HashEye

// SMART CONTRACT SECURITY AUDIT

OpenArchive Save (iOS)

Security assessment by HashEye · prepared for Open Technology Fund

HASHEYE AUDITED

PROJECT	OpenArchive Save (iOS)
CLIENT	Open Technology Fund
CATEGORY	Blockchain
PUBLISHED	October 1, 2022
REPORT ID	research-openarchive-save-ios-2022-10-01-gj2qmq

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 hasheye.io/audits/research-openarchive-save-ios-2022-10-01-gj2qmq.

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

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/hasheye-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, GrrCon, 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 hasheye on Twitter and explore our public repositories at https://github.com/hasheye-io . To engage us directly, visit our "Contact" page at https://www.hasheye.io/contact , or email us at info@hasheye.io . HashEye, Inc. 228 Park Ave S #80688 New York, NY 10003 https://www.hasheye.io info@hasheye.io HashEye 1 OpenArchive iOS Security Assessment PUBLIC

 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 iOS Security Assessment PUBLIC

Table of Contents HashEye 3 OpenArchive iOS Security Assessment PUBLIC

Executive Summary Engagement Overview The Open Technology Fund engaged HashEye to review the security of its OpenArchive Save application for iOS. From October 17 to October 24, 2022, a team of two consultants conducted a security review of the client-provided source code, with two person-weeks 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 did not uncover any significant flaws or defects that could impact system confidentiality, integrity, or availability. However, the audit did uncover a significant flaw that creates a privacy risk for users. Additionally, the audit uncovered some opportunities for improvement from a code maturity perspective. General recommendations include more robust unit testing, CI/CD improvements to remove unnecessary details (such as IP addresses

used for development) from the source tree, and more explicit documentation regarding security guarantees and limitations. 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 11 findings in the original report, eight are fully resolved. All high-severity findings from the original report were resolved, one low-severity finding is partially resolved, and one low-severity finding remains unresolved. Fixing these two findings will help to mitigate attacks that hinder user privacy and reduce the risk of leaking sensitive information to attackers who gain access to the device's filesystem. HashEye 4 OpenArchive iOS Security Assessment PUBLIC

EXPOSURE ANALYSIS Severity Count High 3 Medium 1 Low 5 Informational 2 CATEGORY BREAKDOWN Category Count Access Controls 1 Auditing and Logging 1 Configuration 3 Data Exposure 6 HashEye 5 OpenArchive iOS 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: Alex Useche , Consultant Will Brattain , Consultant alex.useche@hasheye.io will.brattain@hasheye.io Project Timeline The significant events and milestones of the project are listed below. Date Event October 11, 2022 Pre-project kickoff call October 25, 2022 Delivery of report draft February 16, 2023 Delivery of final report with fix review July 11, 2023 Delivery of updated report with new Conclusion section HashEye 6 OpenArchive iOS Security Assessment PUBLIC

Project Goals The engagement was scoped to provide a security assessment of the Save iOS 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? • Could 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 iOS Security Assessment PUBLIC

Project Targets The engagement involved a review and testing of the following target. Save iOS Repository https://github.com/OpenArchive/Save-app-ios Version 9ce7e2755b5d115d9d2d37630de175c9228fcaea Type Swift Platform iOS HashEye 8 OpenArchive iOS 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 iOS simulators • 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: • Analysis of the secure use of dependencies and third-party libraries HashEye 9 OpenArchive iOS 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 Custom keyboards are not disabled Configuration Medium 2 The application does not declare first-party web domains Configuration Low 3 Metadata remains present in cache after disconnecting integration or removing image Data Exposure Low 4 Consider adding an option for requiring a passcode or biometric authentication when re-opening application Access Controls Informational 5 Picture view screen not cleared when app moved to the background Data Exposure Low 6 Logging of Dropbox token Data Exposure High 7 Excessive logging Auditing and Logging Low 8 Identity exposed via WebView persistent data store in IaScrapeViewController.swift Data Exposure High 9 Unencrypted communications permitted Data Exposure High 10 Internal IP address exposed Data Exposure Low 11 NSPinnedDomains property list key not configured for Internet Archive, Dropbox Configuration Informational HashEye 10 OpenArchive iOS 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 11 OpenArchive iOS 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 12 OpenArchive iOS Security Assessment PUBLIC

B. Fix Review Results On January 11, 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 eight of the issues described in this report, has partially resolved one, and has not resolved one. 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 Custom keyboards are not disabled Medium Resolved 2 The application does not declare first-party web domains Low Partially resolved 3 Metadata remains present in cache after disconnecting integration or removing image Low Resolved 4 Consider adding an option for requiring a passcode or biometric authentication when re-opening application Informational Resolved 5 Picture view screen not cleared when app moved to the background Low Unresolved 6 Logging of Dropbox token High Resolved 7 Excessive logging Low Resolved 8 Identity exposed via WebView persistent data store in IaScrapeViewController.swift High Resolved 9 Unencrypted communications permitted High Resolved HashEye 13 OpenArchive iOS Security Assessment PUBLIC

10 Internal IP address exposed Low Resolved HashEye 14 OpenArchive iOS Security Assessment PUBLIC

Detailed Fix Review Results 1. Custom keyboards are not disabled Resolved in commit a485109 . By default, third-party keyboards are now disabled. Users can still configure third-party keyboards if they wish to from the Save application settings. 2. The application does not declare first-party web domains Partially resolved in commit a485109 . The application now declares third-party domains for archive.org . However, it is still possible to navigate to other domains when adding the Dropbox integration. In this case, this is the responsibility of the Dropbox API. Nonetheless, the finding is marked as partially resolved to highlight the potential risk that exists due to the limits imposed by the Dropbox API. 3. Metadata remains present in cache after disconnecting integration or removing image Resolved in commits f762cd2 and b1fe1a5 . Caching was disabled everywhere in the application, so metadata is no longer available when disconnecting integrations or removing files from the application. 4. Consider adding an option for requiring a passcode or biometric authentication when re-opening the application Resolved in commit 9b19e4f . The Open Technology Fund implemented a new feature that allows users to enforce face ID or passcode authentication to be able to use the application. 5. Picture view screen not cleared when app moved to the background Unresolved in commit 40c3794 . The application view continues not to be cleared when the application is moved to the background, based on our testing of version 2.8.2 of the Save application. 6. Logging of Dropbox token Resolved in commit ac84067 . The Dropbox token is no longer logged to STDOUT . 7. Excessive logging Resolved in commit 06fa697 . Calls to print() were replaced with debugPrint() , which only prints to the console when the application runs in debug mode. 8. Identity exposed via WebView persistent data store in IaScrapeViewController.swift Resolved in commit effafae . WKWebView is now configured to use non-persistent storage. 9. Unencrypted communications permitted Resolved in commit 1a8ff12 . The application now enforces the use of HTTPS for network communications. HashEye 15 OpenArchive iOS Security Assessment PUBLIC

10. Internal IP address exposed Resolved. The Open Technology Fund responded with the following: While it may look like it exposes some information, this is merely an internal IP address of a virtual machine running on one dev's machine. The address is semi-ephemeral and doesn't constitute valuable information, but is rather a decoy. As a result, the finding is marked as resolved. HashEye 16 OpenArchive iOS Security Assessment PUBLIC

C. Certificate Pinning Considerations During the review, HashEye noted that the NSPinnedDomains property list key was not configured for Internet Archive or Dropbox. Consequently, integrations for Dropbox and Internet Archive could establish communications with third-party domains. Our initial recommendation was to implement certificate pinning for the expected third-party domains via the NSPinnedDomain s property list key. However, further investigation into this potential issue, as well as discussions we had with OpenArchive, raised concerns related to certificate pinning. While certificate pinning enhances protection from machine-in-the-middle attacks, our discussions with the OpenArchive teams and investigations revealed the following concerns: • The domains for which certificate pinning is suggested are not controlled by the Open Technology Fund or OpenArchive. • As a result, pinning the certificates would require dynamic code to check certificate chains for domains controlled by Dropbox and Internet Archive. This would require a significant amount of infrastructure work that could lead to mistakes and vulnerabilities. • Dropbox controls an entire server farm. Pinning certificates for Dropbox could lead to broken functionality. Other issues pertaining to certificate pinning have been exposed by Digicert in the article “ Stop Certificate Pinning ,” including risks such as the compromise of keys. Additionally, according to the OWASP Mobile Application Security guide, “Pinning is a recommended practice, especially for MASVS-L2 apps. However, developers must implement it exclusively for the endpoints under their control and be sure to include backup keys (aka. backup pins) and have a proper app update strategy.” In this case, the types of apps referred to as “MASVS-L2 apps” are those developed by the following industries (quoted directly from the Mobile Application Security Verification Standard): • Health-Care Industry: Mobile apps that store personally identifiable information that can be used for identity theft, fraudulent payments, or a variety of fraud schemes. For the US healthcare sector, compliance considerations include the Health Insurance Portability and Accountability Act (HIPAA) Privacy, Security, Breach Notification Rules and Patient Safety Rule. • Financial Industry: Apps that enable access to highly sensitive information like credit card numbers, personal information, or allow the user to move funds. These apps warrant additional security controls to prevent fraud. Financial apps need to ensure HashEye 17 OpenArchive iOS Security Assessment PUBLIC

compliance to the Payment Card Industry Data Security Standard (PCI DSS), Gramm Leech Bliley Act and Sarbanes-Oxley Act (SOX). Although the risk of machine-in-the-middle attacks could still exist, we believe it is vital to consider the risk mentioned in the resources above. However, in this case, the OpenArchive Save application does not fit the above categories, and the OpenArchive team does not control the domains for which certificate pinning was initially recommended. As such, the finding was removed from the report. HashEye 18 OpenArchive iOS Security Assessment PUBLIC