings

No	Description	Severity	Status
1	Missing restriction on direct deposits to `VirtualFrontierBankContract` contracts	Minor	Acknowledged
2	Attackers can DoS the chain by sending multiple IBC coins	Minor	Acknowledged
3	Long-term `DenomMetaData` accumulation can DoS the chain	Minor	Acknowledged
4	Faulty `DenomMetadata` halts `VirtualFrontierContracts` deployment	Minor	Acknowledged

Detailed Findings

1. Missing restriction on direct deposits to `VirtualFrontierBankContract` contracts

Severity: Minor

In `ethermint:app/ante/eth.go:392-433`, the `NewVirtualFrontierContractDecorator` `AnteHandler` decorator denies users from depositing coins into `VirtualFrontierBankContract` contracts by rejecting Ethereum transactions with a non-zero value.

However, it is still possible to send funds to `VirtualFrontierBankContract` contracts through the `Bank` module.

As a consequence, while direct deposits are prevented, alternative methods for transferring funds remain accessible.

Recommendation

We recommend either blacklisting `VirtualFrontierBankContract` addresses in the `Bank` module or introducing a verification process in the `AnteHandler`.

Status: Acknowledged

2. Attackers can DoS the chain by sending multiple IBC coins

Severity: Minor

In `ethermint:x/evm/keeper/abci.go:30-34`, if the current chain id is prefixed with `ethermint_`, i.e., denoting that the chain is some sort of testnet, the `BeginBlocker` executes the `DeployVirtualFrontierBankContractForAllBankDenomMetadataRecords` function.

This function iterates through all the `DenomMetaData` retrieved from the Bank module, identifying those marked with `/ibc` denoms, and deploying a new `VirtualFrontierBankContract` for each of them.

As a consequence, attackers could exploit this process by initiating a large volume of ICS-20 transfers, causing the chain to store a substantial amount of `DenomMetaData` items during packet processing in ibc-go.

This could lead to a slowdown of nodes, consensus timeouts, and, potentially, halts of the chain.

We classify this issue as minor because this feature is only activated on the Ethermint dev chain.

Recommendation

We recommend handling the deployment of contracts in batches across multiple blocks.

Status: Acknowledged

3. Long-term DenomMetaData accumulation can DoS the chain

Severity: Minor

In ethermint:x/evm/keeper/virtual_frontier_contract.go:128-158, the DeployVirtualFrontierBankContractForAllBankDenomMetadataRecords function is called in the BeginBlocker, iterating through all DenomMetaData in the Bank module using IterateAllDenomMetaData.

However, as the quantity of DenomMetaData grows over time, this process increasingly slows down the blockchain, potentially leading to consensus timeouts and chain halts in the long run.

We classify this issue as minor because this feature is only activated on the Ethermint dev chain.

Recommendation

We recommend implementing a differential queue for tracking newly created DenomMetaData, rather than iterating through the entire set.

Status: Acknowledged

4. Faulty DenomMetadata halts VirtualFrontierContracts deployment

Severity: Minor

In ethermint:x/evm/keeper/virtual_frontier_contract.go:177-192, the function DeployVirtualFrontierBankContractForAllBankDenomMetadataRecords iterates through all the new DenomMetadata and deploys a VirtualFrontierContract for each one using the DeployNewVirtualFrontierBankContract function.

These actions are first applied to a new temporary context, cachedCtx, before the state changes are committed all at once in the defer function.

However, if a single deployment errors, all subsequent deployments will not be committed in the context.

As a result, the presence of faulty `DenomMetadata` can halt the entire module from further deploying any `VirtualFrontierContract` contracts.

We classify this issue as minor because this feature is only activated on the Ethermint dev chain.

Recommendation

We recommend implementing an error-handling mechanism within `DeployVirtualFrontierBankContractForAllBankDenomMetadataRecords` to ensure that the failure of deploying a single `VirtualFrontierContract` does not prevent the commitment of other deployments.

Status: Acknowledged