AltMe Crypto wallet

Security audit report

Prepared for Web3 Digital Wallet

February 22, 2023

Document management

Revision history

Version	Date	Version details
1.0	January 11, 2022	Initial version
1.1	February 22, 2023	Review after fixes

Table of contents

Project summary

Name	AltMe	
Source	**Repository**	**Revision**
	https://github.com/TalaoDAO/AltMe	889539e95f364ed922c1d5743c82b2be8dc54b34
		3dd9d3aac095fadc0638e3ed9fc5641f906e8f30
Methods	Static analysis Behavioral analysis Unit test coverage analysis	

Glossary

Code injection — An attack that exploits bugs in data processing. A code injection attack introduces ("injects") malicious code into a vulnerable program.

MiTM attack — Man-in-the-middle attack. An attack where the attacker secretly relays and possibly alters the communications between two parties who believe they are directly communicating with each other.

Coverage and scope of work

The audit focused on an in-depth analysis of the AltMe wallet in terms of protection of sensitive data, such as user credentials and other private information which is a key focus in mobile security. The main targets of the analysis were:

- package/secure_storage
- package/key_generator
- package/cryptocurrency_keys
- lib/wallet
- lib/pin_code/cubit
- lib/onboarding/gen_phrase/cubit/
- lib/dashboard/home/tab_bar/tokens/confirm_token_transaction/cubit/
- lib/dashboard/beacon/beacon_operation/cubit/
- lib/dashboard/beacon/beacon_sign_payload/cubit/
- lib/dashboard/drawer/backup_credential/cubit/
- lib/dashboard/drawer/recovery_credential/cubit
- lib/import_wallet/cubit/
- lib/dashboard/drawer/recovery_key/cubit/
- lib/dashboard/drawer/secret_key/cubit/
- lib/beacon/cubit/

We conducted the audit in accordance with the following criteria:

- static analysis
- behavioral analysis of AltMe wallet within mentioned scope
- checks against database of known vulnerabilities
- unit test coverage analysis

Application overview

AltMe is a self-custodial crypto-wallet for Web3 where the user is in charge of all wallet data through the use of SSI technology and secure storage which stores data locally.

AltMe wallet enables:

- to store and manage digital assets and NFTs (currently, for Tezos only)
- to create/import/store Tezos and Ethereum blockchain accounts
- to connect to and interact with dApps by utilizing tzip-10 wallet interaction standard
- to get and store different types of credentials: proofs, membership cards, blockchain accounts
- to create encrypted backup with credentials and restore credentials through backups

Executive overview

Apriorit conducted a security assessment of AltMe in December 2022 - February 2023 to evaluate its current state and risk posture, evaluate exposure to known security vulnerabilities, determine potential attack vectors, and check if any can be exploited maliciously.

Summary of strengths

Building upon the strengths of the available implementation can help TalaoDAO better secure it by continuing these good practices. In this case, a number of positive security aspects were readily apparent during the assessment:

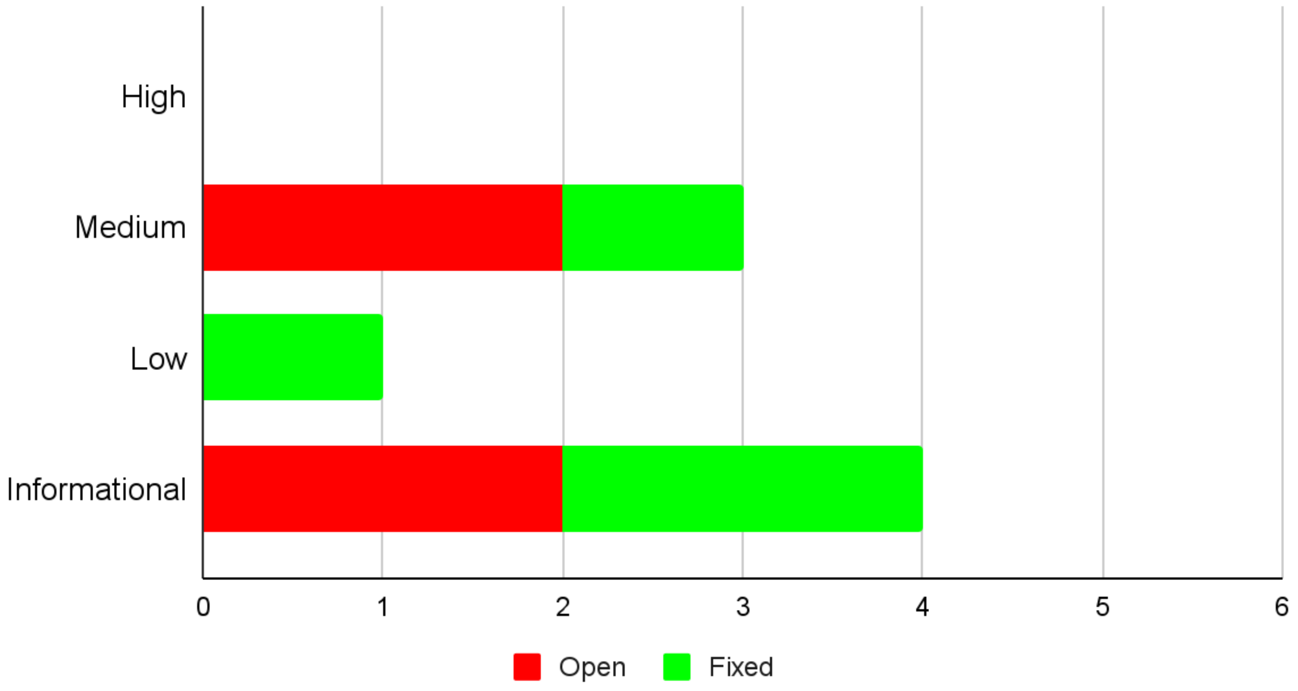
- no high risk vulnerabilities were found;
- sensitive data are stored in secure storage;
- latest flutter SDK release is used;
- the code and project files are well structured, which makes them easy to read and maintain. The code is self-explanatory. The naming policy makes instructions understandable.

Summary of discovered vulnerabilities

During the initial assessment, 3 medium, 1 low and 3 informational vulnerabilities were discovered. After the secondary audit it was discovered that one medium, one low, and two informational vulnerabilities were fixed, and one new informational issue was detected. Overall, two medium, and two informational vulnerabilities are still present. For more detailed information on all of the findings, refer to Appendix A: Detailed Findings.

The chart below shows the distribution of findings.

Vulnerability chart



Summary of medium-risk vulnerabilities and recommendations

For more detailed information on all of the findings, refer to Appendix A: Detailed Findings.

Table 1: Medium-risk vulnerabilities

Risk rating	**Finding name**	**Recommendation**	**Status**
Medium	Secret keys are stored in memory	Store private keys in secure flutter storage	OPEN
Medium	It is possible to bruteforce pin-code	Provide a strategy that limits amount of consecutive attempts to type invalid pin-code within short period of time	OPEN
Medium	Application does not have capabilities to prevent against screenshots	Making screenshots of pages with recovery phrases and private keys should be disabled. Remove sensitive data from view when application goes to background	FIXED

Summary of low-risk vulnerabilities and recommendations

For more detailed information on all of the findings, refer to Appendix A: Detailed Findings.

Table 2: Low-risk vulnerabilities

Risk rating	Finding name	Recommendation	Status
Low	Logging of sensitive data	Delete logging of secret keys from beacon lib	FIXED

Summary of informational-risk vulnerabilities and recommendations

Information issues do not affect security, but they help keep the code clean.

For more detailed information on all of the findings, refer to Appendix A: Detailed Findings.

Table 3: Informational-risk vulnerabilities

Risk rating	Finding name	Recommendation	Status
Informational	Application does not have capabilities to prevent tapjacking attacks	Check attributes that enables to overlay on application screens	OPEN
Informational	Application freezes on Decentralized ID key page	Check if Animation Controller status is managed correctly	OPEN
Informational	Application enables to copy sensitive data to the clipboard	Copying sensitive data from pages with recovery phrases and private keys should be disabled except cases where it's required by the business logic	FIXED
Informational	Application shows wrong wallet status after credentials are restored from backup	Check is wallet is created after backup	FIXED

Security rating

Apriorit reviewed TalaoDAO security posture in regards to the AltMe wallet, and Apriorit consultants identified security strengths as well as vulnerabilities that create low and informational levels of risk. Taken together, the combination of asset criticality, threat likelihood, and vulnerability severity have been used to assign a grade for the overall security of the application. An explanation of the grading scale is included in the second table below. In conclusion, Apriorit recommends that TalaoDAO continue to follow existing good security practices and further improve their security posture by addressing all of the described findings.

	High	**Medium**	**Low**	**Informational**	**Security**	**Grade**
AltMe	0	2	0	2	Moderately secure	B

Security grading criteria

Grade	Security	Criteria description
A	Highly secure	Exceptional attention to security. No high- or medium-risk vulnerabilities and few minor low-risk vulnerabilities.
B	Moderately secure	Good attention to security. No high-risk vulnerabilities and only a few medium- or several low-risk vulnerabilities.
C	Marginally secure	Some attention to security, but security requires improvement. A few high-risk vulnerabilities that can be exploited.
D	Insecure	Significant security gaps exist. A large number of high-risk vulnerabilities.

Test coverage analysis

Unit tests are an essential part of development. They help to find problems in the code that are missed by the compiler before deploying the solution to the production environment.

During the audit, the percentage of unit test coverage for each package was evaluated. The results are presented in the table below.

Package/Lib	Initial coverage
package/secure_storage	100%
package/key_generator	91%
package/cryptocurrency_keys	96.4%
lib/wallet	0%
lib/pin_code/cubit	0%
lib/onboarding/gen_phrase/cubit/	0%
lib/dashboard/home/tab_bar/tokens/confirm_token_transaction/cubit/	0%
lib/dashboard/drawer/backup_credential/cubit/	0%
lib/dashboard/drawer/recovery_credential/cubit	0%
lib/import_wallet/cubit/	0%
lib/dashboard/drawer/secret_key/cubit/	0%

Auxiliary packages are well covered but the functions related to main execution flow are not covered. At least basic test cases should be covered.

package/key_generator uncovered test cases

Function	Description	Status

jwkFromSecretKey, L:188	A jwk is obtained from a secret key	CLOSED
getKeystore, L:198	The keystore is instantiated	OPEN

lib/wallet uncovered test cases

Function	Description	Status
createCryptoWallet, L:95	if ‘mnemonicOrKey’ is a secret key then account created in chain based on a supported key type	OPEN
createCryptoWallet, L:138, L:148	if ‘mnemonicOrKey’ is a mnemonic and blockchain type is recognized then account created based on blockchain type	OPEN
createCryptoWallet, L:158	if ‘mnemonicOrKey’ is a mnemonic and account type is not recognizable then account created on all supported chains	OPEN
editCryptoAccountName, L:338	wallet state and secure storage contain renamed account after the call	OPEN
editCryptoAccountName, L:338	wallet name in credentials list is updated	OPEN
deleteById, L:428	wallet state does not contain credentials with deleted id	OPEN
insertCredential, L:504 updateCredential, L:461	credentials repository and wallet state contain new credential, emits state with new credential	OPEN
resetWallet, L:604	wallet state and secure storage do not contain ssi, did, crypto, profile, credentials data	OPEN
recoverWallet, L:664	old credentials replaced with new credentials	OPEN

lib/pin_code/cubit uncovered test cases

Function	Description	Status

onKeyboardButtonPressed, L:41	emits correct passcode based on input key	OPEN

lib/onboarding/gen_pshrase/cubit uncovered test cases

Function	Description	Status
GenerateSSIAndCryptoAccount, L:41	sets required keys to the storage (mnemonic, ssi, did) validates mnemonic	OPEN

lib/dashboard/home/tab_bar/tokens/confirm_token_transaction/cubit/ uncovered test cases

Function	Description	Status
canConfirmTheWithdrawal, L:36	cannot confirm if amount is less or equal to minimal threshold cannot confirm if wallet is not set	OPEN
sendContractInvocationOperation, L:106	cannot invoke if it is not XTZ and contract address is not set	OPEN
sendContractInvocationOperation, L:150	sends native tokens as well as FA1.2, FA2 tokens	OPEN

lib/dashboard/drawer/backup_credential/cubit/ uncovered test cases

Function	Description	Status
encryptAndDownloadFile, L:60	throws if storage permissions are denied	OPEN
encryptAndDownloadFile, L:74	throws on invalid mnemonic	OPEN
encryptAndDownloadFile, L:80	emits file path on success	OPEN

lib/dashboard/drawer/recovery_credential/cubit uncovered test cases

Function	Description	Status
isMnemonicsValid, L:25	emits correct mnemonics state	OPEN
recoverWallet, L:47	throws error on invalid backup file content	OPEN
recoverWallet, L:63	throws error on missing credentials	OPEN
recoverWallet, L:79	emits success on valid backup file	OPEN

lib/import_wallet/cubit/ uncovered test cases

Function	Description	Status
isMnemonicsOrKeyVali, L:34	emits correct status based on input value: - invalid if empty - invalid if not private key and mnemonic is not correctly - valid for correct mnemonic - valid for private key	OPEN

lib/dashboard/drawer/recovery_key/cubit/ uncovered test cases

Function	Description	Status
loadMnemonic, L:34	emits status success with mnemonics when instantiated	OPEN

lib/dashboard/drawer/secret_key/cubit/ uncovered test cases

Function	Description	Status
initialize, L:13	emits valid secret key based on current crypto index	OPEN

Appendixes

Appendix A. Detailed findings

Risk rating

Our risk ratings are based on the same principles as the Common Vulnerability Scoring System. The rating takes into account two parameters: exploitability and impact. Each of these parameters can be rated as high, medium, or low.

Exploitability — What knowledge the attacker needs to exploit the system and what preconditions are necessary for the exploit to work:

- **High** — Tools for the exploit are readily available and the exploit requires no specialized system knowledge.
- **Medium** — Tools for the exploit are available but have to be modified. The exploit requires specialized knowledge about the system.
- **Low** — Custom tools must be created for the exploit. In-depth knowledge of the system is required to successfully perform the exploit.

Impact — What effect will the vulnerability have on the system if exploited:

- **High** — Administrator-level access and arbitrary code execution or disclosure of sensitive information (private keys, personal information)
- **Medium** — User-level access with no disclosure of sensitive information.
- **Low** — No disclosure of sensitive information. Failure to follow recommended best practices does not result in an immediately visible exploit.

Based on the combination of parameters, an overall risk rating is assigned to a vulnerability.

Vulnerabilities discovered in the application

Open issues

Medium risk
Secret keys are stored in memory
Description:

CryptoAccount contains a list of all accounts along with their private keys.
Sensitive data such as passwords, pincodes, secret keys should not be stored for a long time anywhere except secure storage providers.

Affected code:

/TalaoDAO/AltMe/lib/wallet/cubit/wallet_cubit.dart, createTezosOrEthereumAccount, L:295
/TalaoDAO/AltMe/lib/wallet/cubit/wallet_state.dart, L:9

Recommendation:

Consider associating private key with wallet address and store it in flutter secure storage
/TalaoDAO/AltMe/packages/secure_storage/lib/src/secure_storage.dart

The purpose is to eliminate storing sensitive data in memory for a long period of time. Everytime the private key is needed it can be fetched from storage directly by a unique key.

Details:

OWASP MSTG-STORAGE-10 states "The app does not hold sensitive data in memory longer than necessary, and memory is cleared explicitly after use" See https://github.com/OWASP/owasp-mastg/blob/master/Document/0x05d-Testing-Data-Storage.md#testing-memory-for-sensitive-data-mstg-storage-10 for additional information.

Medium risk

It is possible to bruteforce pin-code

Description:

The pin-code policy is weak as the pin-code contains only 4 digits. Despite the fact that an attacker can pick up a pin code only if he physically owns the device, the weak password policy simplifies password cracking. There are a lot of tools that enable cracking of weak passwords within less than one second. Also, there is no limit on entering the pincode. Therefore, someone can conduct an unlimited amount of attempts and eventually bruteforce it.

Affected code:

All pages that require pin-code confirmation

Recommendation:

It is recommended to increase the length of the pin code to 6 digits. To prevent pin-code brute-force attacks it is mandatory to integrate brute-force protection to block access to the wallet for some period of time after N failed attempts are detected. E.g. block for 30 seconds after 5 failed attempts, then block for 1 minute after the next 5 failed attempts and so on. Then, a user should enter a valid pin-code to open a ledger. Until then, the wallet is blocked and shows a banner with a time counter (even when the app is reopened).

Informational

Application does not have capabilities to prevent tapjacking attacks

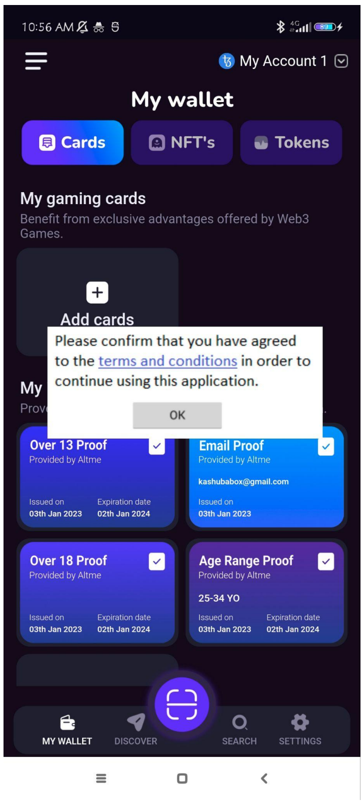
Description:

An attacker can hijack the user's taps and trick him into performing some critical operations that he did not intend to. Screen overlay attacks occur when a malicious application manages to put itself on top of another application which remains working normally as if it were on the foreground. The malicious app might create UI elements mimicking the look and feel as the original app or even the system UI. The intention is

typically to make users believe that they keep interacting with the legitimate app and then try to elevate privileges (e.g by getting some permissions granted), stealthy phishing, capture user taps and keystrokes etc.
Tapjacking (for Android 6.0 (API level 23 and lower)) abuses the screen overlay feature of Android listening for taps and intercepting any information being passed to the underlying activity.

Affected code:

All pages


Please confirm that you have agreed to the terms and conditions in order to continue using this application.

Recommendation:

Tapjacking defense depends on mobile platforms. See the following recommendations on how to prevent tapjacking:
https://github.com/OWASP/owasp-mastg/blob/master/Document/0x05h-Testing-Platform-Interaction.md#testing-for-overlay-attacks-mstg-platform-9

Informational

Application freezes on Decentralized ID key page

Description:

Go to Settings → Self-Sovereign Identity (DID) → Key Decentralized ID → DID private key and click REVEAL. When the counter reaches zero value the expected behavior is to close the **Decentralized ID key** page. But after several attempts to reveal the private key, when the counter approaches 00:00 value the application hangs and does not react when you try to close the page manually (back-leading button is not responsive). The application can be returned to operational state only forcefully by closing and reopening it using the app switcher menu.

Affected code:

The pages that are affected:
- Decentralized ID key, /TalaoDAO/AltMe/lib/dashboard/drawer/manage_did/view/did_private_key.dart, L:8
- EBSI Decentralized ID, /TalaoDAO/AltMe/lib/dashboard/drawer/manage_did/view/did_ebsi_private_key_page.dart , L:9

Recommendation:

Check if Animation Controller status is managed correctly when the counter reaches zero value.

Closed issues

Medium risk

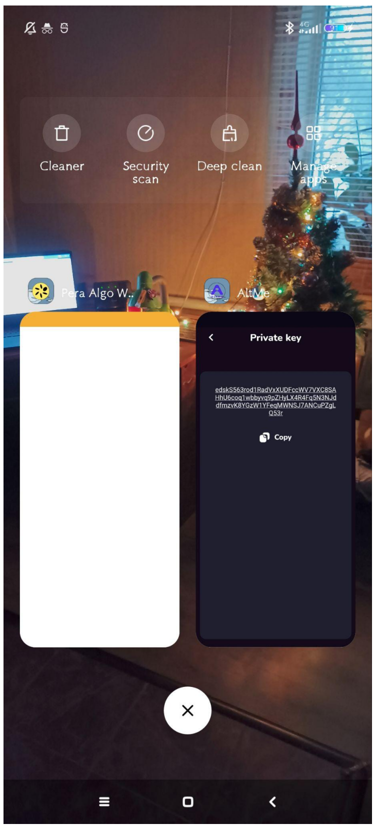
Application does not have capabilities to prevent against screenshots

Description:

Pages that allows to take screenshots of sensitive data directly:
- Recovery phrase (during on-boarding and in settings as well)
- Private key

Consider the scenario where the application shows a page with revealed sensitive data and then goes to the background. Open the **Manage blockchain accounts** page →

choose an account → reveal private key. After entering the pincode, the page displays the private key. Press the **"Home"** button on the device. Then, go to the **app switcher** to display the applications sent to the background and see that the snapshot displays a page with a private key. In the example below, an application on the left deletes sensitive data from view. AltMe on the right leaves sensitive data as is.



edskS563rod1RadVxXUDFccWV7VXC8SAHhU6coq1wbbyvq9pZHyLX4R4Fq5N3NJddfmzvK8YGzW1YFeqMWNSJ7ANCuPZgLQ53r

A malicious application, coupled with tapjacking, can force the user to execute this scenario and in the process copy a snapshot from the folder where minimized application snapshots are stored and send it over the network.

Affected code:

Basically, all pages. But only pages with sensitive data should be protected

Recommendation:

Normally, mobile applications delete sensitive data from snapshots. To prevent screenshot for specific screens see flutter_windowmanager: https://pub.dev/packages/flutter_windowmanager

Low risk

Logging of sensitive data

Description:

There are many legitimate reasons to create log files on a mobile device, such as keeping track of crashes, errors, and usage statistics. Log files can be stored locally for an unlimited period of time when the app is offline and sent to the endpoint once the app is online. However, logging sensitive data may expose the data to attackers or malicious applications, and it might also violate user confidentiality.

Affected code:

/lib/dashboard/beacon/beacon_operation/cubit/beacon_operation_cubit.dart, L:303

Recommendation:

Delete logging of secret key from beacon operation implementation.

Details:

OWASP MSTG-STORAGE-3 states "No sensitive data is written to application logs."
See CWE-532: Insertion of Sensitive Information into Log File https://cwe.mitre.org/data/definitions/532.html

Informational

Application enables to copy sensitive data to the clipboard

Description:

It is recommended to prohibit copying of sensitive data to clipboard as other applications can access it. The clipboard is accessible system-wide and is therefore shared by apps. This sharing can be misused by malicious apps to get sensitive data that has been stored in the clipboard.

Affected code:

Pages:

- Manage Decentralized ID

- Private Key
- Recovery phrase

Recommendation:

If it’s required by a business logic, copying secret keys to the clipboard can be allowed. However, recovery phrases should never be copied to the clipboard.

Informational

Application shows wrong wallet status after credentials are restored from backup

Description:

Credentials can be successfully restored from the backup.
However:
- accounts are not restored. Accounts are mentioned as credentials and listed in the list of “My blockchain account” in the dashboard but missing from Blockchain Settings→Manage blockchain accounts, and they need to be exported again;
- the app does not allow you to navigate through wallet features right after credentials restoration and shows the banner that proposes to import or create a wallet (look like the on-boarding page when wallet does not exist). The banner disappears when the application is reopened.

Affected code:

/home/docker/projects/github/TalaoDAO/AltMe/lib/wallet/cubit/wallet_cubit.dart, recoverWallet, L:669

Recommendation:

It is recommended to check if a wallet is created and emit status based on such a check. See /lib/splash/helper_function/is_wallet_created.dart, isWalletCreated, L:10

Appendix B. Description of methodologies

Mobile applications security checks

Apriorit uses a comprehensive and methodical approach to assess the security of mobile applications. We take the following steps to find vulnerabilities, expose weaknesses, and identify deviations from accepted best practices in assessed applications. Notes and results from these testing steps are included in the corresponding section of the report.

Our security audit includes the following stages:

1. **Discovery and threat modeling**. The first step is to perform reconnaissance and information gathering to decide how resources can be used in the most secure way. It is important to obtain a thorough understanding of the logic of application, and the environment they operate within so tests can be targeted appropriately. Within this stage, the following tasks were completed:

    a. Identify technologies

    b. Analyze the specification, whitepaper, and application source code

    c. Create a map of relations packages and libraries

    d. Research and analyze standard implementations for functionality

    e. Review permissions for each role

2. **Data validation**. Inputs are triggers to the logic but are also the source of most high-risk attacks. This step ensures that data provided to the application is processed and checked. All cases of invalid or unexpected data should be handled appropriately.

3. **Data storage and privacy requirements**

   The protection of sensitive data, such as user credentials and private information, is a key focus in mobile security. Firstly, sensitive data can be unintentionally exposed to other apps running on the same device if operating system mechanisms like IPC are used improperly. Data may also unintentionally leak to cloud storage, backups, or the keyboard cache. Additionally, mobile devices can be lost or stolen more easily compared to other types of devices, so an adversary gaining physical access is a more likely scenario. In that case, additional protections can be implemented to make retrieving the sensitive data more difficult.

   The vast majority of data disclosure issues can be prevented by following simple rules provided by OWASP MASVS.

4. **High-quality software development standards**. The development of high-quality software requires teams to follow the best practices of coding standards. This will help to avoid or mitigate the most common mistakes during development that lead to security vulnerabilities. It will also help with traceability and root cause analysis. This stage includes:

   a. Manual code review and evaluation of code quality

   b. Unit test coverage analysis

Disclaimer

Apriorit has audited the security of mobile applications provided in accordance with industry best practices as of the date of this report. This audit report provides information for the purposes of minimizing security risks, but it cannot guarantee the security of the audited applications.

The report should not be used as the sole means of validating the security of audited applications. We recommend running several independent security audits as well as a public bug bounty program to ensure application security.

This report does not provide any financial or investment advice with regards to the audited applications or their associated project(s). It is advisable to consult with a financial advisor and conduct an independent assessment before making any investment decisions.