OBELISK

Part of Tibereum Group

AUDITING REPORT

Version Notes

Version	No. Pages	Date	Revised By	Notes
1.0	Total: 84	2022-02-16	Zapmore, Donut	Audit Final

Audit Notes

Audit Date	2021-11-30 - 2022-02-15
Auditor/Auditors	Donut, MrTeaThyme, 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	OB65969010

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

Version Notes 2

Audit Notes 2

Disclaimer 2

Obelisk Auditing 3

Audit Information 3

Project Information 6

Audit of Summit v2 7

Summary Table 8

Findings 11

Manual Analysis 11

Deity Balances Update Out Of Order 11

Emergency Withdraw Has Too Many Dependencies 13

No Way To Call Emergency Withdraw On Underlying Contracts 15

Setting Signature Specific Delay Uses The General Delay 16

Off-Chain Randomness Cannot Be Guaranteed 18

Randomness Seed Round Can Be Longer Than SubCartographer Round 19

VRFModule And ElevationHelper Can Be Disconnected 21

Passthroughs Do Not Account For Fees When Enacting Or Retiring 23

Elevation Rewards Duplicated For First Interacted Round 25

Everest Used For Emissions Is Different From Balance 26

Expedition Emissions Can Be Extended Indefinitely 28

Multiple Expeditions Cannot Rollover ElevationHelper 30

Initial Round Still Can Generate Rewards 32

No Limit For Protocol Values 34

Minimum Timelock Duration Insufficient 35

Alloc Multiplier Is 8 Bit Integer 36

Random Numbers Are Re-Used Between Elevations 37

Protocol Values Should Be Public 38

Unbounded Loop 39

Changing Everest Lock Times Uses Incorrect Multiplier 40

Changing Everest Lock Time Limits Does Not Affect Existing Lock Times 41

Lock Duration Can Only Be Raised By Increasing Increments 43

Updating User Everest Requires Locking Or Withdraw Interaction 44

Expeditions Can Be Enabled Or Disabled During A Round 46

Bonus Timestamp Not Set On Deposit 47

Initial Referral BP Is Out Of Bounds 48

Referral Cycles Not Prevented 49

Identical Contracts 50

Tokens Are Transferred To Burn Address Indirectly (Gas Optimization) 51

Totem Supplies Always Returns 10 Values 52

Redundant Check For Active Pool (Gas Optimization) 54
Redundant Check When Setting Summit Per Second (Gas Optimization) 55
Hypothetical Rewards Calculation Does Not Match Rollover 56
Error Message Doesn't Reflect Check 57
Resetting Deposit Timestamp For Tax Can Be Avoided 59
Setter Does Not Change Intended Variable 60
Linear Scaling Incorrect When Lower Bound Value Exceeds Upper Bound Value 61
Supplies Updated Incorrectly When Changing Both Deity and Safety Factor 62
Static Analysis 63
Contract Values Can Be Constant Or Immutable (Gas Optimization) 63
Unused Library 64
No Events Emitted For Changes To Protocol Values 65
Unused Variables 67
Unused Functions And Modifiers 68
Missing Zero Checks 69
On-Chain Analysis 70
Treasuries And LP Generation Are Externally Owned Accounts 70
Missing Signature Specific Delay 71
Changes To Deployed Contract 73

External Addresses 74
Externally Owned Accounts 74
Admin 74
Expedition Treasury 74
LP Generator 75
Treasury 75
Trusted Seeder 75
External Contracts 76
Deposit Tokens 76
SummitV1 77
Beefy Vaults 77

Appendix A - Reviewed Documents 79
Revisions 80
Imported Contracts 81

Appendix B - Risk Ratings 82

Appendix C - Finding Statuses 82

Appendix D - Audit Procedure 83

Project Information

Name	Summit
Description	Summit Defi is bringing new and unique features to the #DeFi space, starting off with what we are calling "Yield Multiplying" launching first on $FTM
Website	https://ftm.summitdefi.com/
Contact	https://twitter.com/SummitDefi
Contact information	@architect_dev on TG
Token Name(s)	N/A
Token Short	N/A
Contract(s)	See Appendix A
Code Language	Solidity
Chain	Fantom

> **After working through, Closing and Mitigating issues found, the current implementation of the contracts is as stated.**

Obelisk was commissioned by Summit on the 29th of November 2021 to conduct a comprehensive audit of Summits' contracts. The following audit was conducted between the 30th of November 2021 and the 15th of February 2022. Two of Obelisk's security experts went through the related contracts manually using industry standards to find if vulnerabilities could be exploited either by the project team or users.

The code is mostly written from scratch and it's then natural to have a multitude of findings when a new pair of eyes go through the code. The auditors gave suggestions on solving the findings they found and the project team was eager to listen and solve the majority of the issues.

Overall, all **High-Risk** issues are closed.

Most **Medium-Risk** findings are either Closed, Mitigated. Issue #5 and #8 are Partially Mitigated which means they work in the current setup, however, care should be taken when the setup is changed. Issue #45 refers to that the treasury addresses, as well as the lp generator address, are externally owned accounts, currently without multising, which needs to be considered.

All **Low-Risk** issues besides issue #19 are Closed. Low-Risk issue #19 adds no risk to losing funds as the update can be done independently in case it's needed.

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 Summit project**.** This document is a summary of the findings that the auditors found.

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

Summary Table

Finding	ID	Severity	Status
Deity Balances Update Out Of Order	#0001	High Risk	Closed
Emergency Withdraw Has Too Many Dependencies	#0002	High Risk	Closed
No Way To Call Emergency Withdraw On Underlying Contracts	#0003	High Risk	Closed
Setting Signature Specific Delay Uses The General Delay	#0004	High Risk	Closed
Off Chain Randomness Cannot Be Guaranteed	#0005	Medium Risk	Partially Mitigated
Randomness Seed Round Can Be Longer Than SubCartographer Round	#0006	Medium Risk	Closed
VRFModule And ElevationHelper Can Be Disconnected	#0007	Medium Risk	Mitigated
Passthroughs Do Not Account For Fees When Enacting Or Retiring	#0008	Medium Risk	Partially Mitigated
Elevation Rewards Duplicated For First Interacted Round	#0009	Medium Risk	Closed
Everest Used For Emissions Is Different From Balance	#0010	Medium Risk	Mitigated
Expedition Emissions Can Be Extended Indefinitely	#0011	Medium Risk	Closed
Multiple Expeditions Cannot Rollover ElevationHelper	#0012	Medium Risk	Mitigated
Initial Round Still Can Generate Rewards	#0013	Medium Risk	Closed
No Limit For Protocol Values	#0014	Medium Risk	Closed
Minimum Timelock Duration Insufficient	#0015	Low Risk	Mitigated

Alloc Multiplier Is 8 Bit Integer	#0016	Low Risk	Closed
Random Numbers Are Re-Used Between Elevations	#0017	Low Risk	Closed
Protocol Values Should Be Public	#0018	Low Risk	Closed
Unbounded Loop	#0019	Low Risk	Open
Changing Everest Lock Times Uses Incorrect Multiplier	#0020	Low Risk	Closed
Changing Everest Lock Time Limits Does Not Affect Existing Lock Times	#0021	Low Risk	Closed
Lock Duration Can Only Be Raised By Increasing Increments	#0022	Low Risk	Closed
Updating User Everest Requires Locking Or Withdraw Interaction	#0023	Low Risk	Closed
Expeditions Can Be Enabled Or Disabled During A Round	#0024	Low Risk	Closed
Bonus Timestamp Not Set On Deposit	#0025	Low Risk	Closed
Initial Referral BP Is Out Of Bounds	#0026	Low Risk	Closed
Referral Cycles Not Prevented	#0027	Informational	Closed
Identical Contracts	#0028	Informational	Closed
Tokens Are Transferred To Burn Address Indirectly (Gas Optimization)	#0029	Informational	Open
Totem Supplies Always Returns 10 Values	#0030	Informational	Closed
Redundant Check For Active Pool (Gas Optimization)	#0031	Informational	Closed
Redundant Check When Setting Summit Per Second (Gas Optimization)	#0032	Informational	Closed
Hypothetical Rewards	#0033	Informational	Closed

Calculation Does Not Match Rollover			
Error Message Doesn't Reflect Check	#0034	Informational	Closed
Resetting Deposit Timestamp For Tax Can Be Avoided	#0035	Informational	Closed
Contract Values Can Be Constant Or Immutable (Gas Optimization)	#0036	Informational	Closed
Unused Library	#0037	Informational	Closed
No Events Emitted For Changes To Protocol Values	#0038	Informational	Closed
Unused Variables	#0039	Informational	Closed
Unused Functions And Modifiers	#0040	Informational	Closed
Missing Zero Checks	#0041	Low Risk	Closed
Setter Does Not Change Intended Variable	#0042	Low Risk	Closed
Linear Scaling Incorrect When Lower Bound Value Exceeds Upper Bound Value	#0043	Low Risk	Closed
Supplies Updated Incorrectly When Changing Both Deity and Safety Factor	#0044	Medium Risk	Closed
Treasuries And LP Generation Are Externally Owned Accounts	#0045	Medium Risk	Open
Missing Signature Specific Delay	#0046	Low Risk	Mitigated
Changes To Deployed Contract	#0047	Informational	Closed

Findings

Manual Analysis

Deity Balances Update Out Of Order

FINDING ID	#0001
SEVERITY	High Risk
STATUS	Closed
LOCATION	Rev4 - ExpeditionV2.sol -> 715-742

```
1     function selectDeity(uint8 _newDeity)
2         public
3         nonReentrant validDeity(_newDeity)
  expeditionInteractionsAvailable
4     {
5         // ...
6
7         // Update user deity in state
8         user.deity = _newDeity;
9         user.deitySelected = true;
10        user.deitySelectionRound =
  elevationHelper.roundNumber(EXPEDITION);
11
12        // Update user's interaction in this expedition
13        _updateUserRoundInteraction(user);
14
15        // Transfer deitied everest from previous deity to new deity
16        if (user.entered) {
17            expeditionInfo.supplies.deity[user.deity] -=
  user.deitiedSupply;
18            expeditionInfo.supplies.deity[_newDeity] +=
  user.deitiedSupply;
19        }
20
21        // ...
22    }
```

DESCRIPTION	The users' deity is set before the expedition info is modified leading to no change in deity supplies. The function merely subtracts user balance from the new deity then adds it back. This can ultimately lead to users being unable to withdraw from the EverestToken because *updateUserEverest()* may

	revert.
RECOMMENDATION	Update the user deity after balance transfer occurs.
RESOLUTION	The deity supplies are now updated in the correct order.

Emergency Withdraw Has Too Many Dependencies

FINDING ID	#0002
SEVERITY	High Risk
STATUS	Closed
LOCATION	Rev-2 - CartographerOasis.sol -> 406-418

```
    function emergencyWithdraw(address _token, address _userAdd)
        external override
        nonReentrant onlyCartographer poolExists(_token)
validUserAdd(_userAdd)
        returns (uint256)
    {
        return _unifiedWithdraw(
            poolInfo[_token],
            userInfo[_token][_userAdd],
            userInfo[_token][_userAdd].staked,
            _userAdd,
            false
        );
    }
```

LOCATION	Rev-2 - CartographerElevation.sol -> 1038-1051

```
    function emergencyWithdraw(address _token, address _userAdd)
        external override
        nonReentrant onlyCartographer poolExists(_token)
validUserAdd(_userAdd)
        returns (uint256)
    {
        return _unifiedWithdraw(
            poolInfo[_token],
            userInfo[_token][_userAdd],
            userInfo[_token][_userAdd].staked,
            _userAdd,
            false,
            true
        );
    }
```

DESCRIPTION	The *emergencyWithdraw()* functions are reliant on functionality of multiple other contracts including *Cartographer*, *ElevationHelper*, all other sub cartographers, and the pool's associated passthrough.

	The purpose of an emergency withdrawal function should be to allow users to extract deposited funds in case of broken logic elsewhere in the contracts.
RECOMMENDATION	Simplify the emergency withdrawal process to have the minimum effects and interactions between only the *Cartographer*, *CartographerOasis*, and passthrough. Do not update the pool when doing an emergency withdrawal.
RESOLUTION	The *emergencyWithdraw()* functions no longer update the pool, reducing the dependency of the function. Project team comment: “Removing the updatePool from the EmergencyWithdraw pipeline is as far as we can go in slimming down functionality. Any further removal of functionality may allow the contract's internal states to become out of sync”

No Way To Call Emergency Withdraw On Underlying Contracts

FINDING ID	#0003
SEVERITY	High Risk
STATUS	Closed
LOCATION	Rev-2 - Cartographer.sol

DESCRIPTION	The top-level cartographer contract does not expose an emergency withdrawal functionality to end-users and third-party protocols. This can lead to loss of user funds in the event of a protocol malfunction or if the underlying protocol malfunctions.
RECOMMENDATION	Implement an emergency withdrawal function.
RESOLUTION	An emergency withdrawal function call was added to the Cartographer.

Setting Signature Specific Delay Uses The General Delay

FINDING ID	#0004
SEVERITY	High Risk
STATUS	Closed
LOCATION	Rev-4 - Timelock.sol -> 59-67

```
1     function setFunctionSpecificDelay(string memory signature_, uint
  specificDelay_) public {
2         require(msg.sender == address(this), "Timelock::setDelay:
  Call must come from Timelock.");
3         require(specificDelay_ >= delay, "Timelock::setDelay:
  Signature specific delay must exceed base delay.");
4         require(specificDelay_ <= MAXIMUM_DELAY, "Timelock::setDelay:
  Signature specific delay must not exceed maximum delay.");
5
6         signatureSpecificDelay[sigToHash(signature_)] =
  specificDelay_;
7
8         emit NewSpecificDelay(signature_, specificDelay_);
9     }
```

DESCRIPTION	Suppose a secure function is set with a delay of 10 days, with the general delay being 1 day. The delay of that secure function can be changed to 1 day, after the general delay. Then the formerly secure function can be called 1 day after that. For example, to call a dangerous function, the following steps can be taken: • queue *setFunctionSpecificDelay("dangerousFunction()",1 day)* • wait 1 day • execute *setFunctionSpecificDelay("dangerousFunction()",1 day)* • queue *dangerousFunction()* • wait 1 day • execute *dangerousFunction()* In general, a specific delay of more than double the general delay will not be effective.
RECOMMENDATION	Apply the function-specific delay to queuing changes of function-specific delays.

RESOLUTION	Using *queueTransaction* was changed to use the existing delay when calling *setFunctionSpecificDelay()*.

Off-Chain Randomness Cannot Be Guaranteed

FINDING ID	#0005
SEVERITY	Medium Risk
STATUS	Partially Mitigated
LOCATION	Rev-2 - SummitVRFModule.sol -> 160-173

```
1     function receiveSealedSeed(bytes32 _sealedSeed)
2         public
3         onlyTrustedSeeder
4     {
5         require(nextSeedRoundAvailable(), "Already sealed seeded");
6
7         // Increment seed round and set next seed round end timestamp
8         seedRound += 1;
9         seedRoundEndTimestamp += (baseRoundDuration *
  seedRoundDurationMult);
10
11        // Store new sealed seed for next round of round rollovers
12        sealedSeed[seedRound] = _sealedSeed;
13        futureBlockNumber[seedRound] = block.number + 1;
14    }
```

DESCRIPTION	Randomness generated off-chain cannot be guaranteed as truly random. A malicious actor in control of the protocol may manipulate the operation of the trusted seeder. Additionally, if the seed is not generated in the 120s timeframe, the seed will effectively be null.
RECOMMENDATION	Use a cryptographically verifiable randomness oracle. Note: The noted implementation is not a verifiable randomness oracle, despite being called SummitVRFModule.
RESOLUTION	The contract can be replaced with a chainlink randomness oracle when it is made available. The contract was also renamed to SummitTrustedSeederRNGModule.

Randomness Seed Round Can Be Longer Than SubCartographer Round

FINDING ID	#0006
SEVERITY	Medium Risk
STATUS	Closed
LOCATION	Rev-2 - ElevationHelper.sol -> 56

```
    function setElevationRoundDurationMult(uint8 _elevation, uint8 _mult)
        public
        onlyOwner elevationOrExpedition(_elevation)
    {
        require(_mult > 0, "Duration mult must be non zero");
        pendingDurationMult[_elevation] = _mult;
    }
```

LOCATION	Rev-2 - ElevationHelper.sol -> 324

```
        uint256 rand = ISummitVRFModule(summitVRFModuleAdd).getRandomNumber(roundNumber[_elevation]);
```

LOCATION	Rev-2 - SummitVRFModule.sol -> 56

```
    uint256 seedRoundDurationMult = 2;
```

LOCATION	Rev-2 - SummitVRFModule.sol -> 151-154

```
1     function getRandomNumber(uint256 roundNumber) public view
  override returns (uint256) {
2         return uint256(keccak256(abi.encode(roundNumber,
  unsealedSeed[seedRound], futureBlockHash[seedRound])));
3     }
```

DESCRIPTION	The ElevationHelper uses the SummitVRFModule to get a random number for each round of a given elevation. However, if the elevation round is incremented faster than the randomness round, the values of *unsealedSeed* and *futureBlockHash* will be reused until a new seed is provided. This can be done by setting *durationMult* to 1.
RECOMMENDATION	Ensure that the randomness rounds cannot be longer than the elevation rounds. Add a check to ensure that rounds can only be rolled over if a new random number is ready.
RESOLUTION	Random number round duration is now 1 hour and as such cannot be longer than elevation rounds.

VRFModule And ElevationHelper Can Be Disconnected

FINDING ID	#0007
SEVERITY	Medium Risk
STATUS	Mitigated
LOCATION	Rev-2 - ElevationHelper.sol -> 266-271

```
1     function setSummitVRFModuleAdd (address _summitVRFModuleAdd)
2         public onlyOwner
3     {
4         require(_summitVRFModuleAdd != address(0), "SummitVRFModule
  missing");
5         summitVRFModuleAdd = _summitVRFModuleAdd;
6     }
```

LOCATION	Rev-2 - SummitVRFModule.sol -> 99-104

```
1     function setElevationHelper (address _elevationHelper)
2         public onlyOwner
3     {
4         require(_elevationHelper != address(0), "ElevationHelper
  missing");
5         elevationHelper = _elevationHelper;
6     }
```

DESCRIPTION	The ElevationHelper and SummitVRFModule contracts can be disconnected from each other, breaking the protocol's functionality. Changing the SummitVRFModule can be used to cause the round timings of the two contracts to go out of sync. In particular, *SummitVRFModule.elevationHelper* can call *setSeedRoundEndTimestamp()*.
RECOMMENDATION	Ensure the noted setter functions are called only once.
RESOLUTION	This behavior is intended as the randomness module will be upgraded when chainlink is available. A timelock was added to delay any inappropriate changes to the contract settings. When implementing the chainlink randomness, ensure

	that functionality exists to sync the upgraded randomness module to the *ElevationHelper*. Project team comment: “The only upgrade that will be made to the VRFModule will be to replace it with an on-chain chainlink VRF, which will not need the seed round timestamp for its functionality.”

Passthroughs Do Not Account For Fees When Enacting Or Retiring

FINDING ID	#0008
SEVERITY	Medium Risk
STATUS	Partially Mitigated
LOCATION	Rev-2 - BeefyVaultV2Passthrough.sol -> 158-162

```
        uint256 tokenBalance = passthroughToken.balanceOf(address(this));

        // Return collective user's amount back to cartographer
        uint256 usersWithdrawn = tokenBalance > balance ? balance : tokenBalance;
        passthroughToken.safeTransfer(cartographer, usersWithdrawn);
```

LOCATION	Rev-2 - BeefyVaultV6NativePassthrough.sol -> 154-164 Rev-2 - BeefyVaultV6Passthrough.sol -> 154-164

```
        // Withdraw all from the vault
        uint256 sharesBalance = IERC20(vault).balanceOf(address(this));
        if (sharesBalance > 0) {
            IBeefyVault(vault).withdrawAll();
        }

        uint256 tokenBalance = passthroughToken.balanceOf(address(this));

        // Return collective user's amount back to cartographer
        uint256 usersWithdrawn = tokenBalance > balance ? balance : tokenBalance;
        passthroughToken.safeTransfer(cartographer, usersWithdrawn);
```

LOCATION	Rev-2 - MasterChefPassthrough.sol -> 103-105

```
        uint256 cartographerBalance = passthroughToken.balanceOf(cartographer);
        passthroughToken.safeTransferFrom(cartographer, address(this), cartographerBalance);
        IMasterChef(masterChef).deposit(masterChefPid, cartographerBalance);
```

LOCATION	Rev-2 - MasterChefPassthrough.sol -> 154-164

```
1         uint256 stakedAmount = balance();
2
3         // Withdraw all from the masterChef
4         IMasterChef(masterChef).withdraw(masterChefPid,
  stakedAmount);
5
6         // Return collective user's amount back to cartographer
7         passthroughToken.safeTransfer(cartographer, stakedAmount);
```

DESCRIPTION	The passthrough contracts deposit and withdraw tokens to underlying farming contracts or other vaults. During normal deposit and withdrawal, they take potential fees into account. However, no fees are accounted for when retiring.
RECOMMENDATION	Ensure that user deposits are correctly accounted for when retiring pass-through contracts.
RESOLUTION	Project team comment: “Retire is withdrawing the full amount in the vault. If the fees are covered by the earnings of the vault (withdrawn amount > running balance uint) then the running balance is sent to the cartographer, else the withdrawn amount is sent.” Obelisk Comment: As long as the withdrawal fee has been covered by the current rewards, this will work. The project team should ensure that changing strategies is done sparingly. Note: The delay for retiring strategies will be verified in the on-chain analysis.

Elevation Rewards Duplicated For First Interacted Round

FINDING ID	#0009
SEVERITY	Medium Risk
STATUS	Closed
LOCATION	Rev-2 - CartographerElevation.sol -> 764-772

```
        claimable += userFirstInteractedRoundWinnings(user, poolRoundInfo[pool.token][user.prevInteractedRound], totem);

        // Escape early if user interacted during previous round
        if (user.prevInteractedRound == currRound - 1) return claimable;

        // Add multiple rounds of precomputed mult delta for all rounds between first interacted and most recent round
        claimable += user.staked * (pool.totemRunningPrecomputedMult[totem] - user.winningsDebt) / 1e12;

        return claimable;
```

DESCRIPTION	The user's *winningsDebt* is set to the value of *pool.totemRunningPrecomputedMult[totem]* at the beginning of the round in which they stake. As a result, they will receive rewards from that round as if they staked for the entire round. Note: this can only be done once every other round due to the escape early clause.
RECOMMENDATION	Ensure that users' *winningsDebt* is tracked accurately.
RESOLUTION	The calculation of the winnings debt was changed to explicitly include any "rewards" accrued by the first interaction round. This will negate their existence for future claims and resolve the issue.

Everest Used For Emissions Is Different From Balance

FINDING ID	#0010
SEVERITY	Medium Risk
STATUS	Mitigated
LOCATION	@OpenZeppelin/ERC20.sol -> 36

```
1    mapping(address => uint256) private _balances;
```

LOCATION	EverestToken.sol -> 39-48

```
1    struct UserEverestInfo {
2        address userAdd;
3
4        uint256 everestOwned;
5        uint256 everestLockMultiplier;
6        uint256 lockDuration;
7        uint256 lockRelease;
8        uint256 summitLocked;
9    }
10   mapping(address => UserEverestInfo) public userEverestInfo;
```

LOCATION	EverestToken.sol -> 389-410

```
1    function withdrawLockedSummit(uint256 _everestAmount)
2        public
3        nonReentrant notPanic userEverestInfoExists userOwnsEverest
  userLockDurationSatisfied validEverestAmountToBurn(_everestAmount)
4    {
5        // ...
6        summit.safeTransfer(msg.sender, summitToWithdraw);
7        _burnEverest(msg.sender, _everestAmount);
8        // ...
9    }
```

DESCRIPTION	The EverestToken acts as a normal token, but uses a separate variable from *ERC20._balances* to determine users' emissions.

	Users require the exact amount of EverestToken as an ERC20 to withdraw their locked summit. If users attempt to trade their Everest, they may be unable to retrieve their stake.
RECOMMENDATION	Incorporate the *transfer* and *transferFrom* functionalities of the ERC20 token into the contract's behaviour. Alternatively, remove the ERC20 inheritance.
RESOLUTION	Project team comment: “We have added whitelisted transfer addresses for the EVEREST token, only transfers with a whitelisted address as either the source or destination will be permitted. This prevents users from accidentally losing their EVEREST token. The requirement to keep EVEREST tokens to unlock their SUMMIT will be made as clear to all users through the frontend.”

Expedition Emissions Can Be Extended Indefinitely

FINDING ID	#0011
SEVERITY	Medium Risk
STATUS	Closed
LOCATION	Rev2 - ExpeditionV2.sol -> 358-368

```
    function _recalculateExpeditionTokenEmissions(ExpeditionToken storage expedToken)
        internal
        returns (bool)
    {
        uint256 fund = expedToken.token.balanceOf(address(this)) - expedToken.markedForDist;

        expedToken.emissionsRemaining = fund;
        expedToken.roundEmission = fund == 0 ? 0 : fund / expeditionRunwayRounds;

        return fund > 0;
    }
```

LOCATION	ExpeditionV2.sol -> 381-389

```
    function addExpeditionFunds(address _token, uint256 _amount)
        public nonReentrant
    {
        require (_token == address(expeditionInfo.summit.token) || _token == address(expeditionInfo.usdc.token), "Invalid token to add to expedition");
        IERC20(_token).safeTransferFrom(msg.sender, address(this), _amount);
        _recalculateExpeditionEmissions();

        emit ExpeditionFundsAdded(_token, _amount);
    }
```

DESCRIPTION	The function *addExpeditionFunds()* can be called each round by any account. Each time, the round emissions will be re-calculated using *expeditionRunwayRounds*, effectively extending the rounds while reducing the emission rate. If this is called each

	round, it will result in an exponentially decaying emission rate.
RECOMMENDATION	Separate the functionality of funding the emissions and extending the emissions.
RESOLUTION	The recommended changes have been implemented.

Multiple Expeditions Cannot Rollover ElevationHelper

FINDING ID	#0012
SEVERITY	Medium Risk
STATUS	Mitigated
LOCATION	Rev-4 - ElevationHelper.sol -> 316-323

```
1     function validateRolloverAvailable(uint8 _elevation)
2         external view
3     {
4         // Elevation must be unlocked for round to rollover
5         require(block.timestamp >= unlockTimestamp[_elevation],
  "Elevation locked");
6         // Rollover only becomes available after the round has ended,
  if timestamp is before roundEnd, the round has already been rolled
  over and its end timestamp pushed into the future
7         require(block.timestamp >= roundEndTimestamp[_elevation],
  "Round already rolled over");
8     }
```

LOCATION	Rev-4 - ExpeditionV2.sol -> 496-515

```
1     function rollover()
2         public
3     {
4         // Ensure that the expedition is ready to be rolled over,
  ensures only a single user can perform the rollover
5         elevationHelper.validateRolloverAvailable(EXPEDITION);
6
7         // ...
8     }
```

LOCATION	Rev-4 - EverestToken.sol -> 424-431

```
1     function addEverestExtension(address _extension)
2         public
3         onlyOwner
4     {
5         require(_extension != address(0), "Missing extension");
6         require(everestExtensions.length() < 3, "Max extension cap
  reached");
7         everestExtensions.add(_extension);
8     }
```

DESCRIPTION	Each expedition contract will attempt to rollover the expedition round of the *ElevationHelper* separately. However, the *ElevationHelper* only allows a single rollover per round per elevation. Note that the EverestToken allows for up to 3 expedition-type contracts.
RECOMMENDATION	Check whether rolling over the elevation helper is necessary before rolling over the expedition contract.
RESOLUTION	Project team comment: “Only a single Expedition that interacts with the ElevationHelper will ever be active concurrently. Other Everest Extensions will be entirely independent of the rollover mechanic, including a DAO, standard farming, etc.”

Initial Round Still Can Generate Rewards

FINDING ID	#0013
SEVERITY	Medium Risk
STATUS	Closed
LOCATION	Rev-4 - ElevationHelper.sol -> 328-346

```
1     function selectWinningTotem(uint8 _elevation)
2         external
3         onlyCartographerOrExpedition
  elevationOrExpedition(_elevation)
4     {
5         // No winning totem should be selected for round 0, which
  takes place when the elevation is locked
6         if (roundNumber[_elevation] == 0) { return; }
7
8         // ...
9     }
```

LOCATION	Rev-4 - CartographerElevation.sol -> 657-686

```
1     function rollover()
2         external override
3         onlyCartographer
4     {
5         uint256 currRound = elevationHelper.roundNumber(elevation);
6         uint8 winningTotem = elevationHelper.winningTotem(elevation,
  currRound - 1);
7
8         // ...
9     }
```

LOCATION	Rev-4 - ExpeditionV2.sol -> 520-546

```
1     function _rolloverExpedition(uint256 _currRound)
2         internal
3     {
4         if (!expeditionInfo.live) return;
5
6         uint8 winningDeity = elevationHelper.winningTotem(EXPEDITION,
  _currRound - 1);
7
8         // ...
9     }
```

DESCRIPTION	The elevation helper does not select a winning totem in the very first round (index 0). However, the elevation and expedition contracts can generate winnings if a pool is active during this round.
RECOMMENDATION	Disable any earnings in round 0 or generate a winning totem/deity.
RESOLUTION	Elevation and expedition pools were set to never generate rewards on the first round they rollover.

No Limit For Protocol Values

FINDING ID	#0014
SEVERITY	Medium Risk
STATUS	Closed
LOCATION	Rev-4 - Cartographer.sol -> 1179-1185

```
1    function setTaxDecayDuration(uint256 _taxDecayDuration)
2        public
3        onlyOwner
4    {
5        taxDecayDuration = _taxDecayDuration;
6        emit SetTaxDecayDuration(_taxDecayDuration);
7    }
```

DESCRIPTION	The following values can be set arbitrarily high, potentially breaking the functionality of the contracts: - *taxDecayDuration*
RECOMMENDATION	Add an upper limit to the values.
RESOLUTION	The recommended changes have been implemented.

Minimum Timelock Duration Insufficient

FINDING ID	#0015
SEVERITY	Low Risk
STATUS	Mitigated
LOCATION	Rev-2 - Timelock.sol -> 23

```
1    uint public constant MINIMUM_DELAY = 6 hours;
```

DESCRIPTION	The minimum delay in the timelock contracts constants is below the minimum recommendation.
RECOMMENDATION	Obelisk recommends a minimum delay of 72 hours.
RESOLUTION	Project team comment: “V1 has shown the Summit team that a 24-hour timelock is not fast enough to respond or fine-tune in the way we like. However some functionality must be put on a longer timelock, and our new more robust function-specific timelock will allow us to handle both well.” Signature-specific timelock delays were added to high-risk functions. Refer to finding #46.

Alloc Multiplier Is 8 Bit Integer

FINDING ID	#0016
SEVERITY	Low Risk
STATUS	Closed
LOCATION	Rev-2 - ElevationHelper.sol -> 285

```
function setElevationAllocMultiplier(uint8 _elevation, uint8 _allocMultiplier)
        public
        onlyOwner allElevations(_elevation)
    {
        require(_allocMultiplier <= 300, "Multiplier cannot exceed 3X");
        pendingAllocMultiplier[_elevation] = _allocMultiplier;
        if (_elevation == OASIS) {
            allocMultiplier[_elevation] = _allocMultiplier;
        }
    }
```

DESCRIPTION	The type of *_allocMultiplier* is uint8 which has a maximum value of 255. Therefore comparing it with 300 will always return true. The alloc multiplier is used as a relative multiplier. Therefore even with a limit of 300, tokens could be weighted at 1, with another at 300 to make their relative weights 1:300.
RECOMMENDATION	Use a larger uint datatype for the alloc multiplier.
RESOLUTION	The recommended changes have been implemented.

Random Numbers Are Re-Used Between Elevations

FINDING ID	#0017
SEVERITY	Low Risk
STATUS	Closed
LOCATION	Rev-2 - ElevationHelper.sol -> 324

```
1         uint256 rand =
  ISummitVRFModule(summitVRFModuleAdd).getRandomNumber(roundNumber[_ele
  vation]);
```

LOCATION	Rev-2 - SummitVRFModule.sol -> 152-154

```
1     function getRandomNumber(uint256 roundNumber) public view
  override returns (uint256) {
2         return uint256(keccak256(abi.encode(roundNumber,
  unsealedSeed[seedRound], futureBlockHash[seedRound])));
3     }
```

DESCRIPTION	The same random seed will be used across multiple elevations which end their rounds concurrently. If the elevations have the same round number as well, the selection of winning totems from each elevation will not be independent.
RECOMMENDATION	Do not re-use random numbers.
RESOLUTION	The elevation number is hashed with the random number to produce a unique random number per elevation.

Protocol Values Should Be Public

FINDING ID	#0018
SEVERITY	Low Risk
STATUS	Closed
LOCATION	• Rev-2 - CartographerOasis.sol -> 55: *Cartographer cartographer;* • Rev-2 - CartographerOasis.sol -> 65: *mapping(address => EnumerableSet.AddressSet) userInteractingPools;* • Rev-2 - CartographerOasis.sol -> 76: *EnumerableSet.AddressSet private poolTokens;* • Rev-2 - CartographerOasis.sol -> 77: *EnumerableSet.AddressSet private activePools;* • Rev-2 - CartographerElevation.sol -> 101: *Cartographer cartographer;* • Rev-2 - CartographerElevation.sol -> 102: *ElevationHelper elevationHelper;* • Rev-2 - CartographerElevation.sol -> 127: *mapping(address => EnumerableSet.AddressSet) userInteractingPools;* • Rev-2 - CartographerElevation.sol -> 154: *EnumerableSet.AddressSet private poolTokens;* • Rev-2 - CartographerElevation.sol -> 155: *EnumerableSet.AddressSet private activePools;* • Rev-4 - Cartographer.sol -> 120: *address[] tokensWithAllocation;* • Rev-4 - EverestToken.sol -> 52: *EnumerableSet.AddressSet everestExtensions;* • Rev-4 - ExpeditionV2.sol -> 104: *ElevationHelper elevationHelper;* • Rev-7 @OpenZeppelin/AccessControl.sol -> 102: *mapping(bytes32 => RoleData) private _roles;*

DESCRIPTION	Variables critical to the operation of the protocol should be public or have an associated view function.
RECOMMENDATION	Add getter functions or change the values to be public.
RESOLUTION	The recommended changes have been implemented.

Unbounded Loop

FINDING ID	#0019
SEVERITY	Low Risk
STATUS	Open
LOCATION	• Rev-2 - CartographerOasis.sol -> 229-231: *for (uint16 index = 0; index < poolTokens.length(); index++) {* • Rev-2 - CartographerElevation.sol -> 444-446: *for (uint16 index = 0; index < poolTokens.length(); index++) {*

DESCRIPTION	Iterating over an unbounded array can cause transactions to revert due to the gas limit.
RECOMMENDATION	Provide a limit to the size of the array. Alternatively, pass a lower and upper index as parameters and iterate over a range.
RESOLUTION	The pools can be updated independently in case the mass updating of pools hits the gas limit. Project team comment: “If this ever runs into the gas limit, each pool can be updated independently. It is not necessary for the functionality of SUMMIT, only a convenience.”

Changing Everest Lock Times Uses Incorrect Multiplier

FINDING ID	#0020
SEVERITY	Low Risk
STATUS	Closed
LOCATION	Rev4 - EverestToken.sol -> 136

```
1        minLockTime = _lockTimeDays * 24 * 365;
```

LOCATION	Rev4 - EverestToken.sol -> 141

```
1        maxLockTime = _lockTimeDays * 24 * 365;
```

DESCRIPTION	Changing the lock time limits is implied to be in days (eg. a multiplier of *3600*24*) but instead uses a multiplier of *24*365*. This will cause the lock limits to be much smaller than anticipated. Note that *setLockTimeRequiredForTaxlessSummitWithdraw* and *setLockTimeRequiredForLockedSummitDeposit* use incorrect limits as well by assuming that their inputs are in days.
RECOMMENDATION	Use the correct multipliers.
RESOLUTION	The recommended changes have been implemented.

Changing Everest Lock Time Limits Does Not Affect Existing Lock Times

FINDING ID	#0021
SEVERITY	Low Risk
STATUS	Closed
LOCATION	Rev-2 - EverestToken.sol -> 134-143

```
1     function setMinLockTime(uint256 _lockTimeDays) public onlyOwner {
2         require(_lockTimeDays <= maxLockTime && _lockTimeDays >= 1 &&
  _lockTimeDays <= 30, "Invalid minimum lock time (1-30 days)");
3         minLockTime = _lockTimeDays * 24 * 365;
4         emit SetMinLockTime(_lockTimeDays);
5     }
6     function setMaxLockTime(uint256 _lockTimeDays) public onlyOwner {
7         require(_lockTimeDays >= minLockTime && _lockTimeDays >= 7 &&
  _lockTimeDays <= 730, "Invalid maximum lock time (7-730 days)");
8         maxLockTime = _lockTimeDays * 24 * 365;
9         emit SetMaxLockTime(_lockTimeDays);
10    }
```

LOCATION	Rev-2 - EverestToken.sol -> 154-163

```
1     function setLockTimeRequiredForTaxlessSummitWithdraw(uint256
  _lockTimeDays) public onlyOwner {
2         require(_lockTimeDays >= minLockTime && _lockTimeDays <=
  maxLockTime && _lockTimeDays >= 1 && _lockTimeDays <= 30, "Invalid
  taxless summit lock time (1-30 days)");
3         lockTimeRequiredForTaxlessSummitWithdraw = _lockTimeDays;
4         emit
  SetLockTimeRequiredForTaxlessSummitWithdraw(_lockTimeDays);
5     }
6     function setLockTimeRequiredForLockedSummitDeposit(uint256
  _lockTimeDays) public onlyOwner {
7         require(_lockTimeDays >= minLockTime && _lockTimeDays <=
  maxLockTime && _lockTimeDays >= 1 && _lockTimeDays <= 90, "Invalid
  locked summit lock time (1-90 days)");
8         lockTimeRequiredForClaimableSummitLock = _lockTimeDays;
9         emit
  SetLockTimeRequiredForLockedSummitDeposit(_lockTimeDays);
10    }
```

DESCRIPTION	The lock times are intended to be between the minimum and maximum lock times. However, changing the minimum and maximum lock times can result in the lock

	times being outside the expected bounds.
RECOMMENDATION	Update the lock times whenever changing the minimum or maximum lock times.
RESOLUTION	The lock time mechanism was changed to use an inflection point and a general lock time range. These are now strictly ordered from minLockTime to inflectionLockTime to maxLockTime.

Lock Duration Can Only Be Raised By Increasing Increments

FINDING ID	#0022
SEVERITY	Low Risk
STATUS	Closed
LOCATION	EverestToken.sol -> 280

```
1         require(_lockDuration >= everestInfo.lockDuration, "Lock
  duration must strictly increase");
```

DESCRIPTION	The duration of the lock can only increase even when extending the lock, thereby resetting its start time. This can prevent small incremental increases to the lock release time. If the value of *lockTimeRequiredForClaimableSummitLock* is reduced, it may cause a temporary loss of functionality in *SummitLocking.harvestWinnings()* and *ExpeditionV2._harvestExpedition()*.
RECOMMENDATION	Allow the lock duration to increase in smaller increments.
RESOLUTION	Project team has stated this is intentional. Project team comment: “The increaseLockDuration should only allow the user to increase. Users being able to reduce their lock duration would allow users to get the full benefit of the locking and then reduce to a short time to unlock their summit early. To reduce their lock duration they'll have to wait until their lock matures, then withdraw and lock with a new initial amount. Alternatively, a 2nd account can be used, but this is working as intended”

Updating User Everest Requires Locking Or Withdraw Interaction

FINDING ID	#0023
SEVERITY	Low Risk
STATUS	Closed
LOCATION	Rev-2 - ExpeditionV2.sol -> 670-695

```
    function updateUserEverest(uint256 _everestAmount, address
_userAdd)
        external override
        onlyEverestToken
    {
        // ...
    }
```

LOCATION	Rev-2 - EverestToken.sol -> 451-458

```
    function _updateEverestExtensionsUserEverestOwned(UserEverestInfo
storage user)
        internal
    {
        // Iterate through and update each extension with the user's
everest amount
        for (uint8 extensionIndex = 0; extensionIndex <
everestExtensions.length(); extensionIndex++) {

 BaseEverestExtension(everestExtensions.at(extensionIndex)).updateUse
rEverest(user.everestOwned, user.userAdd);
        }
    }
```

DESCRIPTION	Updating a user's everest in the *ExpeditionV2* contract requires a user interaction from the EverestToken which may neither be possible nor desirable for a user. As *ExpeditionV2* contracts are connected and disconnected from the *EverestToken*, it may cause users' information in the *ExpeditionV2* contracts to be out of sync.
RECOMMENDATION	Ensure that the values between the contracts are properly synchronized. One method could be to add a public function to allow

	users to manually update their everest amounts from the *ExpeditionV2* contract. Another could be to use the ERC20 functionality to stake in the Expedition.
RESOLUTION	Functionality was added to allow for users to manually update their EverestAmount value.

Expeditions Can Be Enabled Or Disabled During A Round

FINDING ID	#0024
SEVERITY	Low Risk
STATUS	Closed
LOCATION	Rev-2 - ExpeditionV2.sol -> 670-695

```
    /// @dev Turn off an expedition
    function disableExpedition()
        public
        onlyOwner
    {
        require(expeditionInfo.live, "Expedition already disabled");
        expeditionInfo.live = false;

        emit ExpeditionDisabled();
    }

    /// @dev Turn on a turned off expedition
    function enableExpedition()
        public
        onlyOwner
    {
        require(!expeditionInfo.live, "Expedition already enabled");
        expeditionInfo.live = true;

        emit ExpeditionEnabled();
    }
```

DESCRIPTION	Expedition rounds can be enabled and disabled during a round. This differs from the Elevation contract which updates its active state only after a rollover.
RECOMMENDATION	Change the Expedition to use a similar mechanism to update its live state after rollover.
RESOLUTION	Project team was stated that this behavior is intentional.

Bonus Timestamp Not Set On Deposit

FINDING ID	#0025
SEVERITY	Low Risk
STATUS	Closed
LOCATION	Rev-4 - Cartographer.sol -> 858-860

```
1         if (tokenLastWithdrawTimestampForBonus[msg.sender][_token] ==
  0) {
2             tokenLastWithdrawTimestampForBonus[msg.sender][_token] ==
  block.timestamp;
3         }
```

DESCRIPTION	The bonus timestamp is not assigned in *Cartographer.deposit()*. Note the use of == as opposed to =.
RECOMMENDATION	Use the correct assignment operator.
RESOLUTION	The recommended changes have been implemented.

Initial Referral BP Is Out Of Bounds

FINDING ID	#0026
SEVERITY	Low Risk
STATUS	Closed
LOCATION	Rev-4 - Cartographer.sol -> 110

```
1     uint256 public referralsSummitBP = 20;
```

LOCATION	Rev-4 - Cartographer.sol -> 287-293

```
1     function setSummitDistributionBPs(uint256 _referralsBP, uint256
  _treasuryBP) public onlyOwner {
2         // Require dev emission less than 25% of total emission
3         require(_treasuryBP  <= 250 && _referralsBP <= 5, "Invalid
  Distributions");
4
5         referralsSummitBP = _referralsBP;
6         treasurySummitBP = _treasuryBP;
7     }
```

DESCRIPTION	The initial value of *referralsSummitBP* is set to 0.2% which is greater than the limit in its setter function.
RECOMMENDATION	Ensure the initial value and the setter's bounds match.
RESOLUTION	The referral limit was increased to 0.4%.

Referral Cycles Not Prevented

FINDING ID	#0027
SEVERITY	Informational
STATUS	Closed
LOCATION	Rev-2 - SummitReferrals.sol -> 34-40

```
1     function createReferral(address referrerAddress) public {
2         require(referrerAddress != msg.sender, "Cant refer
  yourself");
3         require(referrerOf[msg.sender] == address(0), "Already been
  referred");
4         require(referrerOf[referrerAddress] != msg.sender, "No
  reciprocal referrals");
5         referrerOf[msg.sender] = referrerAddress;
6         emit ReferralCreated(referrerAddress, msg.sender);
7     }
```

DESCRIPTION	Self-referrals, reciprocal referrals, and 3-way referrals are prevented. However, four or more addresses may refer to each other in a cycle.
RECOMMENDATION	Disallow referrers from also being referees, in essence, once an account has made a referral they can no longer become someone else's referral. This will not prevent referred accounts from going on to become referrers themselves but it does prevent existing referees from forming a cycle.
RESOLUTION	The referral system was removed.

Identical Contracts

FINDING ID	#0028
SEVERITY	Informational
STATUS	Closed
LOCATION	Rev-2 - BeefyVaultV6NativePassthrough.sol Rev-2 - BeefyVaultV6Passthrough.sol

DESCRIPTION	The noted files are identical except for the contract name.
RECOMMENDATION	Remove one of the contracts and deploy separately.
RESOLUTION	*BeefyVaultV6NativePassthrough.sol* was removed.

Tokens Are Transferred To Burn Address Indirectly (Gas Optimization)

FINDING ID	#0029
SEVERITY	Informational
STATUS	Open
LOCATION	Rev-2 - SummitToken.sol -> 26-27

```
1         oldSummit.safeTransferFrom(msg.sender, address(this),
  _amount);
2         oldSummit.safeTransfer(burnAdd, _amount);
```

DESCRIPTION	The burned amount when swapping from Summit v1 tokens is transferred twice.
RECOMMENDATION	Transfer the burned tokens directly to the burn address. *oldSummit.safeTransferFrom(msg.sender, burnAdd, _amount);*
RESOLUTION	Project team comment: “The gas increase is negligible and user approves the new Token address instead of the burn address”

Totem Supplies Always Returns 10 Values

FINDING ID	#0030
SEVERITY	Informational
STATUS	Closed
LOCATION	Rev-2 - CartographerElevation.sol -> 288-302

```
function totemSupplies(address _token) public view poolExists(_token)
returns (uint256[10] memory) {
        ElevationPoolInfo storage pool = poolInfo[_token];
        return [
            elevation >= 1 ? pool.totemSupplies[0] : 0,
            elevation >= 1 ? pool.totemSupplies[1] : 0,
            elevation >= 2 ? pool.totemSupplies[2] : 0,
            elevation >= 2 ? pool.totemSupplies[3] : 0,
            elevation >= 2 ? pool.totemSupplies[4] : 0,
            elevation >= 3 ? pool.totemSupplies[5] : 0,
            elevation >= 3 ? pool.totemSupplies[6] : 0,
            elevation >= 3 ? pool.totemSupplies[7] : 0,
            elevation >= 3 ? pool.totemSupplies[8] : 0,
            elevation >= 3 ? pool.totemSupplies[9] : 0
        ];
    }
```

LOCATION	Rev-2 - CartographerElevation.sol -> 319-354

```
    function totemRoundRewards(address _token)
        public view
        poolExists(_token)
        returns (uint256[11] memory)
    {
        // ...
        uint256[11] memory finalTotemRewards;
        // ...
        return finalTotemRewards;
    }
```

LOCATION	Rev-2 - ElevationHelper.sol -> 422-452

```
    function historicalWinningTotems(uint8 _elevation) public view
allElevations(_elevation) returns (uint256[20] memory) {

        // Early escape OASIS winners, as they don't exist
        if (_elevation == OASIS) {
            return [uint256(0), 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0];
        }

        uint256 round = roundNumber[_elevation];
        return [
            // ...
        ];
    }
```

DESCRIPTION	The noted functions return a fixed element array which may contain unnecessary blank entries.
RECOMMENDATION	Return a dynamic array with the exact number of elements.
RESOLUTION	The recommended changes have been implemented.

Redundant Check For Active Pool (Gas Optimization)

FINDING ID	#0031
SEVERITY	Informational
STATUS	Closed
LOCATION	Rev-2 - CartographerElevation.sol -> 364-377

```
1     function _markPoolActive(ElevationPoolInfo storage pool, bool
  _active)
2         internal
3     {
4         if (pool.active == _active) return;
5
6         // ...
7     }
```

LOCATION	Rev-2 - CartographerElevation.sol -> 430

```
1             if (_live && !pool.active) _markPoolActive(pool, true);
```

LOCATION	Rev-2 - CartographerElevation.sol -> 679

```
1         if (!pool.live && pool.active) _markPoolActive(pool, false);
```

DESCRIPTION	The *active* flag is checked within *_markPoolActive()* and does not need to be checked again before calling it.
RECOMMENDATION	This would also clarify the relationship between *pool.active* and *pool.live*
RESOLUTION	The recommended changes have been implemented.

Redundant Check When Setting Summit Per Second (Gas Optimization)

FINDING ID	#0032
SEVERITY	Informational
STATUS	Closed
LOCATION	Rev-4 - Cartographer.sol -> 277-282

```
1  function setTotalSummitPerSecond(uint256 _amount) public onlyOwner {
2          // Must be less than 1 SUMMIT per second
3          require(_amount >= 0 && _amount < 1e18, "Invalid emission");
4
5          summitPerSecond = _amount;
6      }
```

DESCRIPTION	The value of *_amount* can't be less than 0 since it is an unsigned integer.
RECOMMENDATION	Remove the redundant check.
RESOLUTION	The recommended changes have been implemented.

Hypothetical Rewards Calculation Does Not Match Rollover

FINDING ID	#0033
SEVERITY	Informational
STATUS	Closed
LOCATION	Rev-2 - CartographerElevation.sol -> 523-554

```
function hypotheticalRewards(address _token, address _userAdd)
    public view
    poolExists(_token) validUserAdd(_userAdd)
    returns (uint256, uint256)
{
    // ...
}
```

LOCATION	Rev-2 - CartographerElevation.sol -> 604-625

```
function rollover()
    external override
    onlyCartographer
{
    // ...
}
```

DESCRIPTION	The winnings calculated by *hypotheticalRewards()* takes into account: - the user's stake - the user's totem's stake - the total stake in the pool However, the winnings calculated by *rollover()* use the average winning multiplier across all pools.
RECOMMENDATION	Because this is a UI function, this does not present a security risk. However, the mismatch in calculations may give users the wrong impression.
RESOLUTION	Functionality changed to use correct formulas.

Error Message Doesn't Reflect Check

FINDING ID	#0034
SEVERITY	Informational
STATUS	Closed
LOCATION	Rev4 - ExpeditionV2.sol -> 337

```
function setExpeditionDeityWinningsMult(uint256 _deityMult) public
onlyOwner {
        require(_deityMult >= 100 && _deityMult <= 500, "Invalid
runway rounds (7-90)");
        //...
}
function setExpeditionRunwayRounds(uint256 _runwayRounds) public
onlyOwner {
        require(_runwayRounds >= 7 && _runwayRounds <= 90, "Invalid
runway rounds (7-90)");
        //...
}
```

LOCATION	Rev4 - Cartographer.sol -> 1188-1195

```
    function setBaseMinimumWithdrawalTax(uint16
_baseMinimumWithdrawalTax)
        public
        onlyOwner
    {
        require(_baseMinimumWithdrawalTax <= 100, "Minimum tax
outside 0%-10%");
        baseMinimumWithdrawalTax = _baseMinimumWithdrawalTax;
        emit SetBaseMinimumWithdrawalTax(_baseMinimumWithdrawalTax);
    }
```

DESCRIPTION	The error message does not match captured error at the noted locations.
RECOMMENDATION	Update error handling to match. Note: It might be easier to read for the check (100 <= example && example <= 500)

RESOLUTION	The recommended changes have been implemented.

Resetting Deposit Timestamp For Tax Can Be Avoided

FINDING ID	#0035
SEVERITY	Informational
STATUS	Closed
LOCATION	Rev-4 - Cartographer.sol -> 863-865

```
1         if (_amount > (_userTokenStakedAmount(_token, msg.sender) *
  taxResetOnDepositBP / 10000)) {
2             tokenLastDepositTimestampForTax[msg.sender][_token] =
  block.timestamp;
3         }
```

DESCRIPTION	The withdrawal tax is reset whenever a user deposits an amount greater than a given threshold based on their current stake. This can be avoided by depositing in multiple smaller steps.
RECOMMENDATION	Confirm whether this behavior is intentional or changes the threshold mechanism.
RESOLUTION	Project team has stated this is intentional. Project team comment: “This behavior is intentional. We want to prevent users from depositing a tiny amount of token to allow the withdrawal tax to decrease, and only then depositing a large stack of funds. Though this can be skirted easily, it is much harder to do so in the specific circumstances we are trying to avoid.”

Setter Does Not Change Intended Variable

FINDING ID	#0042
SEVERITY	Low Risk
STATUS	Closed
LOCATION	Rev-5 - EverestToken.sol -> 145-149

```
1    function setInflectionLockTime(uint256 _lockTimeDays) public
  onlyOwner {
2        require(_lockTimeDays >= minLockTime && _lockTimeDays <=
  maxLockTime && _lockTimeDays >= 7 && _lockTimeDays <= 365, "Invalid
  inflection lock time (7-365 days)");
3        minLockTime = _lockTimeDays * daySeconds;
4        emit SetInflectionLockTime(_lockTimeDays);
5    }
```

DESCRIPTION	The inflection lock time is not modified in the function *setInflectionLockTime()*.
RECOMMENDATION	Ensure that the correct variable is changed in the setter.
RESOLUTION	The recommended changes have been implemented.

Linear Scaling Incorrect When Lower Bound Value Exceeds Upper Bound Value

FINDING ID	#0043
SEVERITY	Low Risk
STATUS	Closed
LOCATION	Rev-5 - libs/SummitMath.sol -> 7-19

```
1     function scaledValue(uint256 scalar, uint256 minBound, uint256
  maxBound, uint256 minResult, uint256 maxResult)
2         internal pure
3         returns (uint256)
4     {
5         require(minBound <= maxBound, "Invalid scaling range");
6         if (minResult == maxResult) return minResult;
7         if (scalar <= minBound) return minResult;
8         if (scalar >= maxBound) return maxResult;
9         if (maxResult > minResult) {
10            return (((scalar - minBound) * (maxResult - minResult) *
  1e12) / (maxBound - minBound) / 1e12) + minResult;
11        }
12        return (((maxBound - scalar) * (minResult - maxResult) *
  1e12) / (maxBound - minBound) / 1e12);
13    }
```

DESCRIPTION	Only the linearly scaling amount is included in the case of *maxResult* being less than *minResult*.
RECOMMENDATION	Include the non-scaling in the result, in this case, *maxResult*.
RESOLUTION	The recommended changes have been implemented.

Supplies Updated Incorrectly When Changing Both Deity and Safety Factor

FINDING ID	#0044
SEVERITY	Medium Risk
STATUS	Open
LOCATION	Rev-9 - ExpeditionV2.sol -> 833-842

```
        if (user.entered) {
            // Transfer deitied everest from previous deity to new deity
            expeditionInfo.supplies.deity[prevDeity] -= user.deitiedSupply;
            expeditionInfo.supplies.deity[_newDeity] += user.deitiedSupply;

            // Remove safe and deitied everest from existing supply states
            expeditionInfo.supplies.safe = expeditionInfo.supplies.safe - existingSafeSupply + user.safeSupply;
            expeditionInfo.supplies.deitied = expeditionInfo.supplies.deitied - existingDeitiedSupply + user.deitiedSupply;
            expeditionInfo.supplies.deity[user.deity] = expeditionInfo.supplies.deity[user.deity] - existingDeitiedSupply + user.deitiedSupply;
        }
```

DESCRIPTION	The values of *expeditionInfo.supplies.deity[]* are updated incorrectly when changing both deity and safety factors at the same time. This can result in other users being unable to select their deity, change their safety factor, and potentially even update their Everest amounts.
RECOMMENDATION	The correct changes should be: • *deity[prevDeity] -= existingDeitiedSupply;* • *deity[_newDeity] += user.deitiedSupply;*
RESOLUTION	The recommended changes have been implemented.

Static Analysis

Contract Values Can Be Constant Or Immutable (Gas Optimization)

FINDING ID	#0036
SEVERITY	Informational
STATUS	Closed
LOCATION	• Rev-2 - SummitLocking.sol -> 22: *uint256 public epochDuration = 3600 * 24 * 7;* • Rev-2 - SummitVRFModule.sol -> 56: *uint256 seedRoundDurationMult = 2;* • Rev-4 - Cartographer.sol -> 136: *uint256 public taxResetOnDepositBP = 500;* • Rev-5 - EverestToken.sol -> 29: *uint256 public daySeconds = 24 * 3600;*

DESCRIPTION	Variables that 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	All values have either been deprecated or changed to a constant.

Unused Library

FINDING ID	#0037
SEVERITY	Informational
STATUS	Closed
LOCATION	Rev-2 - libs/UQ112x112.sol

DESCRIPTION	The UQ112x112 library is never used.
RECOMMENDATION	Remove the library.
RESOLUTION	The library was removed.

No Events Emitted For Changes To Protocol Values

FINDING ID	#0038
SEVERITY	Informational
STATUS	Closed
LOCATION	• Rev-2 - ElevationHelper.sol -> 266-271: *function setSummitVRFModuleAdd (address _summitVRFModuleAdd) public onlyOwner* • Rev-2 - ElevationHelper.sol -> 271-281: *function setElevationRoundDurationMult(uint8 _elevation, uint8 _mult) public onlyOwner elevationOrExpedition(_elevation)* • Rev-2 - ElevationHelper.sol -> 285-294: *function setElevationAllocMultiplier(uint8 _elevation, uint8 _allocMultiplier) public onlyOwner allElevations(_elevation)* • Rev-2 - SummitVRFModule.sol -> 99-104: *function setElevationHelper (address _elevationHelper) public onlyOwner* • Rev-2 - SummitVRFModule.sol -> 109-112: *function setTrustedSeederAdd(address _trustedSeeder) public override onlyCartographer* • Rev-2 - SummitVRFModule.sol -> 117-119: *function setSeedRoundEndTimestamp(uint256 _seedRoundEndTimestamp) public override onlyElevationHelper* • Rev-4 - Cartographer.sol -> 270-273: *function setRolloverRewardInNativeToken(uint256 _reward) public onlyOwner* • Rev-4 - Cartographer.sol -> 277-282: *function setTotalSummitPerSecond(uint256 _amount) public onlyOwner* • Rev-4 - Cartographer.sol -> 287-293: *function setSummitDistributionBPs(uint256 _referralsBP, uint256 _treasuryBP) public onlyOwner* • Rev-4 - SummitLocking.sol -> 102-107: *function setYieldLockEpochCount(uint8 _count)* • Rev-5 - Cartographer.sol -> 280-283: *function setRolloverRewardInNativeToken(uint256 _reward) public 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	Recommended changes have been implemented.

Unused Variables

FINDING ID	#0039
SEVERITY	Informational
STATUS	Closed
LOCATION	• Rev-2 - CartographerOasis.sol -> 77: *EnumerableSet.AddressSet private activePools;* • Rev-2 - CartographerElevation.sol -> 104: *bool public elevationEnabled;*

DESCRIPTION	The noted variables are not used.
RECOMMENDATION	Remove the variables or incorporate them into the contract functionality.
RESOLUTION	The variables were removed.

Unused Functions And Modifiers

FINDING ID	#0040
SEVERITY	Informational
STATUS	Closed
LOCATION	• Rev-4 BaseEverestExtension.sol -> 16-21: *function getUserEverest(address _userAdd) internal view returns (uint256)* • Rev-4 CartographerElevation.sol -> 739-748: *function totemPrecomputedMultForRound(ElevationPoolInfo storage pool, uint8 _totem, uint256 _roundIndex) internal view returns (uint256)* • Rev-4 EverestToken.sol -> 114-117: *modifier validUserAdd(address _userAdd)* • Rev-4 ExpeditionV2.sol -> 274-277: *modifier elevationHelperRoundRolledOver()*

DESCRIPTION	The noted functions and modifiers are never used.
RECOMMENDATION	Remove the unused functions and modifiers or incorporate them into the contract functionality.
RESOLUTION	The recommended changes have been implemented.

Missing Zero Checks

FINDING ID	#0041
SEVERITY	Low Risk
STATUS	Closed
LOCATION	• Rev-4 contracts/Cartographer.sol -> 180-186: *constructor(address _treasuryAdd, address _expeditionTreasuryAdd)*

DESCRIPTION	The contract address values can be set to zero address in various constructors, initializers, and setter functions. Zero addresses may cause incorrect contract behavior.
RECOMMENDATION	Add a check to ensure contract values are never set to invalid zero addresses.
RESOLUTION	The recommended changes have been implemented.

On-Chain Analysis

Treasuries And LP Generation Are Externally Owned Accounts

FINDING ID	#0045
SEVERITY	Medium Risk
STATUS	Open
LOCATION	Cartographer 0x71210E72D065C19406913cD706e964A9f21856D4

DESCRIPTION	The treasury addresses, as well as the lp generator address, are externally owned accounts. A malicious actor in control of these accounts can take funds intended for project purposes. expeditionTreasuryAdd 0x00676eF184C36EBf73d0F3059D2a6909F02AA893 lpGeneratorAdd 0x50963e3c0899584a3EBD3226d976210aF0e42349 treasuryAdd 0x474332025Dd20D5F09FFd766b317F98A872D71e0
RECOMMENDATION	Set these addresses to an appropriate contract. A multi-sig wallet can work as well.
RESOLUTION	Project team comment: “The treasuries are currently managed by the Summit Team, as we have been managing them for the duration of V1. As we refine our strategies for the treasuries we will develop contracts that handle these treasuries trustlessly.”

Missing Signature Specific Delay

FINDING ID	#0046
SEVERITY	Low Risk
STATUS	Mitigated
LOCATION	Timelock 0x191528B779Ada279145D42350226bF75c0c73715

DESCRIPTION	The following functions are sensitive and should have a longer time delay - *setExpeditionTreasuryAdd(address)* - *setLpGeneratorAdd(address)* - *setElevationHelper(address)* Note: the *expeditionTreasuryAdd* and *lpGeneratorAdd* are currently EOAs. The following specific delays were found without the full function signature: - *transferOwnership* - 7 days - *renounceOwnership* - 7 days - *setPendingAdmin* - 7 days - *migrateSummitOwnership* - 7 days - *setTotalSummitPerSecond* - 3 days - *setTokenPassthroughStrategy* - 3 days - *retireTokenPassthroughStrategy* - 3 days - *upgradeSummitRNGModule* - 3 days - *setYieldLockEpochCount* - 3 days
RECOMMENDATION	Add a new specific time delay for the functions noted. Obelisk recommends a delay of at least 72 hours.
RESOLUTION	The correct signature-specific delays were added. 7 day delays: ● *migrateSummitOwnership(address)* ● *renounceOwnership()* ● *setPendingAdmin(address)* ● *transferOwnership(address)* 3 day delays ● *retireTokenPassthroughStrategy(address)* ● *setElevationHelper(address)* ● *setExpeditionTreasuryAdd(address)*

- *setLpGeneratorAdd(address)*
- *setTokenPassthroughStrategy(address,address)*
- *setTotalSummitPerSecond(uint256)*
- *setYieldLockEpochCount(uint8)*
- *upgradeSummitRNGModule(address)*

Changes To Deployed Contract

FINDING ID	#0047
SEVERITY	Informational
STATUS	Closed
LOCATION	ElevationHelper 0xDfBb673787DfC1477b0ca7890887136a13296811

DESCRIPTION	The expedition round duration was changed from 6 hours to 24 hours. The delays before the rounds at each elevation begin were also changed.
RECOMMENDATION	No changes are necessary.
RESOLUTION	N/A

External Addresses

Externally Owned Accounts

Admin

ACCOUNT	0x3a7679E3662bC7c2EB2B1E71FA221dA430c6f64B
USAGE	0x191528B779Ada279145D42350226bF75c0c73715 *Timelock.admin* - Variable 0x71210E72D065C19406913cD706e964A9f21856D4 *Cartographer.getRoleMember(PAUSER_ROLE,0)* 0xC687806Cfd11B5330d7c3aE6f18B18DC71e1083e *EverestToken.getRoleMember(PAUSER_ROLE,0)* 0x94233b479B37FBb41E81C63E27b6C2279646C609 *ExpeditionV2.getRoleMember(PAUSER_ROLE,0)* 0x17EB377C16653523DFAa8402de8A8eEe4832c108 *SummitGlacier.getRoleMember(PAUSER_ROLE,0)* 0x0dDB88e14494546D07fCd94c3f0ef6D3296B1cD7 *SummitTokenV2.getRoleMember(PAUSER_ROLE,0)*
IMPACT	• receives elevated permissions as owner, operator, or other

Expedition Treasury

ACCOUNT	0x00676eF184C36EBf73d0F3059D2a6909F02AA893
USAGE	0x71210E72D065C19406913cD706e964A9f21856D4 *Cartographer.expeditionTreasuryAdd* - Variable
IMPACT	• receives transfer of tokens farmed by project

LP Generator

ACCOUNT	0x50963e3c0899584a3EBD3226d976210aF0e42349
USAGE	0x71210E72D065C19406913cD706e964A9f21856D4 *Cartographer.lpGeneratorAdd* - Variable
IMPACT	• receives transfer of tokens farmed by project

Treasury

ACCOUNT	0x474332025Dd20D5F09FFd766b317F98A872D71e0
USAGE	0x71210E72D065C19406913cD706e964A9f21856D4 *Cartographer.treasuryAdd* - Variable
IMPACT	• receives transfer of tokens farmed by project

Trusted Seeder

ACCOUNT	0x7E1e4354de68B644c30b40F983f66aF60042fF69
USAGE	0xFCcf00CCdeb8964c0F996f332F5A0f763d571d27 *SummitTrustedSeederRNGModule.trustedSeeder* - Variable
IMPACT	• receives elevated permissions as owner, operator, or other

External Contracts

These contracts are not part of the audit scope.

Deposit Tokens

ADDRESS	SpookyLP WFTM TOMB 0x2A651563C9d3Af67aE0388a5c8F89b867038089e SpookyLP WFTM TSHARE 0x4733bc45eF91cF7CcEcaeeDb794727075fB209F2 SpookyLP WFTM BOO 0xEc7178F4C41f346b2721907F5cF7628E388A7a58 WeightedPool2Tokens BPT-BEETS-FTM 0xcdE5a11a4ACB4eE4c805352Cec57E236bdBC3837 WeightedPool GRAND-ORCH 0xd47D2791d3B46f9452709Fa41855a045304D6f9d WeightedPool2Tokens FTM-OPERA 0xcdF68a4d525Ba2E90Fe959c74330430A5a6b8226 SpookyToken BOO 0x841FAD6EAe12c286d1Fd18d1d525DFfA75C7EFFE WeightedPool BPT-L1TOKEN 0x9af1F0e9aC9C844A4a4439d446c1437807183075 SpookyLP WFTM 2SHARE 0x6398ACBBAB2561553a9e458Ab67dCFbD58944e52 SpookyLP WFTM 2OMB 0xbdC7DFb7B88183e87f003ca6B5a2F81202343478
USAGE	0x71210E72D065C19406913cD706e964A9f21856D4 *Cartographer.tokensWithAllocation* - Variable 0x8047C5Bed363FE1bf458eC3E20E93A3c28A07b8d *CartographerOasis.getPools* - Variable 0x1805922e7F82fc9DbAd8E2435C146ba605C4a25d Plains 0x64F8a1DBC20f132159605Ad8d7111e75EA702358 Mesa 0x93af6a3882aAF4112Fc404E30277b39452F44cf6 Summit *CartographerElevation.getPools* - Variable
IMPACT	• ERC20 Token

SummitV1

ADDRESS	0x8F9bCCB6Dd999148Da1808aC290F2274b13D7994
USAGE	0x0dDB88e14494546D07fCd94c3f0ef6D3296B1cD7 *SummitTokenV2.oldSummit* - Initialized
IMPACT	• ERC20 Token

Beefy Vaults

ADDRESS	BeefyVaultV6 - SpookyLP WFTM TOMB 0x27c77411074ba90cA35e6f92A79dAd577c05A746 BeefyVaultV6 - SpookyLP WFTM TSHARE 0xae94e96bF81b3a43027918b138B71a771D381150 BeefyVaultV6 - SpookyLP WFTM BOO 0xEe3a7c885Fd3cc5358FF583F2DAB3b8bC473316f BeefyVaultV6 - WeightedPool2Tokens BPT-BEETS-FTM 0xAe0AB718971bb2BAd88AE6Bdc4D0eA63F3CD53Ee BeefyVaultV6 - WeightedPool GRAND-ORCH 0x0ab24Bfc2503bB536ad667c00685BBB70fA90433 BeefyVaultV6 - WeightedPool2Tokens FTM-OPERA 0xB40c339e2b0a8513152F68082D3c87314E03776D BeefyVaultV6 - SpookyToken BOO 0x15DD4398721733D8273FD4Ed9ac5eadC6c018866 BeefyVaultV6 - WeightedPool BPT-L1TOKEN 0x0139C853539bF1EDf221cf9d665F282C2701335a BeefyVaultV6 - SpookyLP WFTM 2SHARE 0x03668Bd5dc63B1e15c39619b599091A4f68cAFB3 BeefyVaultV6 - SpookyLP WFTM 2OMB 0xf3A72885cB383543AEE60f44Ca51C760f0bC3b9b
USAGE	BeefyVaultV6Passthrough - SpookyLP WFTM TOMB 0x3B4C96337f62EE0Bc70a30F523b15de15dBAEF9E BeefyVaultV6Passthrough - SpookyLP WFTM TSHARE 0xCccfd006145dDDfC7Bb431329D3620a4d601936b BeefyVaultV6Passthrough - SpookyLP WFTM BOO 0xf0D80E3E76Acc03E37976673aF6730866afF0305 BeefyVaultV6Passthrough - WeightedPool2Tokens BPT-BEETS-FTM 0x8D82120DD86a54AfC4b8cf749C46c2a46717Ce90 BeefyVaultV6Passthrough - WeightedPool GRAND-ORCH 0x4776Bc42C56B8d5c53B5EBC4D306CDFD480c926f BeefyVaultV6Passthrough - WeightedPool2Tokens FTM-OPERA 0x79A54A9502aEf31768e2a0702ca8eA8Ea7AEa20C BeefyVaultV6Passthrough - SpookyToken BOO 0x5722b6F09848354E2A7539FE509eEe708155658b

	BeefyVaultV6Passthrough - WeightedPool BPT-L1TOKEN 0xde223f1dE2a62345d48a76250E767aCfAa3792B3 BeefyVaultV6Passthrough - SpookyLP WFTM 2SHARE 0xa9db0459648e1754AB4811815f771a99e2d92392 BeefyVaultV6Passthrough - SpookyLP WFTM 2OMB 0xF136B1aE80362DEd39d7cf4A6172E3F67260b276 *BeefyVaultV6Passthrough.passthroughToken* - Initialized
IMPACT	• receives transfer of tokens deposited by users • impacts ability to deposit or withdraw tokens

Appendix A - Reviewed Documents

Document	Address
interfaces/IPancakeFactory.sol	N/A
interfaces/IPancakeRouter.sol	N/A
interfaces/IPassthrough.sol	N/A
interfaces/ISubCart.sol	N/A
interfaces/ISummitRNGModule.sol	N/A
interfaces/IUniswapV2Pair.sol	N/A
interfaces/PancakeFactory.sol	N/A
libs/ERC20Mintable.sol	N/A
libs/Multicall.sol	N/A
libs/SummitMath.sol	N/A
libs/UQ112x112.sol	N/A
BaseEverestExtension.sol	N/A
BeefyVaultV2Passthrough.sol	N/A
BeefyVaultV6NativePassthrough.sol	N/A
BeefyVaultV6Passthrough.sol	SpookyLP WFTM TOMB 0x3B4C96337f62EE0Bc70a30F523b15de15dBAEF9E SpookyLP WFTM TSHARE 0xCccfd006145dDDfC7Bb431329D3620a4d601936b SpookyLP WFTM BOO 0xf0D80E3E76Acc03E37976673aF6730866afF0305 WeightedPool2Tokens BPT-BEETS-FTM 0x8D82120DD86a54AfC4b8cf749C46c2a46717Ce90 WeightedPool GRAND-ORCH 0x4776Bc42C56B8d5c53B5EBC4D306CDFD480c926f

	WeightedPool2Tokens FTM-OPERA 0x79A54A9502aEf31768e2a0702ca8eA8Ea7AEa20C SpookyToken BOO 0x5722b6F09848354E2A7539FE509eEe708155658b WeightedPool BPT-L1TOKEN 0xde223f1dE2a62345d48a76250E767aCfAa3792B3 SpookyLP WFTM 2SHARE 0xa9db0459648e1754AB4811815f771a99e2d92392 SpookyLP WFTM 2OMB 0xF136B1aE80362DEd39d7cf4A6172E3F67260b276
Cartographer.sol	0x71210E72D065C19406913cD706e964A9f21856D4
CartographerElevation.sol	Plains 0x1805922e7F82fc9DbAd8E2435C146ba605C4a25d Mesa 0x64F8a1DBC20f132159605Ad8d7111e75EA702358 Summit 0x93af6a3882aAF4112Fc404E30277b39452F44cf6
CartographerOasis.sol	0x8047C5Bed363FE1bf458eC3E20E93A3c28A07b8d
ElevationHelper.sol	0xDfBb673787DfC1477b0ca7890887136a13296811
EverestToken.sol	0xC687806Cfd11B5330d7c3aE6f18B18DC71e1083e
ExpeditionV2.sol	0x94233b479B37FBb41E81C63E27b6C2279646C609
MasterChefPassthrough.sol	N/A
SummitGlacier.sol	0x17EB377C16653523DFAa8402de8A8eEe4832c108
SummitReferrals.sol	N/A
SummitToken.sol	0x0dDB88e14494546D07fCd94c3f0ef6D3296B1cD7
SummitTrustedSeederRNGModule.sol	0xFCcf00CCdeb8964c0F996f332F5A0f763d571d27
Timelock.sol	0x191528B779Ada279145D42350226bF75c0c73715

Revisions

Revision 1	19a1b63e4348d84593534ffdcdbf7f0e764f90d0
Revision 2	71710fd897fd2728785d115dd6913509776eeb18
Revision 3	d05bd64f6dc6f784bd3f0530557dc6fe7db8b983

Revision 4	14f80a616eb9394cab289c06a95802823049dab0
Revision 5	b526774a1e7104f189c5b4259609eb5776901d3c
Revision 6	4af984faa5c0f15252dcad3c77f2564910071c90
Revision 7	301aed1111a4758b665d5d03d293cdda55da43db
Revision 8	9e221b0a455c85fe8c0fea011936f717b5dbc982
Revision 9	963423dc28131412dc830cb8eb04c81bd0f07f1e
Revision 10	c2c32fbd166058a45a44399f67e817cd71883f22

Imported Contracts

OpenZeppelin	4.3.0

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 blockchain. 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