



# AUDIT REPORT

---

May, 2025

For



# Table of Content

Executive Summary	03
Number of Security Issues per Severity	04
Checked Vulnerabilities	05
Techniques and Methods	06
Types of Severity	07
Types of Issues	08
 <b>Medium Severity Issues</b>	09
1. Insecure Direct Object Reference	09
 <b>Low Severity Issues</b>	11
1. Clickjacking	11
2. No/Misconfigured SPF Record	12
3. API_KEYS leaked	13
Closing Summary & Disclaimer	14

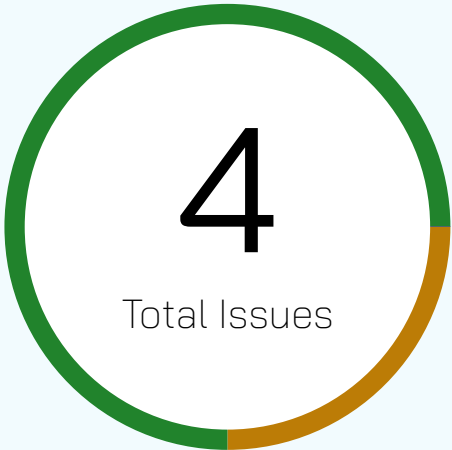


# Executive Summary

Project name	Allo
Audit Scope	Web Application
Overview	Allo is a decentralized platform designed for on-chain capital allocation, enabling communities and organizations to create, manage, and participate in funding pools using innovative allocation strategies. The platform is built on top of the Allo Protocol, which is known for powering democratic and transparent capital distribution.
In Scope	<a href="https://app.allo.xyz/">https://app.allo.xyz/</a>
Initial Review	21st April 2025 - 2nd May 2025
Updated Code Received	16th May 2025
Review 2	19th May 2025 - 20th May 2025



# Number of Issues per Severity



High	0 (0.00%)
Medium	1 (25.00%)
Low	3 (75.00%)
Informational	0 (0.00%)

		Severity			
		High	Medium	Low	Informational
Issues	Open	0	0	0	0
	Resolved	0	1	3	0
	Acknowledged	0	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 & Methods

Throughout the pentest of applications, care was taken to ensure:

- Information gathering – Using OSINT tools information concerning the web architecture
- Information leakage, web service integration, and gathering other associated information
- Related to web server & web services
- Using Automated tools approach for Pentest like Nessus, Acunetix etc.
- Platform testing and configuration
- Error handling and data validation testing
- Encryption-related protection testing
- Client-side and business logic testing

## Tools and Platforms used for Pentest:

Burp Suite

Nabbu

Dirbuster

DNSEnum

Turbo Intruder

SQLMap

Acunetix

Nmap

Horusec

Neucil

Metasploit

Postman

Netcat

Nessus

and many more...

# 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

<div>Open</div> <p>Security vulnerabilities identified that must be resolved and are currently unresolved.</p>	<div>Resolved</div> <p>Security vulnerabilities identified that must be resolved and are currently unresolved.</p>
<div>Acknowledged</div> <p>Vulnerabilities which have been acknowledged but are yet to be resolved.</p>	<div>Partially Resolved</div> <p>Considerable efforts have been invested to reduce the risk/ impact of the security issue, but are not completely resolved.</p>



# Medium Severity Issues

## Insecure Direct Object Reference

**Resolved**

### Description

An Insecure Direct Object Reference (IDOR) vulnerability has been discovered in the application, which allows an attacker to access unauthorized data by manipulating the unique identifier of a resource. This can occur when the application uses user-supplied data to access or manipulate a resource without proper validation or access controls.

In the current scenario, we can change the account id and address with victim's details and we should be able to view the information.

### Vulnerable URLs

[https://app.allo.xyz/api/exchange/orders?account\\_id=18e4a952-43dc-400a-be89-ffac7c9390da](https://app.allo.xyz/api/exchange/orders?account_id=18e4a952-43dc-400a-be89-ffac7c9390da)  
<https://app.allo.xyz/api/exchange/accounts/balance/18e4a952-43dc-400a-be89-ffac7c9390da>  
[https://app.allo.xyz/api/exchange/orders/?account\\_id=18e4a952-43dc-400a-be89-ffac7c9390da](https://app.allo.xyz/api/exchange/orders/?account_id=18e4a952-43dc-400a-be89-ffac7c9390da)  
<https://app.allo.xyz/api/exchange/orders/18e4a952-43dc-400a-be89-ffac7c9390da/volume>  
<https://app.allo.xyz/api/exchange/positions/18e4a952-43dc-400a-be89-ffac7c9390da>  
<https://app.allo.xyz/api/exchange/wallets>

### Impact




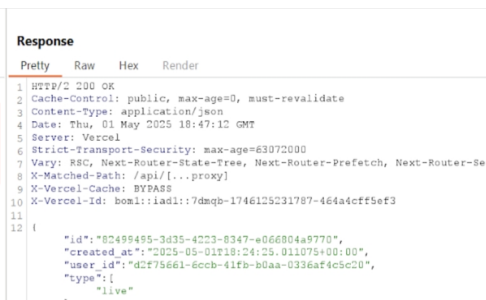

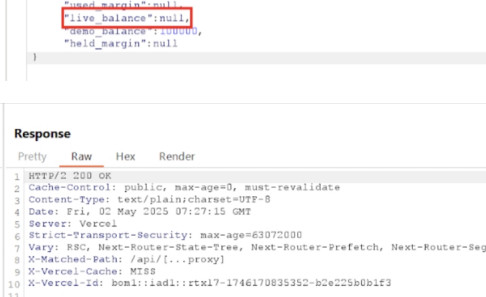



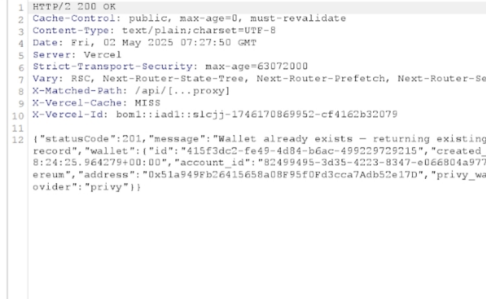
An attacker can exploit this vulnerability to access sensitive data such as account balance, used margin, held margin, wallet id etc.

### Recommendations

Add Authorization bearer for each api call to be seen only by the user that needs to see that data and not any user or even unauthorized requests.

### Proof of Concept



Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
							
							
							
							
							

## 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 URL

<https://app.allo.xyz/>

### 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

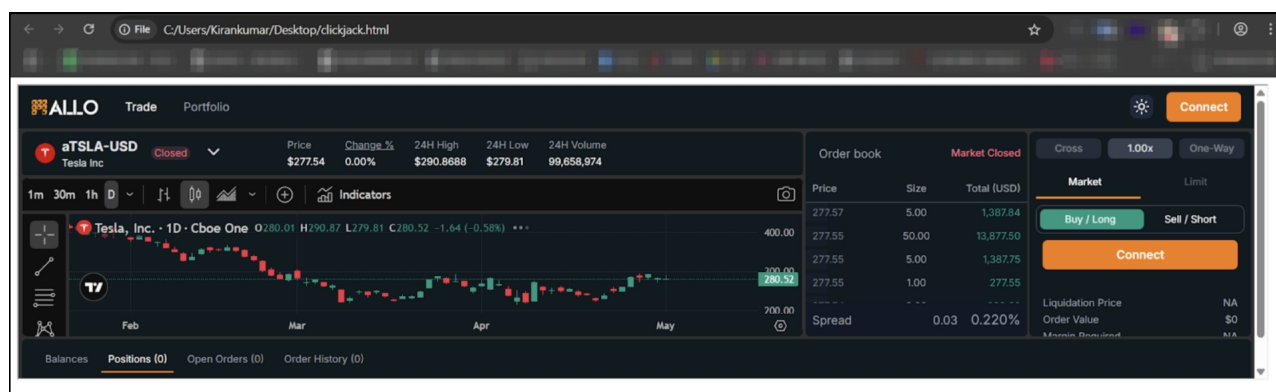
### Impact

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://app.allo.xyz/></html>
```



## No/Misconfigured SPF Record

**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.

### Vulnerable URL

[https://\\*.allo.xyz/](https://*.allo.xyz/)

### 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.

### POC

Create an html file and add the below code.  
<html><iframe src= https://app.allo.xyz/></html>

SPF record lookup and validation for: allo.xyz

SPF records are published in DNS as TXT records.

The TXT records found for your domain are:

GOOGLE-SITE-VERIFICATION=740JRR6K08NXBMAIVNRSTTNH9CL9KRIF3XNRURS-PAG  
google-site-verification=p1txlFAabqP8gFjgpqeQjeWfJYXnosIPZloNOK33YzU

No valid SPF record found.



## API\_KEYS leaked

**Resolved**

### Description

Two production-grade credentials were discovered hard-coded in JavaScript bundles that ship to end-users:

Key:-

WXETMuFUQmqybHuRkSgxv:25B8LJHSfpG6LVjR2ytU5Cwh7Z4Sch2ocoU

Used for WalletConnect Cloud – Project ID & Secret for dApp sessions

LzbasoBiLqItex3VkcQ7LRmL4PtftiZ1EMJrizrgfonWN6byJReu/l6yrUoo3zLW

Used for Rainbow “enhanced-provider” – REST/RPC enrichment API

### Vulnerable File

[https://app.allo.xyz/\\_next/static/chunks/4e88bc13-d1e7543251f64590.js?  
dpl=dpl\\_9VQubE585FVquspi9FfmytGtPrbh](https://app.allo.xyz/_next/static/chunks/4e88bc13-d1e7543251f64590.js?dpl=dpl_9VQubE585FVquspi9FfmytGtPrbh)

### Impact

Attacker can open relay connections, masquerade as our dApp, spam pairing requests, and harvest wallet addresses or push malicious transaction requests.

### Recommendation

Revoke / rotate both keys via WalletConnect Cloud & Rainbow dashboards.  
Generate fresh credentials and store them in the secret-manager backing our CI/CD (GitHub Secrets / Vault).

Patch and redeploy all environments (prod, staging).

### POC

```
  }({
    baseUrl: "https://enhanced-provider.rainbow.me",
    headers: {
      "x-api-key": void 0 !== _ && void 0 !== _.env && _.env.RAINBOW_PROVIDER_API_KEY || "LzbasoBiLqItex3VkcQ7LRmL4PtftiZ1EMJrizrgfonWN6byJReu/l6yrUoo3zLW"
    }
  });

}, {
  label: "Meld.io",
  name: "meld",
  feeRange: "1-2%",
  url: "https://meldcrypto.com",
  supportedChains: ["eip155", "solana"]
}]
, n = "WXETMuFUQmqybHuRkSgxv:25B8LJHSfpG6LVjR2ytU5Cwh7Z4Sch2ocoU"
, o = {
  FOUR_MINUTES_MS: 24e4,
  TEN_SEC_MS: 1e4
```



# Closing Summary

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

Some issues of medium, low severity were found, In the end, Allo Team resolved all issues.

# 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



<b>7+</b> Years of Expertise	<b>1M+</b> Lines of Code Audited
<b>\$30B+</b> Secured in Digital Assets	<b>1400+</b> Projects Secured

Follow Our Journey



# AUDIT REPORT

---

May, 2025

For



Canada, India, Singapore, UAE, UK

[www.quillaudits.com](http://www.quillaudits.com)

[audits@quillaudits.com](mailto:audits@quillaudits.com)