DECENTRALIZED APPLICATION CODE REVIEW AND SECURITY ANALYSIS REPORT

SCRAMBLE
Nov 30th, 2023

Hacken OÜ

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Table of Contents

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Confidentiality Statement

This document contains sensitive and proprietary information belonging to the customer, including details about the customer's intellectual property and potential vulnerabilities, as well as methods for exploiting these vulnerabilities. The information contained in this document is intended for use by the customer and may be disclosed to the public at the customer's discretion.

The unauthorized reproduction, distribution, or sharing of this document or its contents is strictly prohibited. This includes sharing the information with any third-party individuals or organizations without the prior written consent of the customer.

In the event of a breach of confidentiality, all parties with access to this information must immediately notify the customer and take appropriate steps to remedy the situation.

This confidentiality agreement will remain in effect until the information contained in this document becomes public knowledge through no fault of any party who has access to it, or until the customer revokes it.

By accessing or using this document, all parties agree to be bound by the terms and conditions outlined in this confidentiality agreement and to maintain the confidentiality of the information contained therein.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Introduction

Hacken OÜ (Consultant) was contracted by Scramble (Customer) to conduct a Decentralized Application Code Review and Security Analysis to identify security weaknesses and provide remediation recommendations. This report presents the security assessment findings of the customer's applications.

Customer	Scramble
Website	Scramble

UID	Scramble		
#	**Date**	**Occasion**	**Score**
1	Oct 27, 2023	Review	6.421
2	Nov 30, 2023	Remediation review	8

Approved By
Stephen Ajayi - dApp Audit Technical Lead, s.ajayi@hacken.io
Luciano Ciattaglia - Services Director, l.ciattaglia@hacken.io

Scope

The scope of the project is review and security analysis of applications in the following repositories:

1. https://github.com/wavect/ScrambleWallet Commit (e3d05222) for findings evaluation and (b9dd4f59da6aa7a0449507922eb409b9f27c2aa3/beta-release branch) commit for remediation phase.

The source code of these applications underwent a secure code review and static application security testing. Additionally, it was checked for commonly known and industry-specific vulnerabilities. Considered items include but are not limited to:

- Overconfidence in a node (or node provider)
- Failure to account for a blockchain branching out
- Incorrect validation of ENS records
- Weak authentication via message signing
- Unsafe private key storage
- XSS/SQL injections from the blockchain data
- Misuse of checksum addresses
- Blockchain data inconsistency
- Incorrect integration with a smart contract and/or blockchain network

- Usage of wrong data types
- Deprecated, vulnerable, or outdated Web 3.0 libraries
- *Important: backend API is not in scope.*

System Overview

Scramble is a browser extension wallet that supports various cryptocurrencies and NFTs, including TRON. It is a modular application, with each frontend route having its controller/service that can communicate with internal development APIs and other public services. This architecture makes Scramble highly extensible and adaptable, allowing it to add support for new cryptocurrencies and web3 features.

Scramble https://github.com/wavect/ScrambleWallet

Table. Public Views

Public View	Description
GET /	Intro Screen
GET /locked	Locked Screen
GET /activity:id	List activities
GET /assets:id	List balance per id {BTC/ETH,…}
GET /dapps:id	List dApps per id {BTC/ETC,…}
GET /nfts/:id?	List NFTs per id {}
GET /send/:id?	Sent transactions
GET /verify-transaction/:id?	Check transactions (before sending it)
GET /swap/:id?	Swap current crypto per other one by id.
GET /swap-best-offer/:id?	Suggest the best offer per current currency.
GET /swap-best-offer-hw/:id?	Suggest the best offer per current currency. (Including TRON/HW)
GET /add-network	List all networks and add/delete them into your list wallets.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Executive Summary

The overall system score represents a high-level rating of the system's security and consists of the following metrics: Security, Code Quality, and Documentation Quality. For more information on how this score is calculated, see the "Scoring" section of the methodology.

The overall system score is **6.421** out of **10**

To improve the overall system score, see "Security" and "Code Quality" sections of the executive summary.

After the remediation phase of findings:

The overall system score is **8** out of **10**

Documentation Quality

This score reflects the quality of the functional requirements and technical documentation provided for the review.

- The application's desired behavior is not documented.
- The functional requirements are not clear and concise.
- The functional requirements are up-to-date.
- The technical documentation needs to contain clear instructions on how to set up the environment and run the application.

The documentation quality score is **5.76** out of **10**

To improve the documentation quality score, include instructions on how to set up the environment locally and update the functional requirements to the latest version.

Code Quality

This score reflects the quality of the source code provided for the review. It includes test coverage, adherence to the style guides, and how easy it is to follow the code.

- The core functionality needs to be fully covered with tests.
- The source code contains hard-to-follow logic flows.
- The source code is not properly formatted.
- The source code does not adhere to the style guides.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

The code quality score is **6.45** out of **10**

Security

This score reflects how well the system addresses three core information security tenets: confidentiality, integrity, and availability.

3 high, **4** medium, **10** low and **10** informational severity issues were identified after the initial code review and security analysis.

After the remediation phase, the system had **2** medium, **6** low and **12** informational severity issues, and:

The security score is **8.5** out of **10**

A detailed description of found issues can be found in the "Issue Overview" section of the report.

To improve the security score medium issues should be fixed.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Definitions

Table. Issue Severity Definition

Severity	Description
Critical	These issues present a major security vulnerability that poses a severe risk to the system. They require immediate attention and must be resolved to prevent a potential security breach or other significant harm.
High	These issues present a significant risk to the system, but may not require immediate attention. They should be addressed in a timely manner to reduce the risk of the potential security breach.
Medium	These issues present a moderate risk to the system and cannot have a great impact on its function. They should be addressed in a reasonable time frame, but may not require immediate attention.
Low	These issues present no risk to the system and typically relate to the code quality problems or general recommendations. They do not require immediate attention and should be viewed as a minor recommendation.

Table. Issue Status Definition

Status	Description
New	The issue was presented to the customer. The remediation after the initial discovery was not yet made.
Reported	The issue was not fixed as a result of the remediation. The customer was informed of the risks associated with it.
Fixed	The issue was fixed and does not present a risk to the system.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Issue Overview

■■■ High

H01. Use of own crypto

Common Weakness Enumeration	CWE-1240
Status	Fixed commit a8bd117dfef467c8f3e90e491ff0db56244e430b

Description:
The application utilizes its custom encryption implementation for data security and integrity verification. For the Message Authentication Code (MAC), it relies on keccak256 applied to the concatenated string and ciphertext. While keccak256 is resistant to hash extension attacks, it is recommended to implement an HMAC construction for added security. Moreover, it's crucial to note that the initialization vector (IV) is not included in the integrity check. Consequently, an attacker could potentially provide an alternate IV, leading to the decryption of nonsensical data and potentially compromising the underlying business logic.

PoC.

```
utils/encrypt.ts
 const mac = keccak256(
    bufferToHex(
                  Buffer.concat([Buffer.from(derivedKey.slice(16,    32)),
sparams.ciphertext])
    )
  );
            if      (mac      !==      sparams.mac)      throw      new
Error(Errors.OtherErrors.WrongPassword);
  const decipher = createDecipheriv(
    sparams.cipher,
    derivedKey.slice(0, 16),
    sparams.iv
  );
  return runCipherBuffer(decipher, sparams.ciphertext);
```

Impact:
Bypassing mac controls through a forged initialization vector (IV) may compromise application logic and facilitate further attacks on user assets.

Recommendation:
Utilize exclusively verified libraries and their API for encryption/decryption and integrity checks.

Fixed Details: The customer has switched to the simple AES-GCM method from the https://www.npmjs.com/package/cryptr package. Even if the package does not recommend encrypted passwords, it is used for the keyring and we do not see a significant risk of violation of vendor recommendation.

H02. Hardcoded TRON Pro API Key

Common Weakness Enumeration	CWE-321
Status	**False Positive**

Description:
The application relies on a hardcoded Tron API key. TronGrid provides a comprehensive range of full-node HTTP APIs and extended APIs for the TRON network. To ensure fair allocation of requested resources, all API requests must include the API Key as a parameter. Requests lacking API Key will either face severe limitations or not receive a response at all.
Currently, each Account per day is permitted a maximum of 100,000 per day, with the possibility of adjustments based on related requirements in the future. If a user exceeds this limit while using the API key the, the access frequency per second will be substantially restricted to around 5 queries per seconds (5qps). Going beyond this limit will result in access denied, and a 503 error will be returned.

Fore more information, please refer to:
https://developers.tron.network/reference/select-network

PoC.

```
utils/src/tronweb.ts

 tronWeb.setHeader({ "TRON-PRO-API-KEY": "15cdaea*****268a", }); // TODO:
appropriate secret handling

There is also unclear hardcoded Tron address:

if (!privateKey) { // TODO #MOUN-85
tronWeb.setAddress("TYRHB9Cna8DrfRvVCbybb1kxsxLCAFsTG3"); }
```

Impact:
The attacker may attempt to do a DoS on the service by exhausting the daily quota for the Tron API key.

Recommendation:
Proxy the traffic through your own service and implement their anti-automation controls. Do not hardcode keys into extensions. If any key is required on the user side, get it dynamically with an AJAX request, this facilitates rotation of the key without the requirement to upgrade the extension binary.

False Positive details: It was confirmed by customers that it has a high availability “Pay as you go” plan which can cover a high request rate with their key.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

H03. Vendor-Disapproved Usage of Tron's "Sign" Function

Common Weakness Enumeration	CWE-1240
Status	**False Positive**

Description:
The application code utilizes the TronWeb.Trx.sign routine, which is accompanied by a warning on its page:
https://developers.tron.network/reference/sign

WARNING
Do not use this in any web / user-facing applications. This will expose the private key.

On the other hand, the function signMessageV2 does not carry such warning:
https://developers.tron.network/reference/signmessagev2

PoC.
signers/tron/src/index.ts:

```
async sign(msgHash: string, keyPair: KeyPair): Promise<string> {
 if (msgHash.trim().startsWith("{")) { // is serialized transaction object
const

{data, network} = JSON.parse(msgHash); let sig = null; try { if
(data?.raw_data?.contract[0]?.type === "TransferAssetContract" ||
data?.raw_data?.contract[0]?.type === "TransferContract") { // sign native
|| TRC-10 sig = new TronWeb( network.node, network.node, network.node,
keyPair.privateKey.substring(2)).trx.sign(data,
keyPair.privateKey.substring(2)); }
```

Impact:
During the audit period, we could not ascertain how the `sign` function exposes the private key leaving us with mere speculations about potential side channels or other leakage methods. Considering that the extension is accessible to any script in any frame, we can assume that attackers might attempt to exploit this vulnerability. An exposed private key represents a significant risk, as it could lead to the theft of user assets from the walle.

Recommendation:
To address this issue, it is advisable to seek clarification from Tron developers regarding the risks associated with the `*sign*` function. Consider switching to the use of `*signMessageV2*` exclusively as a safer alternative.

False positive details: The thread modeling provided by the Hacken team clarified the sign function. It is possible to confirm no observable threat can execute weaknesses inside it.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Medium

M01. Too Wide Manifest Permissions

Common Weakness Enumeration	CWE-732
Status	Partially Fixed with 4ee9dedcd640c5a2bc2f7654af1b5ab86e7826e8

Description:
The application manifest supports extension loading on all pages and filesystem files, particularly content_scripts and the framing of the scripts. Here are the specific concerns:

- Unsafe HTTP (unencrypted) is allowed to load the extension. This ultimately enables the attacker to compromise it with a high probability, as it is very simple to inject hacker's scripts into the application and then use them to substitute transaction details and interact with the extension in other ways.
- Filesystem objects are allowed to load the extension. This, at the very least, facilitates phishing attacks when the user is tricked into downloading and opening an HTML file, which then initiates interaction with the extension.
- The use of the MAIN world for the content script carries certain risks. When using the "MAIN" world, the host page can access and interfere with the injected script. By default, it is set to "ISOLATED," which provides a unique execution environment for the content script. An isolated world is a private execution environment that isn't accessible to the page or other extensions. A practical consequence of this isolation is that JavaScript variables in an extension's content scripts are not visible to the host page or other extensions' content scripts. This concept was initially introduced with the launch of Chrome to provide isolation for browser tabs.
- The content script will be injected into all frames, not just the topmost one. If the financial app allows framing (as with DAPP using the wallet), the attacker may utilize clickjacking to interact with it from the owning frame. This makes the DAPP a high-risk target. We do not anticipate that many DAPPs will permit framing.
- Web-accessible resources do not use dynamic URLs. When a resource is listed in web_accessible_resources, it can be accessed using the URL format chrome-extension://<your-extension-id>/<path/to/resource>. In Manifest V3, Chrome can employ a dynamic URL by setting use_dynamic_url to true. The dynamic ID is generated per session and regenerated upon browser restart or extension reload. The risk is minimal for a web-accessible JS script that is not susceptible to clickjacking.
- Non-existent files (*.js.map) are included in the list of web accessible resources.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

PoC. See *manifest.json* file in the extension folder.

near-validator/routes/transfer-assets.ts

```
{ "matches": [ "file://*/*", "http://*/*", "https://*/*" ], "js": [ "scripts/inject.js" ], "run_at": "document_start", "all_frames": true, "world": "MAIN" },
```

Impact:
Taking into account all of the issues mentioned above, an attacker could potentially deceive the user into visiting their malicious page or downloading a harmful file, subsequently enabling them to interact with the extension and Dapp to compromise both.

Recommendation: Implement the following controls:

- Allow only HTTPS sites.
- Propagate the extension only to the top frame.
- Use ISOLATED worlds.
- Use dynamic URLs for web-accessible resources.
- Remove *.js.map from web-accessible resources.

Partially Fixed details: While the Metamask is the most popular wallet, it does not mean that it is the most secure one. It is always the balance of usability, ease of development, and security. As for acknowledged a 0.5 security impact score should be given.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

M02. Mnemonic in Memory Immediately After Unlocking

Common Weakness Enumeration	CWE-316
Status	**Fixed** (commit 485b51399adc7dbe3021b0150e055c3b795b3643)

Description:
Right after unlocking the application, it's possible to find the mnemonic in clear text in the memory. While this does require the attacker to have access to the computer (a very powerful position), it significantly facilitates a full account compromise.

PoC. Unlock the wallet, then use Chrome devtools to dump the heap of the extension. Search in the dump file for the mnemonic string.

Impact:
The mnemonic stored in clear text memory upon unlocking the application exposes a risk. An attacker with computer access can compromise user accounts, potentially leading to unauthorized access and financial loss.

Locking the extension clears the mnemonic from memory, which partially reduces the risk. However, the wallet should provide some resistance even against time-limited unattended access attacks and make it more challenging for attackers (or malware) to identify the mnemonic.

Recommendation: Implement the following best practices:

- Do not store mnemonic in clear text in memory, use some form of obfuscation
- Clear the mnemonic from memory whenever it is not required.

Fix details: We did not find the mnemonics in the heap after the fix.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

M03. Insufficient Origin Check for Tab-originated Messages

Common Weakness Enumeration	CWE-940
Status	**Reported**

Description:
The application utilizes the https://github.com/zikaari/webext-bridge setNamespace mechanism to facilitate communication with the content scripts. For security reasons, if the application wishes to receive or send messages to or from the window context, one of the extension's content scripts must invoke allowWindowMessaging(<namespace: string>) to unlock message routing. The namespace is hardcoded in the extension and is named "539d9437b5301cc13079a517bfe4fa9b661dae9e4b00c5393aef075a8425d561" (//keccak256("scramble")).
Unlike chrome.runtime.sendMessage and chrome.runtime.connect, which require the extension's manifest to specify which sites are allowed to communicate with the extension, webext-bridge lacks such restrictions by design. This means any webpage, whether intended or not, can perform actions like sendMessage(msgId, data, 'background') or something similar, as long as it follows the same protocol used by webext-bridge and uses the same namespace as the extension.

Since the application loads content scripts in each window/tab/frame, and these scripts are executed in the MAIN world, an attacker may attempt to interact with existing objects and send their own communication messages. For example, another extension that injects its script may try to interact with the wallet extension, or a malicious iframe may do the same.

PoC. Due to the time constraints during testing, we did not actively interfere with the existing communications from our injected script. However, we manually modified the scripts being injected, added debugging logs, and were able to observe all web-ext traffic.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Impact:
The absence of origin checking creates a broad attack surface. For instance, if the extension is unlocked and contains a vulnerable function for conducting transactions, an attacker could attempt to trigger it. Furthermore, they may try to manipulate transaction parameters, such as the destination address, in an effort to steal user assets. Due to the lack of a functioning Proof of Concept (PoC) and identified functions that cause harm without a confirmation window, we have assigned a MEDIUM-level severity to this finding.

Recommendation:
Ensure that the background service only accepts messages from the content script to mitigate this security risk.

Reported details: The verification is only based on the namespace string parameter, which is hardcoded in the extension and publically known to the attacker. He can use its own initialization code with the same namespace string and contact the backend of the extension. webext-bridge framework is fundamentally weaker than Chrome's native mechanism. Requested clarification from customer

Low

L01. Vulnerable Dependencies

Common Weakness Enumeration	CWE-1395
Status	Fixed (b9dd4f59da6aa7a0449507922eb409b9f27c2aa3 commit)

Description:
The dependency analyzer tools have identified several weaknesses, with the most severe ones being:

***Semver is vulnerable* to Regular Expression Denial of Service (ReDoS):**
Reference: Github-node-semver Commit

PoC. We were unable to find a way to exploit these issues or the others identified by npm audit and Snyk. However, it is important to note that using vulnerable libraries can negatively impact the application reputation and potentially facilitate attacks involving other vulnerabilities.

Recommendation:
We recommend updating all dependencies with known vulnerabilities to ensure the security and stability of the wallet.

Fix details: After the fix there were no vulnerable dependencies found (except one false positive).

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

L02. Improper Input Validation at Views/Routes

Common Weakness Enumeration	CWE-20
Status	**Acknowledged**

Description:
Input validation is a frequently used technique for safeguarding against potentially harmful inputs, ensuring that they are safe for processing within the code or when interacting with other components. When software lacks proper input validation, an attacker can manipulate the input in a way not anticipated by the rest of the application. This can result in unintended information being passed to various parts of the system, potentially leading to altered control flow, unauthorized resource control, or even arbitrary code execution.

PoC. The extension allows malicious users to inject non-valid data at the following routes:

- /assets/:id
- /nfts/:id
- /verify-transaction/:id
- /swap/:id

Impact:
An attacker could leverage malicious input to modify data or disrupt control flow unexpectedly, potentially leading to attacks including Cross-Site Scripting (XSS) and Server-Side Request Forgery (SSRF) attacks.

Recommendation:
It is important for the applications to validate inputs before rendering components in the front end. Otherwise, Specifically, for each "id" parameter, validation should be carried out using an allowlist (enum) that specifies accepted values, such as "BTC" or "ETH." This approach will effectively prevent malicious users from executing attacks through these routes.

Acknowledged details: this low logic error will not be considered impacted by the app.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

L03. The Use of Weak RC4 and AES-CTR Crypto Algorithms

Common Weakness Enumeration	CWE-327
Status	Fixed (a8bd117dfef467c8f3e90e491ff0db56244e430b commit)

Description:
The application currently utilizes weak crypto algorithms, specifically RC4 and AES-CTR, in the `*packages/utils/src/encrypt.ts*` file.

PoC.

```
 const scryptParams = {
  cipher: "aes-128-ctr",
  kdf: "scrypt",
  dklen: 32,
  n: 262144,
  r: 8,
  p: 1,
};
```

Impact:

- AES-128: This has a 128-bit key, with no known purely cryptanalytic attack better than brute force, and is thus much better than the password.
- CTR mode: With AES's 128-bit block, the primary concern is key-stream reuse, which would require a flawed random number generator.

Recommendation: Implement the following controls:

- Use AES-256 GCM (Galois/Counter Mode) encryption to ensure both confidentiality and integrity of the data. This mode provides a higher level of security compared to AES-128 CTR.

Fix details: The customer has changed AES-128 to AES-GCM.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

L04. Sensitive Data Exposure through Clipboard

Common Weakness Enumeration	CWE-200
Status	**Acknowledged**

Description:
The application's current feature allowing users to copy both the seed phrase and the password to the clipboard poses a significant security risk. Malicious software or malware could actively monitor the clipboard for sensitive information like seed phrases and passwords. If an attacker gains access to both the seed phrase and the password, they can potentially recover the entire wallet, gaining control over the associated funds. It is crucial to implement measures that prevent such vulnerabilities and protect user data.

Impact:
This could lead to leakage of seed phase and password, which could lead to the compromise of the wallet.

Recommendation: Implement the following controls:

- Remove the option to copy the seed phrase to the clipboard to prevent potential exposure of sensitive information.

Acknowledged details: According to the customer other wallets have this finding. Low vulnerabilities have no security score impact.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

L05. Connection Allowed Under a Locked Wallet

Common Weakness Enumeration	CWE-287
Status	Fixed (commit 646fb8a28ddb2411b273b0c6ef8f456cf6e8d1ba)

Description:
The application allows the wallet to be connected even when it is in a locked state.

PoC. Visit *https://metamask.github.io/test-dapp/* when the wallet is locked, and click ‘Connect’ on the web page.

Impact:
An attacker could potentially connect to the Dapp site (e.g., in an unattended browser) and at least view the user's account. While sending money may require wallet unlock, this still presents a security risk. Therefore, we have marked this finding as LOW.

Recommendation: Implement the following controls:

- Do not allow interaction with DAPP pages if the wallet is in a locked state.

Fix details: we were not able to .connect to the DApp if the wallet was locked after the fix

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

L06. UnlimitedStorage Permission in the Extension Manifest

Common Weakness Enumeration	CWE-770
Status	**Acknowledged**

Description:
By default, extensions are subject to standard quota restrictions on storage, which can be assessed by calling `**navigator.storage.estimate()**`. Storage may also be evicted in rare cases of heavy memory pressure. The "unlimitedStorage" permission, affecting both extension and web storage APIs, exempts extensions from these quota restrictions and eviction.

PoC. You can observe the use of “unlimitedStorage” in the manifest file in the extension folder.

Impact:
A vulnerability in the blockchain or interaction with a malicious website could potentially lead to a denial of service on the user's computer. An attacker might attempt to call the extension's storage routines to continuously fill the disk.

Recommendation: To address this issue, we recommend implementing the following control:

- Explicitly notify the user about the extension's use of unlimited storage quota to ensure transparency.
- Consider the user's normal storage quota (without unlimited storage) and utilize navigator.storage.persist() to protect against eviction. This will help safeguard the user's system and prevent potential misuse.

For more details, please refer to the documentation here: Chrome Extensions - Storage and Cookies.

Acknowledged details: Customer just acknowledges this bug.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

L07. Excessive `tabs` permission in the manifest

Common Weakness Enumeration	CWE-732
Status	**Acknowledged**

Description:
The extension currently employs the "tabs" permission in its manifest. This permission grants access to privileged fields of the Tab objects used by various APIs, including Chrome's tabs and chrome.windows. However, it's not necessary to declare this permission to utilize those APIs. Instead, using the "activeTab" permission allows the extension to operate on the currently active tab, but only after user approval on the selected tab.

PoC. You can see the "tabs" permission in the manifest.json file in the extension folder.

Impact:
If the extension becomes compromised, for example, through an XSS attack in its content script, it could potentially affect all sites the user navigates to. On the other hand, the "activeTab" permission has a significantly reduced impact, as it only applies to the small subset of sites the user has recently approved. Furthermore, if the extension is compromised, the attacker would need the user to invoke the extension before gaining access. This access is temporary and only persists while the tab is open or until the user navigates away or closes the tab.

Recommendation:
To enhance security, it is advisable to use the "activeTab" permission. This permission grants the extension temporary access to the currently active tab when the user invokes the extension, such as by clicking its action. Access to the tab remains in effect only while the user is on that page and is automatically revoked when the user navigates away or closes the tab.

For more information, refer to the following resources:
Declare Permissions
activeTab Manifest Permission

Acknowledged details: Low findings do not impact the final score.

L08. Blacklist of Phishing Domains Not In Use

Common Weakness Enumeration	CWE-732
Status	**Reported**

Description:
The application currently lacks an implementation of a blacklist for phishing Dapps. Implementing such a blacklist is a best practice as it serves to increase users' awareness and provides warnings about potential issues with Dapps, ultimately reducing the risks associated with their assets. Additionally, this proactive approach can enhance trust in the extension and attract more active users.

PoC.To demonstrate the importance of blacklisting phishing Dapp domains, you can take any domain from a source like https://github.com/phantom/blocklist/blob/master/blocklist.yaml, add it to the /etc/hosts file, and point it to “127.0.0.1”. Then, host any Dapp on that domain, such as the Metamask test Dapp, and attempt to connect to it.

Impact:
The absence of phishing Dapp domain blacklisting, coupled with wide access permissions, creates a vulnerability that facilitates hacking attacks against assets held by the extension wallet.

Recommendation: We recommend implementing the following controls:

- Implement a phishing blacklist for known malicious Dapps, using resources such as https://github.com/phantom/blocklist/blob/master/blocklist.yaml.
- Provide users with an "ultra-secure" mode that ensures the wallet is only loaded on a set of whitelisted, known-good Dapps. This can further enhance security and user trust in the extension.

Reported details: The customer was informed that this has not been fixed.

Informational

I01. Statically Hardcoded Package

Common Weakness Enumeration	None
Status	**Acknowledged**

Description:
The extension relies heavily on the webext-bridge library (https://github.com/antfu/webext-bridge) for message passing. While the specific commit from which it was forked is not identified, it appears that Enkrypt integrated it on February 12, 2022 (commit: https://github.com/enkryptcom/enKrypt/commit/986da7608b6638d749ec0bb7bdd0d2d31a682f4b).

Impact:
Although there is no direct security impact associated with this reliance on webext-bridge, maintaining a separate branch of packages can increase the risk that critical operational or security issues may not be promptly addressed and fixed.

Recommendation:
To mitigate potential risks and ensure timely updates, it is advisable to use the webext-bridge package as an external dependency rather than maintaining a separate branch. This approach can help maintain the security and functionality of the extension more effectively.

Acknowledged details: informational findings do not impact the security score.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

I02. Sentry Reporting May Violate User Privacy

Common Weakness Enumeration	None
Status	**Reported**

Description:
The application uses Sentry reporting in its packages/extension folder:

```
packages/extension/src/libs/sentry/sentry.ts:import * as Sentry from
"@sentry/vue"; packages/extension/src/libs/sentry/sentry.ts:import { Vue } from
"@sentry/vue/types/types"; packages/extension/src/libs/sentry/sentry.ts:export
const setupSentryVue = (app: Vue | Vue[], router: Router) => {
packages/extension/src/libs/sentry/sentry.ts: Sentry.init({
packages/extension/src/libs/sentry/sentry.ts: dsn:
"https://f48f98678273128f88402c4bd52b996e@o4505818645725184.ingest.sentry.io/4505
818648477696", packages/extension/src/libs/sentry/sentry.ts: new
Sentry.BrowserTracing({ packages/extension/src/libs/sentry/sentry.ts:
routingInstrumentation: Sentry.vueRouterInstrumentation(router),
packages/extension/src/libs/sentry/sentry.ts: new Sentry.Replay(),
packages/extension/src/libs/sentry/sentry.ts: new
Sentry.BrowserProfilingIntegration(),
```

We can see it in the proxy as well:

```
POST
/api/4505818648477696/envelope/?sentry_key=f48f98678273128f88402c4bd52b996e&sentr
y_version=7&sentry_client=sentry.javascript.vue%2F7.72.0 HTTP/2
Host: o4505818645725184.ingest.sentry.io
Content-Length: 1416
Sec-Ch-Ua: "Chromium";v="118", "Google Chrome";v="118", "Not=A?Brand";v="99"
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/118.0.0.0 Safari/537.36
Sec-Ch-Ua-Platform: "Linux"
Accept: */*
Origin: chrome-extension://kkpahaemdogpgjgcmjaeoggglmgoinci
Sec-Fetch-Site: cross-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io


```
{"event_id":"87cb56a83d3346f6a0943faab6624a0c","sent_at":"2023-10-24T14:44:25.337
Z","sdk":{"name":"sentry.javascript.vue","version":"7.72.0"}}
{"type":"replay_event"}
{"type":"replay_event","replay_start_timestamp":1698158628.949,"timestamp":169815
8665.299,"error_ids":[],"trace_ids":[],"urls":[],"replay_id":"87cb56a83d3346f6a09
43faab6624a0c","segment_id":5,"replay_type":"session","request":{"url":"chrome-ex
tension://kkpahaemdogpgjgcmjaeoggglmgoinci/action.html#/locked","headers":{"User-
Agent":"Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/118.0.0.0
Safari/537.36"}},"event_id":"87cb56a83d3346f6a0943faab6624a0c","environment":"pro
duction","sdk":{"integrations":["InboundFilters","FunctionToString","TryCatch","B
readcrumbs","GlobalHandlers","LinkedErrors","Dedupe","HttpContext","BrowserTracin
g","Replay","BrowserProfilingIntegration"],"name":"sentry.javascript.vue","versio
n":"7.72.0"},"platform":"javascript"}
{"type":"replay_recording","length":426}
{"segment_id":5}
xœl »Â0Eÿå ¨CKé#Y™ø†ªC *µI ºåßq0áÉÇºü\  º.`
è^Íïá&§²Óƒ7èa,`˜ÃtÝ™>hñ÷7D¬& ã‹…>-â§déU ²ËÚ¿'ÁÜ>ÏBÞRnþ½ {ÏÞ±X
q Á7ö¨Ò˜D1-´±YVèªU}Õôm£:Õ¥ÿÿÒ¡  Lhã#Sœ>23ùÿÿ¢¡ Liã#3Ü>24·´¨ÿÿ¢¡ Ìhã#sÜ>2Åÿÿ¢¡ Ìiã#
Ü>2Åÿÿ¢¡ ,hã#KÜ>2ùÿÿ|SIÂ0ü
@!Jªü!náVcPP³(      ‡òz¦R
=>Y™qÆ3’ ß ôw’ ÞË¾û˜tƒBâ| ÙSØ$
K4 ‘ìL‰ßyØÍËeÈ.UÃö2¹^~ëq° Ðu›å;t÷rdJÆ=ùì¼C
Jöªú %|ÅJãÉ¬Ä"d#¨F° Âv‹´ j¤P¸UÔ
ÿÿ‹î@áþ
```


Impact:
The impact of the current situation is that there is no immediate security risk associated with the monitoring activities, but it is considered a best practice to inform users about such monitoring and provide them with the option to disable it.

Recommendation: To adhere to best practices and respect user preferences, it is recommended to incorporate the following features into the browser extension:

- Notify the user about monitoring activities: Clearly inform users about any monitoring or data collection activities carried out by the extension.
- Allow the user to turn them off: Provide users with the option to disable monitoring or data collection activities if they choose to do so.

For users within the European Union (EU), it's particularly important to consider compliance with the General Data Protection Regulation (GDPR) when implementing monitoring or data collection features. Ensure that any monitoring activities are GDPR-compliant and respect user privacy and data protection rights.

Reported details: The hacken team could not find this privacy policy. Customer promised that they will amend it to explicitly mention Sentry and other similar tools.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

I03. Wasm Unsafe Eval In The Manifest

Common Weakness Enumeration	None
Status	**Acknowledged**

Description:
The application includes 'wasm-unsafe-eval' in its Content Security Policy (CSP) within the manifest, most likely due to a requirement from the Polkadot project (@polkadot/wasm-crypto": "^7.2.2"). This configuration allows the execution of WebAssembly (wasm) bytecode.

PoC. See manifest.json file in the extension folder.

```
src/manifest/manifest-chrome.json: "extension_pages": "script-src 'self' 'wasm-unsafe-eval'; object-src 'self'"
```

Impact:
In general, executing WebAssembly (wasm) code is considered safe, and the WebAssembly platform technology includes numerous security controls. Therefore, the inclusion of 'wasm-unsafe-eval' is primarily informational and does not pose a significant security risk.

Recommendation:
It is advisable to review whether the inclusion of wasm modules is indeed necessary for the project's functionality. If wasm modules are found to be necessary, the current configuration can be considered acceptable given the security controls in WebAssembly.

Acknowledged details: informational findings do not impact the security score.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

I06. Unfixed Dependency Version

Common Weakness Enumeration	CWE-758
Status	**Acknowledged**

Description:
The application package.json files do not specify fixed dependency versions in some instances. The use of the caret range (^) in versioning allows for changes that do not affect the left-most non-zero element in the [major, minor, patch] tuple. This range permits adjustments presumed to be additive and non-breaking, based on common practices.

PoC.

```
find ./ -iname '*package.json*' | xargs grep '\^' |head
./types/package.json: "@types/node": "^20.6.0",
./types/package.json: "@typescript-eslint/eslint-plugin": "^5.62.0",
./types/package.json: "@typescript-eslint/parser": "^5.62.0",

Approx 1k lines of code with
```

Impact:
The presence of non-fixed dependencies does not result in an immediate security impact. However, it introduces a potential risk where future updates could inadvertently break the security or other essential properties of the software. On the positive side, non-fixed dependencies reduce the manual effort required to update dependency versions.

Recommendation:
Consider implementing features that notify users about monitoring activities and allow them to opt out. This step ensures compliance with GDPR regulations for users within the EU, enhancing transparency and user privacy.

Acknowledged details: informational findings do not impact the security score.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

I08. Incomplete Test Coverage of Cryptography Code

Common Weakness Enumeration	CWE-1240
Status	Reported

Description:
The application's code lacks full test coverage for its cryptography-related components. Specifically, the Tron signer code does not include testing for the msgHash that starts with "{...".

The "hw-ledger" package contains only a basic example test that checks whether (1+2) == 3, which is insufficient for comprehensive test coverage.

PoC.

```
signers/tron/src/index.ts:

async sign(msgHash: string, keyPair: KeyPair): Promise<string> {
 if (msgHash.trim().startsWith("{")) { // is serialized transaction object
const

{data, network} = JSON.parse(msgHash); let sig = null; try { if
(data?.raw_data?.contract[0]?.type === "TransferAssetContract" ||
data?.raw_data?.contract[0]?.type === "TransferContract") { // sign native
|| TRC-10 sig = new TronWeb( network.node, network.node, network.node,
keyPair.privateKey.substring(2)).trx.sign(data,
keyPair.privateKey.substring(2)); }


signers/tron/tests/sign.test.ts:
describe("Tron signing", () => { const echash =
"82ff40c0a986c6a5cfad4ddf4c3aa6996f1a7837f9c398e17e5de5cbd5a12b28"; const
ecprivkey =
"3c9229289a6125f7fdf1885a77bb12c37a8d3b4962d936f7e3084dece32a3ca1"; const
ecpair = { publicKey:
bufferToHex(privateToPublic(hexToBuffer(ecprivkey))), privateKey:
ecprivkey, }; it("it should sign correctly", async () => { const
tronSigner = new Signer(); const signature = await tronSigner.sign(echash,
ecpair); expect(signature).equals(
"0x92b817b813cc8b71bf929097d63326e39e686b9216b8658b31c4c782ef219534614cc13
7aacc0d427a2e9227edcc0e1018a156803918e7eb103ea77b54229d5b1b" ); }); });

hw-wallets/tests/example.test.ts:
```

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

```
import { expect } from "chai"; describe("Simple addition", () => { // the tests container it("it should properly add", async () => { expect(1 + 2).to.be.equals(3); }); })
```

Impact:
While there may not be an immediate security impact, it is imperative to ensure that all signing paths are thoroughly covered with unit tests. This includes testing both positive and negative (malicious) inputs to validate the robustness of the implementation.

Recommendation:
To address this issue, implement comprehensive unit tests that cover all possible paths in the critical signing code. These tests should encompass valid and invalid data to ensure the reliability and security of the implementation.

Reported details: The Hacken team was not able to see the unit test that covers all tron functionality. Customer informed that this was due to the Tron package volatility.

I09. Reliance on Unaudited hw ledger Library

Common Weakness Enumeration	CWE-1240
Status	**Acknowledged**

Description:
The application's code relies on the "https://github.com/Zondax/ledger-substrate-js" library, which lacks readily identifiable audit reports. Furthermore, the vendor's site explicitly mentions that some routines within the library have not been audited. Specifically, the package provides a client library for communicating with Substrate Apps on Ledger Nano S/S+/X devices and includes an "hd_key_derivation" function, which is not audited and depends on external packages.

PoC.
src/ledger/substrate/substrateApps.ts

```
import type { SubstrateApp } from "@zondax/ledger-substrate";
```

Impact:
While there may not be an immediate security impact, the utilization of libraries without a robust audit record poses a potential risk by introducing a vulnerable dependency into the project. This could ultimately facilitate further attacks on user assets.

Recommendation:
To mitigate this risk, it is advisable to conduct a comprehensive security audit of the Zondaz/ledger-substrate library to ensure its reliability and safety within the project.

Acknowledged details: informational findings do not impact the security score.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

I10. Support for Outdated Browsers

Common Weakness Enumeration	CWE-1240
Status	**Acknowledged**

Description:
The application manifest file indicates support for relatively old browsers, such as Chrome 67 from 2018.

PoC. Review the manifest.json file in the extension folder.

```
   "minimum_chrome_version": "97",
And package.json in the extension folder:
"browserslist": { "production": [ "chrome >= 67", "edge >= 79", "firefox
>= 68", "opera >= 54", "safari >= 14" ],
```

Impact:
While it is the user's responsibility to maintain an up-to-date browser version. It is considered a good security practice to make users more aware of the risks associated with using outdated browsers. Older browser versions often contain security vulnerabilities and create an unsafe environment for financial applications, such as the target wallet. It's worth noting that the minimum supported Chrome version for the extension is 97, released in January 2022, making it more than a year old.

Recommendation:
Implement a warning mechanism to alert users about using outdated browser versions. This helps users understand the associated security risk and encourages them to update to more secure browser versions.

Acknowledged details: informational findings do not impact the security score.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

I10. Lack of Exception Handling in Signature Verification

Common Weakness Enumeration	CWE-703
Status	Reported

Description:
The application fails to check for exceptional situations when using the `*ethereumjs-util.ecrecover*` routine for signature verification.

PoC.
Although the `ecrecover` routine source code include at least one exception in:
https://github.com/ethereumjs/ethereumjs-util/blob/master/src/signature.ts#L37

```
export const ecrecover = function(
  msgHash: Buffer,
  v: BNLike,
  r: Buffer,
  s: Buffer,
  chainId?: BNLike
): Buffer {
   const signature = Buffer.concat([setLengthLeft(r, 32), setLengthLeft(s, 32)], 64)
  const recovery = calculateSigRecovery(v, chainId)
  if (!isValidSigRecovery(recovery)) {
    throw new Error('Invalid signature v value')
  }
    const senderPubKey = ecdsaRecover(signature, recovery.toNumber(), msgHash)
  return Buffer.from(publicKeyConvert(senderPubKey, false).slice(1))
}
```

But the application ignores it:
signers/ethereum/src/index.ts:

```
async verify( msgHash: string, sig: string, publicKey: string ): Promise<boolean> { const sigdecoded = fromRpcSig(sig); const rpubkey = ecrecover( hexToBuffer(msgHash), sigdecoded.v, sigdecoded.r, sigdecoded.s ); return bufferToHex(rpubkey) === publicKey; }
```

And similarly in the signers/tron/src/index.ts:

```
async verify( msgHash: string, sig: string, publicKey: string ): Promise<boolean> { const sigdecoded = fromRpcSig(sig); const rpubkey = ecrecover( hexToBuffer(msgHash), sigdecoded.v, sigdecoded.r, sigdecoded.s
```

```
); return bufferToHex(rpubkey) === publicKey; }
```

Impact:
The security impact depends on the specific code that calls the routine. Improper exception handling may potentially allow the bypassing of critical security controls, leading to the compromise of user assets.

Recommendation:
It is advisable to catch exceptions generated by `*ecrecover*` and other underlying routines from dependencies. Log these exceptions and, particularly for verification routines, return a false result when exceptions are encountered.
You can find examples and guidance on how to handle exceptions in the following link:
https://snyk.io/advisor/npm-package/ethereumjs-util/functions/ethereumjs-util.ecrecover

Reported details: The customer team informed us that this one is not fixed at the moment. Also, consider this finding valid.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

I11. Requests to Subdomains of the Main RPC

Common Weakness Enumeration	CWE-941
Status	**Reported**

Description:
The application makes attempts to contact all subdomains of the custom network's RPC endpoint.

PoC. Simply observe the traffic in the proxy when adding a custom network.

596	https://rpc.sepoli	POST	/
597	https://rpc.sepolia	POST	/
598	https://rpc.sepolia.o	POST	/
599	https://rpc.sepolia.or	POST	/
600	https://rpc.sepolia.org	POST	/

```
POST / HTTP/1.1
Host: rpc.sepoli
Content-Length: 74
Sec-Ch-Ua: "Chromium";v="118", "Google Chrome";v="118", "Not=A?Brand";v="99"
Sec-Ch-Ua-Platform: "Linux"
Sec-Ch-Ua-Mobile: ?0
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/118.0.0.0 Safari/537.36
Content-Type: application/json
Accept: */*
Origin: chrome-extension://kkpahaemdogpgjgcmjaeoggglmgoinci
Sec-Fetch-Site: cross-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

{"jsonrpc":"2.0","id":6280578101200454,"method":"eth_chainId","params":[]
```

Impact:
The attacker could potentially set his own domain and impersonate a node of the user. This could significantly impact the security of the wallet, provide a manipulated view of the blockchain, and compromise user assets.

Recommendation:
Restrict connections to the address specified by the user and exclusively use HTTPS endpoints to enhance security and mitigate the risk.

Reported details: This finding was moved from medium to informational. The Customer should exercise awareness among its user base to they always verify the correctness of RPC endpoints. Security impact after reevaluation is "0".

I12. Use of Hard-coded Credentials at Ethereum Libs and Bitcoin tests

Common Weakness Enumeration	CWE-798
Status	**Reported**

Description:
The product contains hard-coded credentials, including passwords and cryptographic keys, which are utilized for its inbound authentication, outbound communication with external components, or encryption of internal data.

PoC. The wallet stores API key in the following files:
/src/providers/ethereum/libs/activity-handlers/providers/okc/index.ts
/src/providers/bitcoin/tests/bitcoin.address.derivation.mocha.ts

Impact:
The use of hard-coded passwords or keys greatly increases the likelihood of malicious users gaining access to the associated account.

Recommendation: Implement the following controls:

- Store the API keys in the backend, and make requests through the backend when necessary.
- If there's a need to use API keys in the frontend, ensure they are encrypted and securely managed to prevent unauthorized access.

Reported details: This finding was moved from Low to Informational. We found that as per OKex documentation the key is linked to the IP, and non-IP linked kei is expiring (https://www.okx.com/docs-v5/en/#overview-v5-api-key-creation). The BTC keys are public.

Out Of Scope

In alignment with Scramble's feedback and collaborative decision-making process, it has been determined that issues like L09, I04, I05, and I07 have been deliberately excluded from the primary report. This decision stems from the recognition that these issues fall under the category of "out of scope." Significantly, this designation has been assigned due to the successful integration of comprehensive fixes tailored to address these specific concerns, and these remediations have been efficiently incorporated into the Beta version of the system. This strategic approach not only ensures a focused and streamlined primary report but also underscores the proactive steps taken to fortify the Beta version against the identified issues.

L09. Absence of Warning for Adding/Changing the Chain

Common Weakness Enumeration	CWE-223
Status	**Fixed in Beta**

Description:
The application does not provide users with a warning when it adds or changes a blockchain network. Users can only discover these changes when they click on the extension icon in the browser.

PoC. To illustrate this issue, one can use any Dapp and switch it to another blockchain network or add a new blockchain.

Impact:
The absence of a warning when adding or changing a blockchain network can potentially facilitate phishing-style attacks. Users might be tricked into spending assets on a network different from what they intended.

Recommendation:
To enhance security and prevent unintended actions, it is advisable to implement a warning system that alerts the user when there is a change in the blockchain network, such as switching to a different chain or adding a new one. This will help users make informed decisions and reduce the risk of accidental actions on the wrong blockchain.

Reported details: We were using the chainlist.org site to connect multiple chains, and we saw the informational pop-up only for the first chain (we tried Eth Mainnet, Goerli, Sepolia).

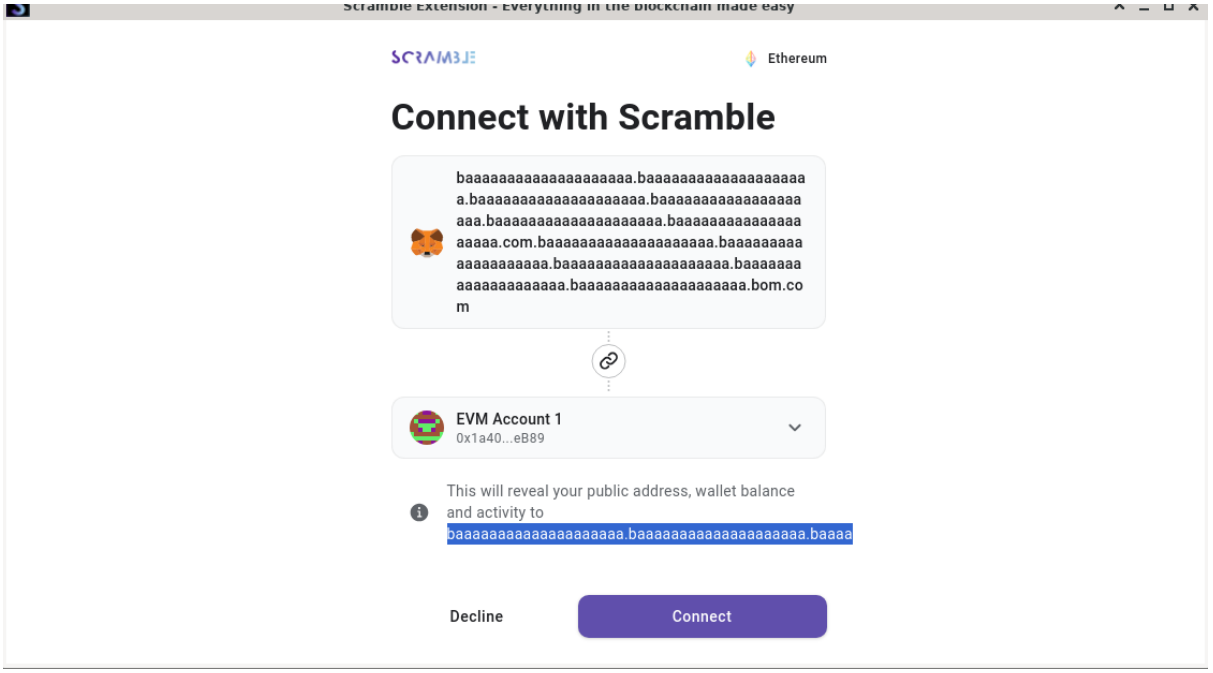
Client's Remark: The fixes for these issue have been successfully integrated into the Beta version.

I04. The long Hostname is Truncated in the Popup Window

Common Weakness Enumeration	CWE-451
Status	**Fixed in Beta**

Description:
The application's user interface truncates long hostnames in the warning message on the Connect string. This truncation can lead to unintended consequences as it reveals the public address, wallet balance, and activity to a potentially incorrect or misleading hostname.

PoC. To demonstrate this issue, create a long hostname in the /etc/hosts file or manipulate DNS resolution using a tool like Burp as a proxy. Interacting with this manipulated hostname will reveal the truncated version in the warning message.



Impact:
There is no immediate security impact as the top frame of the popup correctly displays the host. However this is a sign of poor UI testing.

Recommendation:
To ensure the accuracy of the warning message and prevent potential confusion or misrepresentation, it's advisable to address this truncation issue and display the complete hostname in the warning message. This can help users make informed decisions and avoid unexpected risks when interacting with the application.

Reported details: The Hacken team rechecked it, and it was possible to see hostname truncation. The code just cuts the hostname to 40 characters and does not produce a warning.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Client's Remark: The fixes for this issue have been successfully integrated into the Beta version.

I05. Mewapi Reporting May Violate User Privacy

Common Weakness Enumeration	None
Status	**Fixed in Beta**

Description:
The application employs Mewapi reporting when adding a new blockchain network. This reporting can send information to external servers without notifying the user, potentially impacting user privacy.

PoC. The application uses Mewapi reporting when adding a new chain:
src/libs/metrics/index.ts: fetch("https://partners.mewapi.io/enkrypt-metrics", {
We can see it in the proxy as well:

```
POST /enkrypt-metrics HTTP/2
Host: partners.mewapi.io
Content-Length: 67
Accept: application/json
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/118.0.0.0 Safari/537.36
Content-Type: application/json
Origin: chrome-extension://kkpahaemdogpgjgcmjaeoggglmgoinci
Sec-Fetch-Site: cross-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

{"method":"enkrypt_requests","params":[{"ethereum":{"SEPOLIA":1}}]}
```

Impact:
While there is no immediate security impact, it is advisable to inform users about this data reporting and offer the option to disable it for improved transparency.

Recommendation: Add the following features to the browser extension:

- Notify the user about monitoring activities
- Allow the user to turn them off

For users in the EU consider using a GDPR-compliant Mewapi setup.

Reported details: The hacken team could not find this privacy policy. Customer promised that they will amend it to explicitly mention Sentry and other similar tools.

Client's Remark: The fixes for this issue have been successfully integrated into the Beta version.

I07. Unsafe Mnemonic Handling

Common Weakness Enumeration	CWE-1287
Status	**Fixed in Beta**

Description:
The application is in violation of BIP39 Node.js package recommendations. It does not conduct checks for the mnemonic checksum or the number of words supplied before validation. Instead, it directly calls the package routine. The package's documentation explicitly states that the mnemonic format should be verified before being passed to validation or generation functions. It's important to allow for recovery from mnemonic phrases with invalid checksums (or from wordlists that aren't available). When an invalid checksum is detected, it's advisable to warn the user that the phrase may not have been generated by your app and inquire whether they would like to proceed anyway. This approach enables your app to support phrases from other apps in different languages without having to store all wordlists.
Furthermore, additional checks should be implemented to ensure that the user inputs at least 12 words separated by spaces, like this: phrase.trim().split(/\s+/g).length >= 12.

PoC.

```
signers/ethereum/src/index.ts
async generate(mnemonic: string, derivationPath = ""): Promise<KeyPair> {
const seed = await mnemonicToSeed(mnemonic);
```

Impact:
While there may not be an immediate security impact, this issue significantly impacts the overall user experience of the package.

Recommendation:
Kindly verify the mnemonic format before proceeding with any further actions. In the event of checksum errors, provide a warning to the user. For reference and guidance, please visit: https://www.npmjs.com/package/bip39

Reported details: Customer informed that this one is not resolved. Also, consider this as a valid finding.

Client's Remark: The fixes for this issue have been successfully integrated into the Beta version.

Hacken OÜ
Parda 4, Kesklinn, Tallinn,
10151 Harju Maakond, Eesti,
Kesklinna, Estonia
support@hacken.io

Disclaimers

Hacken Disclaimer

The application given for the audit has been analyzed based on the best industry practices at the time of this report, in relation to cybersecurity vulnerabilities and issues in the application's source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The audit makes no statements or warranties on the security of the code. It also cannot be considered a sufficient assessment regarding the utility and safety of the code, bug-free status, or any other contract statements. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of the decentralized application.

Technical Disclaimer

Decentralized applications are closely integrated with a blockchain platform. The platform, its programming language, and other software related to the application can have vulnerabilities that can lead to hacks. Thus, the audit cannot guarantee the explicit security of the audited application.