QuillAudits

Audit Report
July, 2023

For

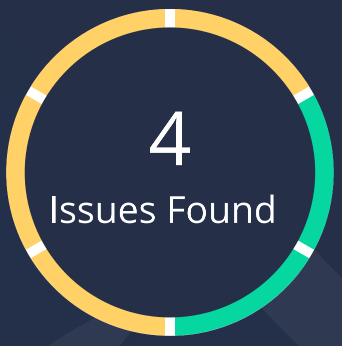
DCOM

Telecom Distribution

Table of Content

Executive Summary

Project Name	Vimo Wallet
Overview	Vimo Wallet is a hot wallet with custodial properties that only provides a transfer to whitelisted funds and is made for the internal use case of the client. This wallet had 2 main properties to be checked out Whitelisting address and transfer of funds.
Timeline	29th June, 2023 - 4th July, 2023
Scope of Audit	The scope of this pentest was to analyse the wallet and API calls for quality, security, and correctness. 1] github.vimolive.com:3000/ 2] https://vaulttest.sarrafapp.com/
In scope	Wallet Source code - Whitelisting and Withdraw (Priority)



🟥 High 🟨 Medium
🟩 Low 🟪 Informational

	High	Medium	Low	Informational
Open Issues	0	0	0	0
Acknowledged Issues	0	0	1	0
Partially Resolved Issues	0	0	0	0
Resolved Issues	0	3	0	0

Techniques and Methods

Throughout the pentest of BIT-Wallet, 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, Wireshark, 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
- DNSenum
- Dirbuster
- SQLMap
- Acunetix
- Neucli
- Nabbu
- Turbo Intruder
- Nmap
- Metasploit
- Horusec
- Postman
- Netcat
- Nessus and many more.

Types of Issues

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

Resolved
These are the issues identified in the initial audit and have been successfully fixed.

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.

Types of Severities

High

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

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

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

Informational

These are 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.

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

Manual Testing

High Severity Issues

No issues found

Medium Severity Issues

1. Input Validation - Address type

Description

The address in the registered wallet is endpoint doesn’t have any validation that the entered address is from the ETH chain or BNB chain or TRX chain. This type of input validation is needed to avoid any confusion to transfer funds and to only be able to add sufficient and correct addresses that are actually needed to be whitelisted and throw an error if they are from an unsupported chain.

Vulnerable File Location: https://vaulttest.sarrafapp.com/register-wallet

Steps to Reproduce

1. Visit this endpoint and add an address with a random agent name.
   Example: 0xabcd.... or TLDadstf..... Or bnbrhrhs35...... and random agent name and click on register

2. You will get that address registered without cross-verifying if the address is from the correct chain.

Recommendation

Set up proper tron-based wallet validation as you have only funds on that chain.
https://github.com/TronWallet/wallet-address-validator

Status

Resolved

2. Weak Id Generation

Description

The “_id” parameter in the storage can be enumerated as only the last 2 digits and one digit in the missile keeps on changing by a +2 of frequency.

Vulnerable File Location: https://vaulttest.sarrafapp.com/register-wallet

Steps to Reproduce

1. Enter the wallet address and agent name once and click on Register
2. Change the details and click on Register again.
3. Both times most characters of _id are same.

This should not be the case as these numbers should be randomly generated _id’s and need to be that way for security issues.

Recommendation

Generate and random _id’s and keep all values changing.

POC

64a2a6b5762d54e38e2aa045
64a2a6d5762d54e38e2aa047

Status

Resolved

3. Incorrect amount transfer

Description

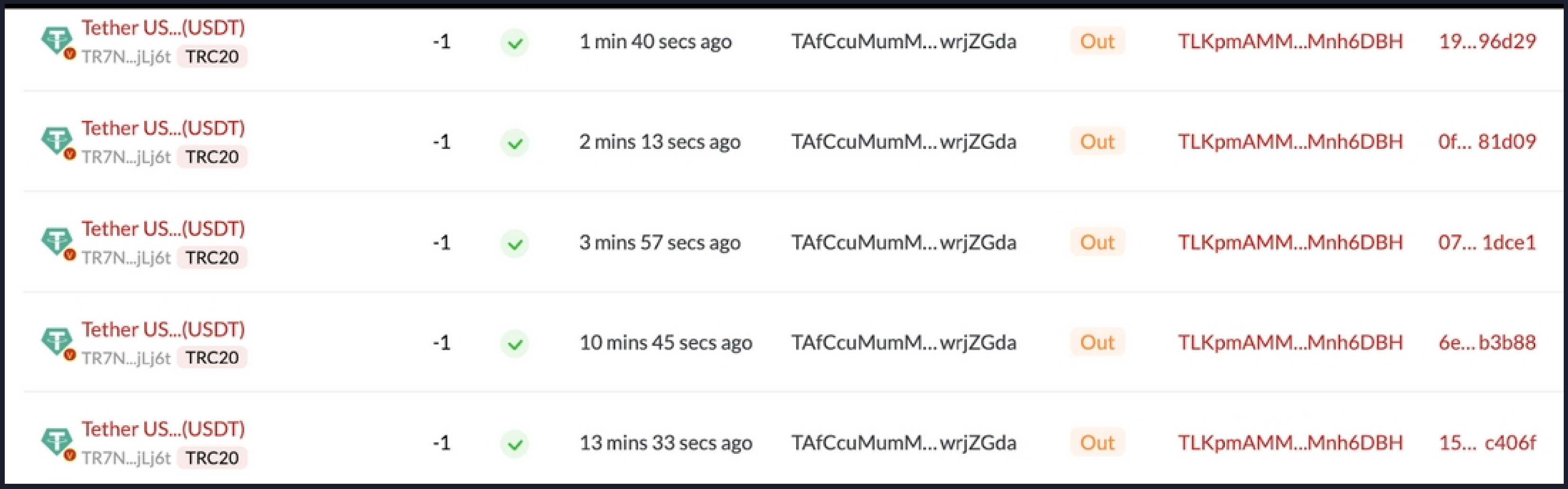
When adding the value in the amount section it only sends 1 USDT even after having a condition to have a value above 2 but if you enter 2.0000001 or 30.00001 it will still send only 1 USDT.

Vulnerable File Location: https://vaulttest.sarrafapp.com/send-transaction

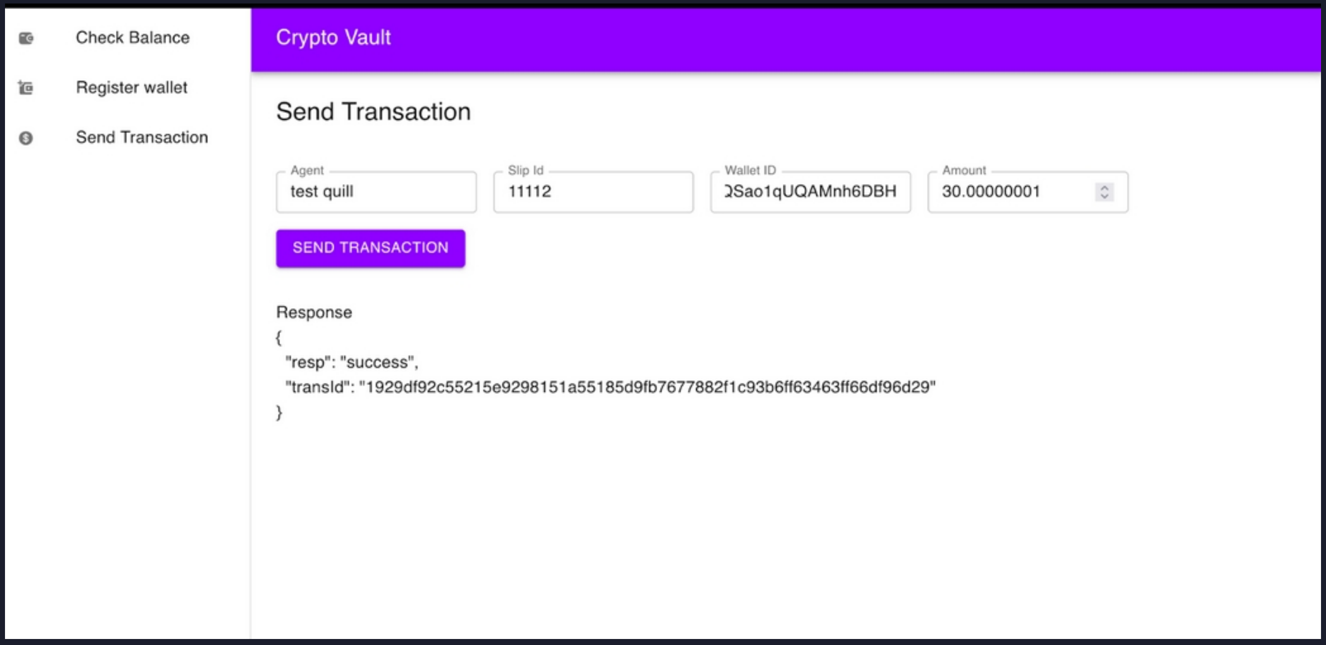
Steps to Reproduce

1. Visit the endpoint above and enter any whitelisted address and enter a slip id and amount to any value above 2 i.e 2.0000001 and click on send
2. It will trigger the Txn as a success and give you a valid txn hash to check.
3. If you check the txn on Explorer it will show that you sent only 1 USDT and not 2.000001

POC



Tether US...(USDT) TR7N...jLj6t TRC20	-1	✓	1 min 40 secs ago	TAfCcuMumM...wrjZGda	Out	TLKpmAMM...Mnh6DBH	19...96d29
Tether US...(USDT) TR7N...jLj6t TRC20	-1	✓	2 mins 13 secs ago	TAfCcuMumM...wrjZGda	Out	TLKpmAMM...Mnh6DBH	0f... 81d09
Tether US...(USDT) TR7N...jLj6t TRC20	-1	✓	3 mins 57 secs ago	TAfCcuMumM...wrjZGda	Out	TLKpmAMM...Mnh6DBH	07... 1dce1
Tether US...(USDT) TR7N...jLj6t TRC20	-1	✓	10 mins 45 secs ago	TAfCcuMumM...wrjZGda	Out	TLKpmAMM...Mnh6DBH	6e... b3b88
Tether US...(USDT) TR7N...jLj6t TRC20	-1	✓	13 mins 33 secs ago	TAfCcuMumM...wrjZGda	Out	TLKpmAMM...Mnh6DBH	15... c406f



Send Transaction

Agent: test quill
Slip Id: 11112
Wallet ID: ɔSao1qUQAMnh6DBH
Amount: 30.00000001

```
Response
{
  "resp": "success",
  "transId": "1929df92c55215e9298151a55185d9fb7677882f1c93b6ff63463ff66df96d29"
}
```

Status

Resolved

Low Severity Issues

4. Multiple Deprecated Libraries in package-lock.json

Description

package-lock Stores files that can be useful for the dependency of the application. This is used for locking the dependency with the installed version. It will install the exact latest version of that package in your application and save it in package. This arises a problem if the dependency used has an exploit in the version mentioned. It can create a backdoor for an attacker.

Vulnerable Dependencies

xml2js
semver
request
passport
moment-timezone
jsonwebtoken
minimatch
dicer
mongoose
qs

Recommended Fix

1) Update all the above mentioned Dependencies
2) Remove any Library Not needed.

Impact

Multiple of these libraries have public exploits and CVE-registered issues that have been patched and can help your application stay more secure from any dependency-vulnerable issues.

Status

Acknowledged

Closing Summary

In this report, we have considered the security of Vimo’s wallet. We performed our audit according to the procedure described above.
Some issues of medium, and low severity were found, Some suggestions and best practices are also provided in order to improve the code quality and security posture.

Disclaimer

QuillAudits Dapp audit is not a security warranty, investment advice, or an endorsement of the Vimo Platform. This audit does not provide a security or correctness guarantee of the audited smart contracts.

The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them. Securing smart contracts is a multi-step process. One audit cannot be considered enough. We recommend that the Vimo Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

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.

850+
Audits Completed

$30B
Secured

800K
Lines of Code Audited

Follow Our Journey

Audit Report
July, 2023

For

DCOM
Telecom Distribution

QuillAudits

Canada, India, Singapore, UAE, UK

www.quillaudits.com

audits@quillhash.com