CONTRACT WOLF

Blockchain Security - Smart Contract Audits

Security Assessment

April 8, 2022

Disclaimer

ContractWolf.io audits and reports should not be considered as a form of project's "advertisement" and does not cover any interaction and assessment from "project's contract" to "external contracts" such as Pancakeswap or similar.

ContractWolf does not provide any warranty on its released reports.

ContractWolf should not be used as a decision to invest into an audited project and is not affiliated nor partners to its audited contract projects.

ContractWolf provides transparent report to all its "clients" and to its "clients participants" and will not claim any guarantee of bug-free code within its **SMART CONTRACT**.

ContractWolf presence is to analyze, audit and assess the client's smart contract's code.

Each company or projects should be liable to its security flaws and functionalities.

Network

Avalanche Chain

Website

https://northpole.money

Twitter

https://twitter.com/NorthPole_money

Discord

https://discord.com/invite/PRye7uMxMD

Medium

https://northpolemoney.medium.com

GitBook

https://docs.northpole.money

Description

Northpole is a lending platform that uses interest-bearing tokens (ibTKNs) as collateral to borrow a USD pegged stablecoin (POLE), that can be used as any other traditional stablecoin.

ContractWolf Engagement

8th of April 2022, **Northpole** engaged and agrees to audit their smart contract's code by ContractWolf. The goal of this engagement was to identify if there is a possibility of security flaws in the implementation of the contract or system.

ContractWolf will be focusing on contract issues and functionalities along with the projects claims from smart contract to their website, whitepaper and repository which has been provided by **Northpole.**

Logo

Contract link

JLPStrategyBoost

- https://snowtrace.io/address/0x5F6b945ED9e4A03FFDb5d675D12224045d7C7e76

CauldronV2Strategys (USDC)

- https://snowtrace.io/address/0xc87ffa864850ef2915cda413fba0292df776ef06

CauldronV2Strategys (AVAX)

- https://snowtrace.io/address/0x1095A73749894176c52d8e3141E713ea1C8092B7

DegenBox

- https://snowtrace.io/address/0xC42BDbcfCc51e54B96b56254B659543B7a74Faf5

Risk Level Classification

Risk Level represents the classification or the probability that a certain function or threat that can exploit vulnerability and have an impact within the system or contract.
Risk Level is computed based on CVSS Version 3.0

Level	Value	Vulnerability
Critical	9 - 10	An Exposure that can affect the contract functions in several events that can risk and disrupt the contract
High	7 - 8.9	An Exposure that can affect the outcome when using the contract that can serve as an opening in manipulating the contract in an unwanted manner
Medium	4 - 6.9	An opening that could affect the outcome in executing the contract in a specific situation
Low	0.1 - 3.9	An opening but doesn’t have an impact on the functionality of the contract
Informational	0	An opening that consists of information’s but will not risk or affect the contract

Auditing Approach

Every line of code along with its functionalities will undergo manual review to check its security issues, quality, and contract scope of inheritance. The manual review will be done by our team that will document any issues that there were discovered.

Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:

    - Review of the specifications, sources, and instructions provided to ContractWolf to make sure we understand the size, scope, and functionality of the smart contract.
    - Manual review of code, our team will have a process of reading the code line-by-line with the intention of identifying potential vulnerabilities and security flaws.

2. Testing and automated analysis that includes:

    - Testing the smart contract functions with common test cases and scenarios, to ensure that it returns the expected results.

3. Best practices review, the team will review the contract with the aim to improve efficiency, effectiveness, clarifications, maintainability, security, and control within the smart contract.

4. Recommendations to help the project take steps to secure the smart contract.

Used Code from other Frameworks/Smart Contracts (Direct Imports)

Imported Packages

JLPStrategyBoost

- Address
- Babylonian
- BaseStrategyV2
- Context
- ERC20
- IBentoBoxMinimal
- IERC20
- IERC20MetaData
- IStrategyV2
- ISushiSwap
- IUniswapV2Pair
- IOracle
- IMasterChefV3
- JLPStrategyBoost
- Ownable
- SafeERC20
- UniswapV2Library
- USTMock
- ERC20Mock

CauldronV2Strategys (USDC)

- BoringERC20
- BoringMath
- BoringMath128
- BoringMath32
- BoringMath64
- BoringOwnable
- BoringOwnableData
- CauldronV2Strategys
- Domain
- ERC20
- ERC20Data
- ERCO20WithSupply
- IBatchFlashBorrower
- IBentoBoxV1
- IERC20
- IFlashBorrower
- IKashiPair
- IMasterContract
- IOracle
- IStrategy
- ISwapper
- JLPStrategy
- POLE
- RebaseLibrary

CauldronV2Strategys (AVAX)

- BoringERC20
- BoringMath
- BoringMath128
- BoringMath32
- BoringMath64
- BoringOwnable
- BoringOwnableData
- CauldronV2Strategys
- Domain
- ERC20
- ERC20Data
- ERCO20WithSupply
- IBatchFlashBorrower
- IBentoBoxV1
- IERC20
- IFlashBorrower
- IKashiPair
- IMasterContract
- IOracle
- IStrategy
- ISwapper
- JLPStrategy
- POLE
- RebaseLibrary

DegenBox

- IERC20
- IFlashBorrower
- IBatchFlashBorrower
- IWETH
- IStrategy
- BoringERC20
- BoringMath
- BoringMath128
- BoringMath64
- BoringMath32
- RebaseLibrary
- BoringOwnableData
- BoringOwnable
- IMasterContract
- BoringFactory
- MasterContractManager
- BaseBoringBatchable
- BoringBatchable
- DegenBox

Description

Optimization enabled: Yes

Contract	Version
JLPStrategyBoost	v0.8.7
CauldronV2Strategys (USDC)	v0.6.12
CauldronV2Strategys (AVAX)	v0.6.12
DegenBox	v0.6.12

Capabilities

Components

JLPStrategyBoost				
Version	Contracts	Libraries	Interfaces	Abstract
1.0	3	4	9	4

CauldronV2Strategys (USDC)				
Version	Contracts	Libraries	Interfaces	Abstract
1.0	7	6	10	1

CauldronV2Strategys (AVAX)				
Version	Contracts	Libraries	Interfaces	Abstract
1.0	7	6	10	1

DegenBox				
Version	Contracts	Libraries	Interfaces	Abstract
1.0	7	6	6	0

Exposed Functions

JLPStrategyBoost				
Version	Public	Private	External	Internal
1.0	20	5	70	47

CauldronV2Strategys (USDC)				
Version	Public	Private	External	Internal
1.0	23	3	100	39

CauldronV2Strategys (AVAX)				
Version	Public	Private	External	Internal
1.0	23	3	100	39

DegenBox				
Version	Public	Private	External	Internal
1.0	17	1	14	24

State Variables

JLPStrategyBoost		
Version	Total	Public
1.0	36	17

CauldronV2Strategys (USDC)		
Version	Total	Public
1.0	54	35

CauldronV2Strategys (AVAX)		
Version	Total	Public
1.0	54	35

DegenBox		
Version	Total	Public
1.0	27	10

Capabilities

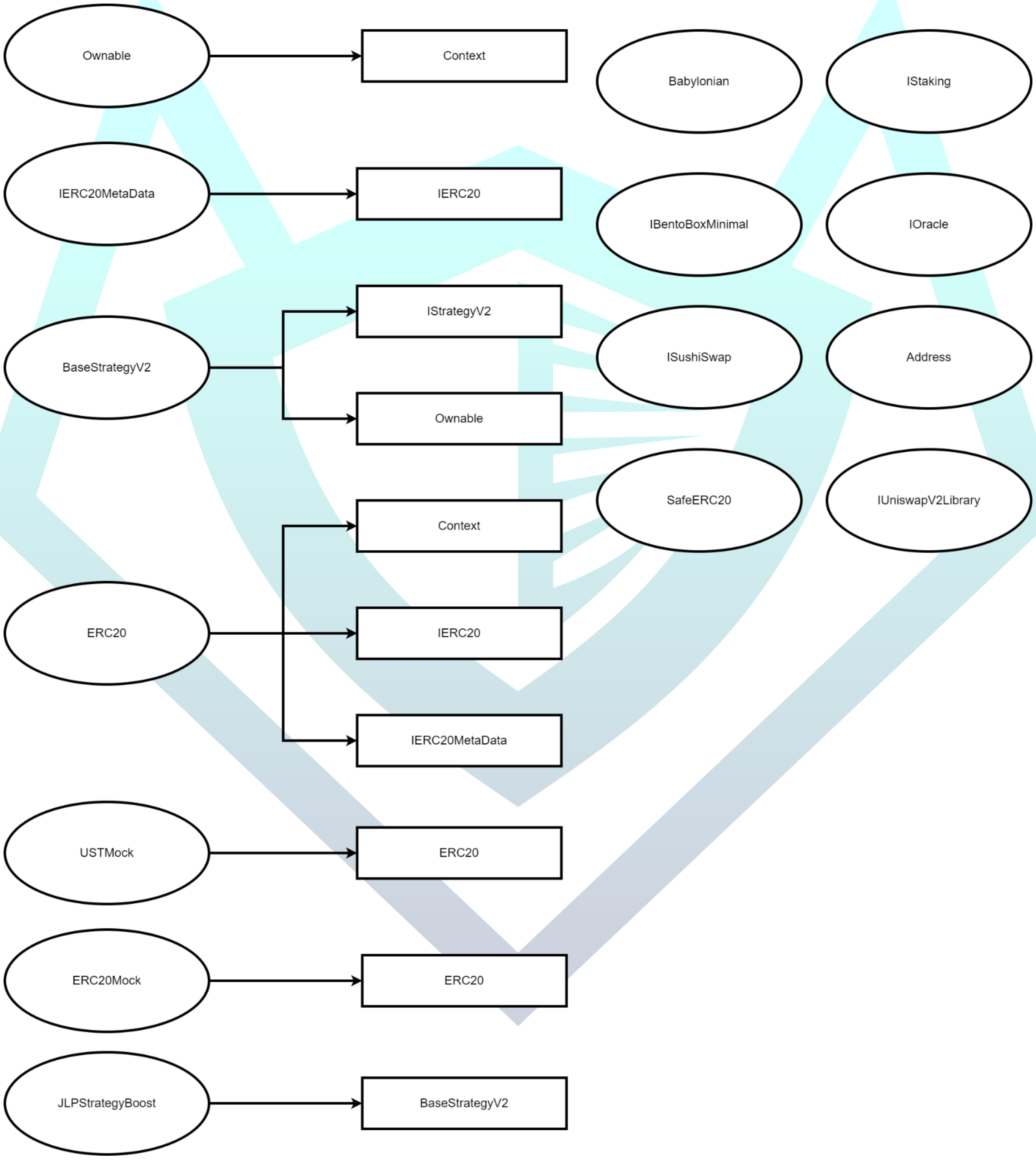
Version	Solidity Versions Observed	Experimental Features	Can Receive Funds	Uses Assembly	Has Destroyable Contracts
1.0	v0.6.12, v0.8.7		Yes	Yes	No

Scope of Work

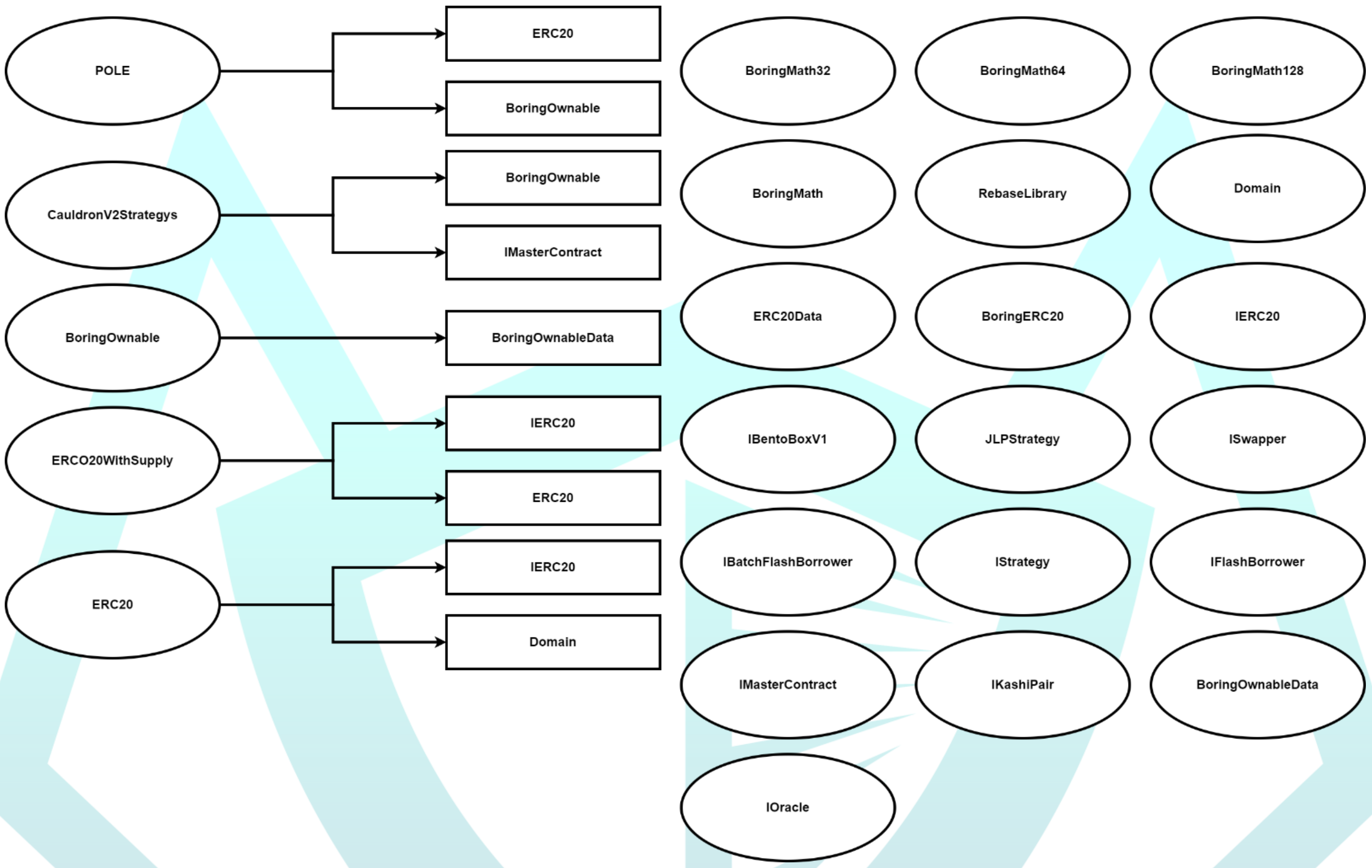
Northpole's team provided us with the files that needs to be tested (Github, Bscscan, Etherscan, files, etc.). The scope of the audit is the main contract.

Inheritance Graph

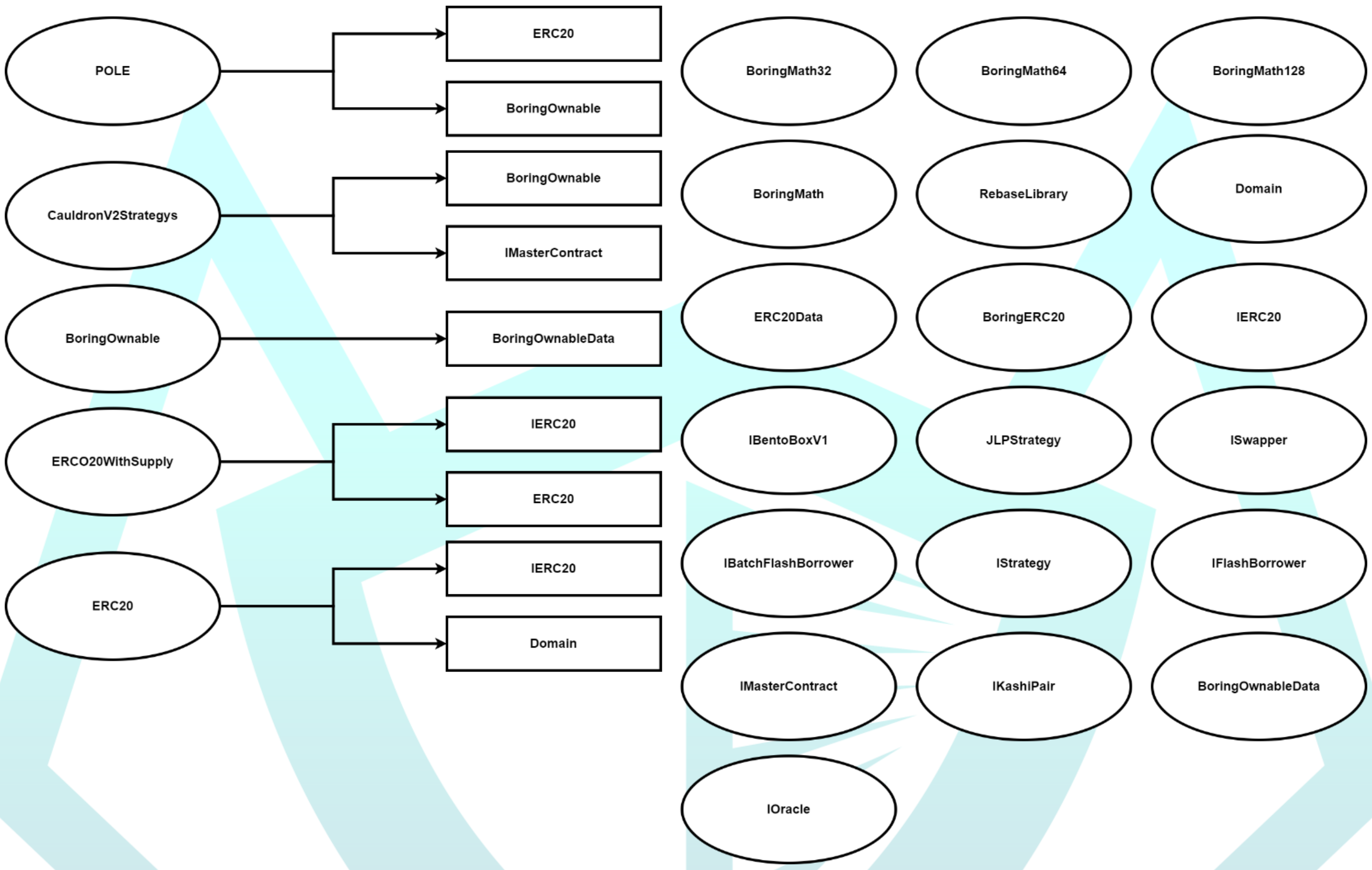
JLPStrategyBoost



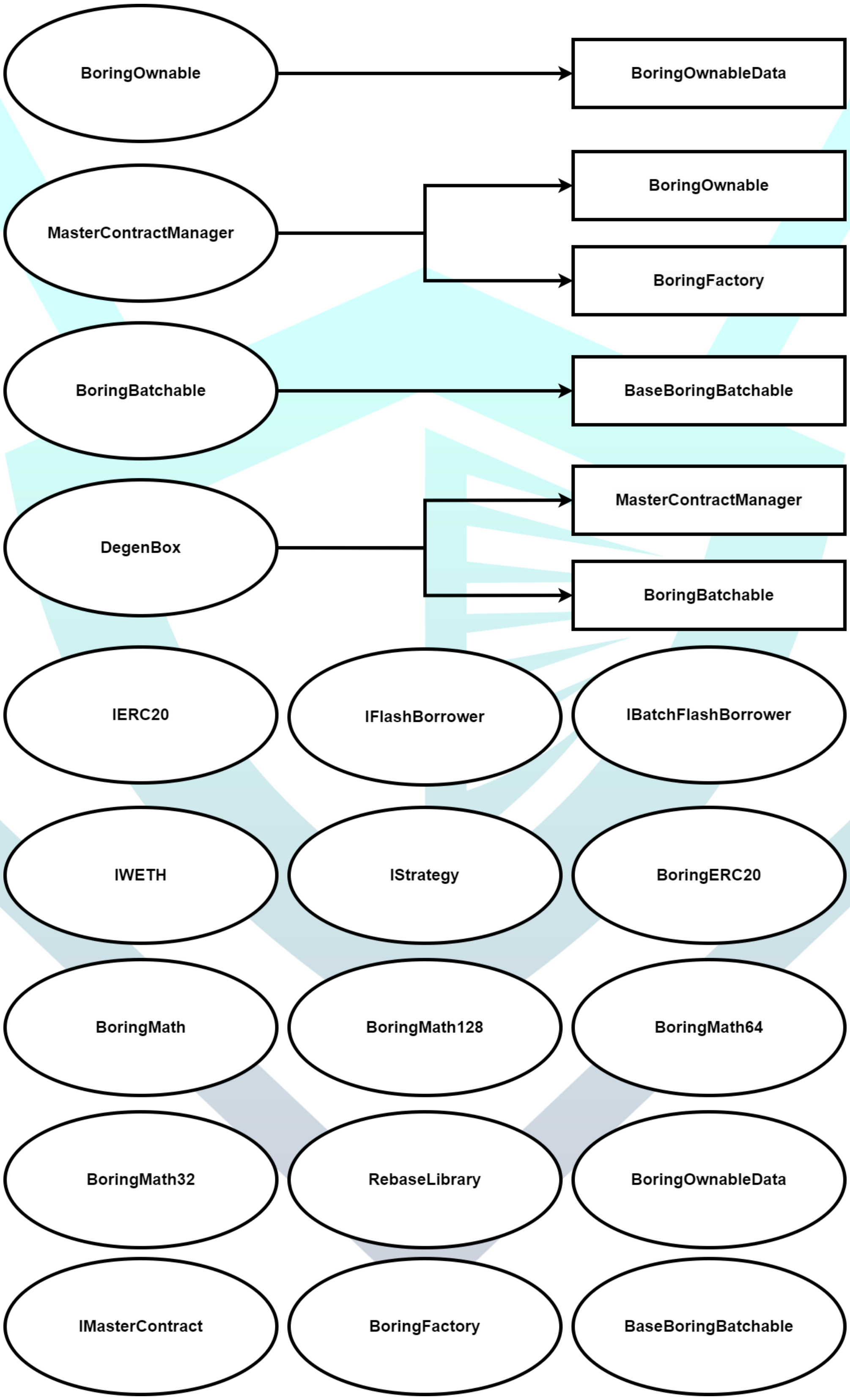
CauldronV2Strategys (USDC)



CauldronV2Strategys (AVAX)



DegenBox


BoringOwnable
BoringOwnableData
MasterContractManager
BoringOwnable
BoringFactory
BoringBatchable
BaseBoringBatchable
DegenBox
MasterContractManager
BoringBatchable
IERC20
IFlashBorrower
IBatchFlashBorrower
IWETH
IStrategy
BoringERC20
BoringMath
BoringMath128
BoringMath64
BoringMath32
RebaseLibrary
BoringOwnableData
IMasterContract
BoringFactory
BaseBoringBatchable

Verify Claims

Correct implementation of Token Standard

Tested	Verified
✓	✗

Function	Description	Exist	Tested	Verified
TotalSupply	Information about the total coin or token supply	✓	✓	✓
BalanceOf	Details on the account balance from a specified address	✓	✓	✓
Transfer	An action that transfers a specified amount of coin or token to a specified address	✓	✓	✓
TransferFrom	An action that transfers a specified amount of coin or token from a specified address	✓	✓	✓
Approve	Provides permission to withdraw specified number of coin or token from a specified address	✓	✓	✓

Function	JLPStrategy Boost	CauldronV2 Strategys (USDC)	CauldronV2 Strategys (AVAX)	DegenBox
Renounce Ownership	✓	–	–	–
Mint Token	✗	✓	✓	–
Block User	–	–	–	–
Burn Token	✗	✗	✗	–
Pause Contract	–	–	–	–

Overall Checkup (Smart Contract Security)

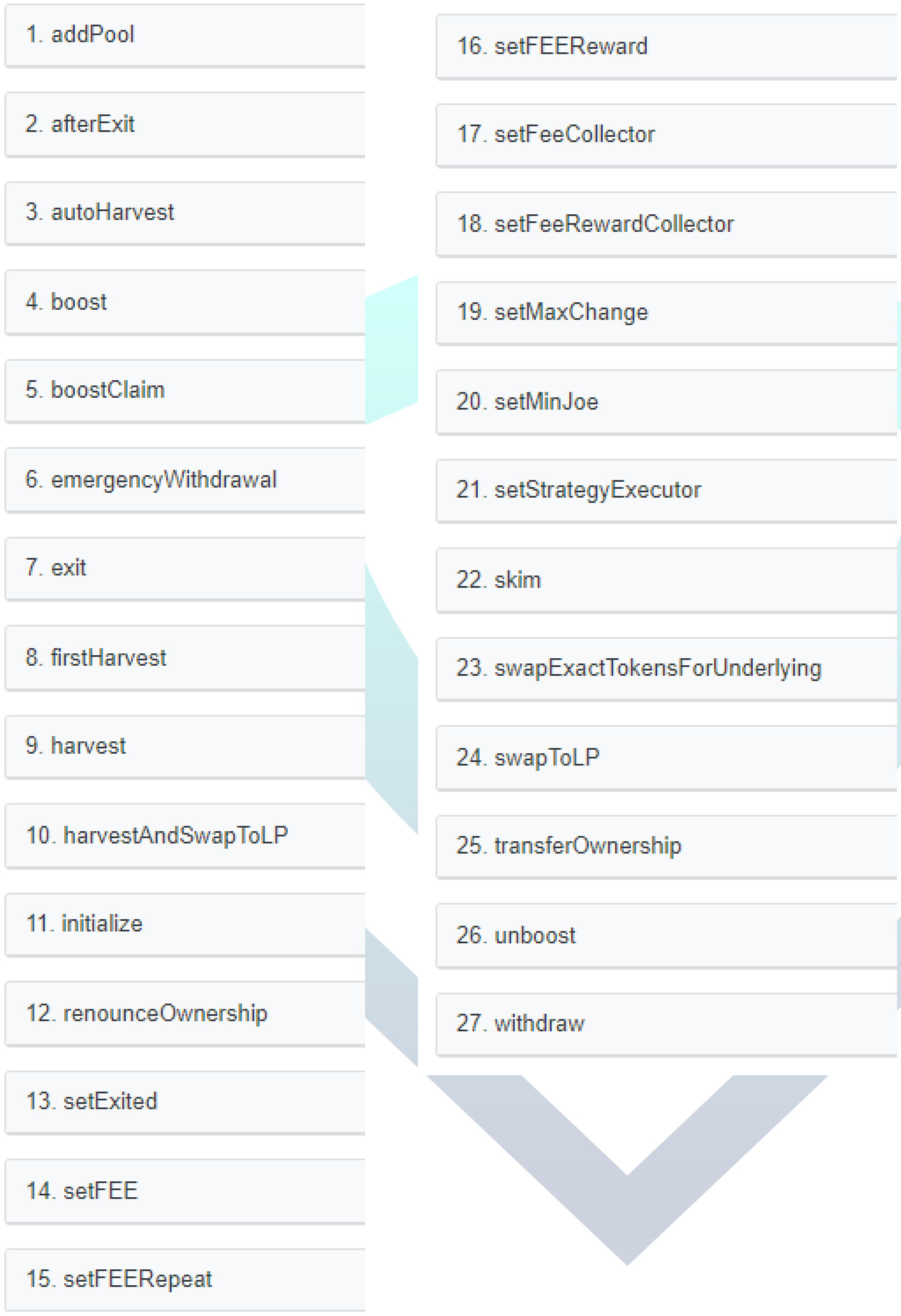
Tested	Verified
✓	✓

Legend

Attribute	Symbol
Verified / Can	✓
Verified / Cannot	✗
Unverified / Not checked	⚑
Not Available	–

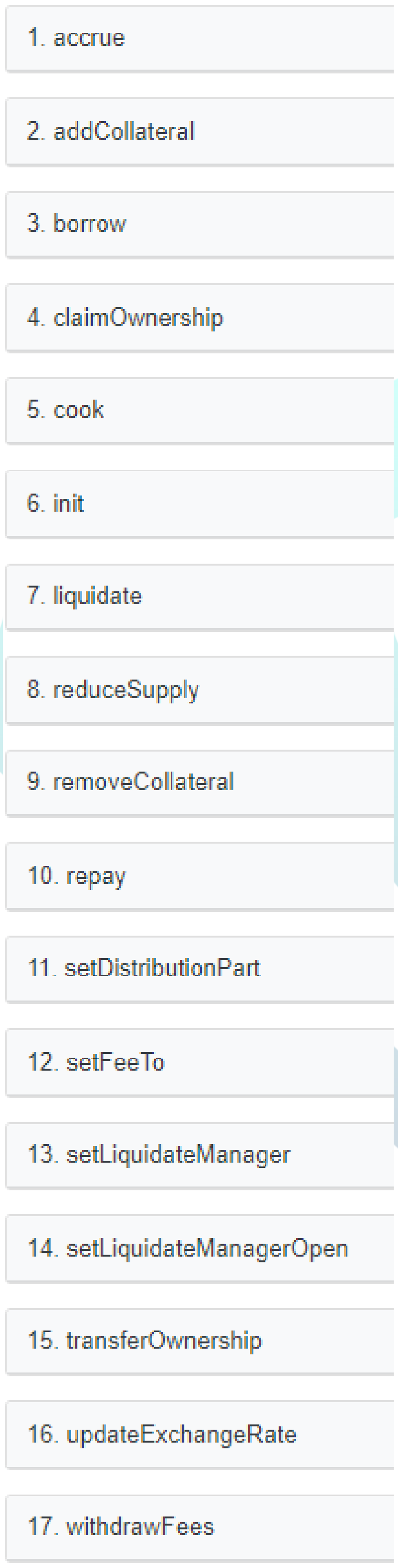
Write Functions of Contract

JLPStrategyBoost



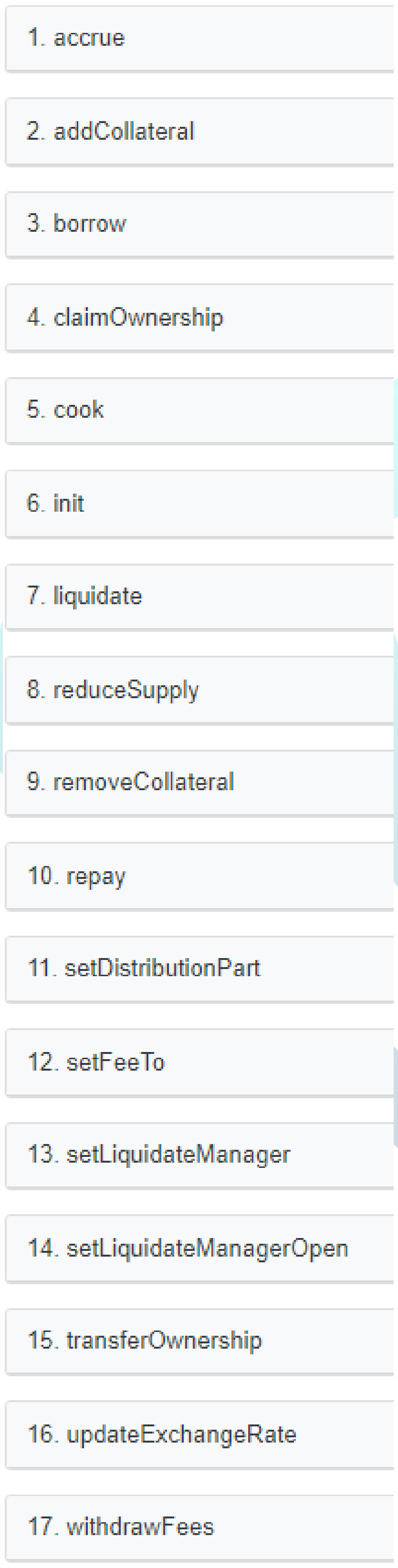
1. addPool
2. afterExit
3. autoHarvest
4. boost
5. boostClaim
6. emergencyWithdrawal
7. exit
8. firstHarvest
9. harvest
10. harvestAndSwapToLP
11. initialize
12. renounceOwnership
13. setExited
14. setFEE
15. setFEERepeat
16. setFEEReward
17. setFeeCollector
18. setFeeRewardCollector
19. setMaxChange
20. setMinJoe
21. setStrategyExecutor
22. skim
23. swapExactTokensForUnderlying
24. swapToLP
25. transferOwnership
26. unboost
27. withdraw

CauldronV2Strategys (USDC)



1. accrue
2. addCollateral
3. borrow
4. claimOwnership
5. cook
6. init
7. liquidate
8. reduceSupply
9. removeCollateral
10. repay
11. setDistributionPart
12. setFeeTo
13. setLiquidateManager
14. setLiquidateManagerOpen
15. transferOwnership
16. updateExchangeRate
17. withdrawFees

CauldronV2Strategys (AVAX)



1. accrue
2. addCollateral
3. borrow
4. claimOwnership
5. cook
6. init
7. liquidate
8. reduceSupply
9. removeCollateral
10. repay
11. setDistributionPart
12. setFeeTo
13. setLiquidateManager
14. setLiquidateManagerOpen
15. transferOwnership
16. updateExchangeRate
17. withdrawFees

DegenBox

1. batch
2. batchFlashLoan
3. claimOwnership
4. deploy
5. deposit
6. flashLoan
7. harvest
8. permitToken
9. registerProtocol
10. setMasterContractApproval
11. setStrategy
12. setStrategyTargetPercentage
13. transfer
14. transferMultiple
15. transferOwnership
16. whitelistMasterContract
17. withdraw

AUDIT PASSED

Low Issues

JLPStrategyBoost		
A floating pragma is set (SWC-103)	L: 3	Address.sol, Context.sol, ERC20.sol, IERC20.sol, IERC20MetaData.sol, Ownable.sol, SafeERC20.sol

Audit Comments

JLPStrategyBoost

- Deployer cannot mint after initial deployment
- Deployer cannot burn
- Deployer cannot block user
- Deployer cannot pause contract
- Deployer can set/update fees with an indefinite amount
- Deployer can set executor users
- Deployer can renounce ownership
- Deployer can transfer ownership
- Deployer can withdraw from contract
- Deployer can modify pool setting
- Deployer can add pool
- Executors can harvest tokens

CauldronV2Strategys (USDC)

- Deployer can transfer ownership
- Deployer can mint
- Deployer can mint to bentoBox
- Deployer cannot block user
- Deployer cannot burn
- Deployer cannot renounce ownership
- Deployer cannot pause contract

CauldronV2Strategys (AVAX)

- Deployer can transfer ownership
- Deployer can mint
- Deployer can mint to bentoBox
- Deployer cannot block user
- Deployer cannot burn
- Deployer cannot renounce ownership
- Deployer cannot pause contract

DegenBox

- Deployer cannot mint after initial deployment
- Deployer cannot burn
- Deployer cannot block user
- Deployer cannot pause contract
- Deployer can transfer ownership

CONTRACTWOLF

Blockchain Security - Smart Contract Audits