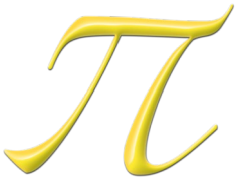

I. Introduction

This document describes the third party software security assessment conducted by Paragon Initiative Enterprises of the JPaseto library.

Our assessment target the Git commit `362924f6b63e695090f37c6e73999361f94404d6` of the master branch on Github.

Report Summary

JPaseto faithfully implements the latest draft of the PASETO specification and uses the published standard test vectors to assure its correctness.

Through the course of our analysis, we did not discover any security vulnerabilities in the JPaseto implementation code.

However, we did discover **1** medium-severity vulnerability in the extension code that integrates JPaseto with BouncyCastle.

II. Vulnerabilities

1. Timing Attack on HMAC Validation for v1.local Decryption

Severity: **Medium**

The `BouncyCastleV1LocalCryptoProvider` class has a method `decrypt()` which is meant to decrypt a ciphertext blob according the PASETO specification for v1 tokens. The PASETO protocol specification says (step 7): “Compare `t` with `t2` using a constant-time string compare function. If they are not identical, throw an exception.”

It’s imperative that the comparison be performed with a constant-time string compare function, since the algorithm in question is AES-256-CTR + HMAC-SHA384.

If a variable-time comparison function is used to verify the HMAC tag, an attacker can learn what the correct HMAC tag should be for a chosen ciphertext based on how long it takes to fail. This can enable an attacker to mint a valid signature for a chosen ciphertext and undermine the integrity of the PASETO.

The implementation currently provided uses `Array.equals()`, which fails fast on the first byte mismatch.

This vulnerability was patched in commit `79df3cbbdf77cfea5e25a12ead81b8837ad5c554`.

III. Other Remarks

In addition to security vulnerabilities, Paragon Initiative Enterprises also looks for non-security bugs in our code audits. We're happy to report that we did not discover any additional bugs in the JPaseto library or the extensions.

Our only additional findings were purely typographical (a typo in README.md, and a variable name that apppears to be a pasto) and had no impact on the implementation.

With the patch we provided for the timing side-channel, we are confident in the correctness and security of JPaseto.