Wallet Application Security Audit Report

Table Of Contents

1 Executive Summary

On 2023.05.16, the SlowMist security team received the OKX team's security audit application for OKX Web3 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	Passed
2	Code decompilation detection	Some Risks
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	Passed
11	SQLite storage security audit	Passed
12	Deeplinks security audit	Passed
13	Client-Based Authentication Security audit	Passed
14	Signature security audit	Passed
15	Deposit/Transfer security audit	Passed

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

3 Project Overview

3.1 Project Introduction

Audit Version

App Name: okx-iOS.ipa

App Link: https://apps.apple.com/us/app/okx-buy-bitcoin-eth-crypto/id1327268470

App Version: 6.15.0

Sha256: e707d76221a9269167e5c8923a1593234079ea086593a9be311027d683b696d3

3.2 Vulnerability Information

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

NO	Title	Category	Level	Status
N1	Decompilation security issues	Code decompilation detection	Suggestion	Confirmed
N2	Missing screen recording detection	Screenshot/screen recording detection	Suggestion	Confirmed
N3	Secret key storage security issue	Secret key storage security audit	Suggestion	Confirmed
N4	Secret key storage security issue	Secret key storage security audit	Suggestion	Confirmed
N5	User interaction issue	User interaction security	Suggestion	Confirmed
N6	The signature information display is incomplete	User interaction security	Suggestion	Confirmed
N7	Suspend and invoke security issue	Suspend evoke security audit	Suggestion	Confirmed
N8	Insufficient backup and recovery authentication	Secret key backup security audit	Suggestion	Confirmed

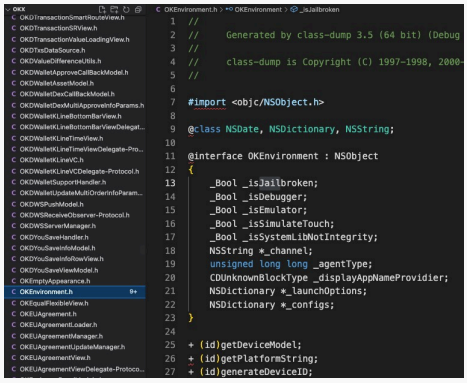
3.3 Vulnerability Summary

[N1] [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

[N2] [Suggestion] Missing screen recording detection

Category: Screenshot/screen recording detection

Content

The OKX wallet is designed to detect when a user takes a screenshot and will prompt them not to do so for security reasons. However, it currently does not detect if the user is recording their screen.

Solution

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

Status

Confirmed

[N3] [Suggestion] Secret key storage security issue

Category: Secret key storage security audit

Content

The wallet mnemonic, private keys, and MPC private key shard information are stored using AES encryption. However, AES encryption algorithm is not the optimal solution as it may be vulnerable to brute-force enumeration.

- MPC-iOS/OKWalletCore/Bridge/OKWBridge.m#line215-217

```
+ (OKWBridgeResult<NSString *> *)encodeData:(NSString *)pwd pwdHash:(NSString
*)pwdHash data:(NSString *)data {
    return [self dispatch:@"encrypt_data" data:@{@"passWord": pwd ? : @"",
@"passWordHash": pwdHash ? : @"", @"data": data ? : @""}];
}
```

- wallet-core/core/core.go#line36-46

```
func EncryptData(pass, hash, data string) (string, int) {
        if !ValidatePass(pass, hash) {
                return "", storage.PASS_ERROR
        }
        aesPass := getAesPass(pass, hash)
        d, err := crypto.Encrypt([]byte(data), aesPass)
        if err != nil {
                return "", storage.DATA_ERROR
        }
        return d, storage.SUCCESS
}
```

- wallet-core/thirdparty/crypto/aes.go#line82-88

```
func Encrypt(rawData, key []byte) (string, error) {
        data, err := AesCBCEncrypt(rawData, key)
        if err != nil {
                return "", err
        }
        return base64.StdEncoding.EncodeToString(data), nil
}
```

Solution

It is recommended to use RSA encryption algorithm for secure storage.

Status

Confirmed

[N4] [Suggestion] Secret key storage security issue

Category: Secret key storage security audit

Content

The getAesPass function invokes MoreHash with an input parameter of 1, indicating the intention to perform multiple hash calculations. However, it seems that the number of calculations in this context is insufficient.

- wallet-core/core/core.go#line71-78

```
func getAesPass(pass string, hash string) []byte {
        h, _ := getHash(pass, hash)
        hb := MoreHash(1, []byte(pass), sha3.NewKeccak256())
        aesPass, _ := hex.DecodeString(h)
        copy(aesPass[:8], hb[:8])
        copy(aesPass[len(aesPass)-8:], hb[len(hb)-8:])
        return aesPass
}
```

- wallet-core/core/core.go#line96-103

```
func MoreHash(count int, value []byte, h hash.Hash) []byte {
        for i := 0; i < count; i++ {
                h.Reset()
                h.Write(value)
                value = h.Sum(nil)
        }
        return value[:32]
}
```

Solution

It is recommended to have an adequate number of iterations for hash computations.

Status

Confirmed

[N5] [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.

Functionality	Support	Notes
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, causing similar address attacks.
Password complexity requirements	✗	Missing password complexity requirements.

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

Solution

It is recommended to add AML, Anti-phishing, Pre-execution and contact whitelisting functions to the application, and password complexity constraints need to be made. It is recommended to remind users to double-check the accuracy of the transfer destination address when it is not in their address book.

Status

Confirmed

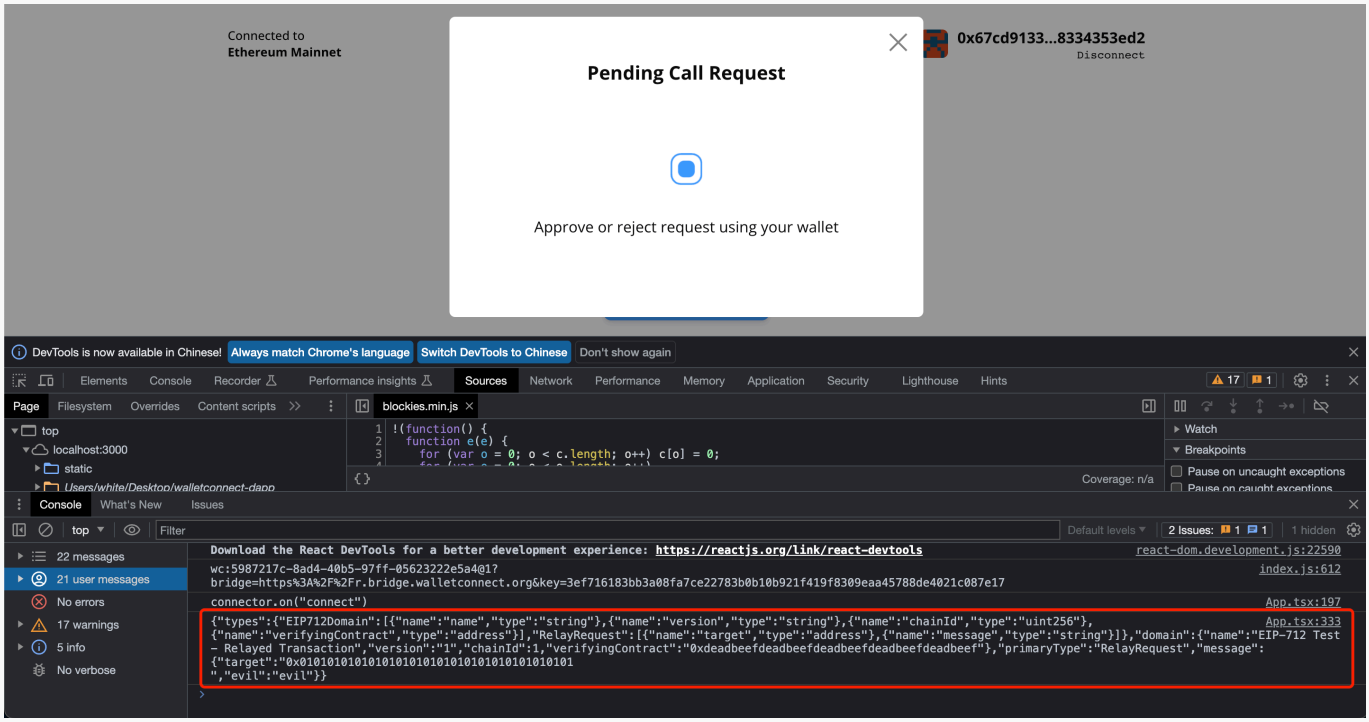
[N6] [Suggestion] The signature information display is incomplete

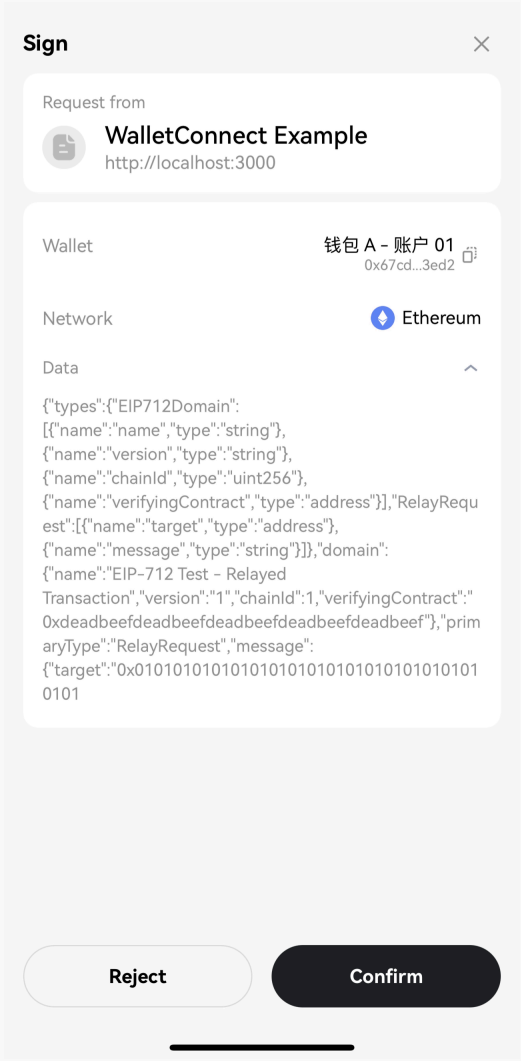
Category: User interaction security

Content

Signed using Wallet Connect, complete and readable signature information is not displayed.

A malicious DApp can deceive users into signing an EIP-712 object different from the one shown in the EIP-712 signature approval preview. To appear less suspicious to the user, the attacker can insert multiple whitespaces between the injected and genuine Permits. When the victim approves the request, the injected Permit is discarded, and the malicious Permit is signed.





Solution

It is recommended to display complete and readable signature information when signing.

Status

Confirmed

[N7] [Suggestion] Suspend and invoke security issue

Category: Suspend evoke security audit

Content

1.No timeout mechanism was found in the wallet app, and the test was suspended for quite a while without re-verification of the password.

2.After the background is suspended, you can continue to use it without verifying the wallet password.

Solution

It is recommended to re-awaken the App after a period of suspension in the background to verify the password.

1. It is recommended to automatically log out when the wallet has not been operated for a long time, and to wake up again requires verification of the password.
2. It is recommended to verify the password when the wallet wakes up again from the background.

Status

Confirmed

[N8] [Suggestion] Insufficient backup and recovery authentication

Category: Secret key backup security audit

Content

When a user loses their old device and attempts to recover their MPC wallet account on a new device, the current authentication relies on OKX exchange account verification and cloud-based account authentication. However, this method cannot guarantee that the recovery process is performed by the wallet owner themselves.

Solution

It is recommended to incorporate biometric authentication, such as facial recognition, to ensure that the recovery is done by the wallet creator.

Status

Confirmed

4 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002306060003	SlowMist Security Team	2023.05.16 - 2023.06.06	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 8 suggestion vulnerabilities. All the findings have been confirmed.

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