Fine penetration tests for fine websites

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Pentest-Report 1Password for Android & iOS 10.2022

Cure53, Dr.-Ing. M. Heiderich, MSc. H. Moesl, P. Einkemmer

Index

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Introduction

"The 1Password you need to remember - With 1Password you only ever need to memorize one password. All your other passwords and important information are protected by your Master Password, which only you know."

From https://1password.com/tour/

This report describes the results of a security assessment of the 1Password complex, specifically targeting the 1Password 8 mobile applications and connected codebase, with both Android and iOS branches covered. Carried out by Cure53 in October 2022, the project included a penetration test and a dedicated audit of the source code.

Registered as *1PW-23,* the project was requested by AgileBits Inc. dba 1Password in August 2022 and then scheduled for mid-autumn 2022 to allow ample time for preparations on both sides. To give some details, Cure53 is very much involved in external testing of the 1Password scope, with this project being the twenty-third instance of security-centered cooperation between the two entities. At the same time, since the 1Password complex is extensive, it is crucial to more precisely state that the mobile applications have been tested by Cure53 exactly three times previously (see the reports labeled *1PW-10, 1PW-19* and *1PW-20*).

As for the precise timeline and specific resources, Cure53 completed the examination in October 2022, namely in CW42. A total of ten days were invested to reach the coverage expected for this assignment. What is more, a team comprising three senior testers has been composed and tasked with this project's preparation, execution and finalization.

For optimal structuring and tracking of tasks, the work was split into two separate work packages (WPs):

- **WP1**: 1Password 8 mobile application codebase and mobile app for Android
- **WP2**: 1Password 8 mobile application for iOS

It can be derived from above that white-box methodology was utilized. Cure53 was supplied with sources, binaries, test-user-accounts, as well as various other means of accessing the scope, as dictated by the project requirements. Additionally, detailed information about the scope was shared with the testing team to make sure the project can be executed in line with the agreed-upon framework.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

The project progressed effectively on the whole. All preparations were done in CW40 to foster a smooth transition into the testing phase. Over the course of the engagement, the communications were done using a private, dedicated and shared Slack channel, which could be joined and used by all relevant personnel from Cure53 and 1Password.

The discussions throughout the test were very good and productive and not many questions had to be asked. Ongoing interactions positively contributed to the overall outcomes of this project. The scope was well-prepared and clear, greatly contributing to the fact that no noteworthy roadblocks were encountered during the test.

Cure53 offered frequent status updates about the test and the emerging findings. Live-reporting was deemed unnecessary, since no findings were actually uncovered during this examination.

It is a very rare occurrence that the testing team actually has no findings to report. In other words, neither security vulnerabilities nor even general weaknesses could be spotted as negatively affecting the targets of this *1PW-23* project. Quite clearly, the total absence of problems must be interpreted as a very good sign and showcases that the 1Password 8 mobile applications boast a very good security posture on the whole. The team of 1Password is hereby congratulated for their excellent work on achieving well-secured and robust applications.

Yet, some caveats should be considered in the context of the evaluation. Specifically, while the scope was clearly defined, Cure53 had difficulty in achieving deep-dives into every aspect of the applications, given the significant complexity. Yet, basic coverage levels over all relevant scope items was achieved.

In the following sections, the report will first shed light on the scope and key test parameters, the available testing material, as well as the structure and content of the WPs. A dedicated chapter on test methodology and coverage then clarifies what the Cure53 team did in terms of attack-attempts, coverage and other test-relevant tasks. Especially in light of no findings, this is intended as a way to transparently guide the 1Password through the tasks and attack attempts completed by the testing team.

Finally, the report will close with broader conclusions pertinent to this October 2022 project. Cure53 elaborates on the general impressions and reiterates the verdict about the 1Password 8 mobile applications and code for Android and iOS branches, basing their opinion on the testing team’s observations and collected evidence.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Scope

- **Penetration tests & Audits against 1Password 8 React Native mobile applications**
  - **WP1**: 1Password 8 mobile application codebase & Mobile app for Android
    - **All important material, sources and binaries were shared with Cure53:**
      - App was made available via sharing the respective binary
      - App binary was shared
      - App source code was provided
    - **Special focus areas:**
      - Mobile-specific security development
      - Critical user-flows
      - Features added in 1Password 8 (best effort)
      - Detailed testing notes that supplied further information about the scope and coverage expectations were shared
  - **WP2**: 1Password 8 mobile app for iOS (respective diff between WP1 and WP2)
    - **All important material, resources and binaries were shared with Cure53:**
      - App was made available via a Testflight invitation
      - App binary was supplied
      - App source code was shared
    - **Special focus areas:**
      - Mobile-specific security development
      - Critical user-flows
      - Features added in 1Password 8 (best effort)
      - Detailed testing notes that supplied further information about the scope and coverage expectations were shared

- **Test-accounts utilized in this assessment:**
  - **Test-user creation URL:**
    - https://start.b5test.com/sign-up/plan
  - **Test environment URL:**
    - https://my.b5test.com
  - **Account #1 (personal):**
    - U: cure53_1pw23_1
  - **Account #2:**
    - U: cure53_1pw23_2
  - **Account #3:**
    - U: cure53_1pw23_3

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

- **Relevant areas in the shared sources:**
  - Android biometrics:
    - *foundation/op-system-auth/android.rs*
    - *foundation/op-enclave/keychain_store.rs*
  - Lock screen handling (iOS):
    - *op-auto-lock/apple.rs*
    - *apple/CoreFoundation/CoreFoundation/AutoLockMonitor.swift*
    - *op-app/src/app/backend/lock_screen.rs*
  - Unlocking the app:
    - *data/op-unlock*
    - */op-data-layer/src/unlock.rs*
  - React Native bridging on iOS:
    - *apple/iOS/ReactNative*
    - *apple/1PasswordFrameworks/Common/React Native*
  - React Native bridging on Android:
    - *android/app/src/main/java/com/onepassword/android/itemdetail/ShareItemFragment.kt*
  - Authentication to 1Password:
    - *op-b5-client/src/internal/keyed_server_connection.rs*
    - *op-b5-client/src/internal/unauthorized_session.rs*
    - *op-app/src/app/backend/signin.rs for some sign-in logic*
  - MFA-related authentication to 1Password
    - *ffi/op-core-android/src/lib.rs*
    - *ffi/op-core-apple/src/lib.rs*
    - *op-app/src/b5ui.rs*
  - Imports handling:
    - *op-app/src/app/backend/import.rs*
    - *op-import-export/src/models/csv.rs*
    - *op-import-export/src/models/onepux.rs*
    - *ui/op-import-export-ui/src/import.rs*
  - Data-moving:
    - *data/op-model-impl/src/lib.rs via prepare_copy_single*
    - *op-app/src/app/backend/item_location.rs*
    - *op-ui/src/item_location* (<-- especially in *move_item*)

- **Test-supporting material was shared with Cure53**
- **All relevant sources were shared with Cure53**

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Test Methodology

This section describes the testing methodology and the resulting coverage of the 1Password mobile applications for iOS and Android, detailing the obtained coverage of the key focus areas. The goal is to facilitate comprehension of the tests performed across the components. Results are presented as pertinent to individual areas of the project's security properties. These areas were either selected by Cure53 or singled out by other involved parties as warranting a closer inspection.

Key areas and coverage

For this audit, the testing team focused on the version 8 release of the 1Password mobile applications for iOS and Android. As this audit has been conducted within a limited timeframe, 1Password provided a specific set of critical features and user-flows that should be prioritized during testing. These areas included the most crucial areas of the application flow, as well as new features that have only been added and released in version 8.

Taking the volume of scope and features into account, the testing team prioritized the following components:

- Mobile-specific security development
- Critical user-flows
- Features added in 1Password 8

After a brief comment on the code structure, each of the above components will be featured in a subchapter of this part of the report.

Source code structure

Before diving into the individual areas, it is important to lay out the basic structure of the codebase. Generally speaking, the 1Password software complex consists of two main repositories.

- The repository that includes the 1Password Server API and the 1Password.com web application
- The repository that includes a shared Rust library, along with the individual client applications for desktop, mobile, and browser extensions

The client applications (with the exception of the web application) use a Rust library, which is also responsible for the communication with the 1Password server API and containing the main code of the critical flows that are part of this review.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

In total, the Rust library is spread across more than 1800 files spanning roughly 400.000 LoC. The 1Password applications for iOS and Android are implemented using Kotlin / Swift and React Native.

- **iOS**: The Apple-specific part is located within the *apple/* folder of the provided codebase and is spread across 700 Swift files. It consists of roughly 57.000 LoC.
- **Android**: The Android-specific part is located within the *android/* folder of the provided codebase and is spread across 6000 Kotlin files. It consists of roughly 45.000 LoC.

The *crate* dependencies defined within the *Cargo.lock* file of the provided 1Password source code, which contained a total of 810 dependencies at the time of this assessment, were analyzed using the well-known *cargo audit*[1] utility. This was done to identify security vulnerabilities reported to the RustSec Advisory Database[2]. It must be positively noted, however, that none of the dependencies contained known vulnerabilities of any type.

Manual code auditing & pentesting

This section comments on the code auditing coverage within the areas of special interest. Cure53 documents the steps undertaken during the audit against the 1Password mobile applications, as the apps were investigated through the specified approaches.

Before the actual code audit, the available documentation was extensively studied to obtain a solid overview of the software compound, in particular its architecture. This helped Cure53 to get a grasp on the possibly problematic areas and potential attack surfaces.

A lot of praise should be given here to the 1Password team, which has been very supportive with pointers towards specific folders and files within the provided codebase. This sped up the auditing process and allowed a stronger focus on the areas of interest.

Mobile-specific security development

The following issues were covered in this realm:

- **Biometric authentication bypass**: 1Password pointed out and provided details with regard to the previously identified issue within the Android application (not spotted by Cure53), which was related to a biometric authentication bypass on

[1] https://docs.rs/cargo-audit/latest/cargo_audit/
[2] https://github.com/RustSec/advisory-db/

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

jailbroken phones. This issue has solely been reviewed based on the provided overview of the fix being designed.
  - In Cure53's view, the proposed solution makes a solid impression. It leverages a recommendable usage of StrongBox (fallback to TEE) for generating an AES-GCM symmetric key when enabling biometric authentication, binding it to the biometric settings.
  - Nevertheless, it is highly recommended to review the implemented fix on the source code level as well as to attempt bypasses through dynamic testing once the proposed remediation has been completed.
- **Lock screen handling**: A previous pentest conducted by Cure53 (*1PW-20*) revealed an issue that made it possible to prevent the auto-lock from triggering (see *1PW-20-004*). The fix for this issue has been reviewed and the issue is found to be properly resolved. Apart from reviewing the provided fix, the testing team conducted further tests and code audits against the lock screen handling in general. No additional issues have been spotted here.

The static source code analysis focused on the following files:

- *foundation/op-system-auth/android.rs*
- *foundation/op-enclave/keychain_store.rs*
- *op-auto-lock/apple.rs*
- *apple/CoreFoundation/CoreFoundation/AutoLockMonitor.swift*
- *op-app/src/app/backend/lock_screen.rs*
- *data/op-unlock folder*
- *op-data-layer/src/unlock.rs*

Critical user-flows

The 1Password team defined two critical user-flows within the application, nominating them to be part of this assessment. Therefore, a particular focus has been laid on these areas.

Authentication to 1Password

The authentication to 1Password using a password plus secret only, with and without MFA, has been analyzed. It is important to note that 1Password explicitly excluded the authentication using SSO Identity Providers (e.g., Okta) from this review, as those components have been reviewed in earlier assessments.

The authentication flow using password plus secret key, including multi-factor authentication, has been reviewed at a source code level. A special focus has been laid on logic issues and coding best practices with regard to cryptography and other security-relevant aspects. No issues were spotted in the allotted time.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

The static source code analysis focused on the following files and folders:

- *op-b5-client/src/internal/keyed_server_connection.rs*
- *op-b5-client/src/internal/unauthorized_session.rs*
- *op-app/src/app/backend/signin.rs for some sign-in logic.*
- *ffi/op-core-android/src/lib.rs*
- *ffi/op-core-apple/src/lib.rs*
- *op-app/src/b5ui.rs*
- *op-b5-client crate.*

Vault access and unlock

The vault unlocking code for iOS and Android has been reviewed, with an emphasis on logic vulnerabilities that may allow attackers (also the unprivileged ones) to get access to user-credentials. No issues were spotted in the allotted time.

The static source code analysis focused on the following files:

- *data/op-unlock*
- */op-data-layer/src/unlock.rs*

Features added in 1Password 8

1Password provided a list of features added in 1Password 8[3], along with a brief description. Whenever applicable, 1Password specified what concerns exist for a given element/feature. For some of the new features, it was required to start using the feature on the web (e.g., this was the case for data imports), but then they needed to be examined for the results of that feature within the mobile applications. The following features have been reviewed as part of this test.

Move items / share items

The 1Password feature for sharing and moving items, as its name denotes, lets the user move items or share them with other users, even if they do not use 1Password.

The static source code analysis focused on the following files:

- **Item-sharing:**
  - *op-app/src/app/backend/share_item.rs*
  - *op-crypto/src/itemshare.rs*
  - *op-item-format/src/item.rs*

[3] Q4-22_1Password_for_iOS_and_Android.pdf

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

- **Moving-data:**
  - *data/op-model-impl/src/lib.rs via prepare_copy_single*
  - *op-app/src/app/backend/item_location.rs*
  - *op-ui/src/item_location (especially in move_item)*

Various tests around specific areas of concerns were invoked. These included - but were not limited to- the attempts discussed next.

Cure53 wished to verify whether it is possible for a user (authenticated or unauthenticated) to access an item they do not have access to. In particular, the following tests were performed:

- An item was shared by creating a public link, which allows anybody having access to this link to view the shared item. No issues were spotted here and anybody having access to the generated link can view the shared item in the defined time frame / until the expiration time has elapsed. Once the link expires, access to the shared item is successfully prevented.
- The additional feature when sharing an item specifies that the item can only be viewed once. This was tested and it was verified that any further attempts to access this item are properly prohibited.
- When sharing an item with a specific person, for instance, the targeted user is required to first verify their email address before being able to view the item. Any attempts to access the shared item as another user, which was not defined as the recipient when sharing the item, failed. Only the targeted user is able to view the shared item once they verified the email address by providing the correct 6-digit verification code sent to the specified email address.

Data imports
The static source code analysis focused on the following files:

- *op-app/src/app/backend/import.rs*
- *op-import-export/src/models/csv.rs*
- *op-import-export/src/models/onepux.rs*
- *ui/op-import-export-ui/src/import.rs*

For data imports, the following test were covered:

- Cure53 wanted to know if it is possible to import malicious data or malicious files that can be used for further exploitation of 1Password. No promising-looking attacks were spotted in the allotted time frame.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

- The testers attempted to upload files containing malicious content, e.g., the well-known EICAR test files. This was not possible and was prohibited, as assured by the fact that the upload functionality properly validates the allowed file-types that were going to be uploaded.
- Additionally, tampering with UUIDs has been performed in the context of uploading files and data. This did not lead to any vulnerabilities. The idea was to confuse data import features, e.g., attempting to upload files into the vault of other users.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Conclusions

This audit featured the 1Password mobile applications for iOS and Android. The main conclusion of this *1PW-23* examination is that the scope is robust, as evidenced by Cure53 not reporting any security-relevant discoveries.

The testing team responsible for completing the project in October 2022, was granted access to the 1Password codebase, which hosts all the code for the Android and iOS mobile applications. Means to reach the backend were also provided, indicating that three members of the Cure53 team could carry out the required tasks.

In order to maximize the outcome of this testing round, Cure53 got access to a *pdf* document detailing the scope, so that all newly added features and key areas could be discussed from a meta-level perspective first. 1Password has been very supportive in identifying specific folders and files within the 1Password codebase. This helped the testing team to speed up the auditing process and focus on the areas of interest. Moreover, Cure53 was in constant communication with 1Password through a dedicated Slack channel and provided frequent status updates. The communication was excellent and assistance was provided whenever requested.

The code reviewed by the Cure53 team left a positive impression and is of good quality. This confirms the observations collected during previous assessments of the 1Password mobile applications for iOS and Android. It was also positively noted that one important finding from the past (*1PW-20-004),* which is also part of the key areas delineated for this assignment, has been properly resolved.

Based on the code review, it must be stated that the 1Password developers have a good grasp of security concepts. The core part of 1Password is written in Rust, which is a language with built-in memory management that can be both safe and unsafe depending on how it is used. It has proven to be a good choice for programmers that do not want to worry about dangling pointers, use-after-free and data race vulnerabilities. In the case of the codebase presented by 1Password for review, it quickly became evident that secure programming guided the development and resulted in a robust state of the code.

As already explained, this round of testing focused on pre-defined focus areas. In particular, Cure53 needed to review an integrated fix related to iOS lock screen handling, carry out a review of the critical user-flows (authentication and vault access + unlock), as well as was tasked with the verification of some newly added features. However, from the list of newly added features Cure53 reviewed and tested the two most important items, namely “move item / share items” and “data import”.

Dr.-Ing. Mario Heiderich, Cure53
Bielefelder Str. 14
D 10709 Berlin
cure53.de · mario@cure53.de

Due to the immense amount of code to be audited and the limited time allocated to this engagement, it is important to stress that this round of testing followed the best effort principle and, therefore, cannot be considered as having exhausted all attack possibilities or avenues for weaknesses. For example, even though the 1Password team was narrowing down the scope to be audited to the file and folder level, the components and critical user-flows still relied on 1Password core. Hence, a comprehensive examination requires a staggering amount of time to fully understand the big picture, including how the key focus areas fit into the overall architecture.

In terms of the security posture in key areas, it can be concluded that the main items and aspects make an exceptionally good impression. Notably, such a great outcome of no findings is not uncommon for 1Password projects. Since the outcome of this project did not result in any new vulnerabilities, Cure53 decided to elaborate on the steps and methodology used during this round of testing in the dedicated Methodology chapter.

Dynamic testing on iOS revealed that installing the iOS application on a jailbroken device is impossible, since only iOS version 15.5 or newer are supported. Otherwise, older releases would have allowed for a deeper inspection and the simulation of real-world adversaries having privileged access to a victim's device running the 1Password application. While there is no public jailbreak available for iOS version 15 and newer, this might happen in the future.

In conclusion, the items placed in scope for this autumn 2022 project left a very positive impression on the Cure53 testers. Nevertheless, it must be reiterated that the repositories and areas in scope could not all be tested in a comprehensive manner, so the obtained results can only be interpreted in regard to the time-boxed nature of this October 2022 assessment. It must be made absolutely clear however that the tested components appeared safe, yet changes within one part of the 1Password system may have unintentional security impact on other parts, including the 1Password mobile applications.

Cure53 would like to thank the AgileBits Inc. dba 1Password team for their excellent project coordination, support and assistance, both before and during this assignment.