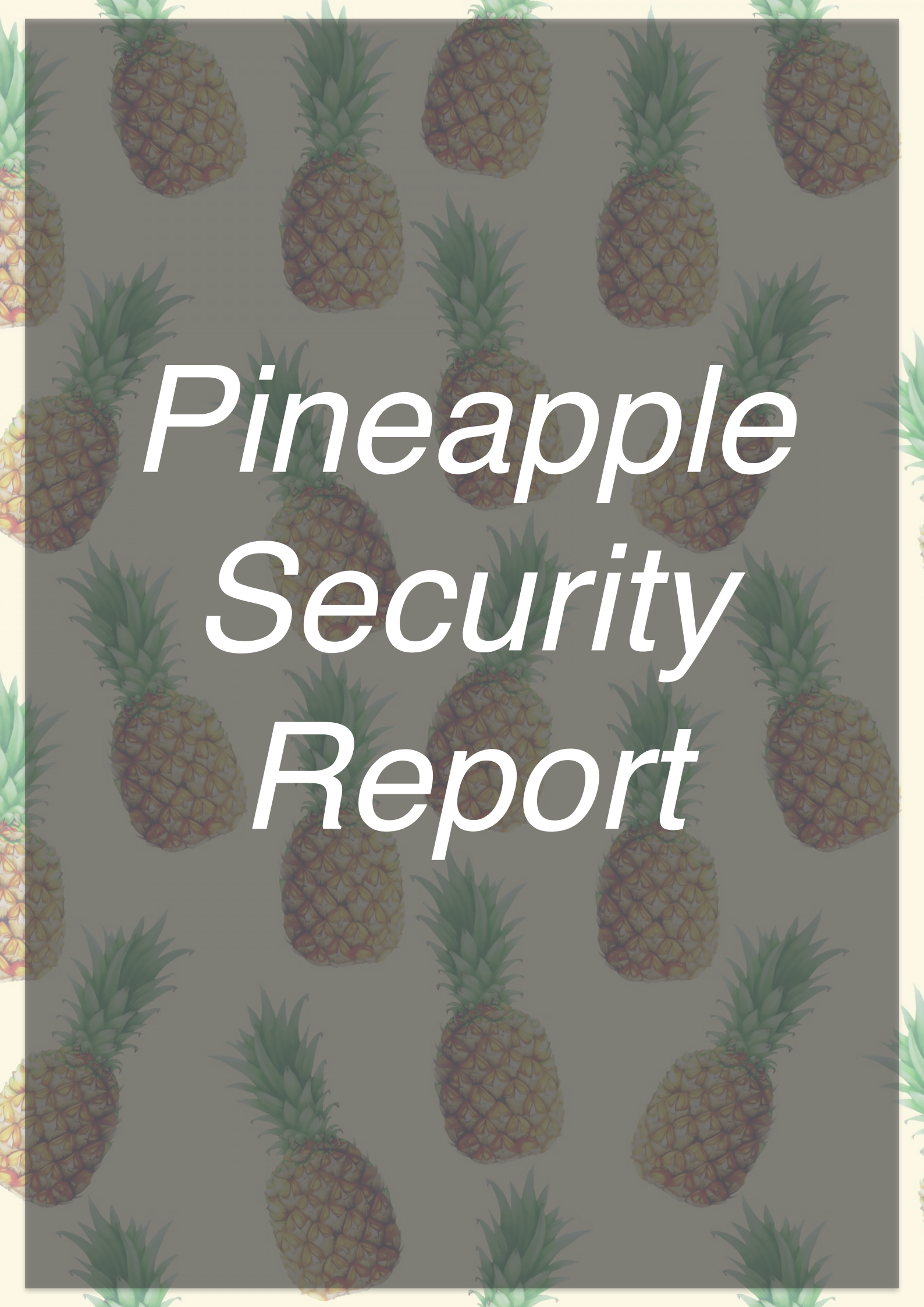
Pineapple
Security
Report

Version: V1.1
Created: 19/05/17
Sponsor: Ali Yavari / Reza Soltanpoor
Author: Paul Davidson

Commerical-in-Confidence

Changelog

Date	Version	Description	Author
19/05/17	1.0	Document Creation	Paul Davidson
22/05/17	1.1	First Draft Finished	Paul Davidson
10/06/17	1.2	Supervisor Changes Added	Paul Davidson

Security Report

In a modern era, Security for Web Applications has become more important than ever.
As OWASP's 2017 Top 10 Report outlines "insecure software is undermining our financial, healthcare, defense, energy, and other critical infrastructure".

And so with this report, I'll aim to outline both the current strengths and weaknesses within our web application, Pineapple, with OWASP's Top 10 Security Risks.

Due to the nature of our chosen web framework, Laravel provides certain security advantages[1].
It has protections for developers against SQL Injection, Cross-Site Request Forgery (CSRF), and Cross-Site Scripting.

A1 (Injection) - SQL Injection is protected by simply encapsulating the supplied data as one string. The example below clearly shows, due to the way in which Laravel validates the data; that this would not be a valid email.

```
SELECT * FROM users WHERE email = 'mrpineapple@gmail.com or 1=1'
```

A8 (Cross-Site Request Forgery) - CSRF is simply preventable through the use of CSRF Tokens, which are generated for every form when included. Only valid tokens can get into the system. We have added tokens to ensure our products security.

A3 (Cross-Site Scripting) - As for Cross-Site Scripting, Laravel doesn't directly insert HTML using the blade brackets syntax. Instead, it is passed to the view, as a String. In our application, we do have points as which we could be vulnerable to this sort of attack; but it is kept to a minimum and not publicly disclosed.

Now for some other security risks:

A6 (Sensitive Data Exposure) - Sensitive Data is not heavily protected due to time constraints of the project. If we had the time, we would have encrypted more of every individuals personal information.

A9 (Using Components with Known Vulnerabilities) - We utilise a variety of components, but have not looked into the various security problems regarding each.

A10 (Underprotected APIs) - We unfortunately have several Unprotected API's for our application, again due to time constraints. If someone was to tap into these specific API's, our game could be manipulated to a various degree. In order to future-proof our game, we would need to invest some time into upgrading our API's to utilise OAUTH (Token based Authentication).

A2 (Broken Authentication and Session Management) - We believe that our session management and authentication is strong enough given time pressures. However it is more than possible that it might be broken, by the right person in the future.

A4 (Broken Access Control) - We believe we don't have any Broken Access Control, allowing only authenticated users into the right places.

All in all, our application should meet the minimum security requirements.

---

[1] http://www.easylaravelbook.com/blog/2015/07/22/how-laravel-5-prevents-sql-injection-cross-site-request-forgery-and-cross-site-scripting/