Audit Report

XION and Account Abstraction Updates 2

v1.0

May 25, 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 Thames Brook Associates, LLC to perform a security audit of XION and Account Abstraction Updates 2.

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 target:

Repository	https://github.com/burnt-labs/xion
Commit	`a9bb0f5099f68474323ed8830c69b00130f8d91a`
Scope	Only changes since our last audit, which was performed on commit `3120374e88dbed49061c4794fe32cbfdb4252ab5`, were in scope.

Repository	https://github.com/burnt-labs/contracts
Commit	`91329a51b99d0322762266cc033112ee9f3e0471`
Scope	Only changes since our last audit, which was performed on commit `574395b76cac43ed13386eb180250a8de50a9f2f`, were in scope.

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

The audit focuses on functionality tied to the Cosmos SDK app chain XION, including its account contract with the addition of features to support account abstraction through multiple methods such as JWT, ETH wallets, Passkeys, and other additional methods. This codebase has been previously audited by Oak Security so the scope of this audit covers the changes implemented since the commit hashes mentioned above.

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	Medium	The code in scope of this audit showed a relatively high complexity due to the use of alternative authentication methods such as JWT.
Code readability and clarity	Medium	-
Level of documentation	Medium	In general, the XION documentation website was extensive, but no specific information pertaining to the new features was provided.
Test coverage	Medium	Overall the project had a medium amount of test coverage with good unit testing, but we were unable to generate test coverage metrics due to a failing test case.

Summary of Findings

No	Description	Severity	Status
1	Front-running audience creation opens phishing and other risks	Major	Resolved
2	Exposing shared keys during audience creation with HS256 Signing	Minor	Resolved
3	Genesis does not validate audience admin addresses	Minor	Resolved
4	Update audience command does not support updating the admin	Minor	Resolved
5	Incomplete validation in update audience function	Minor	Resolved
6	Expired tokens cannot be removed	Minor	Resolved
7	Incomplete NewRegisterCmd implementation for JWK authenticator	Minor	Resolved
8	Overflow checks not enabled for release profile	Minor	Resolved
9	Comitted test private keys	Informational	Acknowledged
10	Messages with empty responses	Informational	Resolved
11	Update empty query response	Informational	Resolved
12	No upper cap on `timeoffset` parameter for JWK	Informational	Acknowledged
13	Missing maximum pagination may lead to RPC server exhaustion	Informational	Resolved

Detailed Findings

1. Front-running audience creation opens phishing and other risks

Severity: Major

The `aud` field, typically representing the URL of the resource server, is a unique index in the system. If an attacker front-runs a legitimate audience creation transaction at `x/jwk/keeper/msg_server_audience.go:19-25`, they can effectively block the legitimate user from creating their intended audience. This can lead to several issues:

- Resource Allocation Disruption: The legitimate user may need to reset their server resource allocator or make other adjustments to their system to work around the blocked audience.
- Phishing Risks: an attacker could use the front-run audience to set up a phishing site. Since the audience field is typically a URL, users might be misled into interacting with the attacker's site.

Recommendation

We recommend using a commit reveal scheme for audience creation.

Status: Resolved

2. Exposing shared keys during audience creation with HS256 signing

Severity: Minor,

When creating an audience at `x/jwk/keeper/msg_server_audience.go:15-40`, it is possible to use shared/symmetric keys (HS256) for signing and verifying tokens. However, this approach would result in exposing the shared keys as the key is stored on a chain.

Recommendation

We recommend validating the `key.Algorithm` to be an asymmetric algorithm before storing the audience.

Status: Resolved

3. Genesis does not validate audience admin addresses

Severity: Minor

The `Validate` function in `x/jwk/types/genesis.go:21` does not properly validate the audience admin addresses. This may create a situation where an invalid admin is added, resulting in the inability to administrate the audience in the future.

Recommendation

We recommend adding address validation in `x/jwk/types/genesis.go:21`.

Status: Resolved

4. Update audience command does not support updating the admin

Severity: Minor

The `CmdUpdateAudience` function within `x/jwk/client/cli/tx_audience.go` allows updating the administrator and key associated with a specific audience index. While it accepts arguments for updating the key, it lacks a parameter for specifying a new administrator through the CLI command. Consequently, the update process assigns the current administrator (obtained from the source address) as the new administrator. This restricts the ability to change the administrator via the CLI command, which poses a significant security risk, particularly in scenarios where the current administrator's credentials are compromised.

Recommendation

We recommend modifying the `CmdUpdateAudience` function to incorporate a dedicated parameter for specifying a new administrator during the update process via the CLI command.

Status: Resolved

5. Incomplete validation in update audience function

Severity: Minor

The `ValidateBasic` function in `x/jwk/types/message_audience.go:53` serves the purpose of validating the `MsgUpdateAudience` message. However, it fails to incorporate validation checks for the `new_admin` parameter within the message. This omission exposes the system to the risk of accepting messages containing invalid administrator data for audiences. The inclusion of such invalid data could hinder the future utilization of the audiences.

Recommendation

We recommend adding a check for the new admin address validation in `x/jwk/types/message_audience.go:53`.

Status: Resolved

6. Expired tokens cannot be removed

Severity: Minor

In the current implementation of the `DeleteAudience` function in `x/jwk/keeper/msg_server_audience.go:72`, expired tokens are not deleted automatically and hence remain stored indefinitely. Moreover, `audience` admins have to pay gas for transactions to delete audiences. This can lead to unnecessary consumption of storage space over time, as the number of expired tokens can potentially grow large.

Recommendation

Transactions to delete these tokens should not be charged a minimum gas fee (`MinGasFee`). The rationale behind this is that the deletion of expired tokens is a maintenance task that benefits the overall system, and thus, should not incur a cost to the user.

Status: Resolved

7. Incomplete `NewRegisterCmd` implementation for JWK authenticator

Severity: Minor

In the current implementation of the `NewRegisterCmd` function located in `x/xion/client/cli/tx.go:246-253`, the `--authenticator [Seckp256|Jwt,required]` flag is mentioned in the usage instructions. This flag is intended to set up the account with a JWT authenticator. As per the contract code, the `addAuthMethod` is invoked by the contract during the initialization message.

However, to add a JWK authenticator during the initialization, additional parameters such as `authenticator`, `audience`, `subject`, and `token` are required. These parameters are not currently being set in the CLI for JWK, which implies that an account can only be created with `AddAuthenticator::Secp256K1` and `ethWallet`. Consequently, the current implementation does not support the creation of an account with a JWK authenticator through the CLI.

Recommendation

We recommend supporting all implemented authenticator initialization processes.

Status: Resolved

8. Overflow checks not enabled for release profile

Severity: Minor

The following packages and contracts do not enable overflow-checks for the release profile:

- `account/Cargo.toml`
- `Cargo.toml`

Recommendation

We recommend enabling overflow checks in the workspace and all packages, including those that do not currently perform calculations. Note that enabling overflow checks in packages other than the workspace manifest will lead to compiler warnings.

Status: Resolved

9. Comitted test private keys

Severity: Informational

The `wasmbindings/keys/jwtRS256.key` file contains an RSA private key. This key appears to be used for testing purposes but it is best practice to avoid committing private keys to the repository.

Recommendation

We recommend removing the `jwtRS256.key` file containing the RSA private key from the repository and adding it to the .gitignore file to prevent accidental commits. Additionally, we recommend managing the key securely outside of the repository, utilizing environment variables or a secure key management solution for testing keys.

Status: Acknowledged

The client states that this is necessary for the CI/CD process and that the keys are only to be used for testing.

10. Messages with empty responses

Severity: Informational

Some messages return empty responses. It is best practice to emit all relevant attributes whenever state changes occur to allow off-chain consumers to react to such updates. The following messages return an empty response.

- `MsgCreateAudienceResponse` in `x/jwk/keeper/msg_server_audience.go:40`
- `MsgUpdateAudienceResponse` in `x/jwk/keeper/msg_server_audience.go:69`

Recommendation

We recommend adding attributes to the messages mentioned above.

Status: Resolved

11. Update empty query response

Severity: Informational

The `ValidateJWT` query function in `x/jwk/keeper/query_validate_jwt.go:47` returns an empty `QueryValidateJWTResponse`. Even though this query is intended to either return a success or a failure, it is best practice to include attributes to improve the user experience.

Recommendation

We recommend updating the `QueryValidateJWTResponse` to return useful information.

Status: Resolved

12. No upper cap on `timeoffset` parameter for JWK

Severity: Informational

In `x/jwk/types/params.go:55-56`, `validateTimeOffset` is used to validate only the type of `timeoffset` and does not validate the value. `timeoffset` is used to adjust the network delays of the clock, and a large offset could erroneously lead to the processing of expired JWTs.

Recommendation

We recommend enforcing upper caps for the `timeoffset` parameter.

Status: Acknowledged

The client has acknowledged this issue, stating that the parameter is currently controlled through governance which will provide adequate assurances that the value is within the expected range.

13. Missing maximum pagination may lead to RPC server exhaustion

Severity: Informational

Due to the lack of an enforcement of an upper limit on the page size in `x/jwk/client/cli/query_audience.go:23`, the RPC server could receive excessive load. If an attacker repeatedly requests large page sizes, response times may slow down and server resources may be exhausted.

Recommendation

We recommend enforcing a maximum page size.

Status: Resolved