OBELISK

Part of Tibereum Group

AUDITING REPORT

Version Notes

Version	No. Pages	Date	Revised By	Notes
1.0	Total: 24	2022-02-11	Zapmore, DoD4uFN	Audit Final

Audit Notes

Audit Date	YYYY-MM-DD - YYYY-MM-DD
Auditor/Auditors	DoD4uFN, ByFixter
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	OB574147293

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. In instances where an auditor or team member has a personal connection with the audited project, that auditor or team member will be excluded from viewing or impacting any internal communication regarding the specific audit.

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	Neverland
Description	“Come with me where dreams are born and time is never planned.”
Website	https://t.co/8YAs3MlmGH
Contact	https://twitter.com/NeverlandFi
Contact information	@XXXX on TG
Token Name(s)	N/A
Token Short	N/A
Contract(s)	See Appendix A
Code Language	Solidity
Chain	Klaytn

Audit of Neverland

> **This is a partial audit of two of Neverlands' contracts which are audited by Obelisk. The project also consists of multiple contracts that were not audited by Obelisk.**

Obelisk was commissioned by Neverland on the 21st of January 2022 to conduct an audit of **Neverlands' EtcBondDepository and StakingHelper contracts**. The following audit was conducted between the 27th of January 2022 and the 10th of February 2022. 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.

The audit was conducted on contracts provided by the Neverland team. There was one Medium Risk issue found, issue #1 where variables can be changed after deployment. Issue #2 and issue #9 are present because of a lack of timelock for sensitive parts. Issue #3 and #4 refer to fail-safes that are missing but are deemed to be of a low-risk nature.

There was an on-chain analysis conducted to match the deployed contracts with the audited ones, and they were a match.

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 Neverland project**.**

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

Summary Table

Finding	ID	Severity	Status
Initialization Function Can Be Called Multiple Times Without Bounds	#0001	Medium Risk	Open
No Timelock	#0002	Low Risk	Open
Use Safe Transfer	#0003	Low Risk	Open
No Upper Bound To Protocol Value	#0004	Low Risk	Open
Inconsistent Code Style	#0005	Informational	Open
Unused Functions And Modifiers	#0006	Informational	Open
Missing Zero Checks	#0007	Informational	Open
No Events Emitted For Changes To Protocol Values	#0008	Informational	Open
Owner Is An EOA	#0009	Low Risk	Open
Unverified Contracts	#0010	Informational	Open

Findings

Manual Analysis

Initialization Function Can Be Called Multiple Times Without Bounds

FINDING ID	#0001
SEVERITY	Medium Risk
STATUS	Open
LOCATION	EtcBondDepository.sol -> 749-768

```
1    function initializeBondTerms(
2        uint _discount,
3        uint _minimumPrice,
4        uint _maxPayout,
5        uint _maxDebt,
6        uint _fee,
7        uint _initialDebt,
8        uint32 _vestingTerm
9    ) external onlyPolicy() {
10       terms = Terms ({
11           discount: _discount,
12           vestingTerm: _vestingTerm,
13           minimumPrice: _minimumPrice,
14           maxPayout: _maxPayout,
15           fee: _fee,
16           maxDebt: _maxDebt
17       });
18       totalDebt = _initialDebt;
19       lastDecay = uint32(block.timestamp);
20   }
```

DESCRIPTION	Function *initializeBondTerms* does not have a mechanism to check for subsequent calls. A subsequent call might change the current terms to any value because there are no boundaries.
RECOMMENDATION	Adjust *initializeBondTerms* to only be callable once. Add bounds for the parameters.
RESOLUTION	N/A

No Timelock

FINDING ID	#0002
SEVERITY	Low Risk
STATUS	Open
LOCATION	• EtcBondDepository.sol -> 781: *function setBondTerms (PARAMETER _parameter, uint _input) external onlyPolicy()* • EtcBondDepository.sol -> 807: *function setStaking(address _staking, bool _helper) external onlyPolicy()* • EtcBondDepository.sol -> 817: *function setApprovedContract(address _contract) external onlyPolicy()* • EtcBondDepository.sol -> 940: *function changeCalc(address _contract) external onlyPolicy()*

DESCRIPTION	Important functions should be limited to a contract under the ownership of a timelock. No timelock has been provided to Obelisk.
RECOMMENDATION	Obelisk recommends a minimum delay of 72 hours.
RESOLUTION	N/A

Use Safe Transfer

FINDING ID	#0003
SEVERITY	Low Risk
STATUS	Open

LOCATION	• EtcBondDepository.sol -> 955: *IERC20(Time).transfer(_recipient, _amount); // send payout*
	• StakingHelper.sol -> 95: *IERC20(Time).transferFrom(msg.sender, address(this), _amount);*
	• StakingWarmup.sol -> 91: *IERC20(MEMOries).transfer(_staker, _amount);*

DESCRIPTION	Direct transfer functions are called.
RECOMMENDATION	Use OpenZeppelin's safe transfer functions. The safe transfer function is used to catch when a transfer fails as well as unusual token behavior.
RESOLUTION	N/A

No Upper Bound To Protocol Value

FINDING ID	#0004
SEVERITY	Low Risk
STATUS	Open
LOCATION	EtcBondDepository.sol -> 782-785

```
1     if ( _parameter == PARAMETER.VESTING ) { // 0
2         require( _input >= 129600, "Vesting must be longer than 36
  hours" );
3         terms.vestingTerm = uint32(_input);
4     }
```

DESCRIPTION	When setting the protocol value *terms.vestingTerm*, there is no upper bound. Since *terms.vestingTerm* is a *uint32* variable, the vesting term can be set up to 2^32 seconds.
RECOMMENDATION	Add an upper bound to the value that *terms.vestingTerm* can be set to.
RESOLUTION	N/A

Inconsistent Code Style

FINDING ID	#0005
SEVERITY	Informational
STATUS	Open
LOCATION	EtcBondDepository.sol -> 870-879

```
1     uint8 bondtype=ITreasury(treasury).reserveType(principle);
2     if(bondtype==1){//50% safety margin
3         uint markDownValue=profit.div(2).add(payout).add(fee);
4         totalDebt=totalDebt.add(markDownValue);
5     }else if(bondtype==2){
6         uint
  markDownValue=ITreasury(treasury).markvalueOf(principle,_amount);
7         totalDebt=totalDebt.add(markDownValue);
8     }else{
9         totalDebt = totalDebt.add( value );
10    }
```

DESCRIPTION	The code style in this area is significantly different from the rest of the contract, for example, with spacing between operators.
RECOMMENDATION	Follow the code standard that is in the rest of the contract.
RESOLUTION	N/A

Static Analysis

Unused Functions And Modifiers

FINDING ID	#0006
SEVERITY	Informational
STATUS	Open
LOCATION	• EtcBondDepository.sol -> 1105-1107: *function refundETH() internal* • EtcBondDepository.sol -> 1113-1117: *function safeTransferETH(address to, uint256 value) internal*

DESCRIPTION	The noted functions and modifiers are never used.
RECOMMENDATION	Remove the unused functions and modifiers or incorporate them into the contract functionality.
RESOLUTION	N/A

Missing Zero Checks

FINDING ID	#0007
SEVERITY	Informational
STATUS	Open
LOCATION	EtcBondDepository.sol -> 723-739

```
1     constructor (
2         address _Time,
3         address _principle,
4         address _treasury,
5         address _DAO,
6         address _bondCalculator
7     ) {
8         require( _Time != address(0) );
9         Time = _Time;
10        require( _principle != address(0) );
11        principle = _principle;
12        require( _treasury != address(0) );
13        treasury = _treasury;
14        require( _DAO != address(0) );
15        DAO = _DAO;
16        bondCalculator = _bondCalculator;
17    }
```

DESCRIPTION	The contract address *bondCalculator* can be set to zero address.
RECOMMENDATION	Add a check to ensure contract values are never set to an invalid zero address.
RESOLUTION	N/A

No Events Emitted For Changes To Protocol Values

FINDING ID	#0008
SEVERITY	Informational
STATUS	Open
LOCATION	• EtcBondCalc.sol -> 282-284: *function addPricefeed(address _token,address _LP) external onlyOwner* • EtcBondCalc.sol -> 285-287: *function addStable(address _token,uint _decimal) external onlyOwner* • EtcBondDepository.sol -> 781-799: *function setBondTerms(PARAMETER _parameter, uint _input) external onlyPolicy()* • EtcBondDepository.sol -> 807-816: *function setStaking(address _staking, bool _helper) external onlyPolicy()* • EtcBondDepository.sol -> 817-819: *function setApprovedContract(address _contract) external onlyPolicy()* • EtcBondDepository.sol -> 940-942: *function changeCalc(address _contract) external onlyPolicy()* • MarkdownCal.sol -> 277-280: *function setHook(address _Time) external* • Staking.sol -> 762-772: *function setContract(CONTRACTS _contract, address _address) external onlyManager()* • Staking.sol -> 778-780: *function setWarmup(uint _warmupPeriod) external onlyManager()* • Treasury.sol -> 399: *function resetReservelist(address[] memory _list) external onlyManager()*

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	N/A

On-Chain Analysis

Owner Is An EOA

FINDING ID	#0009
SEVERITY	Low Risk
STATUS	Open
LOCATION	EtcBondDepository - USDC-USDT 0xf6b61964d474aD05C40709CbdEA9CBd4CE1e9B0A EtcBondDepository - HOOK-USDC 0xDc749F16f433537d56Af82212be2b25a992896B9 EtcBondDepository - DAI-USDT 0x7b8f01dD885B124202EB2f72Efb3c44814320Fcb

DESCRIPTION	The Policy (Owner) of the aforementioned EtcBondDepository contracts is an EOA. The functions that only the Owner can call are : *initializeBondTerms()*, *setBondTerms()*, *setStaking()*, *setApprovedContract()*, *changeCalc()*. Policy (Owner) address 0x1fa5702cEA443906Be280CA7951C89AA1d5BC792
RECOMMENDATION	It is recommended to have a timelock of at least 72h.
RESOLUTION	N/A

Unverified Contracts

FINDING ID	#0010
SEVERITY	Informational
STATUS	Open
LOCATION	EtcBondDepository - USDC-USDT 0xf6b61964d474aD05C40709CbdEA9CBd4CE1e9B0A EtcBondDepository - HOOK-USDC 0xDc749F16f433537d56Af82212be2b25a992896B9 EtcBondDepository - DAI-USDT 0x7b8f01dD885B124202EB2f72Efb3c44814320Fcb StakingHelper 0xd02d05cB7E2e7315839EBF09382b3Fb265D52bD9 StakingHelper With Blacklist 0x641BF9F1E0C28F7214986559f59888AaE2bB6b75

DESCRIPTION	Noted contracts were not verified on the explorer.
RECOMMENDATION	Verification is not possible on the klaytn explorer. We recommend that the contracts be made public so users can check themselves.
RESOLUTION	N/A

External Addresses

Externally Owned Accounts

Owner

ACCOUNT	0x1fa5702cEA443906Be280CA7951C89AA1d5BC792
USAGE	0x7b8f01dD885B124202EB2f72Efb3c44814320Fcb 0xDc749F16f433537d56Af82212be2b25a992896B9 0xf6b61964d474aD05C40709CbdEA9CBd4CE1e9B0A *EtcBondDepository.policy* - Variable
IMPACT	• receives elevated permissions as owner, operator, or other

External Contracts

These contracts are not part of the audit scope.

DAO

ADDRESS	0x94c4Ac7276bB9d65A9982926e9F1A1d0Df112592
USAGE	0x7b8f01dD885B124202EB2f72Efb3c44814320Fcb 0xDc749F16f433537d56Af82212be2b25a992896B9 0xf6b61964d474aD05C40709CbdEA9CBd4CE1e9B0A *EtcBondDepository.DAO* - Immutable
IMPACT	• receives transfer of tokens deposited by users

Bond Calculator

ADDRESS	0x94E676922B4c52Fc06e88bAe96432f79d908fE69
USAGE	0x7b8f01dD885B124202EB2f72Efb3c44814320Fcb 0xDc749F16f433537d56Af82212be2b25a992896B9 0xf6b61964d474aD05C40709CbdEA9CBd4CE1e9B0A *EtcBondDepository.bondCalculator* - Variable
IMPACT	• impacts ability to deposit or withdraw tokens

Principle

ADDRESS	0xc320066b25B731A11767834839Fe57f9b2186f84
USAGE	0x7b8f01dD885B124202EB2f72Efb3c44814320Fcb 0xDc749F16f433537d56Af82212be2b25a992896B9 0xf6b61964d474aD05C40709CbdEA9CBd4CE1e9B0A *EtcBondDepository.principle* - Immutable
IMPACT	• ERC20 Token

Staking

ADDRESS	0xa2dF64c46Dd346721e7834A6bC944b792b8130AB
USAGE	0x641BF9F1E0C28F7214986559f59888AaE2bB6b75 0xd02d05cB7E2e7315839EBF09382b3Fb265D52bD9 *StakingHelper.Staking* - Immutable
IMPACT	• receives transfer of tokens deposited by users

Time Token

ADDRESS	0x8eF50FA375Fc64b9815E51f28F4b83c05D57ac43
USAGE	0x7b8f01dD885B124202EB2f72Efb3c44814320Fcb 0xDc749F16f433537d56Af82212be2b25a992896B9 0xf6b61964d474aD05C40709CbdEA9CBd4CE1e9B0A *EtcBondDepository.Time* - Immutable 0x641BF9F1E0C28F7214986559f59888AaE2bB6b75 0xd02d05cB7E2e7315839EBF09382b3Fb265D52bD9 *StakingHelper.Time* - Immutable
IMPACT	• ERC20 Token

Treasury

ADDRESS	0xD9Ca65f673a93C69E566eA03c3A441705b3B3B6B
USAGE	0x7b8f01dD885B124202EB2f72Efb3c44814320Fcb 0xDc749F16f433537d56Af82212be2b25a992896B9 0xf6b61964d474aD05C40709CbdEA9CBd4CE1e9B0A *EtcBondDepository.treasury* - Immutable
IMPACT	• receives transfer of tokens deposited by users

Appendix A - Reviewed Documents

Document	Address
EtcBondCal.sol	N/A
EtcBondDepository.sol	USDC-USDT 0xf6b61964d474aD05C40709CbdEA9CBd4CE1e9B0A HOOK-USDC 0xDc749F16f433537d56Af82212be2b25a992896B9 DAI-USDT 0x7b8f01dD885B124202EB2f72Efb3c44814320Fcb
MarkdownCal.sol	N/A
Staking.sol	N/A
StakingHelper.sol	0xd02d05cB7E2e7315839EBF09382b3Fb265D52bD9 With Blacklist 0x641BF9F1E0C28F7214986559f59888AaE2bB6b75
StakingWarmup.sol	N/A
Treasury.sol	N/A

Revisions

Revision 1	2b58d5fae9c3c9b30b05a44b366fa9fd3d192ca8

Imported Contracts

No contracts were imported.

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 that 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 contracts 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 practices 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