SLOWMIST

Wallet Application Security Audit Report

Table Of Contents

1 Executive Summary

On 2022.07.15, the SlowMist security team received the team's security audit application for Sender Wallet Android, 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.

Level	Description
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	Passed
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

NO.	Audit Items	Result
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
16	Transaction broadcast security audit	Passed
17	Mnemonic phrase/Private key generation security audit	Passed
18	Mnemonic phrase/Private key storage security audit	Passed
19	Mnemonic phrase/Private key usage security audit	Passed
20	Mnemonic phrase/Private key backup security audit	Passed
21	Mnemonic phrase/Private key destroy security audit	Passed
22	Screenshot/screen recording detection	Passed
23	Paste copy detection	Passed
24	Keyboard keystroke cache detection	Passed
25	Background obfuscation detection	Passed
26	Suspend evoke security audit	Passed
27	AML anti-money laundering security policy detection	Passed
28	Others	Passed

3 Project Overview

3.1 Project Introduction

Audit Version

https://github.com/SenderWallet/sender-wallet-mobile/tree/slowmist-v0.0.1

commit: 7a6323c09edbf15fcc77b87966734b75d7008d53

sender_mobile-v0.0.1.apk(SHA256): 4a3ef0e703f8174b355c04975cb202630c3884cf48673c7f9fe896ab38734bf2

Fixed Version

https://github.com/SenderWallet/sender-wallet-mobile/tree/slowmist-v0.0.1

commit: 1e50dddb7167ca4184f09ca5a2e549fc9a12f4e7

sender_mobile-v0.0.1.apk(SHA256): 2b37110a099a6a56aedbeec59bf2add9c0637f14d83994caf13dbc49f34e51b5

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	Low	Fixed
N3	Runtime environment security recommendations	App runtime environment detection	Suggestion	Fixed
N4	KeyStore lacks PBKDF2 protection	Mnemonic phrase/Private key storage security audit	Medium	Fixed
N5	Redundant code	Others	Suggestion	Fixed

NO	Title	Category	Level	Status
N6	Missing screenshot/screen recording detection	Screenshot/screen recording detection	Suggestion	Fixed
N7	signer is not cleared after expiration	Mnemonic phrase/Private key usage security audit	Medium	Fixed
N8	Missing status flag for transfer	Deposit/Transfer security audit	Low	Fixed
N9	Strengthen reminder	Others	Suggestion	Fixed
N10	Lack of secure keyboard	Keyboard keystroke cache detection	Suggestion	Confirmed
N11	Lack of security reminders	Paste copy detection	Suggestion	Confirmed
N12	Background obfuscation issue	Background obfuscation detection	Suggestion	Fixed
N13	usesCleartextTraffic configuration enhancement	Communication encryption security audit	Suggestion	Fixed
N14	Enhanced mnemonic verification	Business security audit	Suggestion	Confirmed
N15	Lack of AML security policy	AML anti-money laundering security policy detection	Suggestion	Confirmed

3.3 Vulnerability Summary

[N1] [Low] Runtime environment detection issues

Category: App runtime environment detection

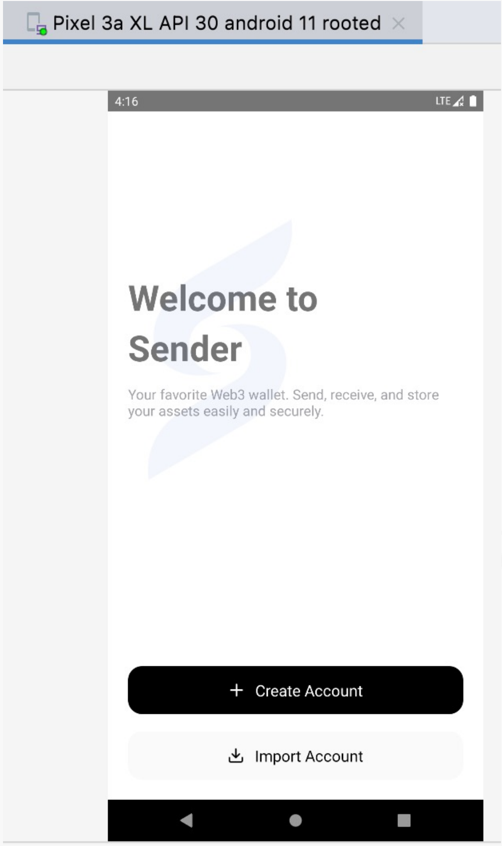
Content

Implemented root and jailbreak detection only in Wallet navigation and Startup navigation. When the App is used for the first time, it will jump to Empty navigation through routing, but Empty navigation does not check the runtime environment of the App. Therefore, the runtime environment may be a security risk when creating/importing a wallet, and lack of a virtual machine detection mechanism.

- src/screens/Startup/index.js#L21

```
useEffect(() => {
  const nextRoute = accounts?.length ? 'Home' : 'Empty';

  if (fromDelete || !keyStore) {
    navigation.replace(nextRoute);
  } else {
    cleanPassword();
    navigation.replace('Unlock', { nextRoute });
  }
}, [navigation]);
```


Pixel 3a XL API 30 android 11 rooted
4:16
LTE
Welcome to Sender
Your favorite Web3 wallet. Send, receive, and store your assets easily and securely.
Create Account
Import Account

android:debuggable is set to true, It should be set to false in the release version.

- AndroidManifest.xml

```
<application android:allowBackup="false"
android:appComponentFactory="androidx.core.app.CoreComponentFactory"
android:debuggable="true" android:icon="@mipmap/ic_launcher"
android:label="@string/app_name"
android:name="com.sender_wallet_mobile.MainApplication"
android:roundIcon="@mipmap/ic_launcher_round" android:theme="@style/AppTheme"
android:usesCleartextTraffic="true">
```

Solution

It is recommended to check the security of the runtime environment when the wallet is running, and set android:debuggable to false.

Status

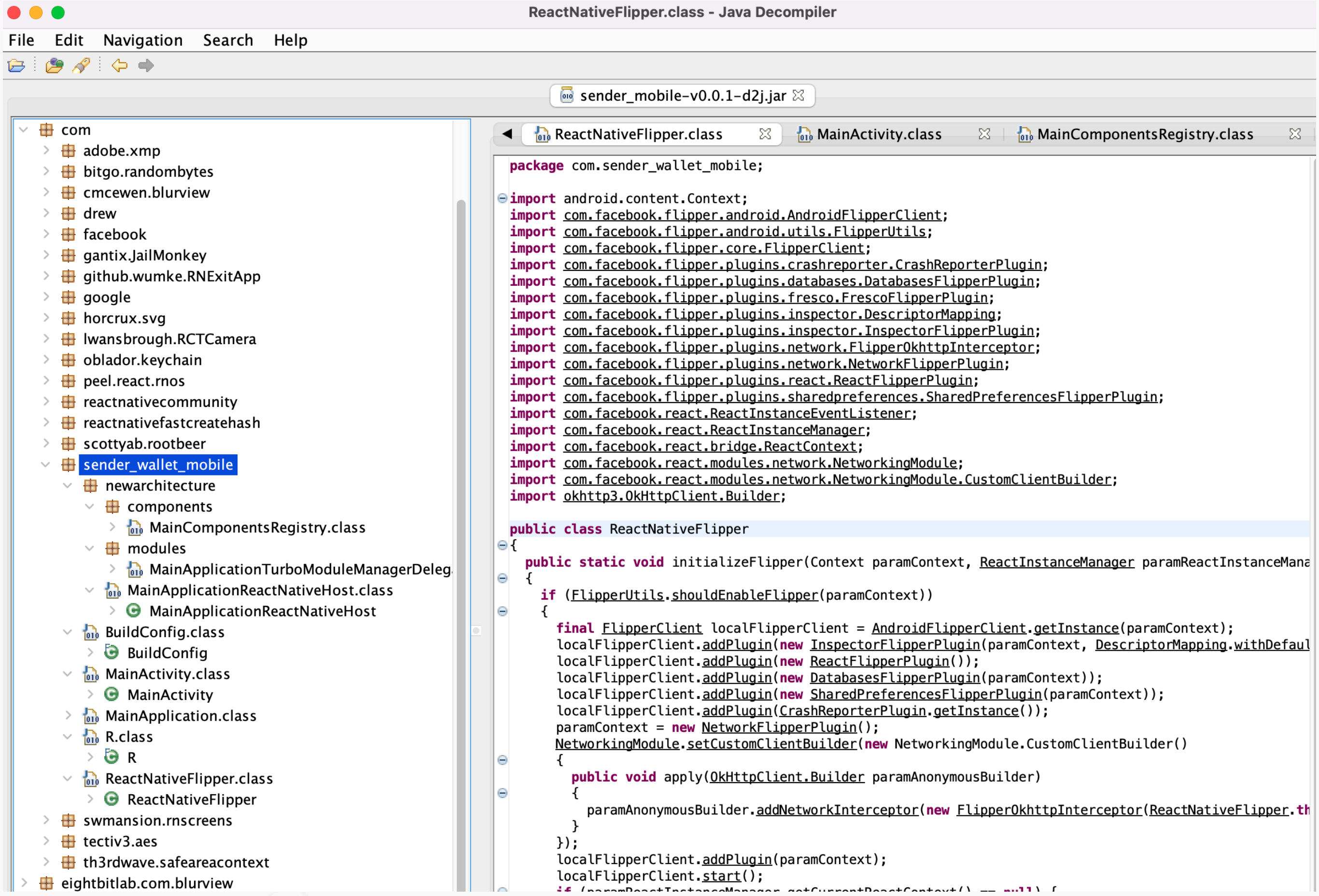
Fixed

[N2] [Low] Decompilation security issues

Category: Code decompilation detection

Content

The App does not obfuscate the code, and the Java code can be obtained by decompiling.

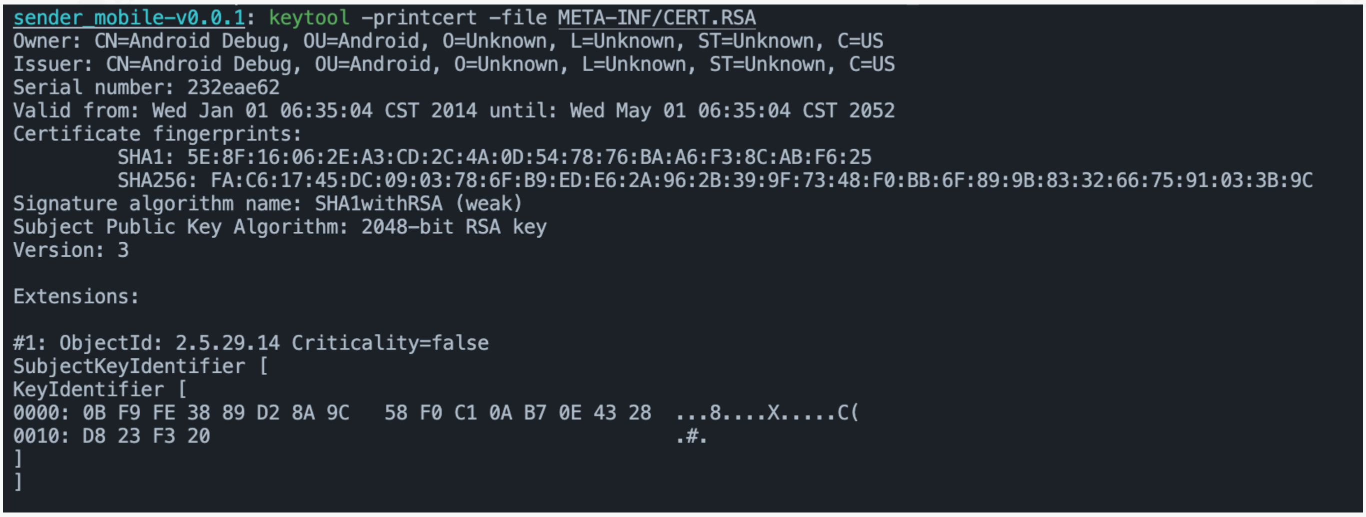


```
package com.sender_wallet_mobile;

import android.content.Context;
import com.facebook.flipper.android.AndroidFlipperClient;
import com.facebook.flipper.android.utils.FlipperUtils;
import com.facebook.flipper.core.FlipperClient;
import com.facebook.flipper.plugins.crashreporter.CrashReporterPlugin;
import com.facebook.flipper.plugins.databases.DatabasesFlipperPlugin;
import com.facebook.flipper.plugins.fresco.FrescoFlipperPlugin;
import com.facebook.flipper.plugins.inspector.DescriptorMapping;
import com.facebook.flipper.plugins.inspector.InspectorFlipperPlugin;
import com.facebook.flipper.plugins.network.FlipperOkhttpInterceptor;
import com.facebook.flipper.plugins.network.NetworkFlipperPlugin;
import com.facebook.flipper.plugins.react.ReactFlipperPlugin;
import com.facebook.flipper.plugins.sharedpreferences.SharedPreferencesFlipperPlugin;
import com.facebook.react.ReactInstanceEventListener;
import com.facebook.react.ReactInstanceManager;
import com.facebook.react.bridge.ReactContext;
import com.facebook.react.modules.network.NetworkingModule;
import com.facebook.react.modules.network.NetworkingModule.CustomClientBuilder;
import okhttp3.OkHttpClient.Builder;

public class ReactNativeFlipper
{
  public static void initializeFlipper(Context paramContext, ReactInstanceManager paramReactInstanceMana
  {
    if (FlipperUtils.shouldEnableFlipper(paramContext))
    {
      final FlipperClient localFlipperClient = AndroidFlipperClient.getInstance(paramContext);
      localFlipperClient.addPlugin(new InspectorFlipperPlugin(paramContext, DescriptorMapping.withDefaul
      localFlipperClient.addPlugin(new ReactFlipperPlugin());
      localFlipperClient.addPlugin(new DatabasesFlipperPlugin(paramContext));
      localFlipperClient.addPlugin(new SharedPreferencesFlipperPlugin(paramContext));
      localFlipperClient.addPlugin(CrashReporterPlugin.getInstance());
      paramContext = new NetworkFlipperPlugin();
      NetworkingModule.setCustomClientBuilder(new NetworkingModule.CustomClientBuilder()
      {
        public void apply(OkHttpClient.Builder paramAnonymousBuilder)
        {
          paramAnonymousBuilder.addNetworkInterceptor(new FlipperOkhttpInterceptor(ReactNativeFlipper.th
        }
      });
      localFlipperClient.addPlugin(paramContext);
      localFlipperClient.start();
```

App certificate signature does not contain basic information about the organization.The debug certificate is used.



```
sender_mobile-v0.0.1: keytool -printcert -file META-INF/CERT.RSA
Owner: CN=Android Debug, OU=Android, O=Unknown, L=Unknown, ST=Unknown, C=US
Issuer: CN=Android Debug, OU=Android, O=Unknown, L=Unknown, ST=Unknown, C=US
Serial number: 232eae62
Valid from: Wed Jan 01 06:35:04 CST 2014 until: Wed May 01 06:35:04 CST 2052
Certificate fingerprints:
         SHA1: 5E:8F:16:06:2E:A3:CD:2C:4A:0D:54:78:76:BA:A6:F3:8C:AB:F6:25
         SHA256: FA:C6:17:45:DC:09:03:78:6F:B9:ED:E6:2A:96:2B:39:9F:73:48:F0:BB:6F:89:9B:83:32:66:75:91:03:3B:9C
Signature algorithm name: SHA1withRSA (weak)
Subject Public Key Algorithm: 2048-bit RSA key
Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 0B F9 FE 38 89 D2 8A 9C   58 F0 C1 0A B7 0E 43 28  ...8....X.....C(
0010: D8 23 F3 20                                        .#.
]
]
```

Solution

It is recommended to harden the App, obfuscate the java code, and supplement the basic information in the release certificate.

Status

Fixed

[N3] [Suggestion] Runtime environment security recommendations

Category: App runtime environment detection

Content

App does not perform virtual machine detection, there is also no detection of the developer mode of the phone. When the developer mode is enabled on the mobile phone, the operating environment may be at risk. The application of the mobile phone can be debugged through the developer mode.

Solution

It is recommended to detect whether the App is running in a virtual machine and detect whether the phone is in developer mode.

Status

Fixed

[N4] [Medium] KeyStore lacks PBKDF2 protection

Category: Mnemonic phrase/Private key storage security audit

Content

PBKDF2 is not used for protection during encryption and decryption.This will lead to the possibility of encryptedData being brute-forced.

- src/utils/crypto.js

```
export const generateHash = (password, salt) => {
  const hmac = CryptoJS.algo.HMAC.create(CryptoJS.algo.SHA512, salt);
  hmac.update(password);
  const hash = hmac.finalize();
  return hash.toString().substring(0, 64);
};
```

```
export const encrypt = (keyStore, key) => Aes.randomKey(16).then((iv) => {
  const text = JSON.stringify(keyStore);
  return (
    Aes.encrypt(text, key, iv, 'aes-256-cbc').then((cipher) => ({
      cipher,
      iv,
    }))
  );
});

export const decrypt = async (This will lead to the possibility of encryptedData
being brute-forced., key) => {
  const message = await Aes.decrypt(encryptedData.cipher, key, encryptedData.iv,
'aes-256-cbc');
  return JSON.parse(message);
};
```

Solution

It is recommended to use PBKDF2 for protection when generating generateHash.

Status

Fixed

[N5] [Suggestion] Redundant code

Category: Others

Content

There are useless commented code in the file and code that is not used in actual business.

- src/screens/Home/Settings/index.js#L83-L88

```
  const settings = [
    // {
    //   onPress: () => { navigation.navigate('Wallet/Manage'); },
    //   icon: require('../../../assets/img/settings-faceid.png'),
    //   label: 'Face ID',
    //   rightComponent: <Image source={require('../../../assets/img/arrow-
right.png')} />,
    // },
```

- src/screens/Home/Settings/index.js#L119-L126

```
  const support = [
    {
      onPress: () => { navigation.navigate('Wallet/Manage'); },
      icon: require('../../../assets/img/settings-help.png'),
      label: 'Help & Support',
      rightComponent: <Image source={require('../../../assets/img/arrow-right.png')}
/>,
    },
  ];
```

- src/screens/Home/Settings/index.js#L185-L196

```
          {/* <Text style={[styles.fontSemiBold, styles.fontSize14,
styles.lineHeight20, { color: '#262626', marginTop: scaleSize(32) }]}>Support</Text>
          {
            _.map(support, (item) => {
              return <SettingItem
                key={item.label}
                onPress={item.onPress}
                icon={item.icon}
                label={item.label}
                rightComponent={item.rightComponent}
              />
            })
          } */}
```

Solution

It is recommended to remove redundant commented code and useless code.

Status

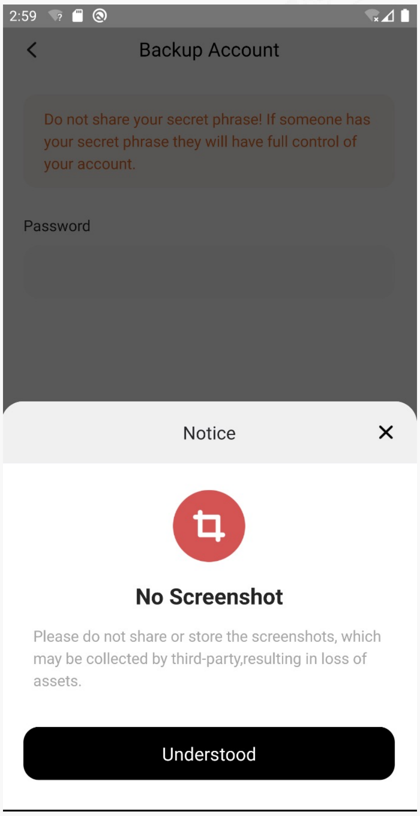
Fixed

[N6] [Suggestion] Missing screenshot/screen recording detection

Category: Screenshot/screen recording detection

Content

The app has a reminder that screenshots are prohibited, but it does not restrict users from taking screenshots and missing screenshot detection and restrictions.



2:59

Backup Account

Do not share your secret phrase! If someone has your secret phrase they will have full control of your account.

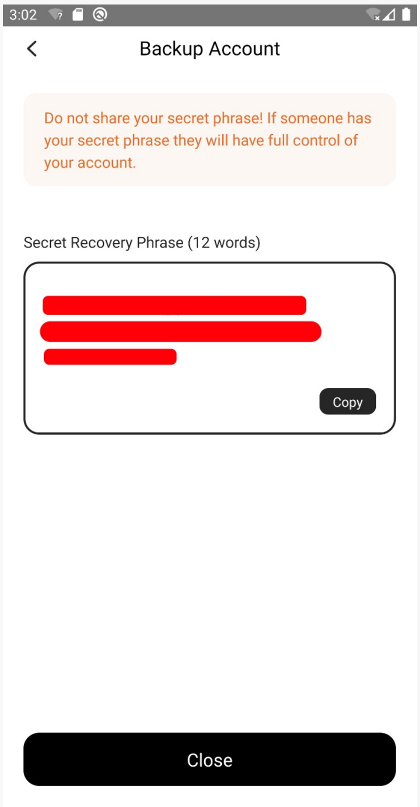
Password

Notice

No Screenshot

Please do not share or store the screenshots, which may be collected by third-party,resulting in loss of assets.

Understood



Backup Account

Do not share your secret phrase! If someone has your secret phrase they will have full control of your account.

Secret Recovery Phrase (12 words)

[redacted]

Solution

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

Status

Fixed

[N7] [Medium] signer is not cleared after expiration

Category: Mnemonic phrase/Private key usage security audit

Content

cleanPassword will be executed after the interval expires but this.signer is not assigned to null.

- src/screens/Home/Wallet/index.js#L201

```
  useEffect(() => {
    const subscription = AppState.addEventListener('change', async (nextAppState) =>
{
      appState.current = nextAppState;
      const currentActiveTime = Date.now();
      if (nextAppState === 'active') {
        const intervalTime = currentActiveTime - activeTime;
        if (intervalTime >= (LOCK_MINUTES * 60000) && keyStore) {
          const hashedPassword = await getPassword();
          if (hashedPassword) {
            cleanPassword();
            // to unlock page
            navigation.push('Unlock');
          }
        }
      } else {
        dispatch(setActiveTime(currentActiveTime));
      }
    });

    return () => {
      subscription.remove();
    };
  }, [activeTime, keyStore, navigation]);
```

- src/core/near.js#L127-L137

```
export class NearService {
  constructor({ config, accountId }) {
    this.viewAccount = getViewAccount({ config, accountId });
    this.signer = null;
    this.config = config;
    this.apiHelper = new ApiHelper({ helperUrl: config.helperUrl });
  }

  setSigner = async ({ mnemonic, accountId }) => {
    const signer = await getSigner({ mnemonic, accountId, config: this.config });
    this.signer = signer;
  };
```

Solution

It is recommended to also clear the value of this.signer when executing cleanPassword.

Status

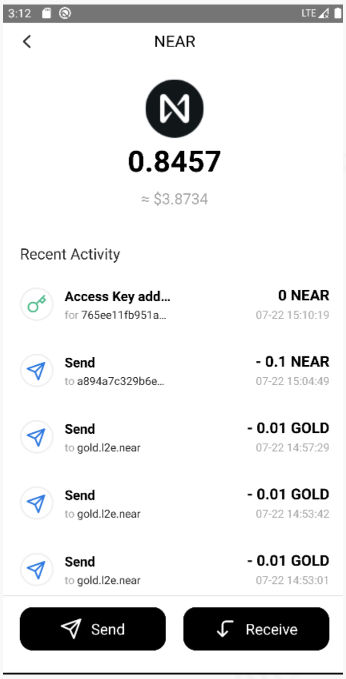
Fixed

[N8] [Low] Missing status flag for transfer

Category: Deposit/Transfer security audit

Content

Transfer failures are not flagged in the app, and incorrect transfer status may be used for scams.All transfers of GOLD tokens are failed.



0.8457

≈ $3.8734

Recent Activity

Access Key add... for 765ee11fb951a...	0 NEAR 07-22 15:10:19
Send to a894a7c329b6e...	- 0.1 NEAR 07-22 15:04:49
Send to gold.l2e.near	- 0.01 GOLD 07-22 14:57:29
Send to gold.l2e.near	- 0.01 GOLD 07-22 14:53:42
Send to gold.l2e.near	- 0.01 GOLD 07-22 14:53:01

Solution

It is recommended to flag the status of the transfer in the app.

Status

Fixed

[N9] [Suggestion] Strengthen reminder

Category: Others

Content

When importing the wallet, if the mnemonic is wrong, the app will not prompt the import error, but stay in the importing.

Solution

It is recommended to remind users after the wallet import fails to avoid confusion for users in the absence of wrong reminders.

Status

Fixed

[N10] [Suggestion] Lack of secure keyboard

Category: Keyboard keystroke cache detection

Content

The app does not use a secure keyboard, mnemonics and passwords may be stolen by the keyboard when using the app.

Solution

It is recommended to add a secure keyboard and use the secure keyboard when entering mnemonics and passwords to avoid sensitive data being recorded.

Status

Confirmed

[N11] [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

It is recommended to remind users that they should record by transcribing instead of directly using the clipboard for copying.

Status

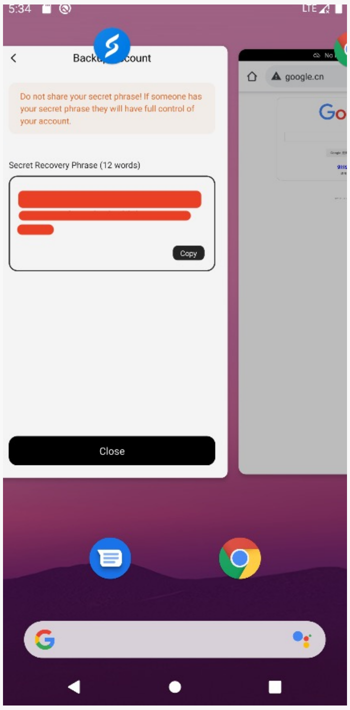
Confirmed

[N12] [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.



Do not share your secret phrase! If someone has your secret phrase they will have full control of your account.

Secret Recovery Phrase (12 words)

[redacted]

Solution

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

Status

Fixed

[N13] [Suggestion] usesCleartextTraffic configuration enhancement

Category: Communication encryption security audit

Content

usesCleartextTraffic is set to true to allow communication using HTTP.

- android/app/src/debug/AndroidManifest.xml#L8

```
    <application
        android:usesCleartextTraffic="true"
        tools:targetApi="28"
        tools:ignore="GoogleAppIndexingWarning">
        <activity android:name="com.facebook.react.devsupport.DevSettingsActivity"
android:exported="false" />
    </application>
```

Solution

It is recommended to set usesCleartextTraffic to false to only allow communication using HTTPS.

Status

Fixed

[N14] [Suggestion] Enhanced mnemonic verification

Category: Business security audit

Content

When creating a wallet, the user is required to confirm whether the mnemonic phrase is backed up completely.The app only requires the user to verify 1 of the 12 mnemonic phrases, and this verification method needs to be strengthened. Because all mnemonics may not be fully backed up with.

Solution

It is recommended to scramble the 12 mnemonics and then let the user reorder the mnemonics, so as to guide the user to verify the correctness of each mnemonic.

Status

Confirmed

[N15] [Suggestion] Lack of AML security policy

Category: AML anti-money laundering security policy detection

Content

The app does not have access to the AML security policy and cannot synchronize malicious addresses to users in a timely manner.

Solution

It is recommended to access the AML security policy to remind users to avoid interacting with malicious addresses.

Status

Confirmed

4 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002207260001	SlowMist Security Team	2022.07.15 - 2022.07.26	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 2 medium risk, 3 low risk vulnerabilities and 10 suggestions.

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.

SLOWMIST

Official Website

www.slowmist.com

✉

E-mail

team@slowmist.com

Twitter

@SlowMist_Team

Github

https://github.com/slowmist