QuillAudits

# AUDIT REPORT

April , 2025

For

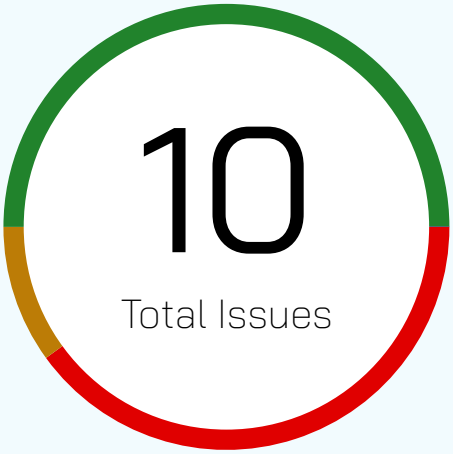
BIGA

# Table of Content

# Executive Summary

| | |
|---|---|
| **Project name** | BIGA arcade |
| **Overview** | BIGA arcade is an innovative world's first P2E based incentive model for games, that elevates gameplay, is sustainable, scalable and promises to socialize a piece of the gaming industry's $300 billion in annual revenues, enabling gamers in a peer to peer environment, to earn a reliable USD denominated income simply by playing games. |
| **Scope of Audit** | Web Application |
| **In Scope** | https://bigarcade.org/ |
| **Review 1** | 3rd March 2025 - 17th March 2025 |
| **Updated changes Received** | 28th March 2025 and 9th April 2025 |
| **Review 2** | 10th April 2025 to 14th April 2025 |
| **Fixed in** | stage.bigarcade.org |

# Number of Issues per Severity




| | |
|---|---|
| High | 4 (40.00%) |
| Medium | 1 (10.00%) |
| Low | 5 (50.00%) |
| Informational | 0 (0.00%) |

Severity

| Issues | High | Medium | Low | Informational |
|---|---|---|---|---|
| Open | 0 | 0 | 0 | 0 |
| Resolved | **3** | **1** | **5** | 0 |
| Acknowledged | **1** | 0 | 0 | 0 |
| Partially Resolved | 0 | 0 | 0 | 0 |

# Checked Vulnerabilities

- Improper Authentication
- Improper Resource Usage
- Improper Authorization
- Insecure File Uploads
- Insecure Direct Object References
- Client-Side Validation Issues
- Rate Limit
- Input Validation
- Injection Attacks
- Cross-Site Scripting (XSS)
- Cross-Site Request Forgery
- Security Misconfiguration
- Broken Access Controls
- Insecure Cryptographic Storage
- Insufficient Cryptography
- Insufficient Session Expiration
- Insufficient Transport Layer Protection
- Unvalidated Redirects and Forwards
- Information Leakage
- Broken Authentication and Session Management
- Denial of Service (DoS) Attacks
- Malware
- Third-Party Components
- And more.

# Techniques and Methods

Throughout the audit of smart contracts, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments, match logic and expected behavior.
- Token distribution and calculations are as per the intended behavior mentioned in the whitepaper.
- Implementation of ERC standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

**The following techniques, methods, and tools were used to review all the smart contracts.**

Structural Analysis

In this step, we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis

A static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

### Code Review / Manual Analysis

Manual Analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

### Gas Consumption

In this step, we have checked the behavior of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

### Tools And Platforms Used For Audit

Remix IDE, Foundry, Solhint, Mythril, Slither, Solidity statistical analysis.

# Types of Severity

Every issue in this report has been assigned to a severity level. There are four levels of severity, and each of them has been explained below

### High Severity Issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality, and we recommend these issues be fixed before moving to a live environment.

### Medium Severity Issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.

### Low Severity Issues

Low-level severity issues can cause minor impact and are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

### Informational Issues

These are four severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

# Types of Issues

**Open**

Security vulnerabilities identified that must be resolved and are currently unresolved.

**Resolved**

Security vulnerabilities identified that must be resolved and are currently unresolved.

**Acknowledged**

Vulnerabilities which have been acknowledged but are yet to be resolved.

**Partially Resolved**

Considerable efforts have been invested to reduce the risk/ impact of the security issue, but are not completely resolved.

# High Severity Issues

## Sensitive Information Disclosure

Acknowledged

**Description**

Sensitive information was being leaked in multiple js files such as REACT_APP_DEFAULT_TOKEN_ADDRESS, REACT_APP_PASSWORD, REACT_APP_CARV_CLIENT_ID, etc. Later the password was used to get access to the staging web application.

**Vulnerable Endpoint**

https://bigarcade.org/static/js/2.9ed994bd.chunk.js
https://bigarcade.org/static/js/main.5414ccee.chunk.js
https://stage.bigarcade.org/static/js/main.58af73b4.js
https://stage.bigarcade.org/static/js/45.6588b500.chunk.js

**Impact**

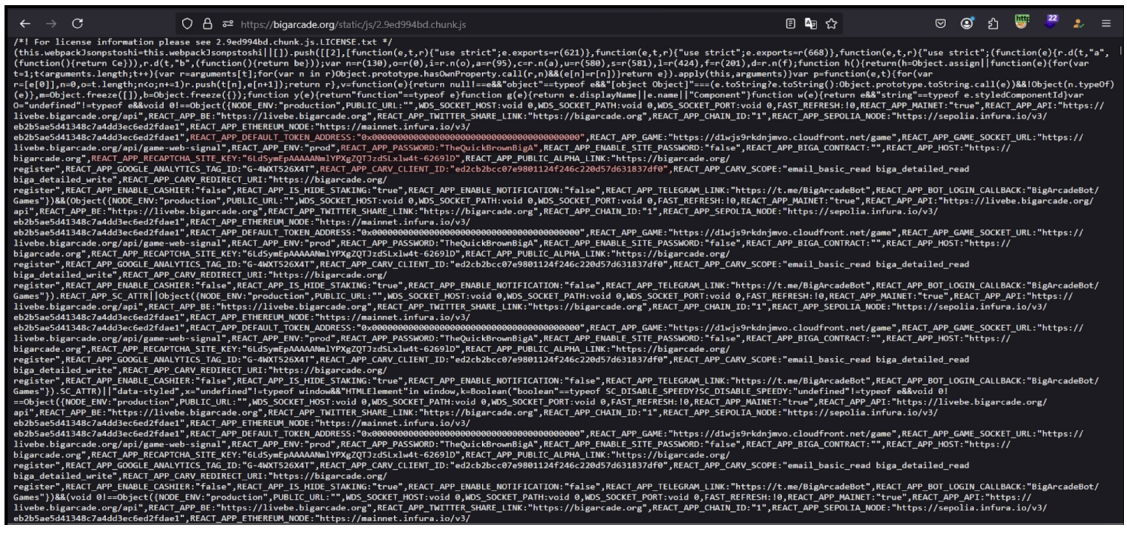
Getting access to the staging application. Staging application is only for internal testing purpose and might contain sensitive information and under development features. It also tends to have higher number of vulnerabilities and pose a potential risk if get exposed.

**Recommendations**

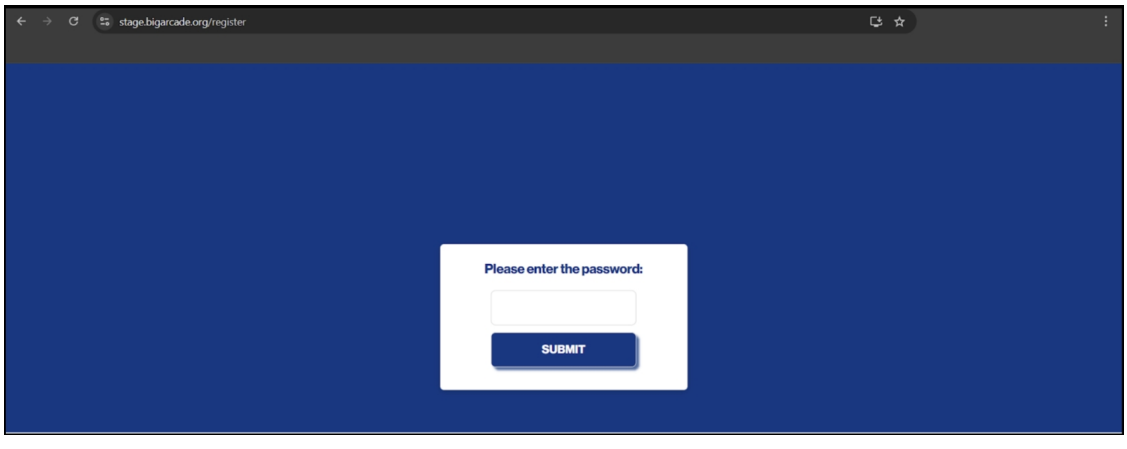
Remove Hardcoded Credentials and shift them to .env

**POC**

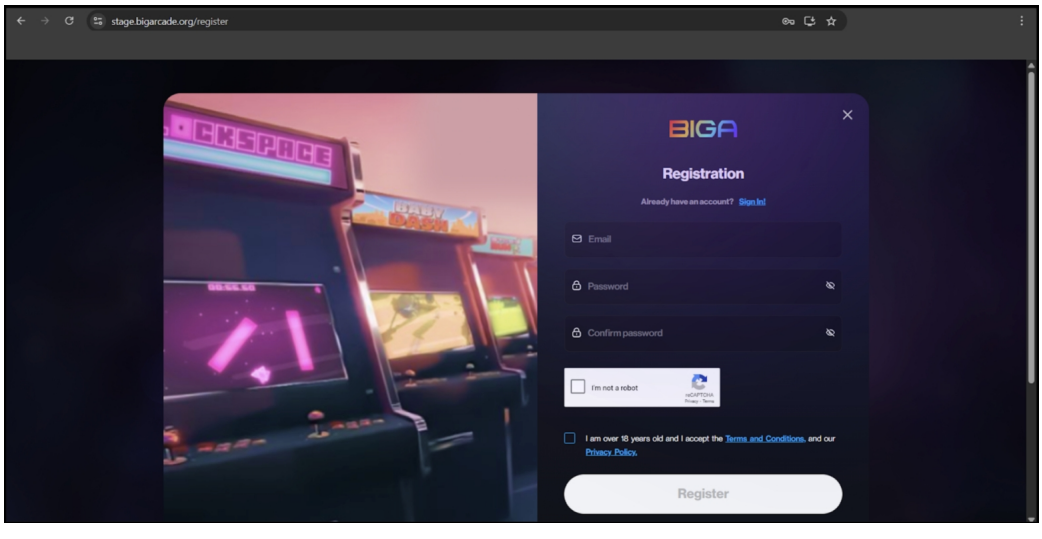
REACT_APP_PASSWORD:"TheQuickBrownBigA"
REACT_APP_DEFAULT_TOKEN_ADDRESS:"
0xcFac93Ed16CE643ea03C0D0001C91736E83223f1" REACT_APP_CARV_CLIENT_ID:"
48e7bf3426120287c15edd132a4ca4575fae383c"



The application asks for password to access https://stage.bigarcade.org/register




Use the password you got in the js analysis and you will get access to the entire application.



## Chained Vulnerabilities Leading to Permanent USER DOS

Resolved

**Description**

the application suffers from multiple vulnerabilities that, when exploited together, can lead to a permanent Denial of Service (DoS) for legitimate users.

Email Enumeration – The “Forgot Password” page reveals whether an email is registered or not, allowing attackers to identify valid and non-registered email addresses.

Account Creation with Non-Registered Emails – Attackers can use non-registered email addresses to create new accounts.

Lack of Rate Limiting on OTP Verification – The application does not enforce rate limits on OTP verification, making it vulnerable to brute-force attacks. Attackers can repeatedly attempt OTPs until verification is successful, enabling automated account creation.

Irreversible Account Deletion – Once logged in, users have the option to permanently delete their account. However, due to improper account handling, deleted accounts cannot be recreated, leading to a permanent lockout.

By automating this attack, an adversary can enumerate all possible email addresses, create accounts, and then delete them, effectively preventing legitimate users from ever registering or using the application.

**Vulnerable Endpoints**

https://bigarcade.org/

**Impact**

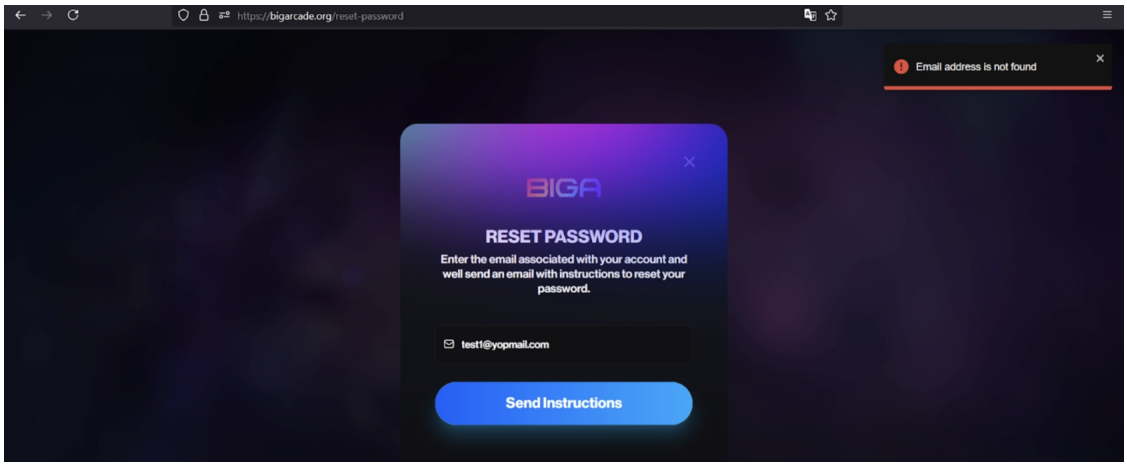
Permanent Denial of Service (DoS) – Legitimate users can be permanently locked out, rendering the application unusable.

Account Takeover Risk – Attackers can create and verify accounts without restriction, increasing the potential for abuse.

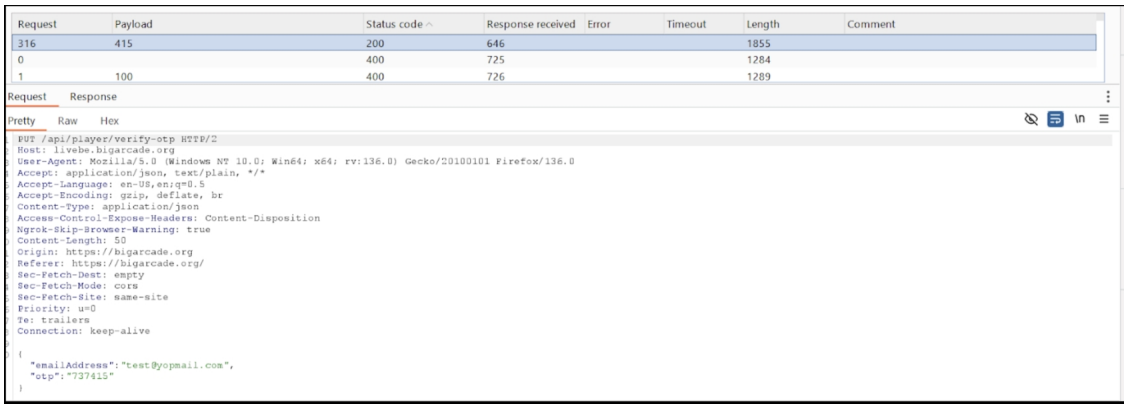
Security & Compliance Issues – The lack of rate limiting and improper account management violate best security practices and may lead to regulatory non-compliance.

**POC**

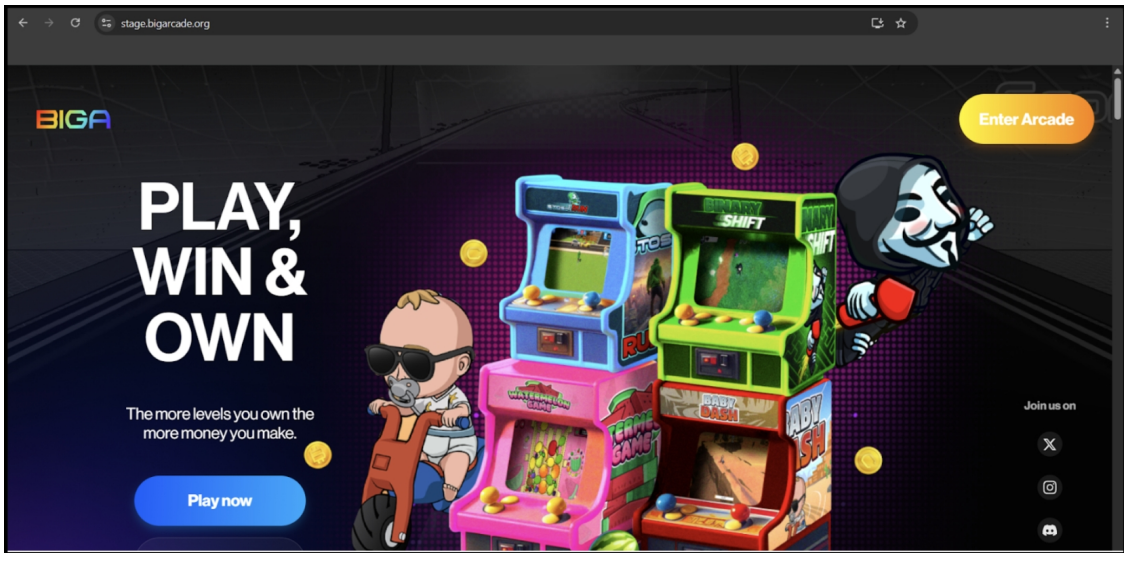
Use password reset page to perform email enumeration. [ Email Address Enumeration – Low ]



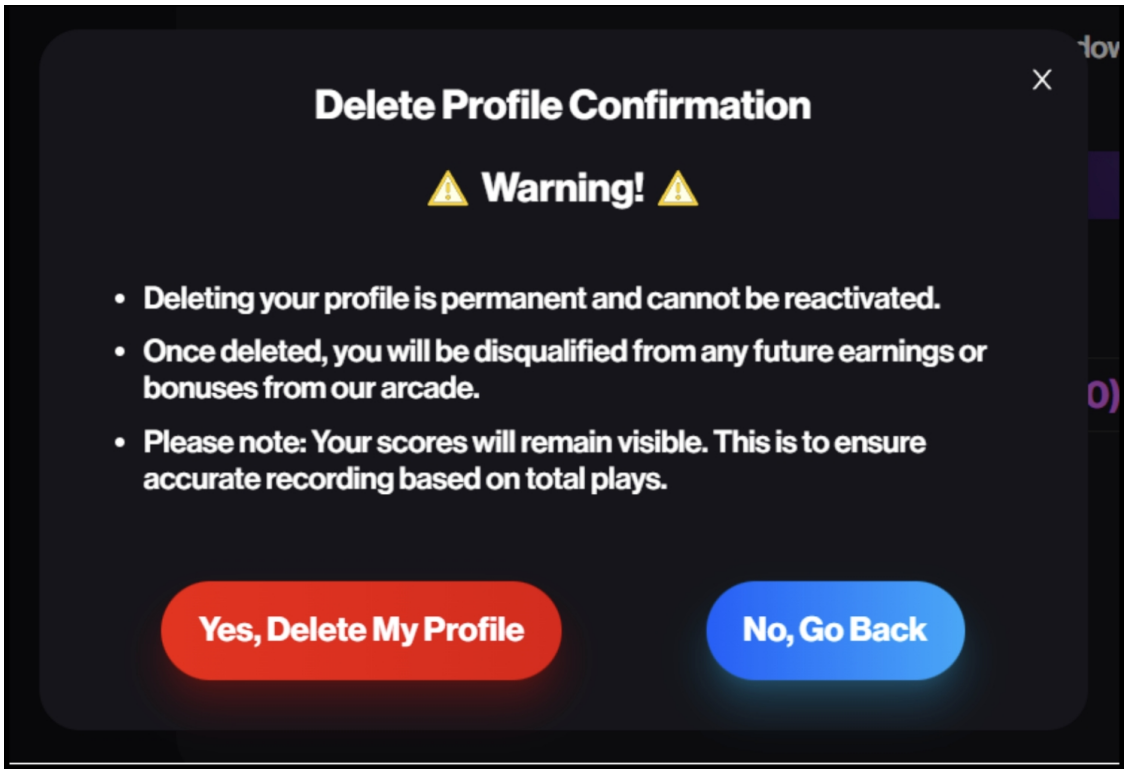

Register using email address which doesn't exists. Since, there is no rate limiting you can bruteforce the otp for bypassing verification. ( OTP Bruteforce on Verify OTP - High)




Login using the signed up credentials at https://stage.bigarcade.org/login



Delete the profile and the user will no longer be able to create the account again. [ No Password & OTP Required for Account Deletion – High ]




**Recommendation**

- Use generic messages for forget password something like “User will receive an email if already registered”

- Setup rate limiting whereever possible such as verify otp, resend otp, send forget password link, etc.

- Implement Account Lockout policy for temporary intervals on multiple login failures.

- Enable 2fa and take confirmation on account delete using password and otp.

- Allow deleted accounts to be recreated or implement a recovery mechanism.

## Login Bruteforce

Resolved

### Description

The application does not enforce rate limiting on login attempts, allowing an attacker to systematically brute-force user credentials. By automating login attempts with commonly used passwords or previously leaked credentials, an attacker can gain unauthorized access to any user account without triggering any security controls.

### Vulnerable Endpoint

https://livebe.bigarcade.org/api/player/login

### Impact

- Increased risk of credential stuffing attacks using breached password databases.
- Compromise of sensitive user information, financial data, or personal communications, depending on the application's functionality.

### POC

Intercept the login request and bruteforce the password.

| Request | Payload | Status code | Response received | Error | Timeout | Length | Comment |
|---|---|---|---|---|---|---|---|
| 877 | 9876 | 201 | 436 | | | 2495 | |
| 0 | | 400 | 491 | | | 1310 | |
| 1 | 9000 | 400 | 603 | | | 1307 | |
| 2 | 9001 | 400 | 583 | | | 1308 | |
| 3 | 9002 | 400 | 694 | | | 1302 | |
| 4 | 9003 | 400 | 618 | | | 1309 | |
| 5 | 9004 | 400 | 652 | | | 1302 | |
| 6 | 9005 | 400 | 645 | | | 1304 | |
| 7 | 9006 | 400 | 631 | | | 1301 | |

Request Response

Pretty Raw Hex

```
POST /api/player/login HTTP/2
Host: livebe.bigarcade.org
Content-Length: 65
Sec-Ch-Ua-Platform: "Windows"
Access-Control-Expose-Headers: Content-Disposition
Sec-Ch-Ua: "Chromium";v="134", "Not:A-Brand";v="24", "Google Chrome";v="134"
Sec-Ch-Ua-Mobile: ?0
Ngrok-Skip-Browser-Warning: true
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/134.0.0.0 Safari/537.36
Accept: application/json, text/plain, */*
Dnt: 1
Content-Type: application/json
Origin: https://bigarcade.org
Sec-Fetch-Site: same-site
```


| Request | Payload | Status code | Response received | Error | Timeout | Length | Comment |
|---|---|---|---|---|---|---|---|
| 877 | 9876 | 201 | 436 | | | 2495 | |
| 0 | | 400 | 491 | | | 1310 | |
| 1 | 9000 | 400 | 603 | | | 1307 | |
| 2 | 9001 | 400 | 583 | | | 1308 | |
| 3 | 9002 | 400 | 694 | | | 1302 | |
| 4 | 9003 | 400 | 618 | | | 1309 | |
| 5 | 9004 | 400 | 652 | | | 1302 | |
| 6 | 9005 | 400 | 645 | | | 1304 | |
| 7 | 9006 | 400 | 631 | | | 1301 | |


Request Response

Pretty Raw Hex

```
Origin: https://bigarcade.org
Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://bigarcade.org/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Priority: u=1, i
Connection: keep-alive

{
  "emailAddress":"daxefok171@excederm.com",
  "password":"Asdf@9876"
}
```

## Recommendation

- Implement rate limiting on login attempts (e.g., limit to 5 failed attempts per minute per IP).
- Implement account lockout or temporary suspension after a certain number of failed attempts.
- Encourage users to enable multi-factor authentication (MFA) to reduce the impact of credential compromise.

## Insecure Account Recovery Restriction Due to Update Limits

Resolved

### Description

The application restricts users from updating their email and password more than once within a 30-day period. This restriction prevents users from recovering their accounts if they get compromised within this timeframe. An attacker who gains access to the account can retain control, as the legitimate user is unable to update credentials or regain access.

### Vulnerable Endpoint

https://livebe.bigarcade.org/api/player/change-password

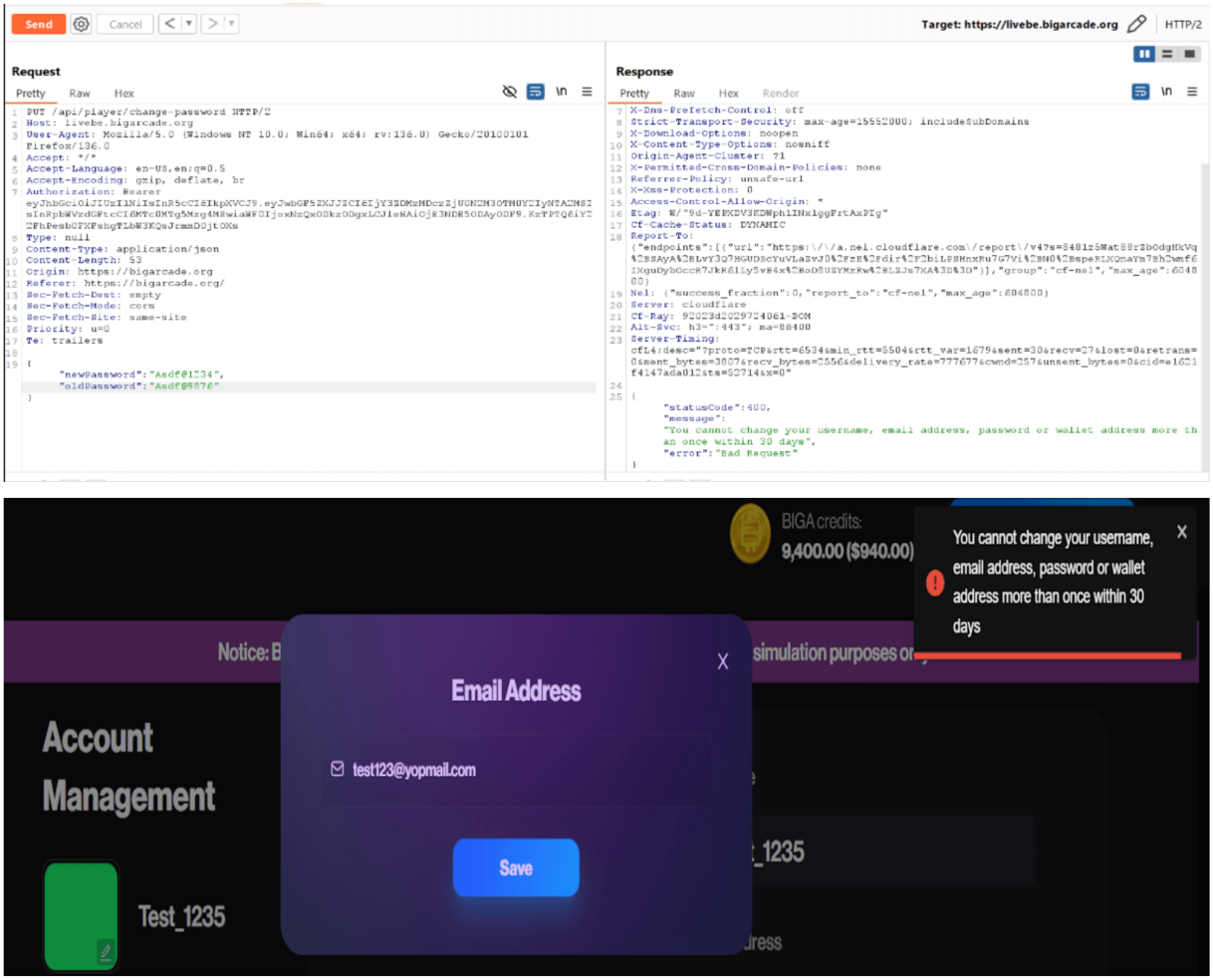
### Impact

1. Users cannot secure their accounts after a compromise, increasing the risk of prolonged unauthorized access.
2. Attackers can maintain control over compromised accounts for an extended period.
3. Legitimate users are locked out without a way to recover their accounts, leading to potential data loss and security risks.

### POC

Password cannot be updated more than once in 30 days.

Email address cannot be updated more than once in 30 days.




## Recommendation

1. Allow users to reset their password and email at any time in case of suspected compromise.
2. Implement an account recovery mechanism, such as identity verification via a secondary email, phone number, or multi-factor authentication (MFA).
3. Introduce risk-based authentication to detect suspicious login attempts and prompt additional verification.
4. Provide emergency support options for users locked out due to security restrictions.

# Medium Severity Issues

## JWT Token Doesn't Expire on Logout

Resolved

### Description

The application does not properly invalidate JWT tokens upon user logout. Even after a user logs out, the token remains valid, allowing an attacker with access to the token to continue making authenticated requests. This increases the risk of session hijacking and unauthorized access if the token is stolen.

### Vulnerable Endpoint

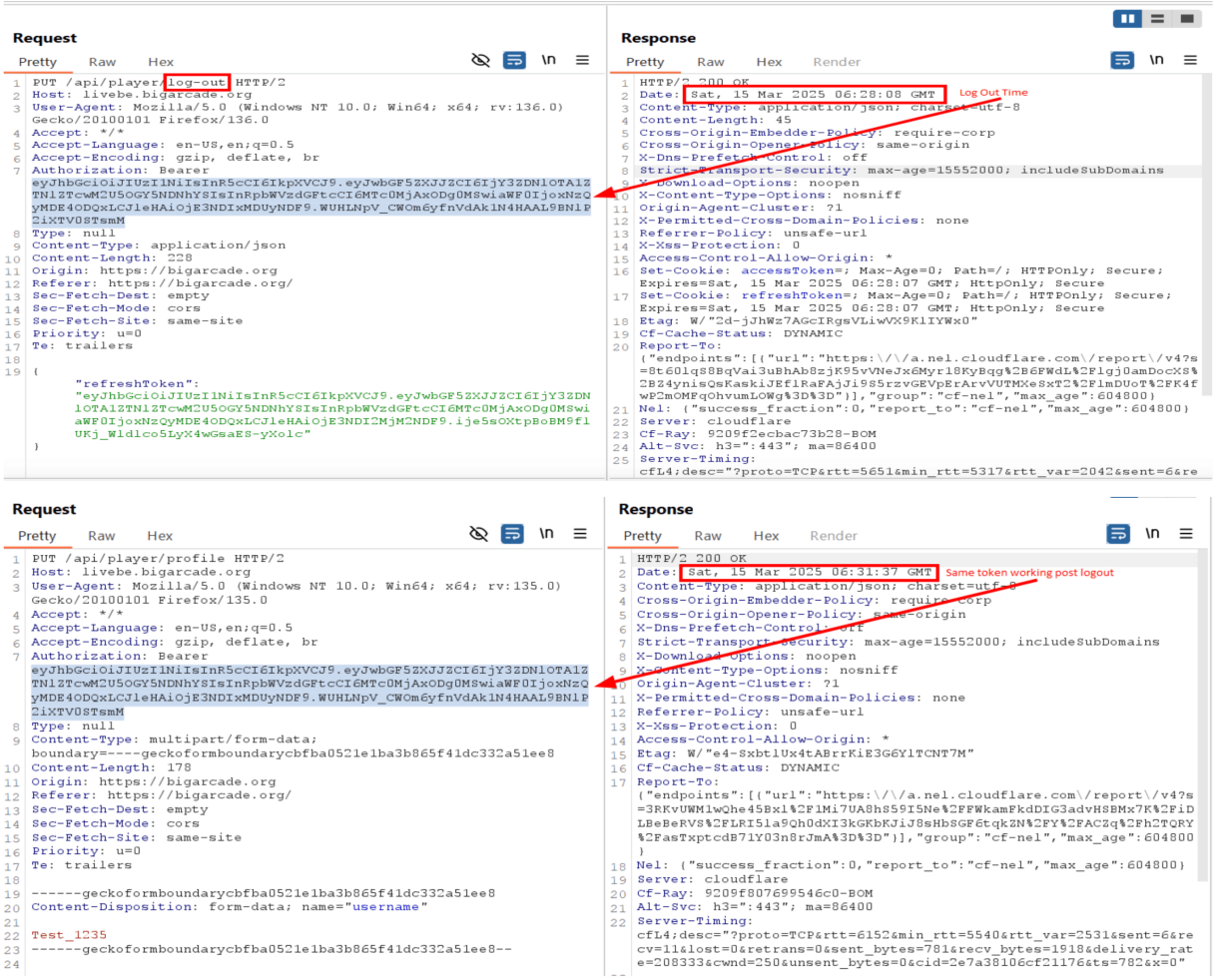
https://livebe.bigarcade.org/api/player/*

### Impact

1. Attackers can reuse a stolen or intercepted JWT token to access user accounts even after logout.
2. Users cannot effectively terminate their sessions, increasing the risk of account compromise.
3. If a user's token is leaked, an attacker can maintain unauthorized access until the token naturally expires.

### POC

1. Log in to the application and capture the JWT token from the authorization header or local storage.
2. Log out of the application.
3. Attempt to use the captured JWT token to access authenticated endpoints.
4. Observe that the token is still valid, and the API responds as if the user is still logged in.
We will be using the same token post logout.




## Recommendation

Use short-lived JWT tokens with refresh tokens to limit exposure.
Invalidate refresh tokens upon logout to prevent reauthentication with an old token.

# Low Severity Issues

## Components with known vulnerabilities

Resolved

### Description

Axios v0.21.4 has multiple known security vulnerabilities, including Cross-Site Request Forgery (CSRF) and CVE-2023-45857. These issues can allow attackers to manipulate requests, perform unauthorized actions, or expose sensitive data.
Cross-Site Request Forgery (CSRF): Axios does not automatically include CSRF protection mechanisms, making it vulnerable to unauthorized request execution if users do not implement additional security measures.
CVE-2023-45857: This vulnerability allows attackers to manipulate HTTP request behavior in a way that could lead to data leakage or unintended server interactions.

### Vulnerable Endpoint

https://bigarcade.org/static/js/2.9ed994bd.chunk.js

### Impact

Attackers may exploit CSRF vulnerabilities to perform actions on behalf of authenticated users without their consent.

### Recommendation

Update to the latest stable version (e.g., v1.6.0 or later) where security patches have been applied.

## Improper CORS Configuration

Resolved

### Description

Cross-Origin Resource Sharing (CORS) is a mechanism that restricts the access of a web page from one domain to resources in another domain. The vulnerability occurs when a web application allows access to resources in a domain that should not be accessible, resulting in information exposure, cross-site scripting (XSS), and other types of attacks.

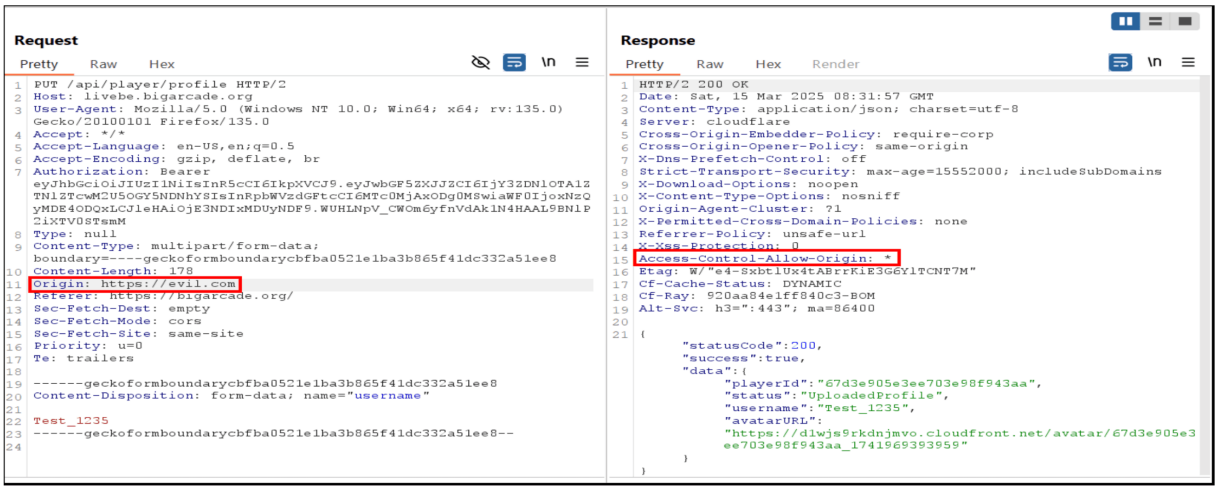
### Impact

CORS vulnerability can lead to sensitive information exposure, cross-site scripting (XSS), and other types of attacks. This can result in unauthorized access to sensitive data, such as user credentials, personal information, and financial data. The impact of this vulnerability can vary depending on the data being exposed and the nature of the attack.

### POC

- Add header "origin:evil.com"
- Check the response headers. You will find atleast one of the following:

1. Access-Control-Allow-Origin: null
2. Access-Control-Allow-Origin: *
3. Access-Control-Allow-Credentials: true



**Recommendation**

To remediate this vulnerability, it is recommended to restrict the domains that are allowed to access resources on your web application. This can be achieved by adding the appropriate CORS headers to the HTTP response from your server. You can also use a Content Security Policy (CSP) to enforce the same-origin policy and prevent the exposure of sensitive data to unauthorized domains. For misconfigured follow the below points :
- Proper configuration of cross-domain requests
- Origins specified in the Access-Control-Allow-Origin header should only be sites that are trusted.
- Avoid using the header Access-Control-Allow-Origin: null.
- Avoid wildcards in internal networks

## Clickjacking

Resolved

### Description

Clickjacking is a type of attack that tricks a user into clicking on a malicious link or button without their knowledge. This can be done by overlaying an invisible layer over a legitimate link or button on a web page, thus making the user unwittingly click the malicious link or button.

### Vulnerable Endpoint

https://bigarcade.org/*

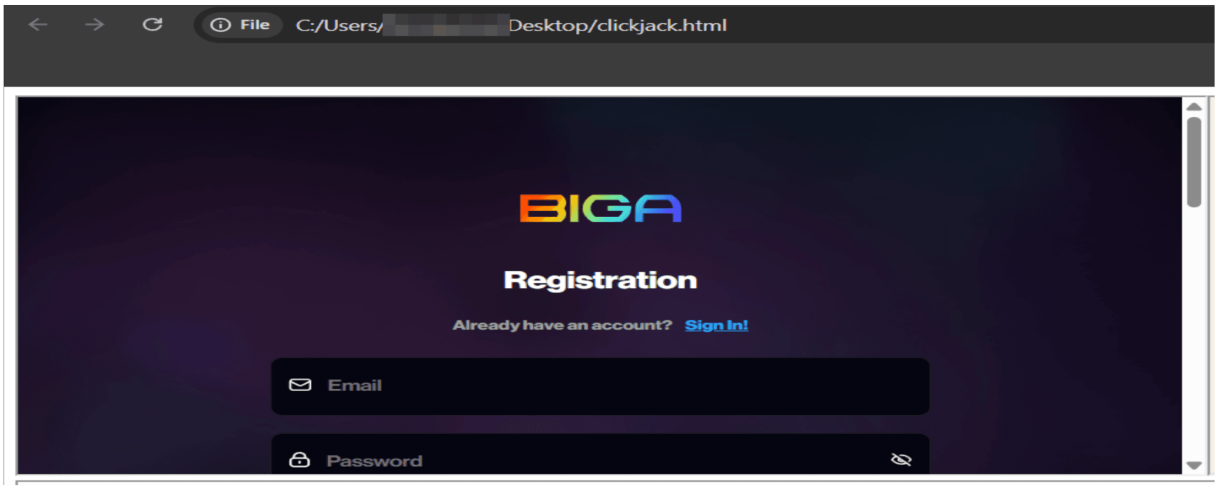
### Impact

If a user is tricked into clicking on a malicious link or button, they may unknowingly give away sensitive information, install malicious software, or be redirected to a malicious website.

### POC

Create an html file and add the below code.

<html><iframe src= https://bigarcade.org/></html>



**Recommendation**

To prevent clickjacking attacks, web developers should use the X-Frame-Options header in their web applications. This header instructs browsers not to render the page in a frame or iframe. Additionally, developers should avoid using legacy code such as ActiveX controls, Flash, and Java Applets as these can be targeted with clickjacking attacks.

## SPF Record Uses SoftFail (~all) Instead of HardFail (-all)

Resolved

### Description

The Sender Policy Framework (SPF) record for the domain is configured with a SoftFail (~all) mechanism instead of a HardFail (-all). This allows unauthorized mail servers to send emails on behalf of the domain, which may increase the risk of email spoofing and phishing attacks. With SoftFail, non-compliant emails are typically marked as spam rather than rejected outright, potentially allowing malicious emails to reach recipients' inboxes

### Impact

Increased risk of email spoofing and phishing attacks.
Attackers can impersonate the domain in emails, leading to brand damage and user deception.

### Recommendation

Update the domain's SPF record to use HardFail (-all) instead of SoftFail (~all) to strictly enforce SPF policies.

```
SPF record lookup and validation for: bigarcade.org

SPF records are published in DNS as TXT records.

The TXT records found for your domain are:
v=spf1 include:sendgrid.net include:_spf.google.com include:calendar-server.bounces.google.com ~all
google-site-verification=Gq-MOjrngqlt_eS6PYzltu4-TE2U643LdWlR0SMj0qw
google-site-verification=ELDjYoeL-FiF9NJUKI7I9yt_f5FFEae0kV6PttmlGzc

Checking to see if there is a valid SPF record.

Found v=spf1 record for bigarcade.org:
v=spf1 include:sendgrid.net include:_spf.google.com include:calendar-server.bounces.google.com ~all
```

## Missing Several Security Headers

Resolved

### Description

The web application is missing essential security headers, and the Strict-Transport-Security (HSTS) header is misconfigured. The observed issues include:
Strict-Transport-Security (max-age=15552000 instead of the recommended 31536000)
Missing security headers:
X-Frame-Options (prevents clickjacking)
X-Content-Type-Options (mitigates MIME-type sniffing)
Content-Security-Policy (prevents XSS and data injection)
Clear-Site-Data (ensures proper session and cache cleanup)
Cross-Origin-Embedder-Policy, Cross-Origin-Opener-Policy, Cross-Origin-Resource-Policy (improves cross-origin security)
Cache-Control (controls caching behavior to prevent sensitive data leaks)

### Impact

The absence of these headers increases the risk of various security threats:
Clickjacking attacks
- MIME-type sniffing vulnerabilities
- Increased susceptibility to cross-site scripting (XSS)
- Potential session hijacking and data leakage
- Insecure cross-origin resource handling

### Recommendation

Correct the HSTS Header:
- Set Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
- Add the Missing Security Headers

# Closing Summary

In this report, we have considered the security of the BIGA arcade. We performed our audit according to the procedure described above.

Some issues of High, Medium, Low, and Informational severity were found. Some suggestions and best practices are also provided to improve the code quality and security posture. In the End, the BIGA arcade Team resolved almost all issues and acknowledged.

# Disclaimer

At QuillAudits, we have spent years helping projects strengthen their smart contract security. However, security is not a one-time event—threats evolve, and so do attack vectors. Our audit provides a security assessment based on the best industry practices at the time of review, identifying known vulnerabilities in the received smart contract source code.

This report does not serve as a security guarantee, investment advice, or an endorsement of any platform. It reflects our findings based on the provided code at the time of analysis and may no longer be relevant after any modifications. The presence of an audit does not imply that the contract is free of vulnerabilities or fully secure.

While we have conducted a thorough review, security is an ongoing process. We strongly recommend multiple independent audits, continuous monitoring, and a public bug bounty program to enhance resilience against emerging threats.

Stay proactive. Stay secure.

# About QuillAudits

QuillAudits is a secure smart contracts audit platform designed by QuillHash Technologies. We are a team of dedicated blockchain security experts and smart contract auditors determined to ensure that Smart Contract-based Web3 projects can avail the latest and best security solutions to operate in a trustworthy and risk-free ecosystem

QuillAudits

| | |
|---|---|
| **7+**<br>Years of Expertise | **1M+**<br>Lines of Code Audited |
| **$30B+**<br>Secured in Digital Assets | **1400+**<br>Projects Secured |

Follow Our Journey

# AUDIT REPORT

April , 2025

For

BIGA

QuillAudits

Canada, India, Singapore, UAE, UK

www.quillaudits.com audits@quillaudits.com