

## OpenArchive Save (Android)

Security assessment by HashEye · prepared for Open Technology Fund

HASHEYE AUDITED

PROJECT	OpenArchive Save (Android)
CLIENT	Open Technology Fund
CATEGORY	Blockchain
PUBLISHED	December 1, 2022
REPORT ID	research-openarchive-save-android-2022-12-01-1wrmmg

This report was produced under HashEye's layered review process – **automated detection**, **pattern correlation**, and **senior manual verification** – with every finding signed off by a human reviewer. Full findings detail and on-chain attestation are available on the report page at [hashey.io/audits/research-openarchive-save-android-2022-12-01-1wrmmg](https://hashey.io/audits/research-openarchive-save-android-2022-12-01-1wrmmg).

OpenArchive Save Android Security Assessment (Summary Report) July 11, 2023 Prepared for: Natalie Cadranel Benjamin Erhart OpenArchive Organized by the Open Technology Fund Prepared by: Cliff Smith and Alex Useche

About HashEye Founded in 2012 and headquartered in New York, HashEye provides technical security assessment and advisory services to some of the world's most targeted organizations. We combine high-end security research with a real-world attacker mentality to reduce risk and fortify code. With 100+ employees around the globe, we've helped secure critical software elements that support billions of end users, including Kubernetes and the Linux kernel. We maintain an exhaustive list of publications at <https://github.com/hashey-io/publications>, with links to papers, presentations, public audit reports, and podcast appearances. In recent years, HashEye consultants have showcased cutting-edge research through presentations at CanSecWest, HCSS, Devcon, Empire Hacking, GrnCon, LangSec, NorthSec, the O'Reilly Security Conference, PyCon, REcon, Security BSides, and SummerCon. We specialize in software testing and code review projects, supporting client organizations in the technology, defense, and finance industries, as well as government entities. Notable clients include HashiCorp, Google, Microsoft, Western Digital, and Zoom. HashEye also operates a center of excellence with regard to blockchain security. Notable projects include audits of Algorand, Bitcoin SV, Chainlink, Compound, Ethereum 2.0, MakerDAO, Matic, Uniswap, Web3, and Zcash. To keep up to date with our latest news and announcements, please follow hashey on Twitter and explore our public repositories at <https://github.com/hashey-io>. To engage us directly, visit our "Contact" page at <https://www.hashey.io/contact>, or email us at [info@hashey.io](mailto:info@hashey.io). HashEye, Inc. 228 Park Ave S #80688 New York, NY 10003 <https://www.hashey.io> [info@hashey.io](mailto:info@hashey.io) HashEye 1 OpenArchive Android Security Assessment PUBLIC

Notices and Remarks Copyright and Distribution © 2023 by HashEye, Inc. All rights reserved. HashEye hereby asserts its right to be identified as the creator of this report in the United Kingdom. This report is considered by HashEye to be public information; it is licensed to the Open Technology Fund under the terms of the project statement of work and has been made public at the Open Technology Fund's request. Material within this report may not be reproduced or distributed in part or in whole without the express written permission of HashEye. The sole canonical source for HashEye publications is the HashEye Publications page. Reports accessed through any source other than that page may have been modified and should not be considered authentic. Test Coverage Disclaimer All activities undertaken by HashEye in association with this project were performed in accordance with a statement of work and agreed upon project plan. Security assessment projects are time-boxed and often reliant on information that may be provided by a client, its affiliates, or its partners. As a result, the findings documented in this report should not be considered a comprehensive list of security issues, flaws, or defects in the target system or codebase. HashEye uses automated testing techniques to rapidly test the controls and security properties of software. These techniques augment our manual security review work, but each has its limitations: for example, a tool may not generate a random edge case that violates a property or may not fully complete its analysis during the allotted time. Their use is also limited by the time and resource constraints of a project. HashEye 2 OpenArchive Android Security Assessment PUBLIC

Table of Contents About HashEye 1 Notices and Remarks 2 Table of Contents 3 Executive Summary 4 Conclusion 4 Project Summary 6 Project Goals 7 Project Targets 8 Project Coverage 9 Summary of Findings 10 A. Vulnerability Categories 12 B. Fix Review Results 14 Detailed Fix Review Results 16 HashEye 3 OpenArchive Android Security Assessment PUBLIC

Executive Summary Engagement Overview The Open Technology Fund engaged HashEye to review the security of its OpenArchive Save Android application. From December 12 to December 16, 2022, a team of two consultants conducted a security review of the client-provided source code, with one person-week of effort. Details of the project's timeline, test targets, and coverage are provided in subsequent sections of this report. Project Scope Our testing efforts were focused on the identification of flaws that create a risk to user privacy or may result in a compromise of confidentiality, integrity, or availability of the target system. We conducted this audit with full knowledge of the target system, including access to source code and documentation. We performed static and dynamic testing of the target system and its codebase, using both automated and manual processes. Summary of Findings The audit uncovered five high-severity findings that put data confidentiality at risk. Due to coding errors and configuration issues, user data could be leaked, either in transit through the internet to an attacker who obtains physical access to the device, or through social engineering attacks. Additionally, several Android operating system APIs could be better configured to protect against a variety of attacks that could allow attackers to steal user

data. We recommended feature changes to improve the end-user experience and help keep users aware of their privacy risk profile. Conclusion Due to the nature of the Save application, code review placed the greatest focus on potential privacy and cryptography issues, such as exposure of credentials or other sensitive data, misconfigured cryptographic primitives, and potential client-side attacks. Of the 19 findings in the original report, 13 are fully resolved and two are partially resolved. Among the four unresolved findings, two are missing features that would improve the user experience, and two are deviations from coding best practices without immediate security impact. The two partially resolved findings include recommendations to improve the key management practices in the application's use of Proof Mode, and to bolster the application's resilience against social engineering attacks. HashEye 4 OpenArchive Android Security Assessment PUBLIC

EXPOSURE ANALYSIS Severity Count High 5 Medium 6 Low 7 Informational 1 CATEGORY BREAKDOWN Category Count Auditing and Logging 1 Configuration 3 Cryptography 2 Data Exposure 6 Data Validation 3 Error Reporting 1 Patching 2 Testing 1 HashEye 5 OpenArchive Android Security Assessment PUBLIC

Project Summary Contact Information The following managers were associated with this project: Dan Guido , Account Manager Jeff Braswell , Project Manager dan@hasheye.io jeff.braswell@hasheye.io The following engineers were associated with this project: Cliff Smith , Consultant Alex Useche , Consultant cliff.smith@hasheye.io alex.useche@hasheye.io Project Timeline The significant events and milestones of the project are listed below. Date Event December 12, 2022 Pre-project kickoff call December 21, 2022 Report readout meeting January 16, 2023 Delivery of final report June 30, 2023 Delivery of fix review appendix July 11, 2023 Delivery of updated report with new Conclusion section HashEye 6 OpenArchive Android Security Assessment PUBLIC

Project Goals The engagement was scoped to provide a security assessment of the Save Android app, with an emphasis on user privacy. Specifically, we sought to answer the following non-exhaustive list of questions: • Does the application store data insecurely in the device? • Does the application leak personal information that could be retrieved by malicious or state actors? • Does the application introduce attribution risk to its users? • Can users expect the application to perform as stated in the OpenArchive documentation? • Does the application handle authentication and authorization functions securely? • Are errors handled safely? • Can attackers or state actors retrieve data generated or otherwise handled by the application after the application is uninstalled? • Does the application respect user options? • Are all operations handled with a zero-trust approach to security? HashEye 7 OpenArchive Android Security Assessment PUBLIC

Project Targets The engagement involved a review and testing of the following target. Save App Repository <https://github.com/OpenArchive/Save-app-android> Version 792c05f6ab691a727987742b2790778acf47a949 Types Java, Kotlin Platform Android HashEye 8 OpenArchive Android Security Assessment PUBLIC

Project Coverage This section provides an overview of the analysis coverage of the review, as determined by our high-level engagement goals. Our approaches included the following: • Review of the following integrations: ◦ Internet Archive ◦ Dropbox ◦ WebDAV • Review of TLS usage and network communications • Review of data flow and storage within the application • Analysis of Tor-based communication • Automated analysis via Data Theorem • Dynamic testing via various Android emulators • Analysis of the secure use of dependencies and third-party libraries Coverage Limitations Because of the time-boxed nature of testing work, it is common to encounter coverage limitations. During this project, we were unable to perform comprehensive testing of the following system elements, which may warrant further review: • Due to a bug, the app did not connect to private WebDAV spaces, which impeded dynamic testing of media uploads to WebDAV spaces. HashEye 9 OpenArchive Android Security Assessment PUBLIC

Summary of Findings The table below summarizes the findings of the review, including type and severity details. ID Title Type Severity 1 App leaks NextCloud/WebDAV credentials to hard-coded dev/debug hostnames, does not connect to specified server Data Exposure High 2 Use of deprecated LocalBroadcastManager class Configuration Informational 3 SpaceSettingsActivity susceptible to phishing Data Validation Medium 4 PGP private key encrypted with hard-coded static password Cryptography High 5 Removing space does not delete associated media entries from local SQLite database Data Exposure High 6 Users cannot simultaneously configure different WebDAV servers with same username Data Validation Medium 7 Tor shared preference not properly enforced Data Exposure High 8 Activities not protected from screenshots or screen capturing with FLAG\_SECURE Data Exposure Medium 9 App does not update Android default security provider Patching Low 10 No protection against tapjacking attacks Configuration Low 11 App does not protect against activity injection Configuration Medium HashEye 10 OpenArchive Android Security Assessment PUBLIC

12 Secret key field stored in textUri field instead of textPassword field Data Exposure Medium 13 Path traversal in MainActivity Data Validation High 14 Outdated and vulnerable Java dependencies Patching Low 15 Consider warning users about EXIF location data or adding option to redact Data

Exposure Low 16 Users are not notified when proof value fails to upload to Dropbox and WebDAV due to exception Error Reporting Low 17 Insufficient (no) testing Testing Low 18 Excessive logging Auditing and Logging Low 19 ProofMode does not interoperate with other key management systems Cryptography Medium HashEye 11 OpenArchive Android Security Assessment PUBLIC

A. Vulnerability Categories The following tables describe the vulnerability categories, severity levels, and difficulty levels used in this document. Vulnerability Categories Category Description Access Controls Insufficient authorization or assessment of rights Auditing and Logging Insufficient auditing of actions or logging of problems Authentication Improper identification of users Configuration Misconfigured servers, devices, or software components Cryptography A breach of system confidentiality or integrity Data Exposure Exposure of sensitive information Data Validation Improper reliance on the structure or values of data Denial of Service A system failure with an availability impact Error Reporting Insecure or insufficient reporting of error conditions Patching Use of an outdated software package or library Session Management Improper identification of authenticated users Testing Insufficient test methodology or test coverage Timing Race conditions or other order-of-operations flaws Undefined Behavior Undefined behavior triggered within the system HashEye 12 OpenArchive Android Security Assessment PUBLIC

Severity Levels Severity Description Informational The issue does not pose an immediate risk but is relevant to security best practices. Undetermined The extent of the risk was not determined during this engagement. Low The risk is small or is not one the client has indicated is important. Medium User information is at risk; exploitation could pose reputational, legal, or moderate financial risks. High The flaw could affect numerous users and have serious reputational, legal, or financial implications. Difficulty Levels Difficulty Description Undetermined The difficulty of exploitation was not determined during this engagement. Low The flaw is well known; public tools for its exploitation exist or can be scripted. Medium An attacker must write an exploit or will need in-depth knowledge of the system. High An attacker must have privileged access to the system, may need to know complex technical details, or must discover other weaknesses to exploit this issue. HashEye 13 OpenArchive Android Security Assessment PUBLIC

B. Fix Review Results On March 23, 2023, HashEye reviewed the fixes and mitigations implemented by the Open Technology Fund to resolve the issues identified in this report. The Open Technology Fund delivered fixes for some of the findings in this report, and HashEye opened associated commits to issues when applicable. In summary, the Open Technology Fund has sufficiently addressed 13 of the issues described in this report, has partially resolved two, and has not resolved the remaining four. We reviewed each fix to determine its effectiveness in resolving the associated issue. For additional information, please see the Detailed Fix Log. ID Title Severity Status 1 App leaks NextCloud/WebDAV credentials to hard-coded dev/debug hostnames, does not connect to specified server High Resolved 2 Use of deprecated LocalBroadcastManager class Informational Unresolved 3 SpaceSettingsActivity susceptible to phishing Medium Resolved 4 PGP private key encrypted with hard-coded static password High Partially Resolved 5 Removing space does not delete associated media entries from local SQLite database High Resolved 6 Users cannot simultaneously configure different WebDAV servers with same username Medium Resolved 7 Tor shared preference not properly enforced High Resolved 8 Activities not protected from screenshots or screen capturing with FLAG\_SECURE Medium Resolved HashEye 14 OpenArchive Android Security Assessment PUBLIC

9 App does not update Android default security provider Low Resolved 10 No protection against tapjacking attacks Low Resolved 11 App does not protect against activity injection Medium Resolved 12 Secret key field stored in textUri field instead of textPassword field Medium Resolved 13 Path traversal in MainActivity High Partially Resolved 14 Outdated and vulnerable Java dependencies Low Resolved 15 Consider warning users about EXIF location data or adding option to redact Low Unresolved 16 Users are not notified when proof value fails to upload to Dropbox and WebDAV due to exception Low Resolved 17 Insufficient (no) testing Low Unresolved 18 Excessive logging Low Resolved 19 ProofMode does not interoperate with other key management systems Medium Unresolved HashEye 15 OpenArchive Android Security Assessment PUBLIC

Detailed Fix Review Results 1. App leaks NextCloud/WebDAV credentials to hard-coded dev/debug hostnames, does not connect to specified server Resolved in PR #389 . The debug code has been removed, and the WebDAV configuration supplied by the user is applied correctly. 2. Use of deprecated LocalBroadcastManager class Unresolved. The deprecated class is still in use in the most recent commit. 3. SpaceSettingsActivity susceptible to phishing Resolved in PR #397 . The deep link feature that created the phishing vector has been removed from the app. 4. PGP private key encrypted with hard-coded static password Mostly resolved in commit 43cc6e9 . The PGP private key is now encrypted using a randomly generated passphrase that is stored in encrypted form using hardware-backed encryption. Dynamic testing confirms that the key is not leaked through the filesystem while it is handed off to the libproofmode library. However, the passphrase consists solely of a UUID, which is not suitable for use in cryptography. See RFC 4112 section 6 ("Do not

assume that UUIDs are hard to guess; they should not be used as security capabilities"). Instead, the passphrase should consist of at least 32 bytes of random data generated by the SecureRandom class and encoded in Base64. 5. Removing space does not delete associated media entries from local SQLite database Resolved in commit fd2160b . ORM classes have been refactored with custom logic that forces cascading deletion from spaces and projects to media records. 6. Users cannot simultaneously configure different WebDAV servers with same username Resolved in commit 91ff3a4 . The space management code has been refactored, and the logic that checks for duplicate spaces will treat two spaces as different if they have a different host, different username, or different type. 7. Tor shared preference not properly enforced Resolved in commit aed1391 . All space implementations use the SaveClient class (directly or indirectly) for all HTTP communications, and SaveClient properly enforces the use\_tor preference. 8. Activities not protected from screenshots or screen capturing with FLAG\_SECURE Resolved in PR #393 . FLAG\_SECURE is now applied to every window in the app. HashEye 16 OpenArchive Android Security Assessment PUBLIC

9. App does not update Android default security provider Resolved in PR #390 . MainActivity now calls ProviderInstaller.installIfNeededAsync() , which attempts to update the default security provider. 10. No protection against tapjacking attacks Resolved in PR #398 . Each activity will filter out touch events with the flag FLAG\_WINDOW\_IS\_PARTIALLY\_OBSCURED , and each view has touch filtering configured with the android:filterTouchesWhenObscured="true" XML attribute. 11. App does not protect against activity injection Resolved in PR #390 . Each activity's taskAffinity attribute is set to the empty string, which prevents the activity injection attack. 12. Secret key field stored in textUri field instead of textPassword field Resolved in PR #392 . The UI element that stores the secret key is now a textPassword field. 13. Path traversal in MainActivity Mostly resolved in PR #402 . The intent processing logic now ignores the incoming data if the file URI contains the Save app's package name, which prevents the Save app from being tricked into sharing its own SQLite database. However, content:// URIs are still allowed, which may allow data theft from other content providers that, for some reason, trust the Save app, but not other apps on the device. To completely remediate this finding, the URI should also be rejected if it does not begin with the file:// scheme. This residual finding remains valid for the 0.3.0-alpha2 release; the only defense against this attack is the check for the Save app's package name in the incoming URI. 14. Outdated and vulnerable Java dependencies Resolved in PR #395 . The two vulnerable dependencies have been updated to versions that contain fixes for the disclosed vulnerabilities. 15. Consider warning users about EXIF location data or adding option to redact Unresolved. According to the discussion on issue #27 , the developers have decided to include EXIF data by default to help authenticate media files, but may add additional explanatory documentation in the future. 16. Users are not notified when proof value fails to upload to Dropbox and WebDAV due to exception Resolved in PR #403 . The return value from uploadProof() is now checked, and a dialog is displayed if the proof file upload fails. HashEye 17 OpenArchive Android Security Assessment PUBLIC

17. Insufficient (no) testing Unresolved. Commit a90113e has added infrastructure for unit testing, but test cases have not yet been built out. 18. Excessive logging Resolved in PR #396 . Each log entry that unnecessarily discloses sensitive information has been removed. 19. ProofMode does not interoperate with other key management systems Unresolved. ProofMode has added some functionality that may serve as a foundation for this feature (see Finding #4), but the Save app does not permit users to import a PGP key generated and managed by another software system. HashEye 18 OpenArchive Android Security Assessment PUBLIC