



SPADE SOLIDITY AUDITS

Volta Dao Audit

December 31, 2021

For :
Volta Dao Team

Website:
www.voltadao.finance

Telegram:
t.me/VoltaDAO

Twitter:
[@SpadeAudits](https://twitter.com/SpadeAudits)

Telegram:
t.me/spadeaudits





Disclaimer

Spade Solidity reports are not, nor should be considered, an “endorsement” or “disapproval” of any particular project or team. These reports are not, nor should be considered, an indication of the economics or value of any “product” or “asset” created by any team or project that contracts Spade to perform a security review.

Spade Solidity Reports do not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

Spade Solidity Reports should not be used in any way to make decisions around investment or involvement with any particular project. These reports in no way provide investment advice, nor should be leveraged as investment advice of any sort.

Spade Solidity Reports represent an extensive auditing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. Spade Solidity’s position is that each company and individual are responsible for their own due diligence and continuous security. Spade Solidity’s goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

What is a Spade Solidity report?

- A document describing in detail an in depth analysis of a particular piece(s) of source code provided to Spade Solidity by a Client.
- An organized collection of testing results, analysis and inferences made about the structure, implementation and overall best practices of a particular piece of source code.
- Representation that a Client of Spade Solidity has indeed completed a round of auditing with the intention to increase the quality of the company/ product’s IT infrastructure and or source code.

OVERVIEW

Project Summary

Project Name	Volta Dao audit
Description	Defi Protocol
Platform	FTM mainnet

Audit Summary

Delivery Date	December 31, 2021
Method of Audit	Static Analysis, Manual Review
Timeline	Story Points - 100

Vulnerability Summary

Total Issues	2
Total Critical	2
Total High	0
Total Medium	0
Total Low	0
Total Informational	0

Executive Summary


Our detailed audit methodology was as follows:

Step 1
A manual line-by-line code review to ensure the logic behind each function is sound and safe from common attack vectors.
Step 2
Simulation of hundreds of thousands of Smart Contract Interactions on a test blockchain using a combination of automated test tools and manual testing to determine if any security vulnerabilities exist.
Step 3
Consultation with the project team on the audit report pre-publication to implement recommendations and resolve any outstanding issues.



Grading

The following grading structure was used to assess the level of vulnerability found within all Smart Contracts:



Threat Level	Definition
Critical	Severe vulnerabilities which compromise the entire protocol and could result in immediate data manipulation or asset loss.
High	Significant vulnerabilities which compromise the functioning of the smart contracts leading to possible data manipulation or asset loss.
Medium	Vulnerabilities which if not fixed within in a set timescale could compromise the functioning of the smart contracts leading to possible data manipulation or asset loss.
Low	Low level vulnerabilities which may or may not have an impact on the optimal performance of the Smart contract.
Informational	Issues related to coding best practice which do not have any impact on the functionality of the Smart Contracts.



About

Volta Dao makes use of code from Olympus Dao. It builds upon its core concepts through additional token utility to increase the staking pressure and protocol interaction. Volta uses the Algorithmic Reserve Currency algorithm.

At a high-level Volta Dao consists of its protocol managed treasury, protocol owned liquidity, bond mechanism, and staking rewards that are designed to control supply expansion.

Bond sales generate profit for the protocol, and the treasury uses the profit to mint Volt and distribute them to stakers. With liquidity bonds the protocol can accumulate its own liquidity.



FTM mainnet Contract addresses

OlympusERC20.sol (Volt Token)	0xAE280713f30942C5956eb0d A21F8feD0394af211
sOlympusERC20.sol (VOLTS token)	0xBECe17fE60ea0c583AC350 a48b1634cc74640F25
Staking.sol (Volt Staking)	0xa51c8576c6C4Af6A5EdF 1Bf485284b63797fe409
StakingHelper.sol (StakingHelper)	0x309E65aec68845447C3b 9288276056a152862a42
StakingDistributor.sol	0xf8368E9Be737004976fCE a9059De147d768ae125
StakingWarmingup.sol	0x70640e64B61063A7cd2311 8fb0187a500B79Aa0C
StandardBondingCalculator (voltBondingCalculator)	0xfbA30002C32E7e8d1Ff6f5 19e4f02B84bD839168
Treasury (VoltTreasury)	0xD55B096C7E268dF9d7d3 35f00a2FcaDf31774Cb3

Scope of Audit

The Spade Tech team were commissioned to perform a security review of the Volta Dao platform. The scope of the audit was widened to include a Post Mortem of certain issues with functionality that emerged at launch.



Volta Dao Team Issues

The Volta Dao Team raised concerns about the competence of their lead smart contract developer who they had hired via a freelance developer platform. When issues with functionality / wallet security emerged, the Volta Dao Team tried to challenge this developer but he allegedly tried to blackmail them. Full details of this incident were published on the Team's twitter account:

<https://twitter.com/VoltaDAO/status/1470819207063511048>

When the audit began this developer was no longer employed and no longer had access to the team multi-sig wallet.

Manual Review/ Post Mortem

It became apparent during the audit process that several critical exploits were present in the protocol. Full redeployment has been recommended to resolve these issues. The issues identified by the Spade Audit Team are outlined below.



Centralisation/ Privilege Issues

CRITICAL ISSUE 1

At the start of the audit The Volta Dao team disclosed that they had control over the following wallets:

MULTI-SIG WALLET	0x98A3d804dD228143c7A8877f3A5a971ff50E6E1b
Team member 1 Multi	0xacD10D4B0FF7fdE5d76C9Bf00c9CA4140273BD0c
Team member 2 Multi	0x0C559A3c89AC91c04a1f2FB76B3bc9Ec5D58D870

The previous developer was said to own the following wallets:

WalletA	0xeeA09d571fC2B70FA5aB50733EF59abddFbEA790
WalletB	0x02cd527F4Fb7263E018763BBE53c2DcAD172a7d4

The Audit identified that the following wallets had privileges within the protocol:

fcbb11fb196f9a4d6d12288f0a2f400fd113162c	RESERVE DEPOSITOR
8d11ec38a3eb5e956b052f67da8bdc9bef8abf3e	RESERVE TOKEN
dabb1f4f38b8c98c36c689f11565a539435d54bc	RESERVE DEPOSITOR
fd52b8c1d4d8abaa7fe9ea7820393d3006031050	LIQUIDITY DEPOSITOR
b38058a9779db9adfed262768c5be493b7476dac	LIQUIDITY TOKEN
421409295e26a3c0a357e076e5b6dfaf8f3aac6f	LIQUIDITY DEPOSITOR
21be370d5312f44cb42ce377bc9b8a0cefla4c83	RESERVE TOKEN
ea8a38aa660e1b15215380207a229dc2eca8c362	RESERVE DEPOSITOR
bece17fe60ea0c583ac350a48b1634cc74640f25	VOLTS
ea8a38aa660e1b15215380207a229dc2eca8c362	REWARD MANAGER
02cd527f4fb7263e018763bbe53c2dcad172a7d4	RESERVE SPENDER
02cd527f4fb7263e018763bbe53c2dcad172a7d4	LIQUIDITY MANAGER
02cd527f4fb7263e018763bbe53c2dcad172a7d4	REWARD MANAGER

Privileges Associated with Each Account

RESERVE DEPOSITOR → Can deposit reserve tokens to the treasury
RESERVE TOKEN → token which can be deposited in the treasury
LIQUIDITY DEPOSITOR → can deposit liquidity tokens to treasury
LIQUIDITY TOKEN → LP token which can be deposited into treasury
REWARD MANAGER → can mint VOLT token

Unauthorised address with this privilege:

Previous developer wallet	0x02cd527F4Fb7263E018763BBE53c2DcAD172a7d4
---------------------------	--

RESERVE SPENDER → can withdraw reserve token from treasury
(ie DAI&WFTM)

Unauthorised address with this privilege:

Previous developer wallet	0x02cd527F4Fb7263E018763BBE53c2DcAD172a7d4
---------------------------	--

LIQUIDITY MANAGER → can withdraw LP token from treasury

Unauthorised address with this privilege:

Previous developer wallet	0x02cd527F4Fb7263E018763BBE53c2DcAD172a7d4
---------------------------	--

RESERVE MANAGER → can withdraw reserve token from treasury

Summary

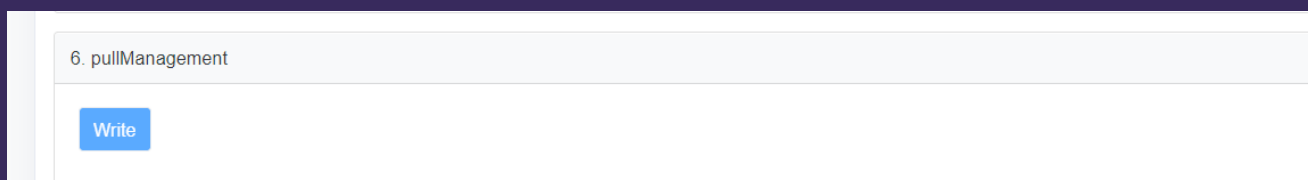
The Spade Audit found that certain unauthorised wallets retained the right to mint Volta Tokens, withdraw FTM/Dai from the treasury and withdraw LP tokens within the treasury. The Volta Dao Team advised that these wallets with privilege belonged to their previous developer, and they were not aware that these privileges existed. Hence all assets within the protocol were found to be at immediate risk of loss.

Issue severity: Critical Issue.

Issue Resolution

To remove these privileges the following steps were recommended:

Step1 : call pullManagement function using

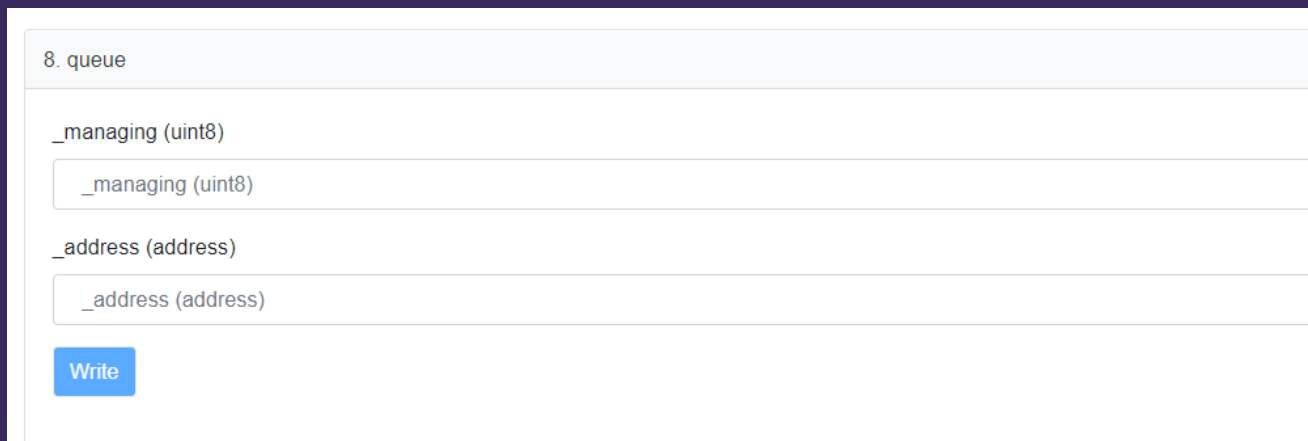


6. pullManagement

Write

[0x98a3d804dd228143c7a8877f3a5a971ff50e6e1b](#) (address), to transfer treasury ownership to team multisig

Step2: remove all other previous dev accounts from privilege using queue and toggle function.



8. queue

_managing (uint8)

_address (address)

Write

12. toggle

CRITICAL ISSUE 2

During the initial deployment of Volta Dao the wallet address 0xe724bd84b4b4c1e1257f5e403999745c1a2036f9 was set permanently as the locking address for the staking pool using the following function:

```
function setContract( CONTRACTS _contract, address _address ) external onlyManager() {  
    if( _contract == CONTRACTS.DISTRIBUTOR ) { // 0  
        distributor = _address;  
    } else if ( _contract == CONTRACTS.WARMUP ) { // 1  
        require( warmupContract == address( 0 ), "Warmup cannot be set more than once" );  
        warmupContract = _address;  
    } else if ( _contract == CONTRACTS.LOCKER ) { // 2  
        require( locker == address(0), "Locker cannot be set more than once" );  
        locker = _address;  
    }  
}
```

Calling this function created a hard-coded exploit within the protocol to enable wallet 0xe724bd84b4b4c1e1257f5e403999745c1a2036f9 to drain the balance of the staking pool at any time.

Token VOLTs

Overview ERC-20

PRICE: \$0.00 @ 0.000000 FTM

FULLY DILUTED MARKET CAP: \$0.00

Total Supply: 6,895,836.211131 VOLTs

Holders: 685 addresses

Transfers: 8,762

Profile Summary

Contract: 0xbece17fe60ea0c583ac350a48b1634cc74640f25

Decimals: 9

Social Profiles: Not Available, Update ?

Transfers Holders Info Contract Analytics Comments

Token Holders Chart

A total of 685 token holders

Rank	Address	Quantity	Percentage	Analytics
1	0xa51c8576c6c4af6a5edf1bf485284b63797fe409	6,766,193.43278184	98.1200%	Analytics
2	0x359ced11bfe99df191e011ec4f1aba2b4a269837	8,812.366083432	0.1278%	Analytics

The Volta Dao team have stated that this wallet belongs to their previous 'rogue' developer who they believe acted maliciously to embed critical exploits within Volta Dao. Due to the nature of this exploit the only way to remove it was to re-deploy the smart contracts. Whilst steps were being taken to secure the treasury using the pull management/ queue and toggle function, the following function was called by wallet 0xe724bd84b4b4c1e1257f5e403999745c1a2036f9.

```

/**
 * @notice provide bonus to locked staking contract
 * @param _amount uint
 */
function giveLockBonus( uint _amount ) external {
    require( msg.sender == locker );
    totalBonus = totalBonus.add( _amount );
    IERC20( VOLTs ).safeTransfer( locker, _amount );
}

```

This enabled the owner of this wallet to steal 1,000,000 VOLTs tokens from the staking contract. 130,000 VOLTs were then unstaked, enabling 130,000 VOLT to be sold for 21,985.7143067525 42846595 DAI.

Txn Hash	Age	From	To	Value	Token
0xcca2d3d61075e46eb2...	8 days 5 hrs ago	0xe724bd84b4b4c1e125...	OUT 0xa51c8576c6c4af6a5e...	100	VOLTs (VOLTs)
0xcca2d3d61075e46eb2...	8 days 5 hrs ago	0xa51c8576c6c4af6a5e...	IN 0xe724bd84b4b4c1e125...	100	VOLT (VOLT)
0x15577d9d66b88fcee9...	8 days 5 hrs ago	0xe724bd84b4b4c1e125...	OUT 0xb38058a9779db9adfe...	130,000	VOLT (VOLT)
0x15577d9d66b88fcee9...	8 days 5 hrs ago	0xb38058a9779db9adfe...	IN 0xe724bd84b4b4c1e125...	21,985.714306752542846595	Dai Stableco... (DAI)
0xa3c283a50ba2ec7f6a...	8 days 5 hrs ago	0xa51c8576c6c4af6a5e...	IN 0xe724bd84b4b4c1e125...	130,000	VOLT (VOLT)
0xa3c283a50ba2ec7f6a...	8 days 5 hrs ago	0xe724bd84b4b4c1e125...	OUT 0xa51c8576c6c4af6a5e...	130,000	VOLTs (VOLTs)
0x891bd97eef2cc35307...	8 days 5 hrs ago	0xeea09d571fc2b70fa5a...	IN 0xe724bd84b4b4c1e125...	1,000,000.77263611	VOLTs (VOLTs)

[Download CSV Export]

The stolen Dai remains in the 'rogue dev' wallet:

<https://ftmscan.com/token/0x8d11ec38a3eb5e956b052f67da8bdc9bef8abf3e?a=0xe724bd84b4b4c1e1257f5e403999745c1a2036f9>

Issue severity: Critical Issue.

Issue Resolution:

To enable a complete redeployment of all smart contracts within protocol the full treasury balance of \$657,307, including all remaining LP tokens, was removed to the following multi-sig wallet owned by the Volta Dao Team.

<https://ftmscan.com/address/0x98a3d804dd228143c7a8877f3a5a971ff50e6e1b#tokentxns>

Conclusion

Due to multiple critical exploits being present in the protocol the Spade Tech team recommends that all current Volta Dao Smart contracts are abandoned and redeployed with fresh contracts.

The owner of address `0xe724bd84b4b4c1e1257f5e403999745c1a2036f9` exploited the protocol and stole \$22k. This balance remains in this wallet and any future movement of the funds will be monitored.

During the process of the Audit/ Post-Mortem investigation the Volta Dao Team worked closely with the Spade Audit team to secure funds contained within the protocol. The Volta Dao Team have accepted the findings/recommendations in full and have expressed a commitment to redeploying a safe, secure and pre-audited platform following consultation with their community.

Total Assets Exploited: **\$21,985**

Total Assets Secured: **\$657,307**

Twitter:
[@SpadeAudits](https://twitter.com/SpadeAudits)

Website:
<https://spadetech.io>

Telegram:
t.me/spadeaudits

Appendix

Finding Categories

Gas Optimization

Gas Optimization findings refer to exhibits that do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Mathematical Operations

Mathematical Operation exhibits entail findings that relate to mishandling of math formulas, such as overflows, incorrect operations etc.

Logical Issue

Logical Issue findings are exhibits that detail a fault in the logic of the linked code, such as an incorrect notion on how block.timestamp works.

Control Flow

Control Flow findings concern the access control imposed on functions, such as owner-only functions being invoke-able by anyone under certain circumstances.

Volatile Code

Volatile Code findings refer to segments of code that behave unexpectedly on certain edge cases that may result in avulnerability.

Data Flow

Data Flow findings describe faults in the way data is handled at rest and in memory, such as the result of a structassignment operation affecting an in-memory struct rather than an instorage one.

Language Specific

Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete .

Twitter:
[@SpadeAudits](https://twitter.com/SpadeAudits)

Website:
<https://spadetech.io>

Telegram:
t.me/spadeaudits

Coding Style

Coding Style findings usually do not affect the generated byte-code and comment on how to make the codebase more legible and as a result easily maintainable.

Inconsistency

Inconsistency findings refer to functions that should seemingly behave similarly yet contain different code, such as a constructor assignment imposing different require statements on the input variables than a setter function.

Magic Numbers

Magic Number findings refer to numeric literals that are expressed in the codebase in their raw format and should otherwise be specified as constant contract variables aiding in their legibility and maintainability.

Compiler Error

Compiler Error findings refer to an error in the structure of the code that renders it impossible to compile using the specified version of the project.

Dead Code

Code that otherwise does not affect the functionality of the codebase and can be safely omitted.

Twitter:
[@SpadeAudits](https://twitter.com/SpadeAudits)

Website:
<https://spadetech.io>

Telegram:
t.me/spadeaudits