F Y E O

Secure Code Review Bifrost Wallet

Towo Labs

June 2023
Version 1.0

Presented by:
FYEO Inc.
PO Box 147044
Lakewood CO 80214
United States

Security Level
Public

Table of Contents

List of Figures

List of Tables

EXECUTIVE SUMMARY

Overview

Towo Labs engaged FYEO Inc. to perform a Secure Code Review of the Bifrost Wallet.

The assessment was conducted remotely by the FYEO Security Team. Testing took place on May 08 - June 16, 2023, and focused on the following objectives:

- To provide the customer with an assessment of their overall security posture and any risks that were discovered within the environment during the engagement.
- To provide a professional opinion on the maturity, adequacy, and efficiency of the security measures that are in place.
- To identify potential issues and include improvement recommendations based on the results of our tests.

This report summarizes the engagement, tests performed, and findings. It also contains detailed descriptions of the discovered vulnerabilities, steps the FYEO Security Team took to identify and validate each issue, as well as any applicable recommendations for remediation.

Key Findings

The following issues have been identified during the testing period. These should be prioritized for remediation to reduce the risk they pose:

- FYEO-BIFROST-ID-01 – Possible spam attack from malicious website
- FYEO-BIFROST-ID-02 – The wallet support unsecure Android versions
- FYEO-BIFROST-ID-03 – Wallet launchMode is set to singleTask
- FYEO-BIFROST-ID-04 – Hex string is not validated in address-derivation
- FYEO-BIFROST-ID-05 – Incorrect use of substr in wallet
- FYEO-BIFROST-ID-06 – Some wallet libraries don't have stack canaries
- FYEO-BIFROST-ID-07 – Transitive dependencies with known vulnerabilities
- FYEO-BIFROST-ID-08 – The wallet uses deprecated services
- FYEO-BIFROST-ID-09 – Unnecessary string copying in address-derivation
- FYEO-BIFROST-ID-10 – Unused file in address-derivation
- FYEO-BIFROST-ID-11 – Wallet may become stuck on pending transaction on GAS price increase

Based on our review process, we conclude that the reviewed code implements the documented functionality.

The re-review has verified that the findings above Informational have all been remediated.

SCOPE AND RULES OF ENGAGEMENT

The FYEO Review Team performed a Secure Code Review Bifrost Wallet. The following table documents the targets in scope for the engagement. No additional systems or resources were in scope for this assessment.

The source code was supplied through private repositories and with their respective commit hash.

Github Repository	Commit hash
https://github.com/TowoLabs/bifrost-wallet	c8c6e8dbf94fe8cdf77963276eca65a0a6094d93
https://github.com/TowoLabs/bifrost-backend	95570a33e3f8230ea9b522d8fd737bb38ac71540
https://github.com/TowoLabs/react-native-address-derivation	485ed588d76046a7543de27c971a6e7fb0d91306
https://github.com/TowoLabs/nft-proxy	563b30b6fef6adb8d56930d8f1aef34ee9cd39d7
https://github.com/TowoLabs/svg-puppeteer	7807d672388010dee67daad05ac6693436489328

Table 1: Scope review

The re-review source code was supplied through private repositories and with their respective commit hash.

Github Repository	Commit hash
https://github.com/TowoLabs/bifrost-wallet	1c85ee2df6e410669028ea7d2a49e2e699bfd82f
https://github.com/TowoLabs/bifrost-backend	3a9bf27f4c8c21f3350c2b69cc8fe1425ff51789
https://github.com/TowoLabs/react-native-address-derivation	c9f8ac768502dc5c37d3345d16746d6b3b297746
https://github.com/TowoLabs/nft-proxy	563b30b6fef6adb8d56930d8f1aef34ee9cd39d7
https://github.com/TowoLabs/svg-puppeteer	7807d672388010dee67daad05ac6693436489328

Table 2: Scope re-review

TECHNICAL ANALYSES AND FINDINGS

During the Secure Code Review Bifrost Wallet, we discovered:

- 3 findings with MEDIUM severity rating.
- 4 findings with LOW severity rating.
- 4 findings with INFORMATIONAL severity rating.

The following chart displays the findings by severity.

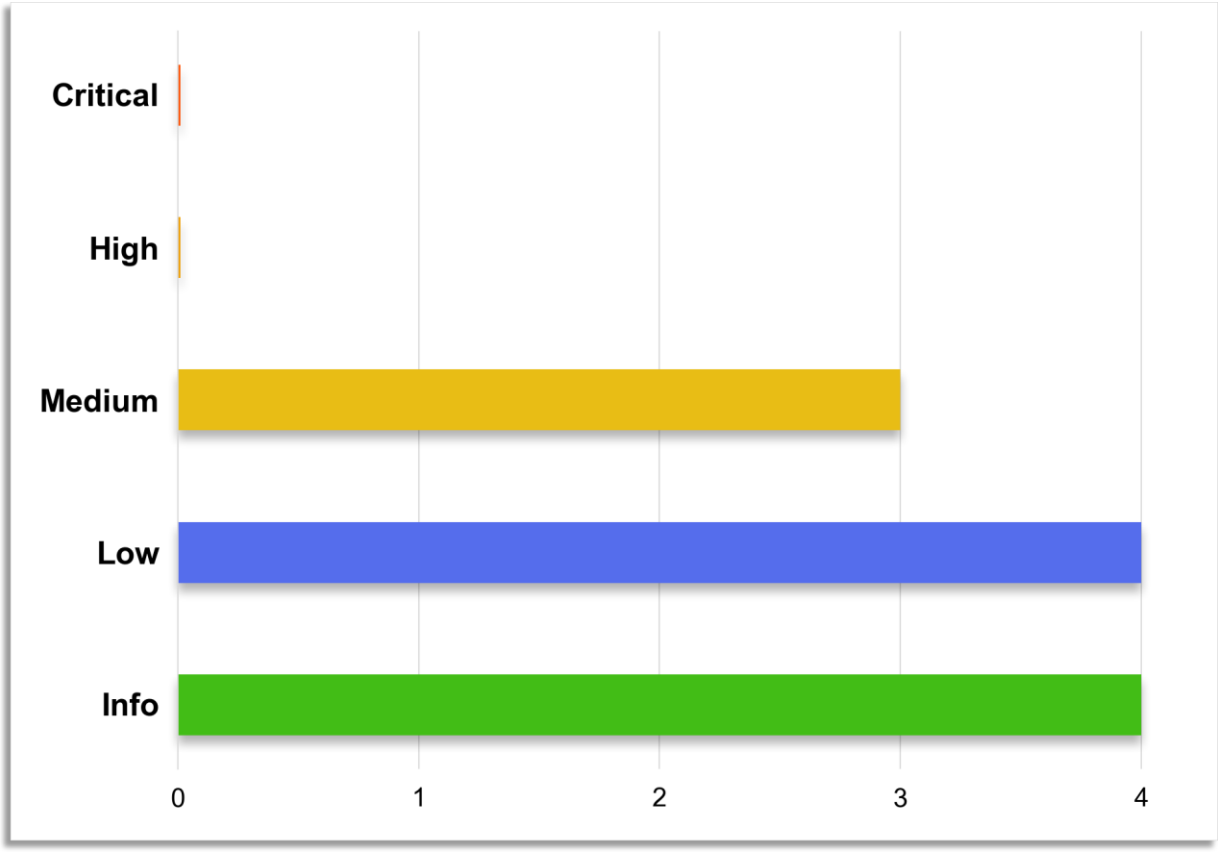


Figure 1: Findings by Severity

Findings

The *Findings* section provides detailed information on each of the findings, including methods of discovery, explanation of severity determination, recommendations, and applicable references.

The following table provides an overview of the findings.

Finding #	Severity	Description
FYEO-BIFROST-ID-01	**Medium**	Possible spam attack from malicious website
FYEO-BIFROST-ID-02	**Medium**	The wallet support unsecure Android versions
FYEO-BIFROST-ID-03	**Medium**	Wallet launchMode is set to singleTask
FYEO-BIFROST-ID-04	**Low**	Hex string is not validated in address-derivation
FYEO-BIFROST-ID-05	**Low**	Incorrect use of substr in wallet
FYEO-BIFROST-ID-06	**Low**	Some wallet libraries don't have stack canaries
FYEO-BIFROST-ID-07	**Low**	Transitive dependencies with known vulnerabilities
FYEO-BIFROST-ID-08	**Informational**	The wallet uses deprecated services
FYEO-BIFROST-ID-09	**Informational**	Unnecessary string copying in address-derivation
FYEO-BIFROST-ID-10	**Informational**	Unused file in address-derivation
FYEO-BIFROST-ID-11	**Informational**	Wallet may stuck on pending transaction on GAS price increase

Table 3: Findings Overview

Technical Analysis

The source code has been manually validated to the extent that the state of the repository allowed. The validation includes confirming that the code correctly implements the intended functionality.

Conclusion

Based on our review process, we conclude that the code implements the documented functionality to the extent of the reviewed code.

Technical Findings

General Observations

The security assessment included 5 repositories:

- Bifrost Wallet is a multi-chain crypto wallet created using React Native.
- Bifrost Backend handles access to other services and also provides data caching.
- Address Derivation - a C++ library that handles address derivation and provides JavaScript interface for wallet code.
- SVG Puppeteer - a microservice for SVG to WEBP conversion.
- NFT Proxy - a microservice for handling NFT metadata.

The audit was focused on code review and its exposure to various types of attacks, including: - Arbitrary code execution - Cross-site scripting - Incorrect storage usage - Insufficient data encryption - Prototype pollution - Zil slip - Denial-of-Service - SQL Injection - Deep Linking

During the analysis, no issues that severely affect the security of the application were found. However, a few inconsistencies and minor problems were identified. Most of the issues identified are related to the overall code quality, and some issues are related to availability and reliability.

Most of the code is covered with tests but this area can be improved further.

Overall, the code is well-documented and nicely structured. It is easy to read and maintain. Function names are self-explanatory. Code is reused when needed, and there is almost no code duplication. The wallet correctly handles secure information.

POSSIBLE SPAM ATTACK FROM MALICIOUS WEBSITE

Finding ID: FYEO-BIFROST-ID-01
Severity: **Medium**
Status: **Remediated**

Description

The wallet has no protection against multiple requests from declined websites. A malicious website may send continuous repeated requests to the wallet application, making it pop up continuously disrupting the users normal web browsing experience.

Proof of Issue

File name: [bifrost-wallet] src/data/actions/browser/ProviderActions.ts

Line number: 229

```
  private handleAccountRequest(): Thunk<string[]> {
    return async dispatch => {
      // TODO: Some dApps may call this method multiple times in quick succession. We
should block such attempts

      // Return approved accounts immediately, if any
      const alreadyApproved = await dispatch(this.glue.getApprovedAccounts());
      if (alreadyApproved.length > 0) {
        return alreadyApproved;
      }

      return dispatch(this.glue.requestAuthorization());
    };
  }
```

Severity and Impact Summary

The malicious or phishing website can spam the wallet until the request gets approved by the user. It can disrupt the user experience by making it difficult to use the wallet for its intended purposes.

Recommendation

It is recommended to validate incoming requests by blacklisting or keeping track of the sites that were already declined.

THE WALLET SUPPORT UNSECURE ANDROID VERSIONS

Finding ID: FYEO-BIFROST-ID-02
Severity: **Medium**
Status: **Remediated**

Description

The application can be installed on Android 5.0+, which makes it vulnerable to the Janus vulnerability. Janus vulnerability targets apps running on Android 5.0-7.0 signed with v1 scheme or apps running on Android 7.0-8.0 signed with v1 and v2/v3 schemes.

Proof of Issue

File name: [bifrost-wallet] build.gradle

Line number: 7

```
minSdkVersion = 21
```

Severity and Impact Summary

An attacker may use this vulnerability to replace the original app with a malicious one during an update. Such an attack creates an almost endless number of new attack vectors.

Recommendation

It is recommended to bump min SDK version to 26 to disallow running the app on insecure devices.

WALLET LAUNCHMODE IS SET TO SINGLETASK

Finding ID: FYEO-BIFROST-ID-03
Severity: **Medium**
Status: **Remediated**

Description

The launchMode refers to a setting in the application's manifest file that determines how the application is launched and behaves when it's already running. Applications with launchMode set to singleTask in the manifest are prone to task hijacking.

Proof of Issue

File name: [bifrost-wallet] android/app/src/main/AndroidManifest.xml

Line number: 19

```
android:launchMode="singleTask"
```

Severity and Impact Summary

Consequences of this vulnerability range from a UI Spoofing attack to a permission harvesting attack. By exploiting task hijacking, attackers can deceive users into thinking they are interacting with a legitimate instance of the mobile wallet application when, in reality, they are interacting with a malicious overlay or interface.

Recommendation

It is recommended to set launchMode to singleInstance to prevent other activities from becoming a part of the app's task.

References

- https://blog.dixitaditya.com/android-task-hijacking

HEX STRING IS NOT VALIDATED IN ADDRESS-DERIVATION

Finding ID: FYEO-BIFROST-ID-04
Severity: **Low**
Status: **Remediated**

Description

The function expects to receive the correct hex string but there is no implicit validation.

Proof of Issue

File name: [react-native-address-derivation] cpp/common/utils.cpp

Line number: 7-20

```
size_t len = str.size() / 2;
std::vector<uint8_t> bufVector(len);

uint8_t *buf = bufVector.data();
for (size_t i = 0; i < len; i++) {
    uint8_t c = 0;
    if (str[i * 2] >= '0' && str[i * 2] <= '9') c += (str[i * 2] - '0') << 4;
    if ((str[i * 2] & ~0x20) >= 'A' && (str[i * 2] & ~0x20) <= 'F')
    c += (10 + (str[i * 2] & ~0x20) - 'A') << 4;
    if (str[i * 2 + 1] >= '0' && str[i * 2 + 1] <= '9')
    c += (str[i * 2 + 1] - '0');
    if ((str[i * 2 + 1] & ~0x20) >= 'A' && (str[i * 2 + 1] & ~0x20) <= 'F')
    c += (10 + (str[i * 2 + 1] & ~0x20) - 'A');
    buf[i] = c;
}
```

Severity and Impact Summary

The incorrect parameter will produce an invalid buffer.

Recommendation

It is recommended to validate the size and the content of the string parameter.

INCORRECT USE OF SUBSTR IN WALLET

Finding ID: FYEO-BIFROST-ID-05
Severity: **Low**
Status: **Remediated**

Description

The `substr` function doesn't change the string but returns a new one. The code ignores return value.

Proof of Issue

File name: [bifrost-wallet] src/data/utils/HexUtils.ts

Line number: 12

```
    if (data.length % 2 !== 0) {
      data.substr(0, data.length - 1);
    }
```

Severity and Impact Summary

The code has no effect. This issue is mitigated in the next instructions by `Buffer.from`.

Recommendation

It is recommended to assign the result of `substr` operation to the `data` variable.

SOME WALLET LIBRARIES DON'T HAVE STACK CANARIES

Finding ID: FYEO-BIFROST-ID-06
Severity: **Low**
Status: **Open**

Description

Stack canaries are used to protect critical stack values against buffer overflow attacks. They act as a guard mechanism against buffer overflow attacks by detecting when the integrity of the stack has been compromised.

Some of the used libraries don't include stack canaries.

By the look of the libraries these seem to be the core react render libraries and should therefore have a limited security impact. Also react native does not seem to offer a way to easily build these libraries with stack canaries

Proof of Issue

```
    - lib/armeabi-v7a/libreact_render_debug.so
    - lib/armeabi-v7a/libbetter.so
    - lib/x86_64/libreact_render_debug.so
    - lib/x86_64/libreact_debug.so
    - lib/x86_64/libbetter.so
    - lib/x86_64/libreact_utils.so
    - lib/x86_64/libreact_render_telemetry.so
    - lib/x86_64/libreactconfig.so
    - lib/arm64-v8a/libreact_render_debug.so
    - lib/arm64-v8a/libreact_debug.so
    - lib/arm64-v8a/libbetter.so
    - lib/arm64-v8a/libreact_utils.so
    - lib/arm64-v8a/libreact_render_telemetry.so
    - lib/arm64-v8a/libreactconfig.so
```

Severity and Impact Summary

Buffer overflow attacks occur when a program writes data beyond the boundaries of a buffer, leading to memory corruption and potential execution of malicious code. Attackers can exploit this vulnerability to gain unauthorized access, execute arbitrary code, or manipulate the program's behavior.

Recommendation

It is recommended to upgrade the problematic libraries or replace them with a version that includes stack canaries.

This is on the roadmap to fix and will be fixed by the team when possible after migrating to Hermes to be able to build the libraries with canaries.

References

This thread shows that the react native libraries are still not built with stack canaries
https://github.com/facebook/react-native/issues/36870

TRANSITIVE DEPENDENCIES WITH KNOWN VULNERABILITIES

Finding ID: FYEO-BIFROST-ID-07
Severity: **Low**
Status: **Remediated**

Description

The project uses dependencies that have known vulnerabilities.

Proof of Issue

```
> yarn audit

82 vulnerabilities found - Packages audited: 2821
Severity: 4 Moderate | 69 High | 9 Critical
```

Severity and Impact Summary

A general analysis of vulnerabilities in dependencies shows that the effect on the code is minimal.

Recommendation

It is recommended to add the following pinned versions to the resolutions in package.json: - “ua-parser-js”: “0.7.33” - https://www.npmjs.com/advisories/1088697 - “pretty-format/ansi-regex”: “5.0.1” - https://www.npmjs.com/advisories/1091190 - “react-native/**/pretty-format/ansi-regex”: “5.0.1” - https://www.npmjs.com/advisories/1091190 - “react-native/*/ansi-regex”: “4.1.1” - https://www.npmjs.com/advisories/1091190 - “**/decode-uri-component”: ”0.2.1” - https://www.npmjs.com/advisories/1091652 - ”@react-native-community/eslint-config/**/json5”: “2.2.2” - https://www.npmjs.com/advisories/1091148 - “babel-loader/loader-utils”: “1.4.2” - https://github.com/advisories/GHSA-76p3-8jx3-jpfq, https://www.npmjs.com/advisories/1091251 - “realm/request/qs”: “6.5.3” - https://www.npmjs.com/advisories/1090135 - “react-native/**/metro/async”: “2.6.4” - https://www.npmjs.com/advisories/1088666

THE WALLET USES DEPRECATED SERVICES

Finding ID: FYEO-BIFROST-ID-08
Severity: **Informational**
Status: **Remediated**

Description

Some URLs in the chain configuration are outdated: - stratosnft.io - the service is shutdown - quixotic.io was renamed to qx.app

Proof of Issue

File name: [bifrost-wallet] src/data/derivation/configuration/chain-configs/Arbitrum.ts

Line number: 60, 87

```
marketplaceUrl: (address: string, id: string) =>
`https://stratosnft.io/asset/${encodeURIComponent(address)}/${encodeURIComponent(id)}`
,
...
marketplaceUrl: (address: string, id: string) =>
`https://testnet.stratosnft.io/asset/${encodeURIComponent(address)}/${encodeURICompone
nt(id)}`,
```

File name: [bifrost-wallet] src/data/derivation/configuration/chain-configs/Optimism.ts

Line number: 60, 87

```
marketplaceUrl: (address: string, id: string) =>
`https://quixotic.io/asset/${encodeURIComponent(address)}/${encodeURIComponent(id)}`,
...
marketplaceUrl: (address: string, id: string) =>
`https://https://testnet.qx.app/asset/${encodeURIComponent(address)}/${encodeURICompon
ent(id)}`,
```

Severity and Impact Summary

Users may experience issues with these services.

Recommendation

It is recommended to replace outdated URLs.

UNNECESSARY STRING COPYING IN ADDRESS-DERIVATION

Finding ID: FYEO-BIFROST-ID-09
Severity: **Informational**
Status: **Remediated**

Description

The `fromhex` function accepts a string parameter by the value that will create a copy. This string is not modified in the function.

Proof of Issue

File name: [react-native-address-derivation] cpp/common/utils.cpp

Line number: 6

```
const std::vector<uint8_t> fromhex(std::string str) {
```

Severity and Impact Summary

The unnecessary copy consumes more memory.

Recommendation

It is recommended to accept this parameter by `const&`.

UNUSED FILE IN ADDRESS-DERIVATION

Finding ID: FYEO-BIFROST-ID-10
Severity: **Informational**
Status: **Remediated**

Description

The code has a header file that is not used, doesn't have protection from multiple includes, and its content is duplicated.

Proof of Issue

File name: [react-native-address-derivation] cpp/common/DerivationStrategy/DerivationResult.h

Severity and Impact Summary

Accidental use of this file may cause compilation problems.

Recommendation

It is recommended to remove unused files.

WALLET MAY STUCK ON PENDING TRANSACTION ON GAS PRICE INCREASE

Finding ID: FYEO-BIFROST-ID-11
Severity: **Informational**
Status: **Open**

Description

The wallet doesn't have the functionality for processing stuck pending transactions. When the gas price rises significantly, transactions with lower gas fees may get stuck in a pending state, waiting for miners to prioritize and include them in the blockchain. This can result in delays in transaction confirmation, causing inconvenience and frustration for wallet users.

Proof of Issue

File name: [bifrost-wallet] src/data/transactions/strategies/EvmTransactionStrategy.ts

Line number: 234

```
      case CustomTxType.REPLACE_TRANSACTION:
        throw new Error("Replace Transaction is not available for EVM chains");
```

Severity and Impact Summary

No further transaction will be processed if there is a pending one.

Recommendation

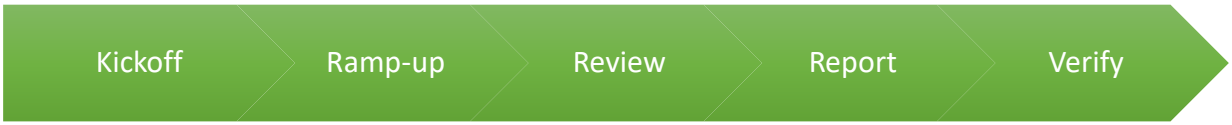
It is recommended to add cancel/speed-up feature for EVM pending transactions.

OUR PROCESS

METHODOLOGY

FYEO Inc. uses the following high-level methodology when approaching engagements. They are broken up into the following phases.

Figure 2: Methodology Flow



KICKOFF

The project is kicked off as the sales process has concluded. We typically set up a kickoff meeting where project stakeholders are gathered to discuss the project as well as the responsibilities of participants. During this meeting we verify the scope of the engagement and discuss the project activities. It's an opportunity for both sides to ask questions and get to know each other. By the end of the kickoff there is an understanding of the following:

- Designated points of contact
- Communication methods and frequency
- Shared documentation
- Code and/or any other artifacts necessary for project success
- Follow-up meeting schedule, such as a technical walkthrough
- Understanding of timeline and duration

RAMP-UP

Ramp-up consists of the activities necessary to gain proficiency on the project. This can include the steps needed for familiarity with the codebase or technological innovation utilized. This may include, but is not limited to:

- Reviewing previous work in the area including academic papers
- Reviewing programming language constructs for specific languages
- Researching common flaws and recent technological advancements

Review

The review phase is where most of the work on the engagement is completed. This is the phase where we analyze the project for flaws and issues that impact the security posture. Depending on the project this may include an analysis of the architecture, a review of the code, and a specification matching to match the architecture to the implemented code.

In this code audit, we performed the following tasks:

1. Security analysis and architecture review of the original protocol
2. Review of the code written for the project
3. Compliance of the code with the provided technical documentation

The review for this project was performed using manual methods and utilizing the experience of the reviewer. No dynamic testing was performed, only the use of custom-built scripts and tools were used to assist the reviewer during the testing. We discuss our methodology in more detail in the following sections.

Code Safety

We analyzed the provided code, checking for issues related to the following categories:

- General code safety and susceptibility to known issues
- Poor coding practices and unsafe behavior
- Leakage of secrets or other sensitive data through memory mismanagement
- Susceptibility to misuse and system errors
- Error management and logging

This list is general and not comprehensive, meant only to give an understanding of the issues we are looking for.

Technical Specification Matching

We analyzed the provided documentation and checked that the code matches the specification. We checked for things such as:

- Proper implementation of the documented protocol phases
- Proper error handling
- Adherence to the protocol logical description

Reporting

FYEO Inc. delivers a draft report that contains an executive summary, technical details, and observations about the project.

The executive summary contains an overview of the engagement including the number of findings as well as a statement about our general risk assessment of the project. We may conclude that the overall risk is low but depending on what was assessed we may conclude that more scrutiny of the project is needed.

We report security issues identified, as well as informational findings for improvement, categorized by the following labels:

- Critical
- High
- Medium
- Low
- Informational

The technical details are aimed more at developers, describing the issues, the severity ranking and recommendations for mitigation.

As we perform the audit, we may identify issues that aren't security related, but are general best practices and steps that can be taken to lower the attack surface of the project. We will call those out as we encounter them and as time permits.

As an optional step, we can agree on the creation of a public report that can be shared and distributed with a larger audience.

Verify

After the preliminary findings have been delivered, this could be in the form of the approved communication channel or delivery of the draft report, we will verify any fixes within a window of time specified in the project. After the fixes have been verified, we will change the status of the finding in the report from open to remediated.

The output of this phase will be a final report with any mitigated findings noted.

Additional Note

It is important to note that, although we did our best in our analysis, no code audit or assessment is a guarantee of the absence of flaws. Our effort was constrained by resource and time limits along with the scope of the agreement.

While assessing the severity of the findings, we considered the impact, ease of exploitability, and the probability of attack. This is a solid baseline for severity determination.

THE CLASSIFICATION OF VULNERABILITIES

Security vulnerabilities and areas for improvement are weighted into one of several categories using, but is not limited to, the criteria listed below:

Critical – vulnerability will lead to a loss of protected assets

- This is a vulnerability that would lead to immediate loss of protected assets
- The complexity to exploit is low
- The probability of exploit is high

High - vulnerability has potential to lead to a loss of protected assets

- All discrepancies found where there is a security claim made in the documentation that cannot be found in the code
- All mismatches from the stated and actual functionality
- Unprotected key material
- Weak encryption of keys
- Badly generated key materials
- Txn signatures not verified
- Spending of funds through logic errors
- Calculation errors overflows and underflows

Medium - vulnerability hampers the uptime of the system or can lead to other problems

- Insecure calls to third party libraries
- Use of untested or nonstandard or non-peer-reviewed crypto functions
- Program crashes, leaves core dumps or writes sensitive data to log files

Low – vulnerability has a security impact but does not directly affect the protected assets

- Overly complex functions
- Unchecked return values from 3rd party libraries that could alter the execution flow

Informational

- General recommendations