Wallet Application Security Audit Report

Table Of Contents

1 Executive Summary

On 2023.03.15, the SlowMist security team received the Hana team's security audit application for Hana wallet iOS, 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/grey box lead, white box assists" 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 wallet application 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 wallet application is manually analyzed to look for any potential issues.

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

NO.	Audit Items	Result
1	App runtime environment detection	Fixed
2	Code decompilation detection	Confirmed
3	App permissions detection	Passed
4	File storage security audit	Passed
5	Communication encryption security audit	Passed
6	Interface security audit	Passed
7	Business security audit	Passed
8	WebKit security audit	Passed
9	App cache security audit	Passed
10	WebView DOM security audit	Fixed
11	SQLite storage security audit	Passed
12	Deeplinks security audit	Passed
13	Client-Based Authentication Security audit	Fixed
14	Signature security audit	Fixed
15	Deposit/Transfer security audit	Passed
16	Transaction broadcast security audit	Passed

NO.	Audit Items	Result
17	Secret key generation security audit	Fixed
18	Secret key storage security audit	Fixed
19	Secret key usage security audit	Passed
20	Secret key backup security audit	Passed
21	Secret key destruction security audit	Passed
22	Screenshot/screen recording detection	Confirmed
23	Paste copy detection	Fixed
24	Keyboard keystroke cache detection	Passed
25	Background obfuscation detection	Fixed
26	Suspend evoke security audit	Passed
27	AML anti-money laundering security policy detection	Passed
28	Others	Fixed
29	User interaction security	Confirmed

3 Project Overview

3.1 Project Introduction

Audit Version

https://github.com/Hana-Technology/hana-app

commit: 2c53c6d260cf46e821e4b5afe8af03303cad5696

hana-app.ipa v1.0.38 (sha256: 971fd14a5e865685581b3a6edd035f8b59bba623e04c748efdcdf325d21b0509)

Fixed Version

https://github.com/Hana-Technology/hana-app

commit: a45f4f2fd881e376d6811d4ac585c61363bcc3c4

hana-app.ipa v1.0.45 (38) (sha256:54d43ce623cfd33f8c8cafd95f3b4801b49d01b5de33522d0455cf6d47178a3c)

3.2 Vulnerability Information

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

NO	Title	Category	Level	Status
N1	Runtime environment detection issues	App runtime environment detection	Low	Fixed
N2	Decompilation security issues	Code decompilation detection	Suggestion	Confirmed
N3	Missing screenshot/screen recording detection	Screenshot/screen recording detection	Suggestion	Confirmed
N4	Lack of security reminders	Paste copy detection	Suggestion	Fixed
N5	Background obfuscation issue	Background obfuscation detection	Suggestion	Fixed
N6	URL validation can be bypassed	WebView DOM security audit	Low	Fixed
N7	Client-Based Authentication issue	Client-Based Authentication Security audit	Low	Fixed
N8	Log leak mnemonic and password	Secret key generation security audit	High	Fixed
N9	Secret key backup issue	Secret key storage security audit	Medium	Fixed
N10	Blind signing lacks security reminder	Signature security audit	Low	Fixed
N11	The wallet address is not fully displayed	Others	Suggestion	Fixed
N12	User interaction issue	User interaction security	Suggestion	Confirmed

3.3 Vulnerability Summary

[N1] [Low] Runtime environment detection issues

Category: App runtime environment detection

Content

Wallet App lacks security alerts for jailbreak detection.

Solution

It is recommended to add an iOS device jailbreak detection and reminder scheme.

Status

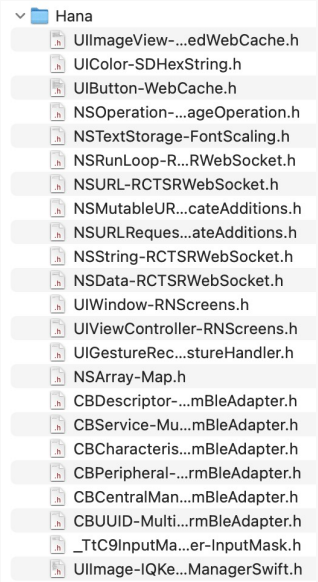
Fixed

[N2] [Suggestion] Decompilation security issues

Category: Code decompilation detection

Content

By dumping the ipa package, you can get the header file without code obfuscation of the header file.



Solution

It is recommended to obfuscate header files.

Status

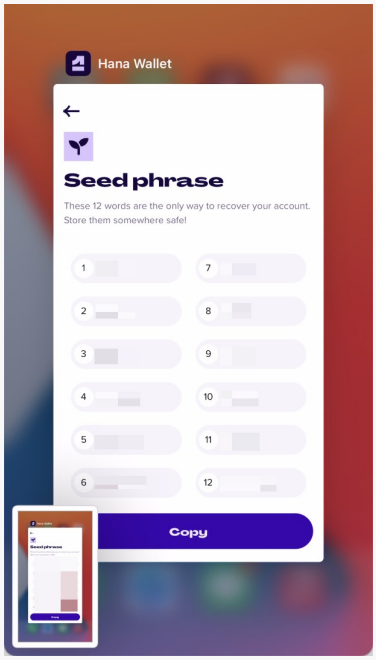
Confirmed

[N3] [Suggestion] Missing screenshot/screen recording detection

Category: Screenshot/screen recording detection

Content

The APP does not have reminders for screenshots, and there are no restrictions on users taking screenshots and recordings.



Solution

It is recommended to add screenshot/screen recording detection and prohibit screenshot/screen recording.

Status

Confirmed

[N4] [Suggestion] Lack of security reminders

Category: Paste copy detection

Content

When exporting wallets, users are allowed to copy mnemonic phrases and the app lacks security reminders, which may be subject to clipboard hijacking attacks.

Solution

Status

Fixed; It is recommended to remind users that they should record by transcribing instead of directly using the

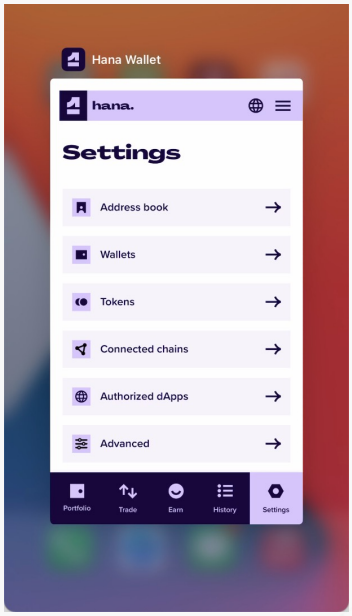
clipboard for copying.

[N5] [Suggestion] Background obfuscation issue

Category: Background obfuscation detection

Content

App UI is not obfuscation when the app is in the background.If the wallet is being exported, the mnemonic phrase may be leaked.



Solution

It is recommended to add an obfuscation mechanism to avoid sensitive data leakage.

Status

Fixed

[N6] [Low] URL validation can be bypassed

Category: WebView DOM security audit

Content

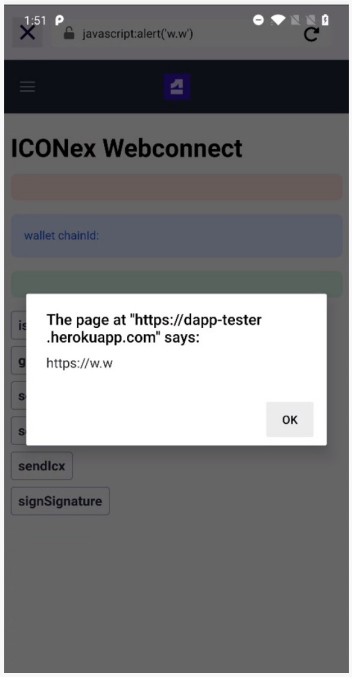
The URL verification is not perfect enough, so it can be accessed through WebView such as "javascript:alert('https://w.w')", "javascript://www.x.com/%0aalert(1)", resulting in abnormal Expected code execution.

- src/screens/browser/Browser.tsx#L334-346

```
function handleChangeUrl() {
  setIsKeyboardOpen(false);
  const hasProtocol = urlInput.includes('://');
  const isUrl = hasProtocol || urlInput.includes('.');

  const uri = isUrl
    ? `${!hasProtocol ? 'https://' : ''}${urlInput}`
    : `https://duckduckgo.com/?q=${urlInput}`;

  setUri(uri);
  setUrlInput(vanityUrl(uri));
  setHttpsUrl(isHttps(uri));
}
```

- src/screens/browser/ICXDApps.tsx#L52-61

```
function handleChangeUrl() {
  const hasProtocol = urlInput.includes('://');
  const isUrl = hasProtocol || urlInput.includes('.');

  const uri = isUrl
    ? `${!hasProtocol ? 'https://' : ''}${urlInput}`
    : `https://duckduckgo.com/?q=${urlInput}`;

  browseTo(uri);
}
```



Solution

It is recommended to enhance the URL matching rules to allow only HTTPS protocol access.

Status

Fixed

[N7] [Low] Client-Based Authentication issue

Category: Client-Based Authentication Security audit

Content

The wallet application is suspended in the background of the phone for a period of time, and the wallet will not be locked.

Solution

It is recommended that after the wallet is suspended in the background for a period of time, re-authentication is required to re-enter the wallet.

Status

Fixed

[N8] [High] Log leak mnemonic and password

Category: Secret key generation security audit

Content

Note: This issue was discovered during development/build and does not exist in the live version.

When the wallet is creating a password, creating a mnemonic, and changing the password, the password and mnemonic will be output in the log, which will be read by other apps.

- src/screens/onboarding/CreatePasscode.new.tsx#L128-129

```
const handleContinue = async (passcode: string, confirmPasscode: string) => {
    console.log('passcode new', passcode);
    console.log('confirmPasscode new', confirmPasscode);
```

- src/screens/onboarding/CreatePasscode.new.tsx#L155-156

```
const seedPhrase = await createVault({ passcode });
console.log('seedPhrase', seedPhrase);
```

- src/screens/onboarding/CreatePasscode.new.tsx#L187-188

```
  console.log('passcode', passcode);
  console.log('confirmPasscode', confirmPasscode);
```

Solution

It is recommended that sensitive information such as passwords and mnemonics be prohibited from being output to the log.

Status

Fixed

[N9] [Medium] Secret key backup issue

Category: Secret key storage security audit

Content

The pbkdf2 hash is stored locally and the wallet has all the ingredients to unlock the secret.

- src/stores/Vault.ts#L133-146

```
async function verifyCredentials(credentials: AuthenticateCredentials) {
  if (credentials.useBiometrics) {
    return true;
  }

  const { encryptionKey } = useInternalState.getState();

  const hashedPasscode = await hash(credentials.passcode!, encryptionKey!);
  const storedPasscode = await SecureStore.getItemAsync(
    STORAGE_KEY.PASSCODE_HASH,
    SecureStoreOptions
  );
  return hashedPasscode === storedPasscode;
}
```

- src/stores/Vault.ts#L608-619

```
async function createVault(credentials: CreateVaultCredentials) {
  const { hasVault } = useVault.getState();

  if (hasVault()) {
    throw new Error('You already have a vault.');
  }

  const encryptionKey = await generateRandomKey();
  await SecureStore.setItemAsync(STORAGE_KEY.ENCRYPTION_KEY, encryptionKey,
SecureStoreOptions);

  const hashedPasscode = await hash(credentials.passcode, encryptionKey);
  await SecureStore.setItemAsync(STORAGE_KEY.PASSCODE_HASH, hashedPasscode,
SecureStoreOptions);
```

Solution

It is recommended to modify the authentication method of unlocking the wallet, and perform authentication and decryption through the pbkdf2 hash input by the user.

Status

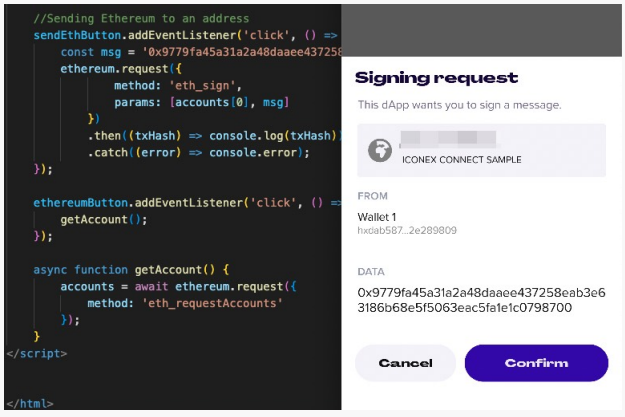
Fixed

[N10] [Low] Blind signing lacks security reminder

Category: Signature security audit

Content

When using eth_sign for the blind signature test, the wallet does not provide a security reminder for the blind signature, and the user may be at risk of being phished.



Solution

It is recommended to detect blind signing and add security reminders.

Status

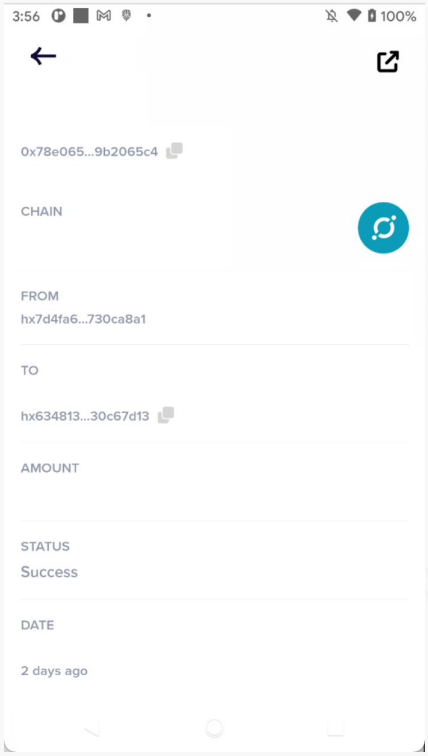
Fixed

[N11] [Suggestion] The wallet address is not fully displayed

Category: Others

Content

The wallet does not fully display the transaction address, and users need to enter the blockchain browser to view the transaction details to know the complete transaction address.This can easily be used for phishing using similar addresses.



Solution

It is recommended that the wallet provide the function of fully displaying the transaction address to avoid phishing to deceive the transaction address.

Status

Fixed

[N12] [Suggestion] User interaction issue

Category: User interaction security

Content

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.
Contact whitelisting	•	The contact whitelisting is not supported.
Password complexity requirements	✓	The password meets the complexity requirements.

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

Solution

It is recommended to enhance user interaction security.

Status

Confirmed

4 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002303310003	SlowMist Security Team	2023.03.15 - 2023.03.31	Passed

Summary conclusion: The SlowMist security team use a manual and SlowMist team's analysis tool to audit the project, during the audit work we found 1 high risk, 1 medium risk, 4 low risks, and 6 suggestion vulnerabilities. And 1 high risk, 1 medium risk, 4 low risks, 3 suggestion vulnerabilities were confirmed and being fixed; We extend our gratitude for Hana Wallet team recognition of SlowMist and hard work and support of relevant staff.

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