AUDIT REPORT

Leap

Leap Wallet Extension

Prepared by SCV-Security

Prepared on 26th August 2023

Table of Contents

Introduction

SCV has been engaged by Leap to conduct a comprehensive security review with the goal of identifying potential security threats and vulnerabilities within the codebase. The purpose of this audit is to evaluate the security posture of the codebase and provide actionable recommendations to mitigate any identified risks. This report presents an overview of the findings from our security audit, outlining areas of concern and proposing effective measures to enhance the codebase's security.

Methodologies

SCV performs a combination of automated and manual security testing based on the scope of testing. The testing performed is based on the extensive experience and knowledge of the auditor to provide the greatest coverage and value to Leap Wallet. Testing includes, but is not limited to, the following:

- Understanding the application and its functionality purpose.
- Deploying SCV in-house tooling to automate dependency analysis and static code review.
- Analyse each line of the code base and inspect application security perimeter.
- Review underlying infrastructure technologies and supply chain security posture.
- Analyse potential sensitive data added to the Document Object Model DOM.
- Examine the presence of attack vectors and conduct assessments to identify potential vulnerabilities, including but not limited to: arbitrary code execution, cross-site scripting, insecure storage usage, and insufficient data encryption.

Scope Functionality

Leap Wallet is a non-custodial web3 wallet for the Cosmos ecosystem. It allows users to store, send, receive, and stake Cosmos-based tokens. Leap Wallet also supports decentralised applications (dApps) built on the Cosmos network.

Scope Submitted

Leap Extension (Chrome)	
Repository	https://github.com/leapwallet/cosmos-extension
Commit	`b68a2091b8d5a555f0da0a1e2e941071f914256f`
Branch	`main`
Components	• Background • Content-script • Password authentication • Key management library integration • NFTs
Key management library	
Repository	https://github.com/leapwallet/leap-keychain
Commit	`ff1130d9d2aaa38deecedd1597fbe603a5524418`
Branch	`main`
Leap Extension Wallet (Penetration Testing)	
Google Store	leap-cosmos-wallet/fcfcfllfndlomdhbehjjcoimbgofdncg
Version	`0.7.11`

Revisions Remediations

Leap Extension (Chrome)	
Repository	https://github.com/leapwallet/cosmos-extension
Commit	`e8a8c9a8ac28f7e3e72018cf1b06189b00904b28`
Branch	`main`
Key management library	

Repository	https://github.com/leapwallet/leap-keychain
Commit	`077f3f420463701dc1738e9329fa7c44ac133a9b`
Branch	`main`
Leap Extension Wallet (Penetration Testing)	
Google Store	leap-cosmos-wallet/fcfcfllfndlomdhbehjjcoimbgofdncg
Version	`0.8.0`

Code Criteria

This section provides an evaluation of specific criteria aspects as described below:

- **Documentation:** Evaluating the presence and comprehensiveness of publicly available or provided explanatory information, diagram flowcharts, comments, and supporting documents to enhance code understanding.
- **Readability:** Assessing how easily the code can be understood and maintained, considering factors such as code structure, naming conventions, and overall organisation.
- **Complexity:** Evaluating the complexity of the code, including factors such as, number of lines, conditional statements, and nested structures.

The status of each criteria is categorised as either **SUFFICIENT** or **NOT-SUFFICIENT** based on the audit assessment. This categorisation provides insights to identify areas that may require further attention and improvement.

Criteria	Status	Notes
Documentation	**NOT-SUFFICIENT**	Documentation was partially available. Several functions and logic flow were lacking in-line comments and can be improved.
Readability	**SUFFICIENT**	The codebase had good readability overall. Outstanding TODOs and hardcoded configurations can be removed from the code base to further improve its readability.
Complexity	**SUFFICIENT**	N/A

Findings Summary

Summary Title	Risk Impact	Status
Wallet mnemonic will be saved in the local disk after LeapWallet import.	SEVERE	RESOLVED
Insufficient iterations on PBKDF2 can make brute force attacks easier.	SEVERE	RESOLVED
Messages via runtime.onMessage events are not validated.	SEVERE	RESOLVED
shouldInjectProvider content-script injection logic is not not comprehensive.	MODERATE	RESOLVED
Low entropy mnemonic for native Leap wallets created during user on-board.	LOW	ACKNOWLEDGED
Contact names and addresses are not unique and can be added multiple times.	LOW	RESOLVED
Dependencies should be pinned in packages.json.	LOW	RESOLVED
Lack of error handling on “Connected Sites” URLs.	LOW	RESOLVED
Vulnerable dependencies in use.	LOW	RESOLVED
Content-script matches in manifest.json are too broad.	LOW	RESOLVED
Arbitrary MsgSignData message view incomplete.	LOW	RESOLVED
Variables are declared but never used.	INFO	RESOLVED
Outstanding TODOS comments and word typos.	INFO	RESOLVED
PasswordManager class has exposed sensitive properties.	INFO	RESOLVED
Extension options declared in the manifest have no effect.	INFO	RESOLVED
Incognito window is not considered by the extension.	INFO	ACKNOWLEDGED
No password prompt confirmation when deleting wallet.	INFO	ACKNOWLEDGED
Unused localStorage theme attribute.	INFO	ACKNOWLEDGED
Content Security Policy (CSP) and manifest host_permissions have no effect.	INFO	PARTIALLY RESOLVED

Findings Technical Details

This section was intentionally removed as per Leap team requested for the reader audience. Vulnerabilities outlined in the Finding Summary provide context on identified items during the audit engagement.

Audit Observations

The audit observations section is intended to present potential findings that are related to the underlying design of the application and would require underlying design changes to remediate that may change the overall functioning of the current implementation.

This section was intentionally removed as per Leap team requested for the reader audience. Vulnerabilities outlined in the Finding Summary provide context on identified items during the audit engagement.

Analysis From Past Incidents

SCV-Security conducted an analysis on past security incidents involving well-known Web3 wallets, aiming to draw a comparison with the security posture of LeapWallet. The analysis took into account the following incidents:

- **MetaMask phishing attack - January 2022**
  - A phishing attack on the MetaMask website resulted in the loss of over $30 million in cryptocurrencies. The attackers created a fake MetaMask website that looked like the real website. When users entered their login credentials on the fake website, the attackers were able to steal them.
- **Forta Security breach - March 2022**
  - Forta Security, a company that provides security alerts for Web3 protocols, was hacked. The attackers gained access to Forta's internal systems and were able to steal sensitive data, including the private keys of some of Forta's customers.
- **WalletConnect phishing attack - April 2022**
  - Another phishing attack targeting the WalletConnect website, resulted in the loss of over $1 million in cryptocurrencies.
- **Phantom Wallet hack - May 2022**
  - The vulnerability allowed the attackers to sign transactions on behalf of users without their knowledge. This allowed the attackers to steal over $6+ million in cryptocurrencies from over 9,000 wallets. The vulnerability was caused by a bug in the way that the wallet handled importing accounts from Slope Finance via supply chain attack. Slope Finance is a Solana-based DeFi platform. The bug allowed the attackers to create a malicious Slope Finance website that could trick users into importing their Phantom Wallet accounts.
- **MathWallet hack - June 2022**
  - The vulnerability was caused by a bug in the way that the wallet handled importing accounts from NFT marketplaces. The bug allowed the attackers to create a malicious NFT marketplace website

that could trick users into importing their MathWallet accounts. This allowed the attackers to steal over $2 million in cryptocurrencies from over 1,000 wallets.

- **Atomic Wallet - June 2022**
    - While the exact vulnerabilities involved are not yet clear, this attack has been attributed to a well-known hacking group called Lazarus, believed to be based in North Korea. The attack resulted in the theft of over $100 million worth of cryptocurrencies from more than 5,000 wallets. The primary method of attack was purported to be phishing campaigns, but the situation remains ambiguous due to reports from some users claiming they did not receive any phishing messages. There's also speculation that the Lazarus group may have compromised the Atomic core developers, allowing them to insert malicious code that executed once wallets were unlocked. The investigation is ongoing, and the precise root cause remains uncertain at this time

Previous incidents have indicated that the underlying reasons for these security breaches can be attributed to any of the following factors:

- Phishing attacks;
- Supply chain attacks;
- Flawed code logic;
- Deliberate and premeditated actions;

Phishing attacks, unfortunately, stand out as one of the most frequent causes of security incidents. Scammers continually devise sophisticated strategies to persuade their intended victims into divulging their passphrase or secrets, with the ultimate aim of stealing funds.

In many cases, these phishing attacks could be mitigated or even prevented with the help of alerts that warn users about potentially unsafe websites. However, implementing such alerts can often be intricate and challenging to consistently maintain.

With regards to Supply Chain Attacks, it is crucial for Leap Wallet to implement a robust access control mechanism for published packages through npm. This mechanism ensures the legitimacy of packages and helps prevent potential vulnerabilities arising from the supply chain.

Additionally, Leap should consider implementing a signing solution, similar to Sigstore, to ensure the origin and integrity of packages. Such a solution would involve digitally signing packages using cryptographic keys held by authorised teams, creating a verifiable chain of trust throughout the supply chain. This approach would strengthen confidence among users, assuring them of the software's authenticity and deterring unauthorised modifications. Furthermore, the utilisation of a signing solution would enhance accountability by enabling efficient tracking of package origins, facilitating prompt responses to security incidents, and ultimately fortifying Leap Wallet's security measures.

In conclusion, SCV-Security has not identified any critical flaws in the code logic of Leap Wallet that would enable or facilitate the described attacks. The storage of private keys and seed phrases within the extension is isolated from the Document Object Model (DOM), preventing their accessibility by dapps and pages accessed by the user.

Best Practices CheckList

As part of the engagement, SCV-Security is providing a checklist to be used for code changes and new implementations. This checklist can be used as a guide for Leap developers, aiming to achieve code consistency and adhering security standards, general security and development best practices. Note that the exact requirements might vary depending on projects and team's preferences.

- ☐ I have performed a self-review of my own code.
- ☐ I have used `yarn audit` to ensure no vulnerable dependencies are added.
- ☐ I have commented on my code, especially in complex areas.
- ☐ I have updated the documentation as needed.
- ☐ My changes generate no new warnings on build.
- ☐ I have added comprehensive tests where necessary.
- ☐ New and existing unit tests pass locally with my changes.
- ☐ I have confirmed that there are no outstanding `//TODO` comments.
- ☐ I have confirmed that there are unusable variables in my code.
- ☐ I have verified that there are no hardcoded variables in my code.
- ☐ I have implemented thorough error handling.
- ☐ I have considered edge cases and boundary conditions.
- ☐ I have followed coding standards and conventions.
- ☐ I have ensured maintainability by avoiding overly complex solutions.
- ☐ I have documented external services, APIs, and third-party libraries used.
- ☐ I have provided descriptive commit messages.
- ☐ I have tested across different browsers and environments.
- ☐ I have reviewed and revised code comments for accuracy and clarity.
- ☐ I have considered performance optimization where applicable.
- ☐ I have requested a colleague's code review for additional feedback.
- ☐ I have ensured that added dependencies are pinned to exact versions.

Conclusion

This audit report conducted by SCV-Security has provided a comprehensive assessment of the Leap Wallet Extension's security posture. The primary objective of this review was to identify potential security threats and vulnerabilities within the codebase and offer actionable recommendations for mitigating these risks.

The findings summary section outlines several risk levels associated with identified issues, ranging from severe to informational. These findings express concerns related to encryption methods, security access controls, code maintainability, and potential user experience improvements. Remediations have been suggested for each issue, with many already resolved.

Audit observations aim to underlying design considerations and potential changes that may enhance the overall functioning of the wallet. Furthermore, the report draws upon past incidents involving Web3 wallets to highlight common attack vectors and vulnerabilities. It underscores the importance of alert mechanisms, supply chain security, and the adoption of code signing solutions as proactive measures to fortify Leap Wallet's security posture.

The Best Practices Checklist serves as a guide for developers to maintain code consistency, adhere to security standards, and follow general best practices in development. It encompasses aspects like code review, documentation, error handling, and more.

SCV-Security outlines its risk assessment methodology, which combines the likelihood and impact of vulnerabilities to calculate overall risk levels.

Overall, this audit report equips Leap Wallet Extension's development team with valuable insights, recommendations, and best practices to bolster the security and reliability of their product. By addressing the identified issues and following the provided guidelines, Leap Wallet can enhance its security posture and provide a safer experience for its users.

Document Control

Version	Date	Notes
-	7th August 2023	Security audit commencement date.
0.1	18th August 2023	Initial report with identified findings delivered.
0.5	25th August 2023	Fixes remediations implemented and reviewed.
1.0	26th August 2023	Audit completed, final report delivered.

Appendices

A. Appendix – Risk assessment methodology

SCV-Security employs a risk assessment methodology to evaluate vulnerabilities and identified issues. This approach involves the analysis of both the LIKELIHOOD of a security incident occurring and the potential IMPACT if such an incident were to happen. For each vulnerability, SCV-Security calculates a risk level on a scale of 5 to 1, where 5 denotes the highest likelihood or impact. Consequently, an overall risk level is derived from combining these two factors, resulting in a value from 10 to 1, with 10 signifying the most elevated level of security risk

Risk Level	Range
CRITICAL	10
SEVERE	From 9 to 8
MODERATE	From 7 to 6
LOW	From 5 to 4
INFORMATIONAL	From 3 to 1

LIKELIHOOD and **IMPACT** would be individually assessed based on the below:

Rate	LIKELIHOOD	IMPACT
5	**Extremely Likely**	Could result in severe and irreparable consequences.
4	**Likely**	May lead to substantial impact or loss.
3	**Possible**	Could cause partial impact or loss on a wide scale.
2	**Unlikely**	Might cause temporary disruptions or losses.
1	**Rare**	Could have minimal or negligible impact.

B. Appendix – Report Disclaimer

This report should not be regarded as an "endorsement" or "disapproval" of any specific project or team. These reports do not indicate the economics or value of any "product" or "asset" created by a team or project that engages SCV-Security for a security review. The audit report does not make any statements or warranties about the code's utility, safety, suitability of the business model, regulatory compliance of the business model, or any other claims regarding the fitness of the implementation for its purpose or its bug-free status. The audit documentation is intended for discussion purposes only. The content of this audit report is provided "as is," without representations and warranties of any kind, and SCV-Security disclaims any liability for damages arising from or in connection with this audit report. Copyright of this report remains with SCV-Security.

THANK YOU FOR CHOOSING

SCV SECURITY

scv.services

contact@scv.services