Security Auditing Report

Reflect Platform (Web/Electron)

Prepared for: Reflect
Prepared by: Luca Carettoni
Date: 04/07/2021

Table of Contents

Revision History

Version	Date	Description	Author
1	04/06/2021	First release of the final report	Luca Carettoni
2	04/07/2021	Peer Review	John Villamil

Contacts

Company	Name	Email
Reflect	Alex MacCaw	alex@alexmaccaw.com
Doyensec, LLC	Luca Carettoni	luca@doyensec.com

Executive Summary

Overview

Reflect engaged Doyensec to perform a security assessment of the Reflect platform. The project commenced on 03/29/2021 and ended on 04/02/2021 requiring 1 security researcher. The project resulted in 10 findings of which 3 were rated as *medium* severity.

The project consisted of a manual web application / ElectronJS security assessment.

Testing was conducted remotely from Doyensec EMEA and US offices.

Scope

Through meetings with Reflect the scope of the project was clearly defined. We list the agreed upon assets below:

- Reflect Electron Application
- Reflect Web Application
- Kiss-crypto Library

The testing took place in the production environment using the latest version of the software at the time of testing (v1.0.7). In detail, this activity was performed on the following releases:

- https://github.com/team-reflect/reflect
  - 338f48e2995da7a31bd4cdbc64cc1bd9e97b34f8
- https://github.com/team-reflect/reflect-electron
  - b193af3ab0dd2242ca425b81606da47a878af1fc
- https://github.com/team-reflect/kiss-crypto
  - 957b28a2803609f7740a97c31c7c20a7b1894fd5

Scoping Restrictions

During the engagement, Doyensec did not encounter difficulties in testing the application.

Testing focused on the targets listed in the scope section only.

Findings Summary

Doyensec researchers discovered and reported 10 vulnerabilities in Reflect. While most of the issues are departure from best practices and low-severity flaws, Doyensec identified 3 issues rated as *medium* severity.

It is important to reiterate that this report represents a snapshot of the security posture of the environment at a point in time.

The findings included insecure settings in ElectronJs' webPreferences, insufficient validation of browser events that could lead to arbitrary navigation, insecure usage of ElectronJs' openExternal and a Cross-Site Scripting issue. In the current setup, these issues represent a moderate risk and cannot be chained together to compromise the security of the user's workstation.

Overall, the security posture of the Internet-facing APIs was found to be in line with industry's best practices.

At the design level, Doyensec found the system to be well architected. Cryptographic primitives and their usage is sound, with no vulnerabilities or misconfigurations identified.

Recommendations

The following recommendations are proposed based on studying Reflect security posture and vulnerabilities discovered during this engagement.

Short-term improvements

- Work on mitigating the discovered vulnerabilities. You can use **Appendix B** - Remediation Checklist to make sure that you have covered all areas

Long-term improvements

- When deriving Argon2 master-key for note encryption, the same seed is always used. This behavior is weak against pre-computation attacks. Having said that, this issue does not represent a security risk in the current setup and should be considered as an hardening best practice only

Methodology

Overview

Doyensec treats each engagement as a fluid entity. We use a standard base of tools and techniques from which we built our own unique methodology. Our 30 years of information security experience has taught us that mixing offensive and defensive philosophies is the key for standing against threats, thus we recommend a *graybox* approach combining dynamic fault injection with an in-depth study of source code to maximize the ROI on bug hunting.

During this assessment, we have employed standard testing methodologies (e.g. OWASP Testing guide recommendations) as well as custom checklists to ensure full coverage of both code and vulnerabilities classes.

Setup Phase

Reflect provided access to the online environment, source code repository and binary for the macOS application.

No additional setup was required besides standard ElectronJS instrumentation (DeveloperTools, MITM).

Tooling

When performing assessments, we combine manual security testing with state-of-the-art tools in order to improve efficiency and efficacy of our effort.

During this engagement, we used the following tools:

- Burp Suite
- Electronegativity
- Curl, netcat and other Linux utilities

Web Application and API Techniques

Web assessments are centered around the data sent between clients and servers. In this realm, the principle audit tool is the Burp Suite, however we also use a large set of custom scripts and extensions to perform specific audit tasks. We focus on authorization, authentication, integrity and trust. We study how data is interpreted, parsed, stored, and relayed between producers and consumers.

We subvert the client with malicious data through reflected and DOM based Cross Site Scripting and by breaking assumptions in trust. We test the server endpoints for injection style flaws including, but not limited to, SQL, template, XML, and command injection flaws. We look at each request and response pair for potential Cross Site Request Forgery and race conditions. We study the application for subtle logic issues, whether they are authorization bypasses or insecure object references. Session storage and retrieval is scrutinized and user separation is thoroughly tested.

Web security is not limited to popular bug titles. Doyensec researchers understand the goals and needs of the application to find ways of breaking the assumed control flow.

Electron Apps Testing

Doyensec has been the first security company to publish a comprehensive security overview of the Electron framework during BlackHat USA 2017. Since then, we have reported dozens of vulnerabilities in the framework itself and popular Electron-based applications.

Thanks to our research efforts, we have extensive experience in analyzing desktop runtime environments based on web technologies. During our testing effort, we will review security mechanisms that ensure isolation between sites,

facilitate web security protections and prevent untrusted remote content to compromise the security of the host. We write custom tools to map out control flow and study an application's behavior and internals. Mapping out attack surface, whether local or remote, is paramount to a successful engagement. Doyensec studies the application's ecosystem, looking for potential downfalls and misconceptions.

Static analysis and instrumentation are important parts of the testing process we use to test an application's response to untrusted data. Doyensec combines manual security testing with a mature Electron.js security testing tool (https://github.com/doyensec/ electronegativity) which will be customized to meet the needs of the target.

We take apart the application looking for privacy leaks and secrets. Storage, transmission, and protection of user information is critical.