Contract Summary and Internal Security Audit Report on Octant V2

(12/23/2024)

BaseStrategy.sol

High-Level Overview

BaseStrategy is an abstract contract that serves as the foundation for implementing yield-generating strategies in a tokenized format. It provides the core infrastructure for strategies to interact with various DeFi protocols while maintaining standardized accounting and security controls.

The contract follows a delegate-call proxy pattern where core functionality is delegated to a TokenizedStrategy implementation contract. This architecture allows for standardized accounting and share calculations while enabling strategy-specific yield generation logic to be implemented separately.

Functionality Breakdown

Major Feature 1: Delegated Strategy Management
- Uses delegate calls to a TokenizedStrategy implementation for core accounting
- Maintains separation between strategy-specific logic and standardized operations
- Implements role-based access control through modifiers (management, keepers, emergency authorized)

Major Feature 2: Yield Generation Framework
- Provides abstract functions that must be implemented by specific strategies
- Handles deployment of funds (_deployFunds), withdrawal (_freeFunds), and harvesting (_harvestAndReport)
- Optional tend mechanism for frequent position management without full harvests

Major Feature 3: Safety Controls
- Emergency withdrawal capabilities
- Deposit and withdrawal limits
- Role-based access control
- Self-check mechanisms through onlySelf modifier

Contract Summary

The contract provides the following main functions:

- function liquidatePosition(uint256 _amountNeeded) external returns (uint256 _liquidatedAmount, uint256 _loss)
- function adjustPosition(uint256 _debtOutstanding) external
- function deployFunds(uint256 _amount) external payable
- function freeFunds(uint256 _amount) external
- function harvestAndReport() external returns (uint256)
- function tendThis(uint256 _totalIdle) external
- function shutdownWithdraw(uint256 _amount) external

Security Analysis

Storage Layout

- tokenizedStrategyImplementation: Address of core implementation
- asset: Underlying ERC20 token
- maxReportDelay: Maximum time between harvests
- TokenizedStrategy: Interface for internal calls

Constants

- ETH: 0xEeeeeEeeeEeEeeEeEeEeeEEEeeeeEeeeeeeeEEeE - Used to represent native ETH
- Implementation slot: 0x360894a13ba1a3210667c828492db98dca3e2076cc3735a920a3ca505d382bbc - EIP-1967 implementation slot

Possible Attack Vectors

Potential Risks

Potential Manipulations

Security Analysis

TokenizedStrategy.sol

High-Level Overview

TokenizedStrategy is a foundational contract that implements ERC4626-like tokenized vault functionality specifically designed for yield-generating strategies. It provides a standardized interface for depositing assets, managing strategy operations, and handling profit/loss reporting while maintaining proper accounting of shares and underlying assets.

The contract uses a unique storage pattern with a custom storage slot to prevent collisions and allow for flexible strategy implementation. It implements comprehensive access control with multiple privileged roles (management, keeper, emergencyAdmin) and includes safety features like emergency shutdown capabilities.

Functionality Breakdown

Major Feature 1: Asset Management

- Handles deposits and withdrawals of underlying assets
- Maintains accounting of shares and total assets
- Implements conversion between shares and assets with proper rounding

- Supports both ERC20 tokens and native ETH

Major Feature 2: Access Control

- Multi-role system with management, keeper, and emergency admin
- Two-step management transfer process for security
- Role-specific permissions for different operations
- Emergency shutdown capabilities

Major Feature 3: Profit Handling

- Standardized profit reporting mechanism
- Keeper-controlled report function for profit/loss recognition
- Support for tending operations
- Protection against PPS manipulation

Major Feature 3: Safety Features

- Emergency shutdown mechanism
- Non-reentrant function protection
- Withdrawal limits and loss parameters
- Comprehensive event emission for transparency

Contract Summary

The contract provides the following main functions:

- function deposit(uint256 assets, address receiver) external payable returns (uint256 shares)
- function mint(uint256 shares, address receiver) external payable returns (uint256 assets)
- function withdraw(uint256 assets, address receiver, address owner) external returns (uint256 shares)
- function withdraw(uint256 assets, address receiver, address owner, uint256 maxLoss) public returns (uint256 shares)
- function redeem(uint256 shares, address receiver, address owner) external returns (uint256)
- function redeem(uint256 shares, address receiver, address owner, uint256 maxLoss) public returns (uint256)
- function shutdownStrategy() external
- function emergencyWithdraw(uint256 amount) external
- function transfer(address to, uint256 amount) external returns (bool)

- function approve(address spender, uint256 amount) external returns (bool)
- function transferFrom(address from, address to, uint256 amount) external returns (bool)
- function permit(address owner, address spender, uint256 value, uint256 deadline, uint8 v, bytes32 r, bytes32 s) external
- function report() external returns (uint256 profit, uint256 loss)
- function tend() external
- function setPendingManagement(address _management) external
- function acceptManagement() external
- function setKeeper(address _keeper) external
- function setEmergencyAdmin(address _emergencyAdmin) external
- function setName(string calldata _name) external

Security Analysis

Storage Layout

- asset: Underlying ERC20 token
- totalAssets: Total assets managed by strategy
- totalSupply: Total shares issued
- management, keeper, emergencyAdmin: Access control addresses
- shutdown: Emergency shutdown flag

Constants

- MAX_BPS = 10_000: Base points for percentage calculations
- MAX_BPS_EXTENDED = 1_000_000_000_000: Extended precision for calculations
- SECONDS_PER_YEAR = 31_556_952: Standard year in seconds
- ENTERED = 2, NOT_ENTERED = 1: Reentrancy guard flags

Possible Attack Vectors

Security Analysis

DragonTokenizedStrategy.sol

High-Level Overview

DragonTokenizedStrategy extends TokenizedStrategy to implement a specialized vault with lockup mechanics and rage quit functionality. The contract introduces voluntary lockup periods for depositors, with a minimum lockup duration of 90 days, and allows for a controlled exit through a rage quit mechanism that enables gradual withdrawals over time.

The key innovation is the balance between commitment (through lockups) and flexibility (through rage quit), while maintaining the core yield-generating functionality of the base strategy.

Functionality Breakdown

Major Feature 1: Lockup Management
- Enforces minimum 3-month lockup period
- Supports extending existing lockups
- Tracks locked shares per user
- Prevents unauthorized early withdrawals

Major Feature 2: Rage Quit System
- Allows users to exit positions gradually
- Converts full lockup to 3-month linear unlock
- Proportional share release over time
- Protection against immediate mass withdrawals

Major Feature 3: Deposit Variations
- Standard deposits (auto-locks all shares)

- Deposits with custom lockup periods
- Mint operations with lockup options
- Protection against deposits during rage quit

Major Feature 3: Withdrawal Controls

- Enforces lockup restrictions
- Handles rage quit unlocking schedule
- Calculates maximum withdrawable amounts
- Prevents unauthorized transfers

Contract Summary

The contract provides the following main functions:

- function initialize(address _asset, string memory _name, address _owner, address _management, address _keeper, address _dragonRouter) external
- function deposit(uint256 assets, address receiver) external payable returns (uint256 shares)
- function mint(uint256 shares, address receiver) external payable returns (uint256 assets)
- function withdraw(uint256 assets, address receiver, address owner, uint256 maxLoss) public returns (uint256 shares)
- function redeem(uint256 shares, address receiver, address owner, uint256 maxLoss) public returns (uint256)
- function depositWithLockup(uint256 assets, address receiver, uint256 lockupDuration) public returns (uint256 shares)
- function mintWithLockup(uint256 shares, address receiver, uint256 lockupDuration) public returns (uint256 assets)
- function initiateRageQuit() external
- function report() external returns (uint256 profit, uint256 loss)

Security Analysis

Storage Layout

- Inherits TokenizedStrategy storage pattern and adds:

Constants

- MINIMUM_LOCKUP_DURATION = 90 days: Minimum required lockup period
- Used for both voluntary lockups and rage quit duration
- Chosen to ensure sufficient commitment while maintaining reasonable liquidity
- Critical for preventing rapid capital flight

Possible Attack Vectors

Security Analysis