OBELISK

Part of Tibereum Group

AUDITING REPORT

Version Notes

Version	No. Pages	Date	Revised By	Notes
1.0	Total:	YYYY-MM-DD	Zapmore, Auditor1	Audit Draft

Audit Notes

Audit Date	YYYY-MM-DD - YYYY-MM-DD
Auditor/Auditors	Auditor1, Auditor2
Auditor/Auditors Contact Information	contact@obeliskauditing.com
Notes	Specified code and contracts are audited for security flaws. UI/UX (website), logic, team, and tokenomics are not audited.
Audit Report Number	OB5XXXXXXXX

Disclaimer

This audit is not financial, investment, or any other kind of advice and is for informational purposes only. This report is not a substitute for doing your own research and due diligence. Obelisk is not responsible or liable for any loss, damage, or otherwise caused by reliance on this report for any purpose. Obelisk has based this audit report solely on the information provided by the audited party and on facts that existed before or during the audit being conducted. Obelisk is not responsible for any outcome, including changes done to the contract/contracts after the audit was published. This audit is fully objective and only discerns what the contract is saying without adding any opinion to it. The audit is paid by the project but neither the auditors nor Obelisk has any other connection to the project and has no obligations other than to publish an objective report. Obelisk will always publish its findings regardless of the outcome of the findings. The audit only covers the subject areas detailed in this report and unless specifically stated, nothing else has been audited. Obelisk assumes that the provided information and material were not altered, suppressed, or misleading. This report is published by Obelisk, and Obelisk has sole ownership of this report. Use of this report for any reason other than for informational purposes on the subjects reviewed in this report including the use of any part of this report is prohibited without the express written consent of Obelisk.

Obelisk Auditing

Defi is a relatively new concept but has seen exponential growth to a point where there is a multitude of new projects created every day. In a fast-paced world like this, there will also be an enormous amount of scams. The scams have become so elaborate that it's hard for the common investor to trust a project, even though it could be legit. We saw a need for creating high-quality audits at a fast phase to keep up with the constantly expanding market. With the Obelisk stamp of approval, a legitimate project can easily grow its user base exponentially in a world where trust means everything. Obelisk Auditing consists of a group of security experts that specialize in security and structural operations, with previous work experience from among other things, PricewaterhouseCoopers. All our audits will always be conducted by at least two independent auditors for maximum security and professionalism.

As a comprehensive security firm, Obelisk provides all kinds of audits and project assistance.

Audit Information

The auditors always conducted a manual visual inspection of the code to find security flaws that automatic tests would not find. Comprehensive tests are also conducted in a specific test environment that utilizes exact copies of the published contract.

While conducting the audit, the Obelisk security team uses best practices to ensure that the reviewed contracts are thoroughly examined against all angles of attack. This is done by evaluating the codebase and whether it gives rise to significant risks. During the audit, Obelisk assesses the risks and assigns a risk level to each section together with an explanatory comment. Take note that the comments from the project team are their opinion and not the opinion of Obelisk.

Table of Contents

Project Information

Name	
Description	
Website	
Contact	
Contact information	@XXXX on TG
Token Name(s)	N/A
Token Short	N/A
Contract(s)	See Appendix A
Code Language	Solidity
Chain	Polygon / BSC

Audit of ShadeCash

The main takeaway will be added here after the audit is completed and the final draft is created.

Obelisk was commissioned by XXXX on the XXXX th of XXXX 2021 to conduct a comprehensive audit of XXXX' contracts. The following audit was conducted between the XXXXth of XXXX 2021 and the XXXXth of XXXX 2021. Two of Obelisk's security experts went through the related contracts manually using industry standards to find if any vulnerabilities could be exploited either by the project team or users.

Findings and other relevant info will be updated at audit completion and added here.

The informational findings are good to know while interacting with the project but don't directly damage the project in its current state, hence it's up to the project team if they deem that it's worth solving these issues.

The team has not reviewed the UI/UX, logic, team, or tokenomics of the XXXX project**.**

Please read the full document for a complete understanding of the audit.

Summary Table

Finding	ID	Severity	Status
Tokens With Transfer Fee Not Supported	#0001	Low Risk	Mitigated
Trusting External Contract	#0002	Low Risk	Open
No Timelock Implemented	#0003	Low Risk	Mitigated
Contract Values Can Be Constant Or Immutable (Gas Optimization)	#0004	Informational	Closed
Multiple Contracts In One File	#0005	Informational	Closed
No Events Emitted For Changes To Protocol Values	#0006	Informational	Closed

Findings

Manual Analysis

Tokens With Transfer Fee Not Supported

FINDING ID	#0001
SEVERITY	Low Risk
STATUS	Mitigated
LOCATION	Rev 1 - LPStaker.sol -> 376-395

```
1     // Deposit LP tokens to for SHADE allocation.
2     function deposit(uint256 amount) public {
3         require(startTime != 0, "Not started");
4
5         UserInfo storage user = userInfo[msg.sender];
6
7         updatePool();
8
9         uint256 pending = user.amount * accSHADEPerShare / 1e12 -
  user.rewardDebt;
10
11        user.amount += amount;
12        user.rewardDebt = user.amount * accSHADEPerShare / 1e12;
13
14        _sendRewards(pending);
15
16        lpToken.safeTransferFrom(address(msg.sender), address(this),
  amount);
17        lpDeposited += amount;
18
19        emit Deposit(msg.sender, amount);
20    }
```

LOCATION	Rev 1 - LPStaker.sol -> 397-416

```
    // Withdraw LP tokens from MasterChef.
    function withdraw(uint256 amount) public {
        UserInfo storage user = userInfo[msg.sender];

        require(user.amount >= amount, "Not enough funds");

        updatePool();

        uint256 pending = user.amount * accSHADEPerShare / 1e12 -
user.rewardDebt;

        user.amount -= amount;
        user.rewardDebt = user.amount * accSHADEPerShare / 1e12;

        _sendRewards(pending);

        lpDeposited -= amount;
        lpToken.safeTransfer(address(msg.sender), amount);

        emit Withdraw(msg.sender, amount);
    }
```

DESCRIPTION	Contract does not support fees on transfer tokens. If the deposited token has a fee on transfer there can be a discrepancy on the actual received amount.
RECOMMENDATION	Ensure that the deposit token used with this contract will never have a transfer fee. Alternatively, check the token balance before and after the transfer to get the actual received amount. In this case, it is necessary to ensure that there cannot be re-entrancy from the token transfer.
RESOLUTION	The project team has stated this contract will never use a token with a transfer fee for the deposit token.

Trusting External Contract

FINDING ID	#0002
SEVERITY	Low Risk
STATUS	Open
LOCATION	Rev 1 - LPStaker.sol -> 368-373

```
1    uint256 shadeReward = masterPending();
2
3    if (shadeReward != 0) {
4        masterChef.withdraw(masterPoolId, 0);
5        accSHADEPerShare += shadeReward * 1e12 / lpDeposited;
6    }
```

DESCRIPTION	If the *masterPending()* returns the wrong value there could be a discrepancy between the received amounts from the masterchef contract. Also if a transfer fee token is the reward then the amount can also differ.
RECOMMENDATION	Check the token balance before and after the transfer to get the actual received amount. In this case, it is necessary to ensure that there cannot be re-entrancy from the token transfer.
RESOLUTION	Project team comment: "Contract designed to work ONLY with ONE predefined token and CURRENT masterChef contract and it can't have any fees.".

No Timelock Implemented

FINDING ID	#0003
SEVERITY	Low Risk
STATUS	Mitigated
LOCATION	Rev 1 - LPStaker.sol -> 467-469

```
function setRewardsStaker(IRewardsStaker newRewardsStaker) external
onlyOwner {
    rewardsStaker = newRewardsStaker;
}
```

DESCRIPTION	The rewardsStaker contract should use a timelock as rewards are sent to this contract. If the rewardsStaker contract isn't working then it could lock deposit and withdraw and users would have to use *emergencywithdraw*.
RECOMMENDATION	Obelisk recommends a timelock delay of at least 72 hours.
RESOLUTION	Project Team has confirmed that a timelock will be applied on the deployed contract.

Static Analysis

Contract Values Can Be Constant Or Immutable (Gas Optimization)

FINDING ID	#0004
SEVERITY	Informational
STATUS	Closed
LOCATION	• Rev 1 - LPStaker.sol -> 299: *IERC20 public shade;* • Rev 1 - LPStaker.sol -> 300: *IMasterChef public masterChef;* • Rev 1 - LPStaker.sol -> 301: *IRewardsStaker public rewardsStaker;*

DESCRIPTION	Variables which do not change during the operation of a contract can be marked *constant* or *immutable* to reduce gas costs and improve code readability.
RECOMMENDATION	Mark these variables as *constant* or *immutable* as appropriate.
RESOLUTION	The recommended changes were implemented. Reviewed in commit [96048adc10c1d304feaef3bb5d01960dcb0f7a5f]

Multiple Contracts In One File

FINDING ID	#0005
SEVERITY	Informational
STATUS	Closed
LOCATION	Rev 1 - LPStaker.sol

DESCRIPTION	The noted files contain multiple contracts.
RECOMMENDATION	Have each contract in its own file.
RESOLUTION	The recommended changes were implemented. Reviewed in commit [96048adc10c1d304feaef3bb5d01960dcb0f7a5f]

No Events Emitted For Changes To Protocol Values

FINDING ID	#0006
SEVERITY	Informational
STATUS	Closed
LOCATION	• Rev 1 - LPStaker.sol -> 476 *function depositToMaster(uint256 pid) external onlyOwner*

DESCRIPTION	Functions that change important variables should emit events such that users can more easily monitor the change.
RECOMMENDATION	Emit events from these functions.
RESOLUTION	The recommended changes were implemented. Reviewed in commit [96048adc10c1d304feaef3bb5d01960dcb0f7a5f]

On-Chain Analysis

Not Analyzed Yet

External Addresses

Externally Owned Accounts

Owner

ACCOUNT	Address
USAGE	0x... *LPStaker.owner* - Variable
IMPACT	• receives elevated permissions as owner, operator, or other

External Contracts

These contracts are not part of the audit scope.

LP Token

ADDRESS	0x3ba80AfDDcdcc301435A8fB8d198cCDb72Bc9a73
USAGE	0x... *LPStaker.lpToken* - Immutable
IMPACT	• ERC20 Token

Shade

ADDRESS	0x3c88baD5dcd1EbF35a0BF9cD1AA341BB387fF73A
USAGE	0x... *LPStaker.shade* - Immutable
IMPACT	• ERC20 Token

MasterChef

ADDRESS	0x8b7bcce67d2566D26393A6b81cAE010762C196B2
USAGE	0x... *LPStaker.masterChef* - Immutable
IMPACT	• impacts ability to deposit or withdraw tokens

Rewards Staker

ADDRESS	TBC
USAGE	0x... *LPStaker.rewardsStaker* - Variable
IMPACT	• impacts ability to deposit or withdraw tokens • receives transfer of tokens deposited or minted by project

Appendix A - Reviewed Documents

Document	Address
LPStaker.sol	N/A

Revisions

Revision 1	Zip file
Revision 2	96048adc10c1d304feaef3bb5d01960dcb0f7a5f

Imported Contracts

Contracts	Version

Appendix B - Risk Ratings

Risk	Description
High Risk	A fatal vulnerability that can cause the loss of all Tokens / Funds.
Medium Risk	A vulnerability that can cause the loss of some Tokens / Funds.
Low Risk	A vulnerability which can cause the loss of protocol functionality.
Informational	Non-security issues such as functionality, style, and convention.

Appendix C - Finding Statuses

Closed	Contracts were modified to permanently resolve the finding.
Mitigated	The finding was resolved by other methods such as revoking contract ownership. The issue may require monitoring, for example in the case of a time lock.
Partially Closed	Contracts were updated to fix the issue in some parts of the code.
Partially Mitigated	Fixed by project specific methods which cannot be verified on chain. Examples include compounding at a given frequency.
Open	The finding was not addressed.

Appendix D - Audit Procedure

A typical Obelisk audit uses a combination of the three following methods:

Manual analysis consists of a direct inspection of the contracts to identify any security issues. Obelisk auditors use their experience in software development to spot vulnerabilities. Their familiarity with common contracts allows them to identify a wide range of issues in both forked contracts as well as original code.

Static analysis is software analysis of the contracts. Such analysis is called “static” as it examines the code outside of a runtime environment. Static analysis is a powerful tool used by auditors to identify subtle issues and to verify the results of manual analysis.

On-chain analysis is the audit of the contracts as they are deployed on the block-chain. This procedure verifies that:

- deployed contracts match those which were audited in manual/static analysis;
- contract values are set to reasonable values;
- contracts are connected so that interdependent contract function correctly;
- and the ability to modify contract values is restricted via a timelock or DAO mechanism. (We recommend a timelock value of at least 72 hours)

Each obelisk audit is performed by at least two independent auditors who perform their analysis separately.

After the analysis is complete, the auditors will make recommendations for each issue based on best practice and industry standards. The project team can then resolve the issues, and the auditors will verify that the issues have been resolved with no new issues introduced.

Our auditing method lays a particular focus on the following important concepts:

- Quality code and the use of best practices, industry standards, and thoroughly tested libraries.
- Testing the contract from different angles to ensure that it works under a multitude of circumstances.
- Referencing the contracts through databases of common security flaws.

Follow Obelisk Auditing for the Latest Information

 ObeliskOrg

 ObeliskOrg

OBELISK

Part of Tibereum Group