RD
AUDITORS

GLITTER FINANCE
PENETRATION AND LOAD TESTING
SUMMARY REPORT

Customer:	Glitter Finance
Prepared on:	18th June 2022

THIS DOCUMENT MAY CONTAIN CONFIDENTIAL INFORMATION ABOUT ITS SYSTEMS AND INTELLECTUAL PROPERTY OF THE CUSTOMER AS WELL AS INFORMATION ABOUT POTENTIAL VULNERABILITIES AND METHODS OF THEIR EXPLOITATION.

THE REPORT CONTAINING CONFIDENTIAL INFORMATION CAN BE USED INTERNALLY BY THE CUSTOMER OR IT CAN BE DISCLOSED PUBLICLY AFTER ALL VULNERABILITIES ARE FIXED - UPON THE DECISION OF THE CUSTOMER.

TABLE OF CONTENTS

Summary

The penetration test is performed to identify loopholes in the application from a security perspective. The aim of this assessment is to discover the vulnerabilities present in the user facing platform, which can pose an information security risk.

Overall, we were able to achieve the goals of the assessment and identify vulnerabilities in the target environment within the time window. There are several findings during the assessment for which the details will be provided in the findings section.

Load testing is a non-functional software testing process in which the performance of software application is tested under a specific expected load. It determines how the software application behaves while being accessed by multiple users simultaneously. The goal of load testing is to improve performance bottlenecks and to ensure stability and smooth functioning of software applications before deployment.

- The maximum operating capacity of an application
- Determine whether the current infrastructure is sufficient to run the application
- Sustainability of application with respect to peak user load
- Number of concurrent users that an application can support, and scalability to allow more users to access it.

It is a type of non-functional testing. In software engineering, load testing is commonly used for the client/server, web-based applications – both intranet and internet.

The assessment is performed from 5th to 16th June 2022.

Introduction

Penetration testing checks your organization’s web-facing assets for security vulnerabilities.

A successful pentest does not only identify the vulnerabilities but also finds different ways to exploit them and anticipates the impact on the system.

Website testing is checking your web application or website for potential bugs before it is deployed live and accessible to the general public. Web testing involves checking for functionality, usability, security, compatibility, performance of the web application or website.

During this stage issues such as that of web application security, the functioning of the site, its access to handicapped as well as regular users and its ability to handle traffic is checked.

Load testing is a type of Performance (Non-Functional) testing. Load testing helps us understand the performance or behaviour of an application/server when various loads are applied. Load testing needs to be performed in normal and peak load conditions.

The aim of load testing is to find the system behaviour when different loads are applied.

As websites and web applications get more feature-rich and complex, performance becomes a major concern for developers and users alike. With studies showing that faster sites result in more engaged users, more sales, and increased traffic, it’s important to pay attention to how quickly you can deliver your site to your users and get it rendered in their browser.

The general term for this area of knowledge is web performance optimization, and over the past few years many best practices, techniques, and technologies have been developed to improve the web experience. Many of these techniques focus on reducing the download size of web pages, optimising JavaScript, and limiting the number of individual HTTP requests a page needs.

Vulnerability Severity

High	Medium	Low	Info
0	0	4	13

Low Vulnerability

1. Missing Security Header: Strict -Transport- Security-CSP (Low)

Risk Description:
The Content-Security-Policy (CSP) header activates a protection mechanism implemented in web browsers which prevents exploitation of Cross-Site Scripting vulnerabilities (XSS). If the target application is vulnerable to XSS, lack of this header makes it easily exploitable by attackers.

Recommendation:
Configure the Content-Security-Header to be sent with each HTTP response in order to apply the specific policies needed by the application.

2. Missing security header: X-Frame-Options(Low)

Risk description:
Because the X-Frame-Options header is not sent by the server, an attacker could embed this website into an iframe of a third-party website. By manipulating the display attributes of the iframe, the attacker could trick the user into performing mouse clicks in the application, thus performing activities without user consent (ex: delete user, subscribe to newsletter, etc). This is called a Clickjacking attack.

Recommendation:
We recommend you to add the X-Frame-Options HTTP header with the values DENY or SAMEORIGIN to every page that you want to be protected against Clickjacking attacks.

3. Missing security header: X-XSS-Protection(Low)

Risk description:
The X-XSS-Protection HTTP header instructs the browser to stop loading web pages when they detect reflected Cross-Site Scripting (XSS) attacks.

Lack of this header exposes application users to XSS attacks in case the web application contains such vulnerability.

Recommendation:
We recommend setting the X-XSS-Protection header to X-XSS-Protection: 1; mode=block.

4. Missing security header: Strict-Transport-Security-HTTPS(Low)

Risk description:
The HTTP Strict-Transport-Security header instructs the browser to initiate only secure (HTTPS) connections to the web server and deny any unencrypted HTTP connection attempts. Lack of this header permits an attacker to force a victim user to initiate a clear-text HTTP connection to the server, thus opening the possibility to eavesdrop on the network traffic and extract sensitive information (e.g., session cookies).

Recommendation:
The Strict-Transport-Security HTTP header should be sent with each HTTPS response. The syntax is as follows: Strict-Transport-Security: max-age= [; includeSubDomains] The parameter max-age gives the time frame for requirement of HTTPS in seconds and should be chosen quite high, e.g. several months. A value below 7776000 is considered as too low by this scanner check. The flag includeSubDomains defines that the policy applies also for sub domains of the sender of the response.

Information

- Website is accessible.
- Nothing was found for directory listing.
- Nothing was found for the HttpOnly cookie flag of cookie.
- Nothing was found for a domain too loose set for cookie- Nothing was found for missing HTTP header - Referer.
- Nothing was found for missing HTTP header - X-Content-Type-Options.
- Nothing was found for the absence of the security.txt file.
- Nothing was found for secure communication.
- Nothing was found for enabled HTTP debug methods.
- Nothing was found for use of untrusted certificates.
- Nothing was found for client access policies.
- Nothing was found for vulnerabilities of server-side software.
- Nothing was found for the Secure flag of cookies.

Unique Injection Points Detected: 2
URLs spidered: 8
Total number of HTTP requests: 18

Functionality Testing

There are a total 11 steps needed to be performed by a user for bridging. Every step is working perfectly.

Load & Server Testing

Scenario 1

Traffic testing: 250 users accessing the web application at the same time.

Engine Count:	- 1
Thread Count:	- 250
Iteration:∞	
Location(s): - US East (Ohio)	
Dedicated Ips:	Not Used
VUH:	- 62.5

Test Stats:

Max Users: 250VU	Avg. Throughput: 401.10HITS/SEC
Errors:	0.00%
Avg. Response Time:	- 522.59 MSEC
Avg. Received Bytes/sec:	1.30MB
Avg. Sent Bytes/sec:	71.68KB

Scenario 2

API Testing: 250 Users connecting their wallet at the same time.

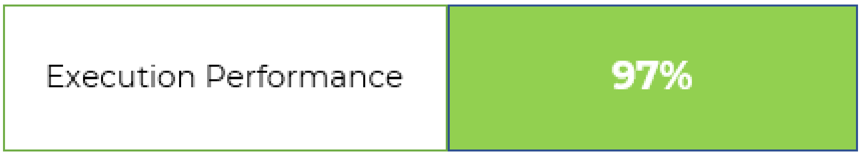
Total Hits: 101848	Avg. Response Time: 589.16MSEC
Max Response: 2062.00MSEC	Min Response: 75.00MSEC
Percentage Error: 0.00%	Total Throughput: 363.98RPS
Avg. Connect Time: 494.99MSEC	Avg. Latency: 589.15MSEC
Total Error Hits: 0	

Test Stats:

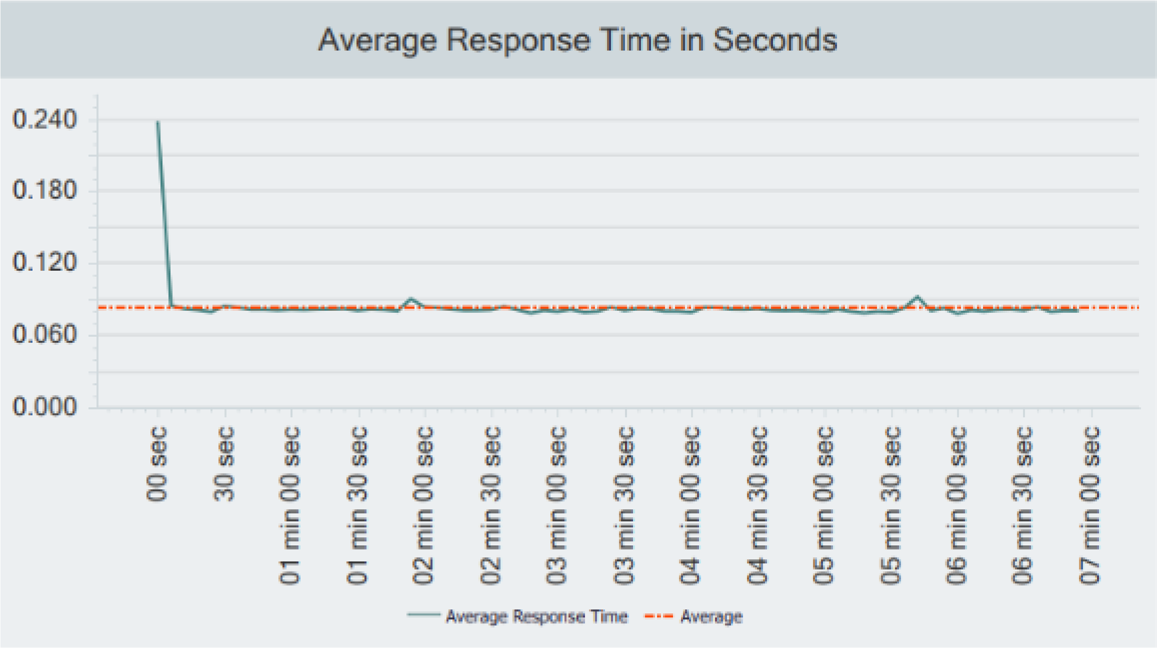
Max Users: 250VU	Avg. Throughput: 363.98HITS/SEC
Errors: 0.00%	Avg. Response Time: 589.16MSEC
Avg. Received Bytes/sec: 241.95KB	Avg. Sent Bytes/sec: 71.09KB

Scenario 3

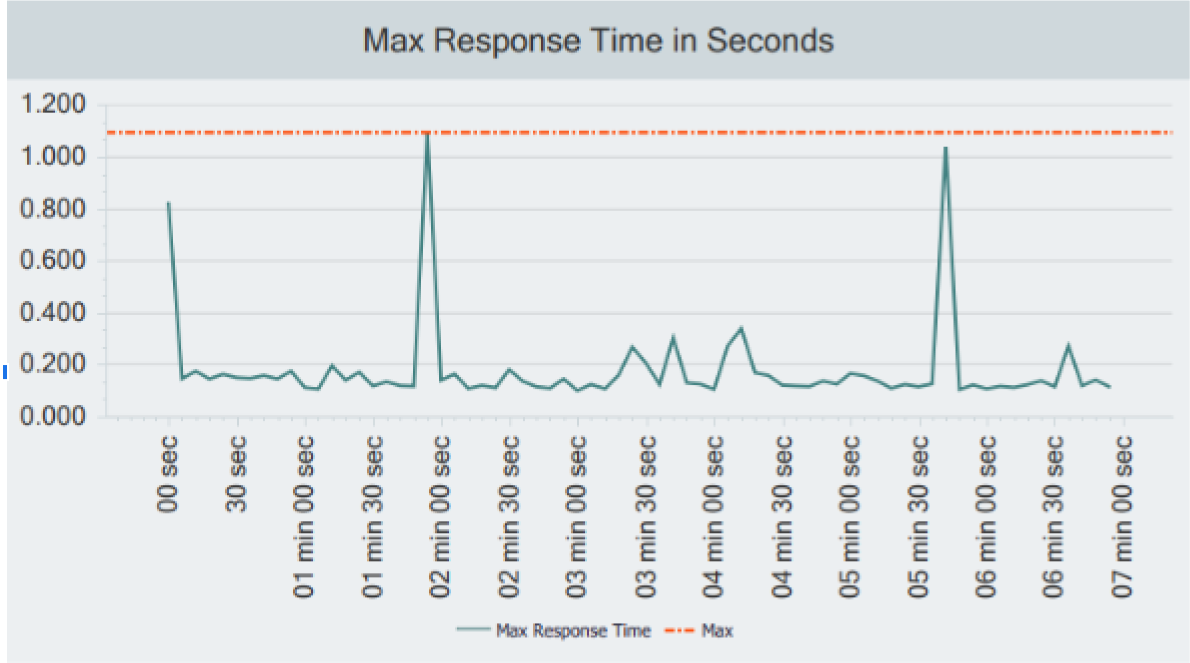
Load Type:	Dynamic Adjustable Curve
Max Users:	100Load Injector
Servers:	2
Successes Sessions:	8940
Failures Sessions:	0
Cpu Limited Sessions:	5
Uncompleted Sessions:	0
Total Sessions:	8940
Average:	0.0826
STDDev:	0.027
Errors:	0
Status:	Finished



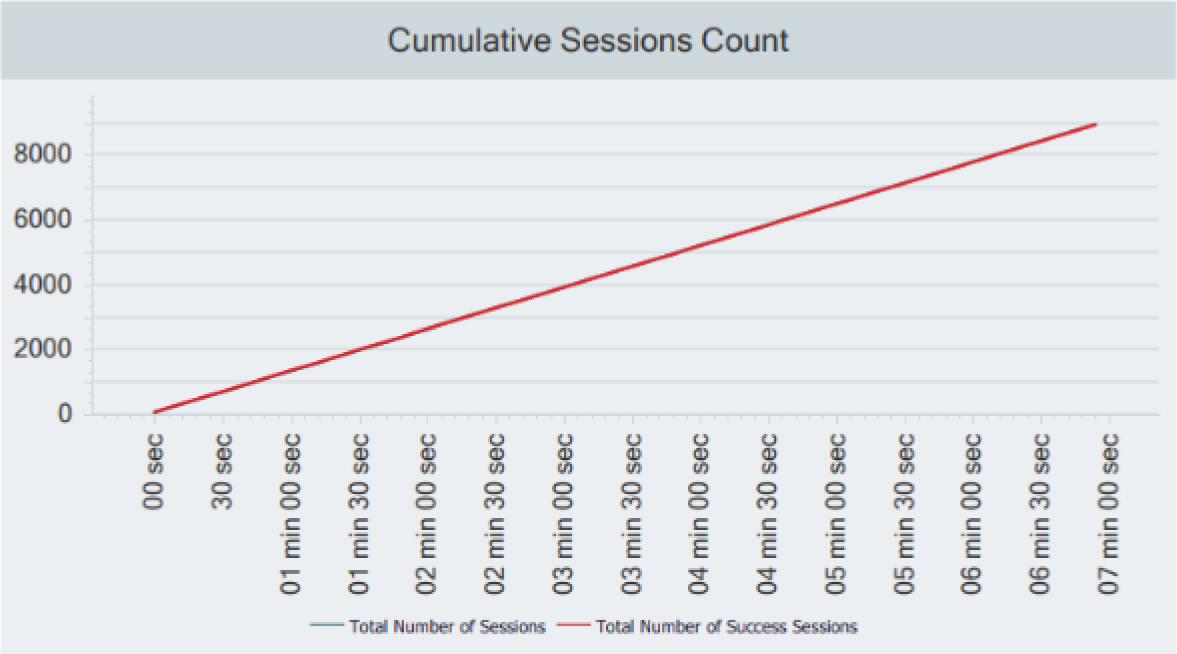
Execution Performance	97%



Average : 0.0826



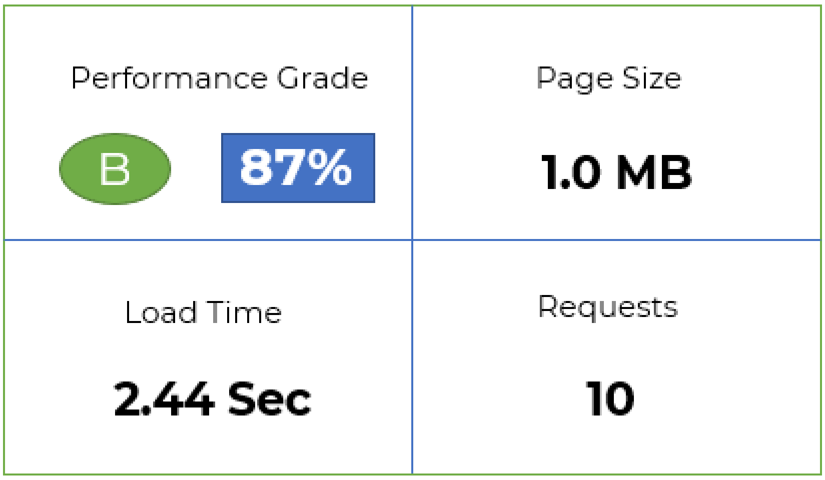
Max : 1.093



SSL SERVER ANALYSE:

SERVER	GRADE
2606:4700:20:0:0:0:ac43:4a68	A
104.26.1.8	A
104.26.0.8	A
2606:4700:20:0:0:0:681a:8	A
2606:4700:20:0:0:0:681a:108	A

PERFORMANCE ANALYSE:



Performance Grade B 87%	Page Size 1.0 MB
Load Time 2.44 Sec	Requests 10

Disclaimers

RD Auditors Disclaimer

The associated code/URLs given for audit have been analysed in accordance with the best industry practices at the date of this report, in relation to cybersecurity vulnerabilities and issues regarding the penetration test, details of which are disclosed in this report.

Because the total number of test cases are unlimited, the audit makes no statements or warranties on the security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bug free status or any other statements of the test.

While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only - we recommend proceeding with several independent audits and a public bug bounty program to ensure security of websites.

RD
AUDITORS

Email: info@rdauditors.com

Website: www.rdauditors.com