Ether Authority

www.EtherAuthority.io
audit@etherauthority.io

SMART CONTRACT

Security Audit Report

Project: Symmetric Protocol
Language: Solidity
Date: June 3rd, 2022

Table of contents

Introduction 4
Project Background 4
Audit Scope 4
Claimed Smart Contract Features 6
Audit Summary 8
Technical Quick Stats 9
Code Quality 10
Documentation 10
Use of Dependencies 10
AS-IS overview 11
Severity Definitions 20
Audit Findings 21
Conclusion 25
Our Methodology 26
Disclaimers 28
Appendix
- Code Flow Diagram 29
- Slither Results Log 43

THIS IS SECURITY AUDIT REPORT DOCUMENT AND WHICH MAY CONTAIN INFORMATION WHICH IS CONFIDENTIAL. WHICH INCLUDES ANY POTENTIAL VULNERABILITIES AND MALICIOUS CODES WHICH CAN BE USED TO EXPLOIT THE SOFTWARE. THIS MUST BE REFERRED INTERNALLY AND ONLY SHOULD BE MADE AVAILABLE TO THE PUBLIC AFTER ISSUES ARE RESOLVED.

Email: audit@EtherAuthority.io

Introduction

EtherAuthority was contracted by Symmetric to perform the Security audit of the Symmetric Protocol smart contracts code. The audit has been performed using manual analysis as well as using automated software tools. This report presents all the findings regarding the audit performed on June 3rd, 2022.

The purpose of this audit was to address the following:

- Ensure that all claimed functions exist and function correctly.
- Identify any security vulnerabilities that may be present in the smart contract.

Project Background

Symmetric is a decentralized exchange (DEX) and automated market maker (AMM).

Audit scope

Name	**Code Review and Security Analysis Report for Symmetric Protocol Smart Contracts**
Platform	**Solidity**
File 1	Authorizer.sol
File 1 MD5 Hash	A14C2F014CD8084E588242A59F517B29
File 2	Vault.sol
File 2 MD5 Hash	BC7432199901BA0E9A43CDE36B1AC190
File 3	VaultAuthorization.sol
File 3 MD5 Hash	EB2CEEF1BE14469E2C11505042A39BC4
File 4	AssetTransfersHandler.sol
File 4 MD5 Hash	102BF0275AB32C9BB96FF31382A46377
File 5	ProtocolFeesCollector.sol
File 5 MD5 Hash	998430C9363547982B9F983B913AD064
File 6	Swaps.sol

File 6 MD5 Hash	1EA05EBB04A7A30D2E9E064428ACBDE6
File 7	PoolRegistry.sol
File 7 MD5 Hash	BAEF612AD577C71AD37D26F8541E9CE7
File 8	WeightedPoolFactory.sol
File 8 MD5 Hash	1DE9205B4E1C6B933572D3836C481FF3
File 9	WeightedPool2TokenFactory.sol
File 9 MD5 Hash	CA8C31B344ADEB488898854D8AD62139
File 10	BalancerHelpers.sol
File 10 MD5 Hash	DF488C795F7951074806EBEAA0CED0C8
File 11	BatchRelayerLibrary.sol
File 11 MD5 Hash	C4CB25C8574E58A04FC3F997EFB2674D
File 12	ManagedPoolFactory.sol
File 12 MD5 Hash	FA3C3E4F9F5604FE95030D9642F1E9BA
File 13	SymmChef.sol
File 13 MD5 Hash	E1FFEA63B84494E4BDFCA5B69E0659F5
File 14	ComplexRewarder.sol
File 14 MD5 Hash	6396175E83DE556241FD9D89730D3D0D
Audit Date	June 3rd,2022

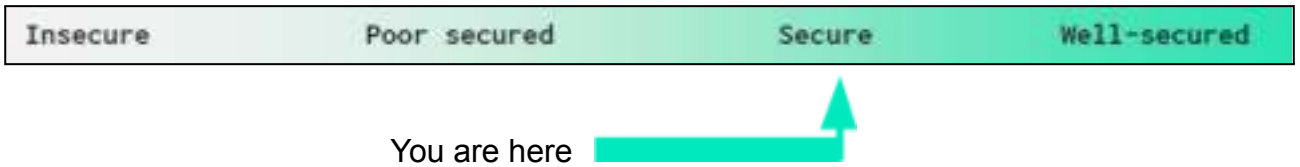
Claimed Smart Contract Features

Claimed Feature Detail	Our Observation
File 1 Authorizer.sol • Authorizer has functions like: revokeRoles, revokeRolesFromMany, etc.	**YES, This is valid.**
File 2 Vault.sol • Vault has functions like: setPaused, etc.	**YES, This is valid.**
File 3 VaultAuthorization.sol • VaultAuthorization has functions like: setAuthorizer, getAuthorizer, etc.	**YES, This is valid.**
File 4 AssetTransfersHandler.sol • AssetTransfersHandler has functions like: _receiveAsset, _sendAsset, etc.	**YES, This is valid.**
File 5 ProtocolFeesCollector.sol • Maximum Protocol Swap Fee Percentage: 50% • Maximum Protocol Flash Loan Fee Percentage: 1%	**YES, This is valid.** **Owner wallet's private key must be handled very securely. Because if that is compromised, then it will create problems.**
File 6 Swaps.sol • Swaps has functions like: swap, batchSwap, etc.	**YES, This is valid.**
File 7 PoolRegistry.sol • PoolRegistry has functions like: registerPool, getPool, etc.	**YES, This is valid.**
File 8 WeightedPoolFactory.sol • WeightedPoolFactory has functions like: create, etc.	**YES, This is valid.**

File 9 WeightedPool2TokenFactory.sol • WeightedPool2TokensFactory has functions like: create, etc.	**YES, This is valid.**
File 10 BalancerHelpers.sol • BalancerHelpers has functions like: queryJoin, queryExit, etc.	**YES, This is valid.**
File 11 BatchRelayerLibrary.sol • BatchRelayerLibrary has inherited BaseRelayerLibrary, AaveWrapping, LidoWrapping, VaultActions, VaultPermit contacts.	**YES, This is valid.**
File 12 ManagedPoolFactory.sol • ManagedPoolFactory has functions like: create, etc.	**YES, This is valid.**
File 13 SymmChef.sol • SymmChef has functions like: poolLength, add, etc.	**YES, This is valid.**
File 14 ComplexRewarder.sol • ComplexRewarder has functions like: massUpdatePools, pendingToken, etc. • The ComplexRewarder owner can set the symm per second to be distributed. • Owner can reclaim/withdraw any tokens (including reward tokens) held by this contract.	**YES, This is valid.**

Audit Summary

According to the standard audit assessment, Customer`s solidity smart contracts are **"Secured"**. Also, these contracts do contain owner control, which does not make them fully decentralized.



We used various tools like Slither, Solhint and Remix IDE. At the same time this finding is based on critical analysis of the manual audit.
All issues found during automated analysis were manually reviewed and applicable vulnerabilities are presented in the Audit overview section. General overview is presented in AS-IS section and all identified issues can be found in the Audit overview section.

We found 0 critical, 0 high, 0 medium and 3 low and some very low level issues.

Investors Advice: Technical audit of the smart contract does not guarantee the ethical nature of the project. Any owner controlled functions should be executed by the owner with responsibility. All investors/users are advised to do their due diligence before investing in the project.

Technical Quick Stats

Main Category	Subcategory	Result
Contract Programming	Solidity version not specified	Passed
	Solidity version too old	Passed
	Integer overflow/underflow	Passed
	Function input parameters lack of check	Moderated
	Function input parameters check bypass	Passed
	Function access control lacks management	Passed
	Critical operation lacks event log	Moderated
	Human/contract checks bypass	Passed
	Random number generation/use vulnerability	N/A
	Fallback function misuse	Passed
	Race condition	Passed
	Logical vulnerability	Passed
	Features claimed	Moderated
	Other programming issues	Passed
Code Specification	Function visibility not explicitly declared	Passed
	Var. storage location not explicitly declared	Passed
	Use keywords/functions to be deprecated	Passed
	Unused code	Passed
Gas Optimization	“Out of Gas” Issue	Passed
	High consumption ‘for/while’ loop	Passed
	High consumption ‘storage’ storage	Passed
	Assert() misuse	Passed
Business Risk	The maximum limit for mintage not set	Passed
	“Short Address” Attack	Passed
	“Double Spend” Attack	Passed

Overall Audit Result: **PASSED**

Code Quality

This audit scope has 14 smart contract files. Smart contracts contain Libraries, Smart contracts, inherits and Interfaces. This is a compact and well written smart contract.

The libraries in the Symmetric Protocol are part of its logical algorithm. A library is a different type of smart contract that contains reusable code. Once deployed on the blockchain (only once), it is assigned a specific address and its properties / methods can be reused many times by other contracts in the Symmetric Protocol.

The Symmetric team has not provided unit test scripts, which would have helped to determine the integrity of the code in an automated way.

Some code parts are well commented on smart contracts. We suggest using Ethereum's NatSpec style for the commenting.

Documentation

We were given a Symmetric Protocol smart contract code in the form of a Github weblink. The hash of that code is mentioned above in the table.

As mentioned above, code parts are well commented. So it is easy to quickly understand the programming flow as well as complex code logic. Comments are very helpful in understanding the overall architecture of the protocol.

Another source of information was its official website https://dev-symmv2-celo.symmetric.exchange/ which provided rich information about the project architecture.

Use of Dependencies

As per our observation, the libraries are used in this smart contracts infrastructure that are based on well known industry standard open source projects.

Apart from libraries, its functions are used in external smart contract calls.

AS-IS overview

Authorizer.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	canPerform	read	Passed	No Issue
3	grantRoles	external	Passed	No Issue
4	grantRolesGlobally	external	Passed	No Issue
5	grantRolesToMany	external	Passed	No Issue
6	grantRolesGloballyToMany	external	Passed	No Issue
7	revokeRoles	external	Passed	No Issue
8	revokeRolesGlobally	external	Passed	No Issue
9	revokeRolesFromMany	external	Passed	No Issue
10	revokeRolesGloballyFrom Many	external	Passed	No Issue
11	hasRole	read	Passed	No Issue
12	getRoleGlobalMemberCou nt	read	Passed	No Issue
13	getRoleGlobalMember	read	Passed	No Issue
14	getRoleMemberCountByC ontract	read	Passed	No Issue
15	getRoleMemberByContract	read	Passed	No Issue
16	getRoleAdmin	read	Passed	No Issue
17	grantRole	write	Passed	No Issue
18	grantRoleGlobally	write	Passed	No Issue
19	revokeRole	write	Passed	No Issue
20	revokeRoleGlobally	write	Passed	No Issue
21	renounceRole	write	Passed	No Issue
22	renounceRoleGlobally	write	Passed	No Issue
23	_setupRole	write	Passed	No Issue
24	_setRoleAdmin	write	Passed	No Issue
25	_grantRole	write	Passed	No Issue
26	_grantRoleGlobally	write	Passed	No Issue
27	_revokeRole	write	Passed	No Issue
28	_revokeRoleGlobally	write	Passed	No Issue

Vault.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	setPaused	external	Passed	No Issue
3	WETH	external	Passed	No Issue

4	authenticateFor	modifier	Passed	No Issue
5	setAuthorizer	external	Passed	No Issue
6	_setAuthorizer	write	Passed	No Issue
7	getAuthorizer	external	Passed	No Issue
8	setRelayerApproval	external	Passed	No Issue
9	hasApprovedRelayer	external	Passed	No Issue
10	_hasApprovedRelayer	internal	Passed	No Issue
11	_canPerform	internal	Passed	No Issue
12	_typeHash	internal	Passed	No Issue
13	flashLoan	external	Passed	No Issue
14	_tokenGiven	write	Passed	No Issue
15	_tokenCalculated	write	Passed	No Issue
16	_getAmounts	write	Passed	No Issue
17	_swapWithPools	write	Passed	No Issue
18	_swapWithPool	write	Passed	No Issue
19	_processTwoTokenPoolSwapRequest	write	Passed	No Issue
20	_processMinimalSwapInfoPoolSwapRequest	write	Passed	No Issue
21	_callMinimalSwapInfoPoolOnSwapHook	write	Passed	No Issue
22	_processGeneralPoolSwapRequest	write	Passed	No Issue
23	queryBatchSwap	external	Passed	No Issue

VaultAuthorization.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	authenticateFor	modifier	Passed	No Issue
3	setAuthorizer	external	access by authenticate	No Issue
4	_setAuthorizer	write	Passed	No Issue
5	getAuthorizer	external	Passed	No Issue
6	setRelayerApproval	external	Function input parameters lack of check	Refer Audit Findings
7	hasApprovedRelayer	external	Passed	No Issue
8	_authenticateFor	internal	Passed	No Issue
9	_hasApprovedRelayer	internal	Passed	No Issue
10	_canPerform	internal	Passed	No Issue
11	_typeHash	internal	Passed	No Issue
12	nonReentrant	modifier	Passed	No Issue
13	_enterNonReentrant	write	Passed	No Issue
14	_exitNonReentrant	write	Passed	No Issue
15	authenticate	modifier	Passed	No Issue
16	_authenticateCaller	internal	Passed	No Issue
17	getActionId	read	Passed	No Issue

18	_canPerform	internal	Passed	No Issue
19	getDomainSeparator	external	Passed	No Issue
20	getNextNonce	external	Passed	No Issue
21	_validateSignature	internal	Passed	No Issue
22	_isSignatureValid	write	Passed	No Issue
23	_typeHash	internal	Passed	No Issue
24	_signature	internal	Passed	No Issue
25	_deadline	internal	Passed	No Issue
26	_calldata	internal	Passed	No Issue
27	_decodeExtraCalldataWord	write	Passed	No Issue
28	_require	write	Passed	No Issue
29	_revert	write	Passed	No Issue
30	whenNotPaused	modifier	Passed	No Issue
31	getPausedState	external	Passed	No Issue
32	_setPaused	internal	Passed	No Issue
33	_ensureNotPaused	internal	Passed	No Issue
34	_ensurePaused	internal	Passed	No Issue
35	_isNotPaused	internal	Passed	No Issue
36	_getPauseWindowEndTime	write	Passed	No Issue
37	_getBufferPeriodEndTime	write	Passed	No Issue

AssetTransfersHandler.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	_receiveAsset	internal	Passed	No Issue
3	_sendAsset	internal	Passed	No Issue
4	_handleRemainingEth	internal	Passed	No Issue
5	receive	external	Passed	No Issue
6	_increaseInternalBalance	internal	Passed	No Issue
7	_decreaseInternalBalance	internal	Passed	No Issue
8	_WETH	internal	Passed	No Issue
9	_isETH	internal	Passed	No Issue
10	_translateToIERC20	internal	Passed	No Issue
11	_translateToIERC20	internal	Passed	No Issue
12	_asIERC20	internal	Passed	No Issue

ProtocolFeesCollector.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue

2	withdrawCollectedFees	external	Function input parameters lack of check	Refer Audit Findings
3	setSwapFeePercentage	external	access by authenticate	No Issue
4	setFlashLoanFeePercentage	external	access by authenticate	No Issue
5	getSwapFeePercentage	external	Passed	No Issue
6	getFlashLoanFeePercentage	external	Passed	No Issue
7	getCollectedFeeAmounts	external	Passed	No Issue
8	getAuthorizer	external	Passed	No Issue
9	_canPerform	internal	Passed	No Issue
10	_getAuthorizer	internal	Passed	No Issue
11	authenticate	modifier	Passed	No Issue
12	_authenticateCaller	internal	Passed	No Issue
13	getActionId	read	Passed	No Issue
14	_canPerform	internal	Passed	No Issue
15	nonReentrant	modifier	Passed	No Issue
16	_enterNonReentrant	write	Passed	No Issue
17	_exitNonReentrant	write	Passed	No Issue

Swaps.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	nonReentrant	modifier	Passed	No Issue
3	_enterNonReentrant	write	Passed	No Issue
4	_exitNonReentrant	write	Passed	No Issue
5	swap	external	Critical operation lacks event log	Refer Audit Findings
6	batchSwap	external	Critical operation lacks event log	Refer Audit Findings
7	_tokenGiven	write	Passed	No Issue
8	_tokenCalculated	write	Passed	No Issue
9	_getAmounts	write	Passed	No Issue
10	_swapWithPools	write	Passed	No Issue
11	_swapWithPool	write	Passed	No Issue
12	_processTwoTokenPoolSwapRequest	write	Passed	No Issue
13	_processMinimalSwapInfoPoolSwapRequest	write	Passed	No Issue
14	_callMinimalSwapInfoPoolOnSwapHook	internal	Passed	No Issue
15	queryBatchSwap	external	Critical operation lacks event log	Refer Audit Findings
16	joinPool	external	Passed	No Issue
17	exitPool	external	Passed	No Issue

18	_toPoolBalanceChange	write	Passed	No Issue
19	_toPoolBalanceChange	write	Passed	No Issue
20	_joinOrExit	write	access by authenticate	No Issue
21	_callPoolBalanceChange	write	Passed	No Issue
22	_processJoinPoolTransfers	write	Passed	No Issue
23	_processExitPoolTransfers	write	Passed	No Issue
24	_validateTokensAndGetBalances	read	Passed	No Issue
25	_unsafeCastToInt256	write	Passed	No Issue

PoolRegistry.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	withRegisteredPool	modifier	Passed	No Issue
3	onlyPool	modifier	Passed	No Issue
4	_ensureRegisteredPool	internal	Passed	No Issue
5	_ensurePoolIsSender	read	Passed	No Issue
6	registerPool	external	Passed	No Issue
7	getPool	external	Passed	No Issue
8	_toPoolId	internal	Passed	No Issue
9	_getPoolAddress	internal	Passed	No Issue
10	_getPoolSpecialization	internal	Passed	No Issue
11	authenticateFor	modifier	Passed	No Issue
12	setAuthorizer	external	access by authenticate	No Issue
13	_setAuthorizer	write	Passed	No Issue
14	getAuthorizer	external	Passed	No Issue
15	setRelayerApproval	external	access by authenticate	No Issue
16	hasApprovedRelayer	external	Passed	No Issue
17	_hasApprovedRelayer	internal	Passed	No Issue
18	_canPerform	internal	Passed	No Issue
19	_typeHash	internal	Passed	No Issue

WeightedPoolFactory.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	create	external	Passed	No Issue
3	getVault	read	Passed	No Issue

4	isPoolFromFactory	external	Passed	No Issue
5	_create	internal	Passed	No Issue
6	getPauseConfiguration	read	Passed	No Issue

WeightedPool2TokenFactory.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	getVault	read	Passed	No Issue
3	isPoolFromFactory	external	Passed	No Issue
4	_create	internal	Passed	No Issue
5	getPauseConfiguration	read	Passed	No Issue
6	create	external	Passed	No Issue

BalancerHelpers.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	queryJoin	external	Function input parameters lack of check, Critical operation lacks event log	Refer Audit Findings
3	queryExit	external	Function input parameters lack of check, Critical operation lacks event log	Refer Audit Findings
4	_validateAssetsAndGetBalances	internal	Passed	No Issue
5	_WETH	internal	Passed	No Issue
6	_isETH	internal	Passed	No Issue
7	_translateToIERC20	internal	Passed	No Issue
8	_translateToIERC20	internal	Passed	No Issue
9	_asIERC20	internal	Passed	No Issue

BatchRelayerLibrary.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	getVault	read	Passed	No Issue
3	getEntrypoint	read	Passed	No Issue
4	setRelayerApproval	external	Function input parameters lack of check	Refer Audit Findings

5	approveVault	write	Function input parameters lack of check	Refer Audit Findings
6	_pullToken	internal	Passed	No Issue
7	_pullTokens	internal	Passed	No Issue
8	_isChainedReference	internal	Passed	No Issue
9	_setChainedReferenceValue`	internal	Passed	No Issue
10	_getChainedReferenceValue	internal	Passed	No Issue
11	_getTempStorageSlot	read	Passed	No Issue
12	wrapAaveDynamicToken	external	Passed	No Issue
13	unwrapAaveStaticToken	external	Passed	No Issue
14	_require	write	Passed	No Issue
15	_revert	write	Passed	No Issue
16	wrapStETH	external	Passed	No Issue
17	unwrapWstETH	external	Passed	No Issue
18	stakeETH	external	Passed	No Issue
19	stakeETHAndWrap	external	Passed	No Issue
20	swap	external	Passed	No Issue
21	batchSwap	external	Passed	No Issue
22	manageUserBalance	external	Passed	No Issue
23	joinPool	external	Passed	No Issue
24	_doJoinPoolChainedReferenceReplacements	write	Passed	No Issue
25	_doWeightedJoinChainedReferenceReplacements	write	Passed	No Issue
26	_doWeightedExactTokensInForBPTOutReplacements	write	Passed	No Issue
27	exitPool	external	Passed	No Issue
28	_doExitPoolChainedReferenceReplacements	write	Passed	No Issue
29	_doWeightedExitChainedReferenceReplacements	write	Passed	No Issue
30	_doWeightedExactBptInForOneTokenOutReplacements	write	Passed	No Issue
31	_doWeightedExactBptInForTokensOutReplacements	write	Passed	No Issue
32	vaultPermit	write	Passed	No Issue
33	vaultPermitDAI	write	Passed	No Issue

ManagedPoolFactory.sol

Functions

Sl.	Functions	Type	Observation	Conclusion

1	constructor	write	Passed	No Issue
2	create	external	Passed	No Issue
3	isPoolFromFactory	external	Passed	No Issue

SymmChef.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	poolLength	read	Passed	No Issue
3	add	write	Function input parameters lack of check, Duplicate LP can be added	Refer Audit Findings
4	set	write	Function input parameters lack of check	Refer Audit Findings
5	setSymmPerSecond	write	access only Owner	No Issue
6	pendingSymm	external	Function input parameters lack of check	Refer Audit Findings
7	massUpdatePools	external	Passed	No Issue
8	updatePool	write	Passed	No Issue
9	deposit	write	Function input parameters lack of check	Refer Audit Findings
10	withdraw	write	Function input parameters lack of check	Refer Audit Findings
11	harvest	write	Function input parameters lack of check	Refer Audit Findings
12	withdrawAndHarvest	write	Function input parameters lack of check	Refer Audit Findings
13	emergencyWithdraw	write	Function input parameters lack of check	Refer Audit Findings
14	transferOwnership	write	access only Owner	No Issue
15	claimOwnership	write	Passed	No Issue
16	onlyOwner	modifier	Passed	No Issue
17	permitToken	write	Passed	No Issue

ComplexRewarder.sol

Functions

Sl.	Functions	Type	Observation	Conclusion
1	constructor	write	Passed	No Issue
2	transferOwnership	write	access only Owner	No Issue
3	claimOwnership	write	Passed	No Issue
4	onlyOwner	modifier	Passed	No Issue
5	onSymmReward	external	Function input parameters lack of check	Refer Audit Findings

6	pendingTokens	external	Function input parameters lack of check	Refer Audit Findings
7	setRewardPerSecond	write	access only Owner	No Issue
8	onlyChef	modifier	Passed	No Issue
9	poolLength	read	Passed	No Issue
10	add	write	access only Owner	No Issue
11	set	write	access only Owner	No Issue
12	reclaimTokens	write	Function input parameters lack of check	Refer Audit Findings
13	pendingToken	read	Function input parameters lack of check	Refer Audit Findings
14	massUpdatePools	external	Passed	No Issue
15	updatePool	write	Passed	No Issue

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to token loss etc.
High	High-level vulnerabilities are difficult to exploit; however, they also have significant impact on smart contract execution, e.g. public access to crucial
Medium	Medium-level vulnerabilities are important to fix; however, they can't lead to tokens lose
Low	Low-level vulnerabilities are mostly related to outdated, unused etc. code snippets, that can't have significant impact on execution
Lowest / Code Style / Best Practice	Lowest-level vulnerabilities, code style violations and info statements can't affect smart contract execution and can be ignored.

Audit Findings

Critical Severity

No Critical severity vulnerabilities were found.

High Severity

No High severity vulnerabilities were found.

Medium

No Medium severity vulnerabilities were found.

Low

(1) Duplicate LP can be added: **SymmChef.sol**

```
/// @notice Add a new LP to the pool. Can only be called by the owner.
/// DO NOT add the same LP token more than once. Rewards will be messed up if you do.
/// @param allocPoint AP of the new pool.
/// @param _lpToken Address of the LP ERC-20 token.
/// @param _rewarder Address of the rewarder delegate.
function add(uint256 allocPoint, IERC20 _lpToken, IRewarder _rewarder) public onlyOwner {
    totalAllocPoint = totalAllocPoint.add(allocPoint);
    lpToken.push(_lpToken);
    rewarder.push(_rewarder);

    poolInfo.push(PoolInfo({
        allocPoint: allocPoint.to64(),
        lastRewardTime: block.timestamp.to64(),
        accSymmPerShare: 0
    }));
    emit LogPoolAddition(lpToken.length.sub(1), allocPoint, _lpToken, _rewarder);
}
```

As per the comments, duplicate lp tokens should not be added into the pool. But there is no validation for duplicate lp tokens. So the owner can add the same lp tokens more than once.

Resolution: We suggest adding validation for duplicate checks for lp tokens in the pool.

(1) Function input parameters lack of check:

Variable validation is not performed in below functions:

VaultAuthorization.sol

- setRelayerApproval = sender , relayer.

ProtocolFeesCollector.sol

- withdrawCollectedFees = recipient.

BalancerHelpers.sol

- queryjoin = recipient , sender
- queryExit = recipient , sender

SymmChef.sol

- add = _lpToken , _rewarder
- set = _rewarder
- pendingSymm = _user
- deposit = to
- withdraw = to
- harvet = to
- withdrawAndHarvest = to
- emergencyWithdraw= to

ComplexRewarder.sol

- onSymmReward = to , _user
- pendingTokens = user
- reclaimTokens = to
- pendingToken = _user

BatchRelayerLibrary.sol

- setRelayerApproval = relayer
- approveVault = token

Resolution: We advise to put validation like integer type variables should be greater than 0 and address type variables should not be address(0).

(2) Critical operation lacks event log:

Missing event log for:

Swaps.sol

- swap
- batchswap
- queryBatchSwap

BalancerHelpers.sol

- queryJoin
- queryExit

Resolution: Write an event log for listed events.

Very Low / Informational / Best practices:

No Informational severity vulnerabilities were found.

Centralization

This smart contract has some functions which can be executed by the Admin (Owner) only. If the admin wallet private key would be compromised, then it would create trouble. Following are Admin functions:

- setAuthorizer: VaultAuthorization can set a new authorizer.
- setRelayerApproval: VaultAuthorization can set relayer approval.
- receive: VaultAuthorization can enable the Vault to receive ETH.
- onSymmReward: ComplexRewarder chef owner can set on symm reward address.
- setRewardPerSecond: ComplexRewarder owner can set the symm per second to be distributed.
- :add: ComplexRewarder owner can add a new LP to the pool.
- set: ComplexRewarder owner can update the given pool's SYMM allocation point and `IRewarder` contract.

- reclaimTokens: ComplexRewarder owner can allow owner to reclaim/withdraw any tokens (including reward tokens) held by this contract.
- withdrawCollectedFees: ProtocolFeesCollector can withdraw collected fees.
- setSwapFeePercentage: ProtocolFeesCollector can set swap fee percentage value.
- setFlashLoanFeePercentage: ProtocolFeesCollector can set flash loan percentage value.
- swap: Swaps authenticate can execute a single swap.
- batchSwap: Swaps authenticate can perform multiple swaps in sequence.
- add: SymmChef owner can add a new LP to the pool.
- set: SymmChef owner can update the given pool's SYMM allocation point and `IRewarder` contract.
- setSymmPerSecond: SymmChef owner can set the symm per second to be distributed.
- setPaused: Vault owner can set paused status.

To make the smart contract 100% decentralized, we suggest renouncing ownership in the smart contract once its function is completed.

Conclusion

We were given a contract code in the form of github weblink. And we have used all possible tests based on given objects as files. We have not observed any major issues in the smart contracts. **So, the smart contracts are ready for the mainnet deployment**.

Since possible test cases can be unlimited for such smart contracts protocol, we provide no such guarantee of future outcomes. We have used all the latest static tools and manual observations to cover maximum possible test cases to scan everything.

Smart contracts within the scope were manually reviewed and analyzed with static analysis tools. Smart Contract's high-level description of functionality was presented in the As-is overview section of the report.

Audit report contains all found security vulnerabilities and other issues in the reviewed code.

Security state of the reviewed contract, based on standard audit procedure scope, is **"Secured"**.

Our Methodology

We like to work with a transparent process and make our reviews a collaborative effort. The goals of our security audits are to improve the quality of systems we review and aim for sufficient remediation to help protect users. The following is the methodology we use in our security audit process.

Manual Code Review:

In manually reviewing all of the code, we look for any potential issues with code logic, error handling, protocol and header parsing, cryptographic errors, and random number generators. We also watch for areas where more defensive programming could reduce the risk of future mistakes and speed up future audits. Although our primary focus is on the in-scope code, we examine dependency code and behavior when it is relevant to a particular line of investigation.

Vulnerability Analysis:

Our audit techniques included manual code analysis, user interface interaction, and whitebox penetration testing. We look at the project's web site to get a high level understanding of what functionality the software under review provides. We then meet with the developers to gain an appreciation of their vision of the software. We install and use the relevant software, exploring the user interactions and roles. While we do this, we brainstorm threat models and attack surfaces. We read design documentation, review other audit results, search for similar projects, examine source code dependencies, skim open issue tickets, and generally investigate details other than the implementation.

Documenting Results:

We follow a conservative, transparent process for analyzing potential security vulnerabilities and seeing them through successful remediation. Whenever a potential issue is discovered, we immediately create an Issue entry for it in this document, even though we have not yet verified the feasibility and impact of the issue. This process is conservative because we document our suspicions early even if they are later shown to not represent exploitable vulnerabilities. We generally follow a process of first documenting the suspicion with unresolved questions, then confirming the issue through code analysis, live experimentation, or automated tests. Code analysis is the most tentative, and we strive to provide test code, log captures, or screenshots demonstrating our confirmation. After this we analyze the feasibility of an attack in a live system.

Suggested Solutions:

We search for immediate mitigations that live deployments can take, and finally we suggest the requirements for remediation engineering for future releases. The mitigation and remediation recommendations should be scrutinized by the developers and deployment engineers, and successful mitigation and remediation is an ongoing collaborative process after we deliver our report, and before the details are made public.

Disclaimers

EtherAuthority.io Disclaimer

EtherAuthority team has analyzed this smart contract in accordance with the best industry practices at the date of this report, in relation to: cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report, (Source Code); the Source Code compilation, deployment and functionality (performing the intended functions).

Due to the fact that the total number of test cases are unlimited, the audit makes no statements or warranties on security of the code. It also cannot be considered as a sufficient assessment regarding the utility and safety of the code, bugfree status or any other statements of the contract. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only. We also suggest conducting a bug bounty program to confirm the high level of security of this smart contract.

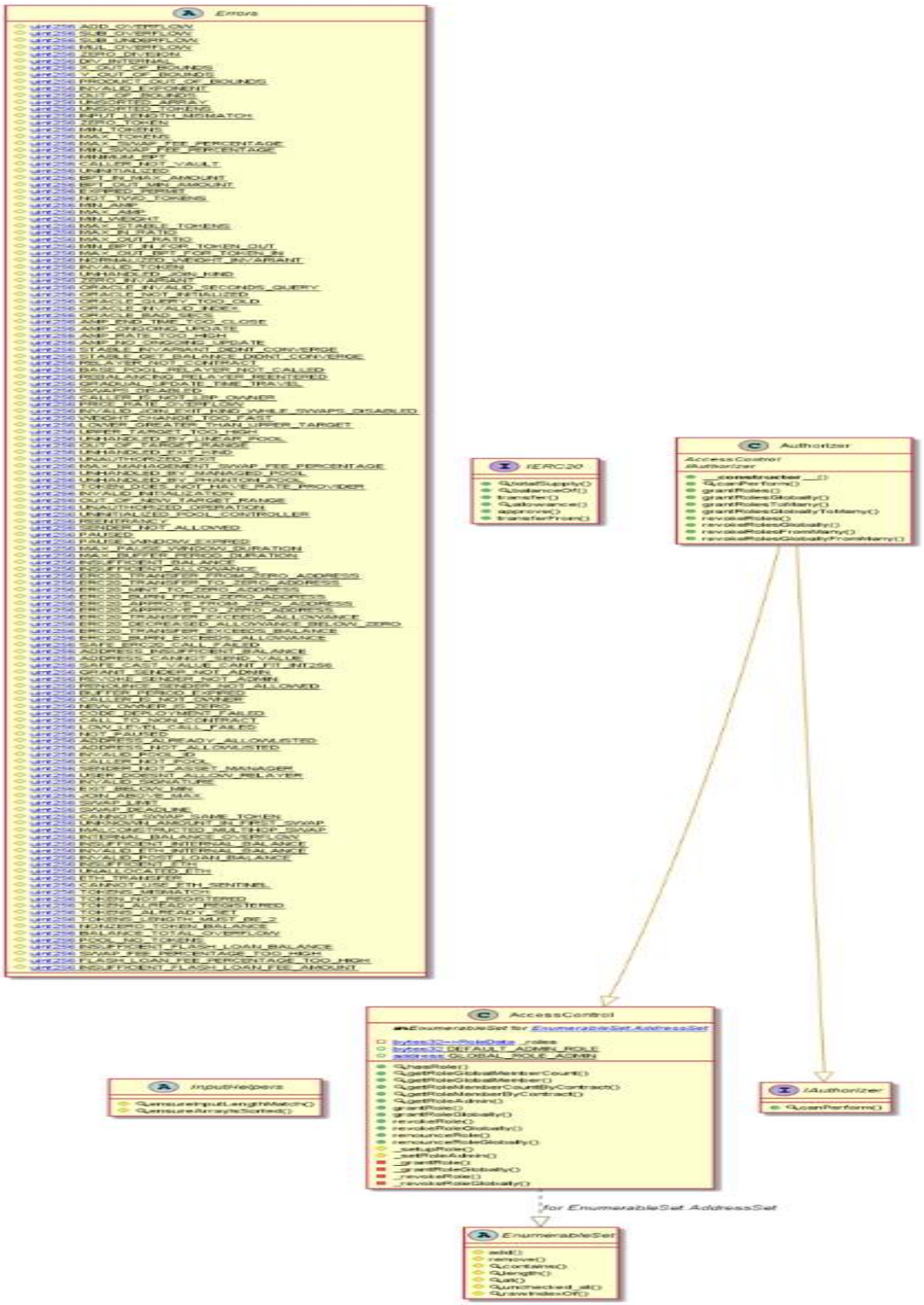
Technical Disclaimer

Smart contracts are deployed and executed on the blockchain platform. The platform, its programming language, and other software related to the smart contract can have their own vulnerabilities that can lead to hacks. Thus, the audit can't guarantee explicit security of the audited smart contracts.

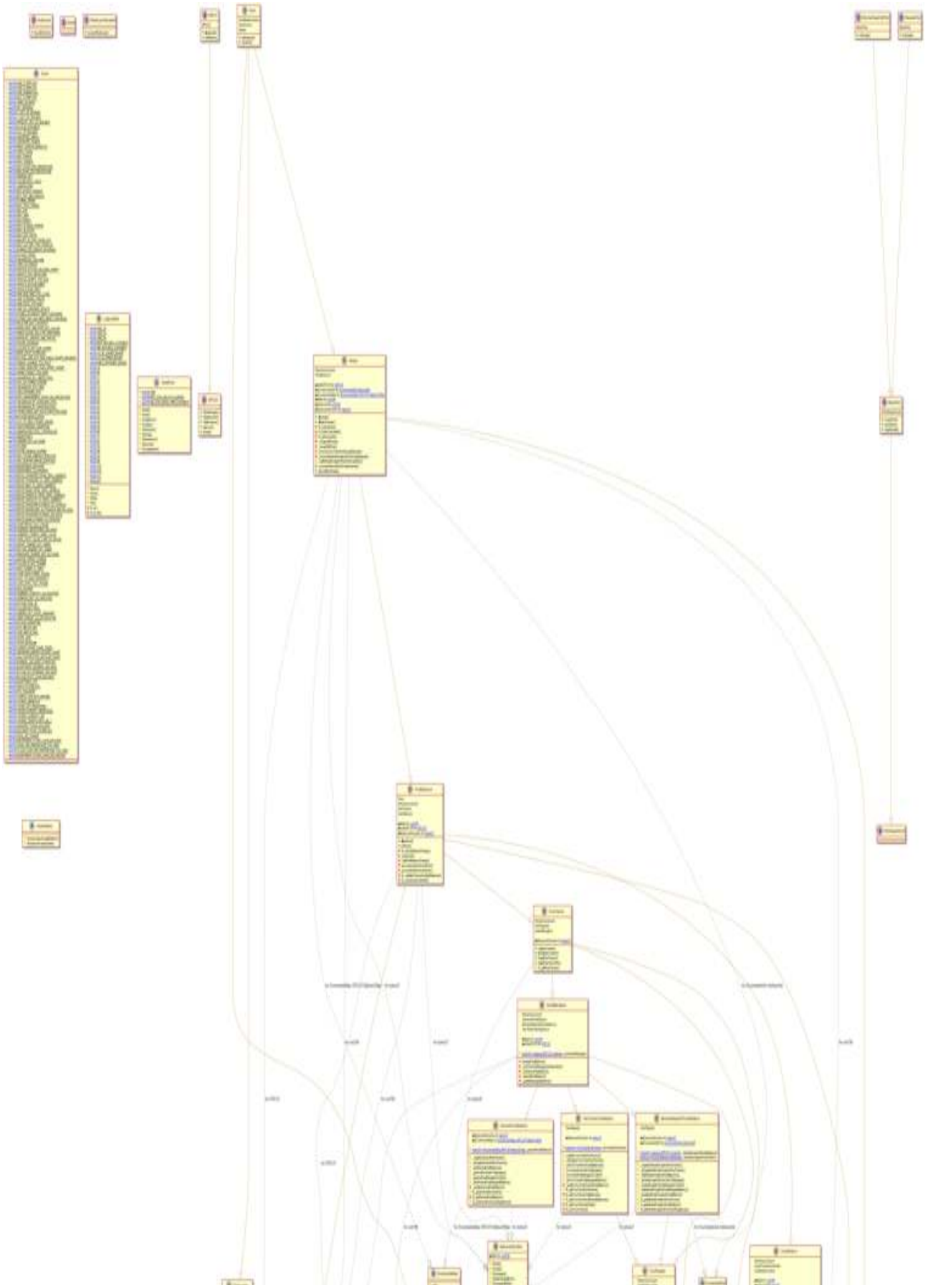
Appendix

Code Flow Diagram - Symmetric Protocol

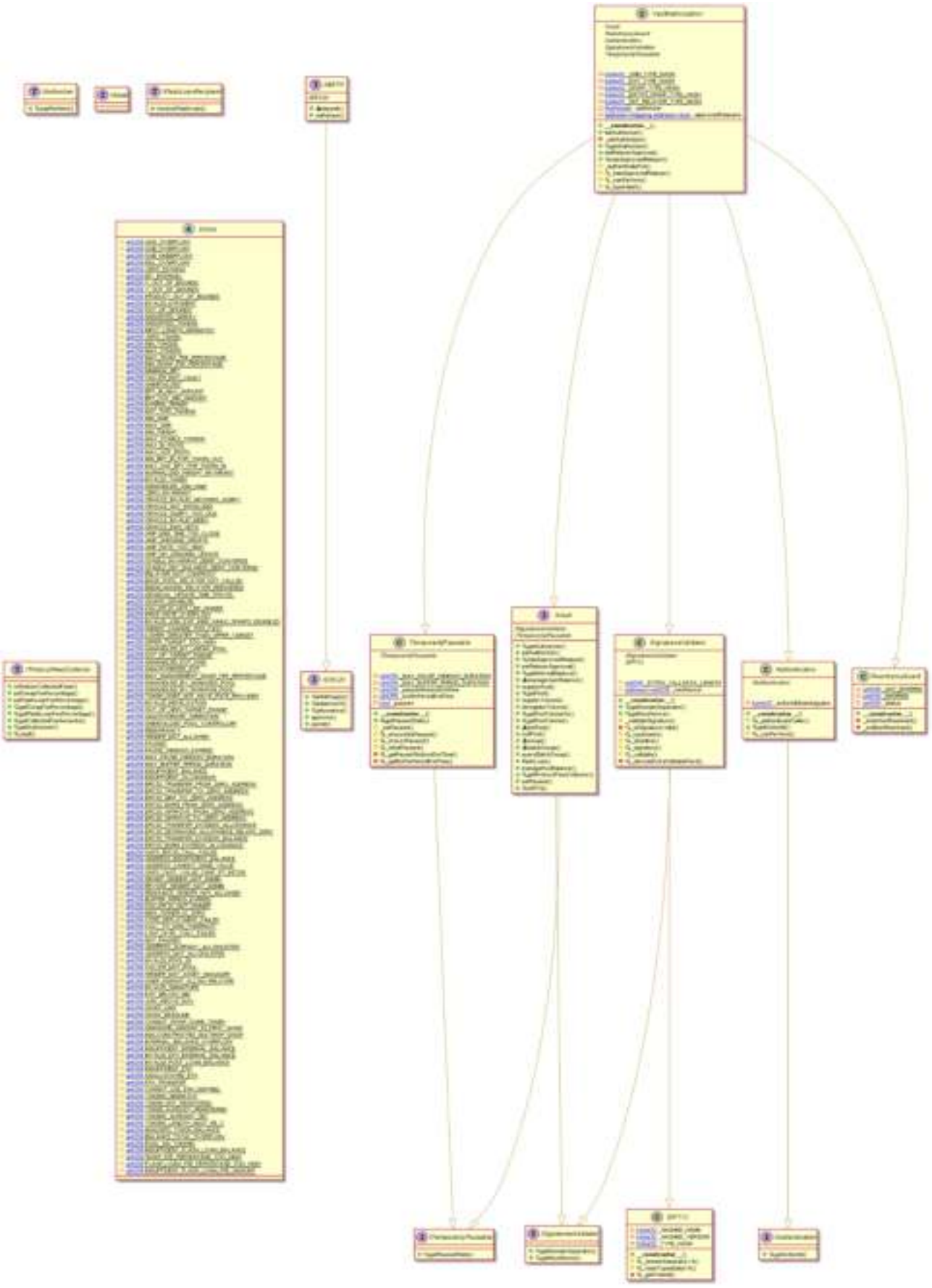
Authorizer Diagram



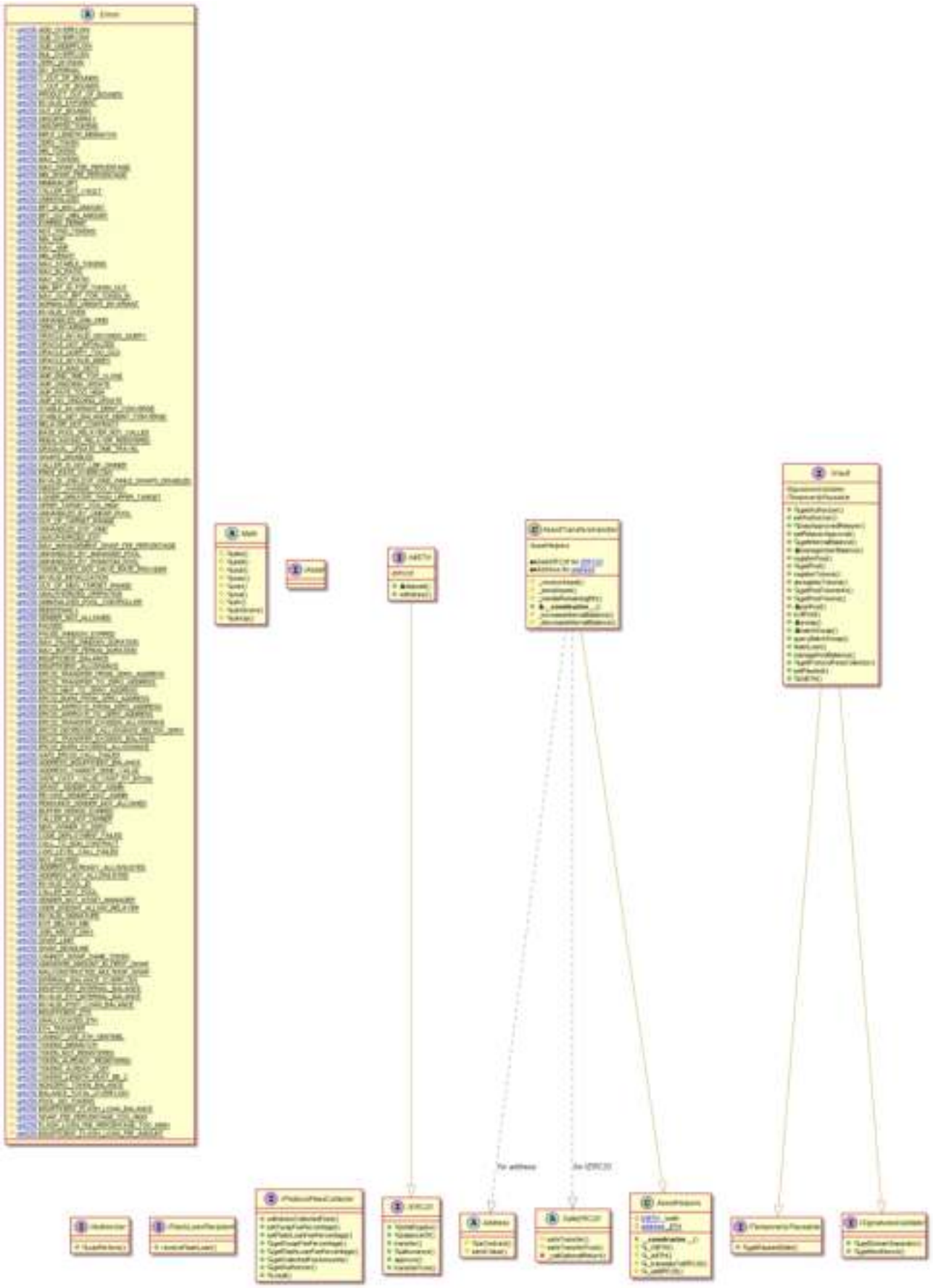
Vault Diagram



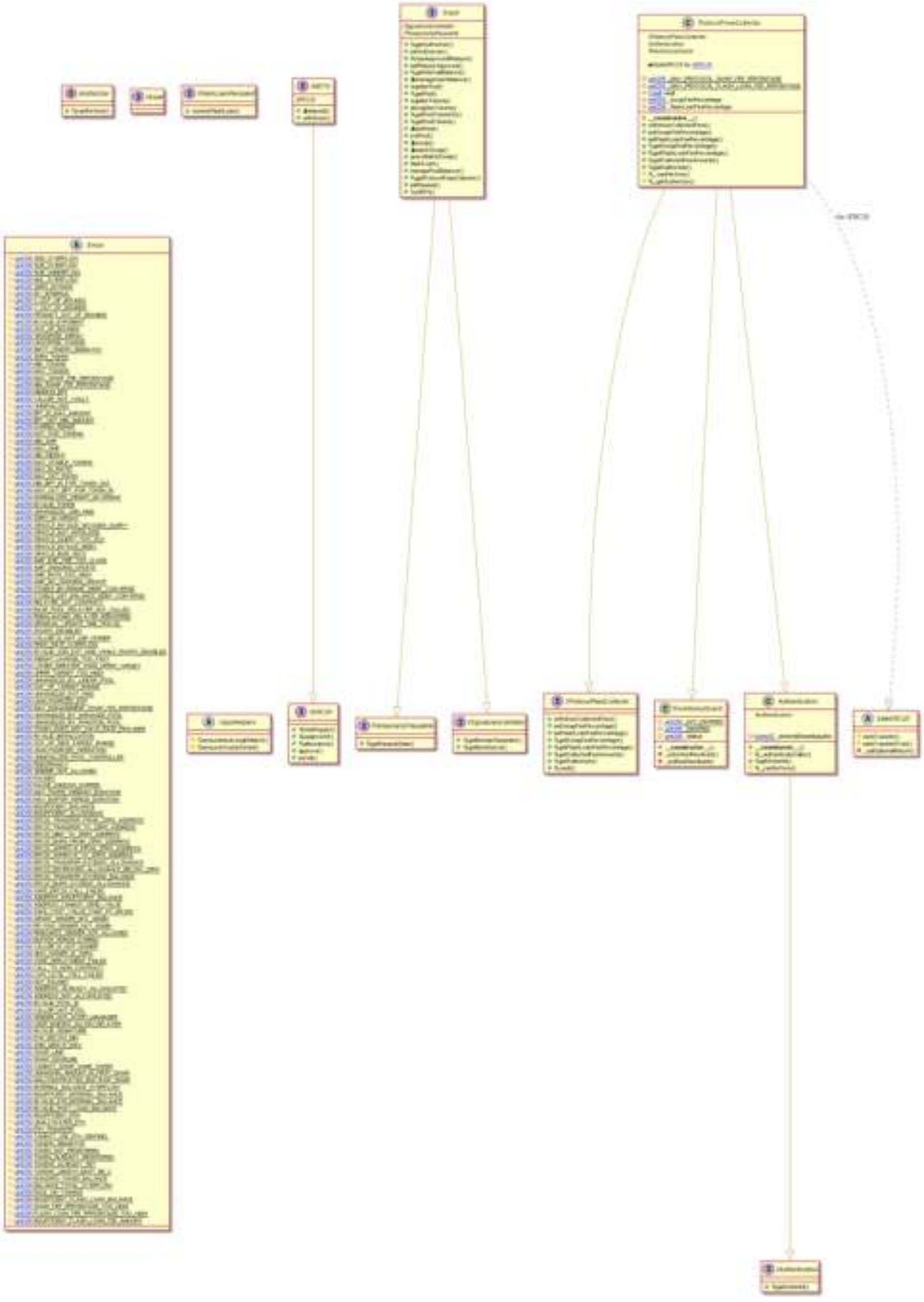
VaultAuthorization Diagram



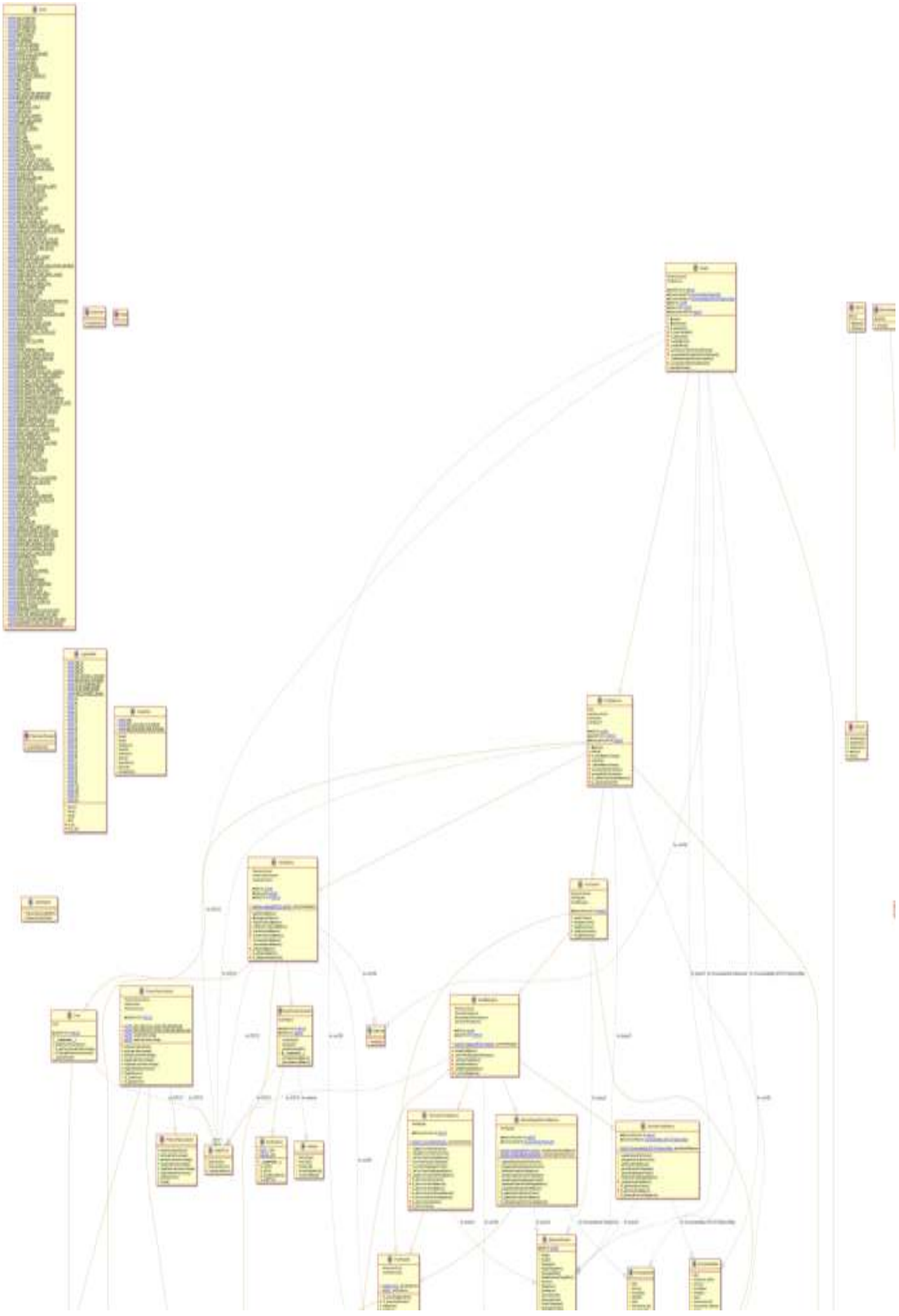
AssetTransfersHandler Diagram



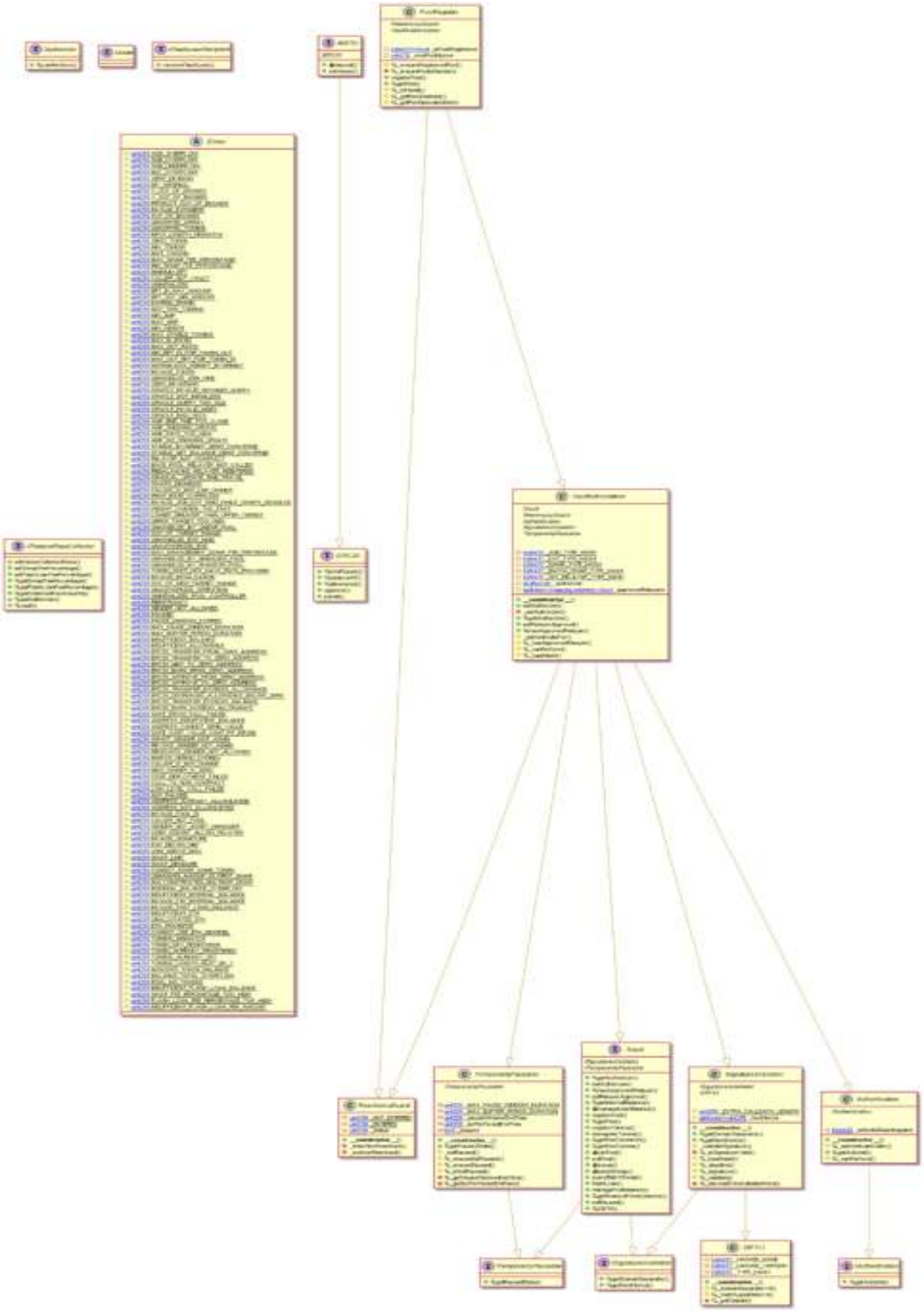
ProtocolFeesCollector Diagram



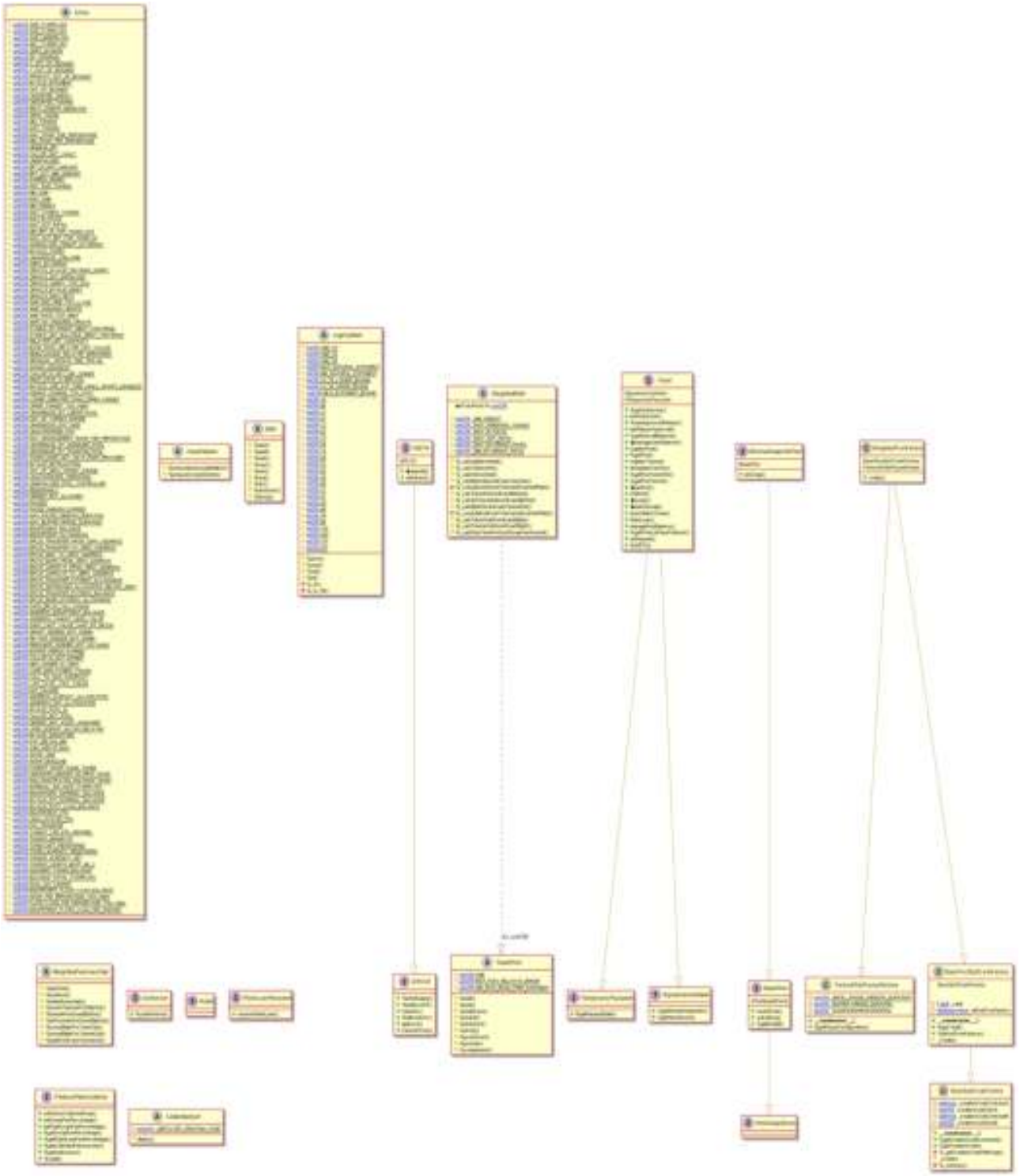
Swaps Diagram



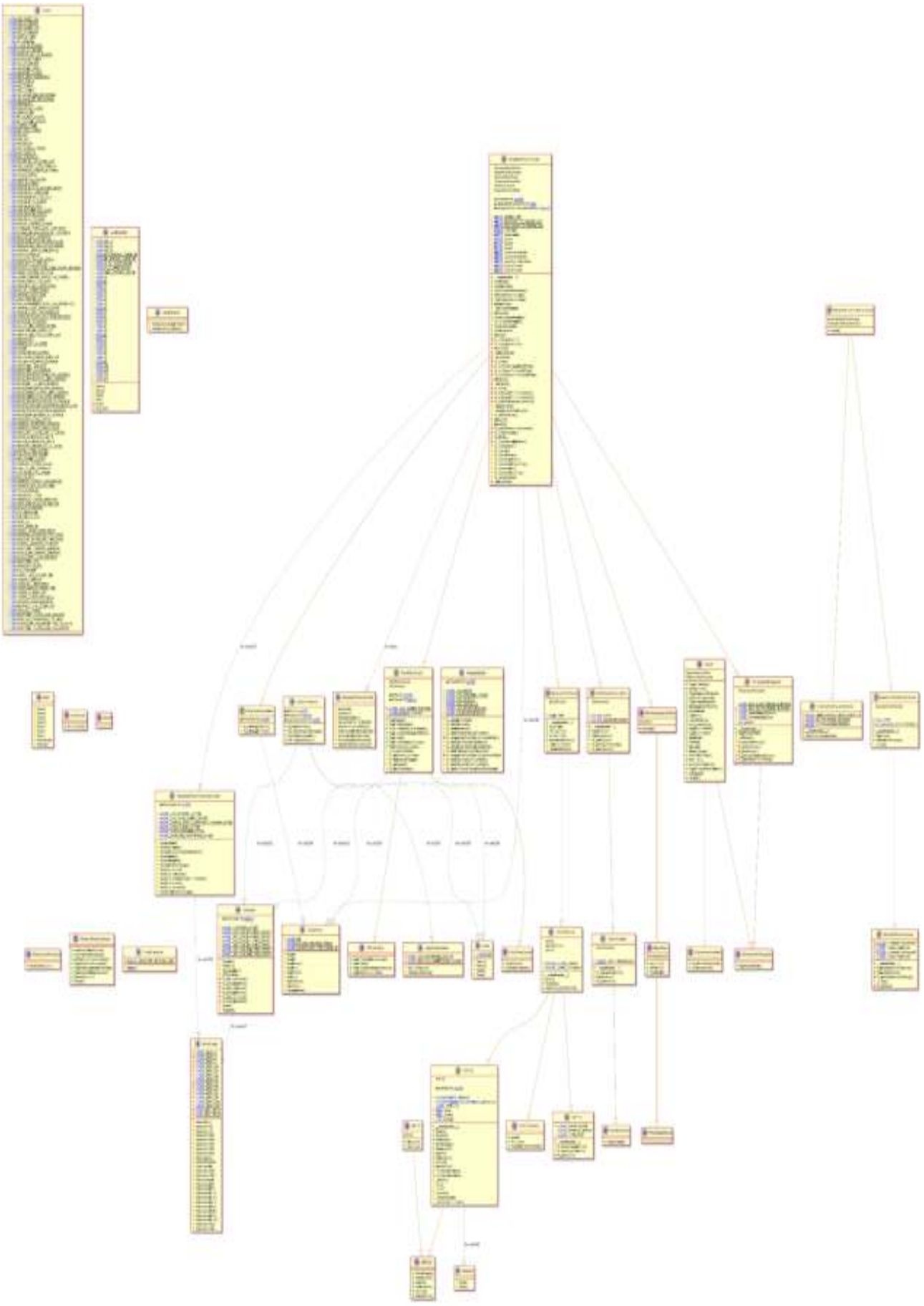
PoolRegistry Diagram



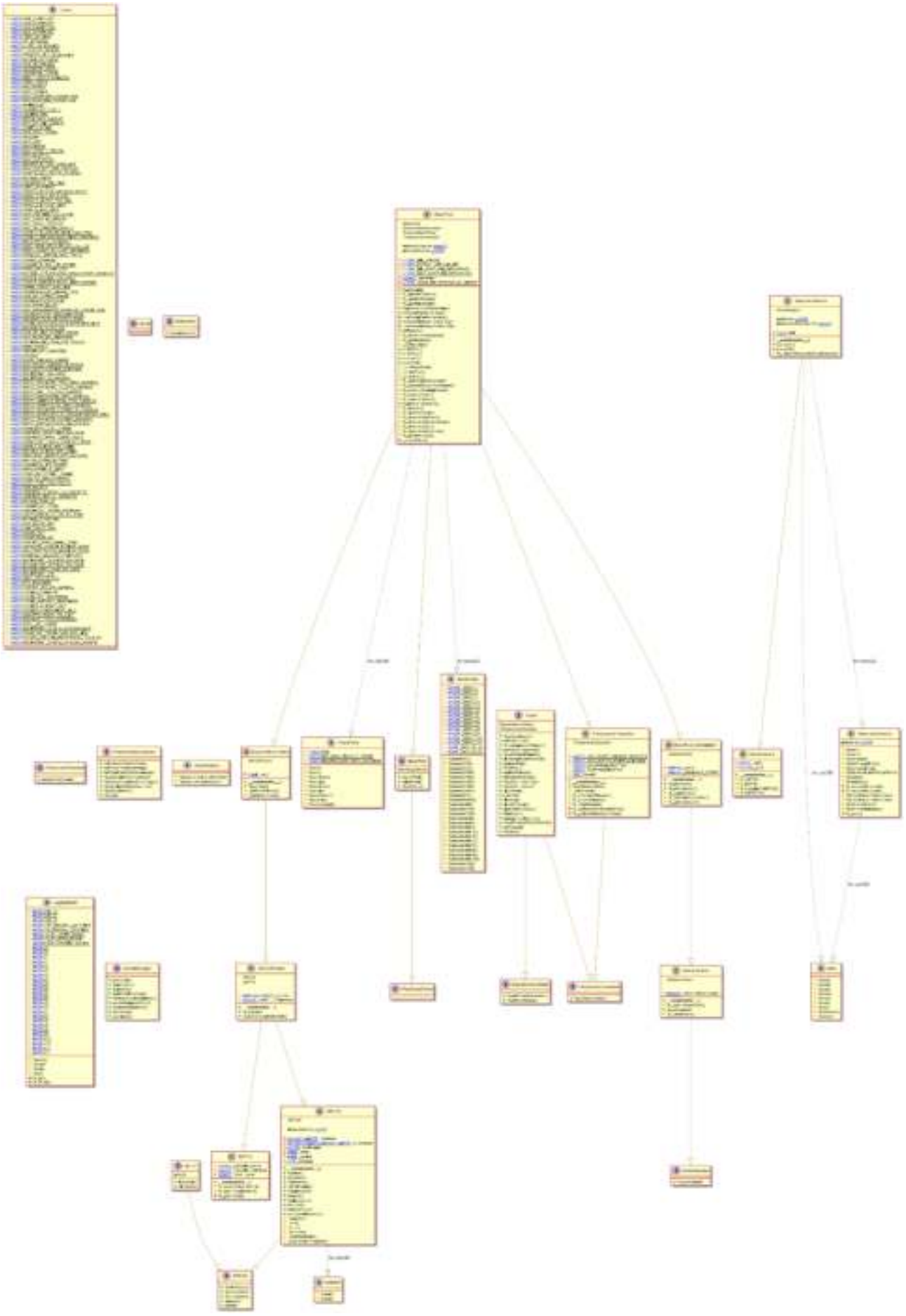
WeightedPoolFactory Diagram



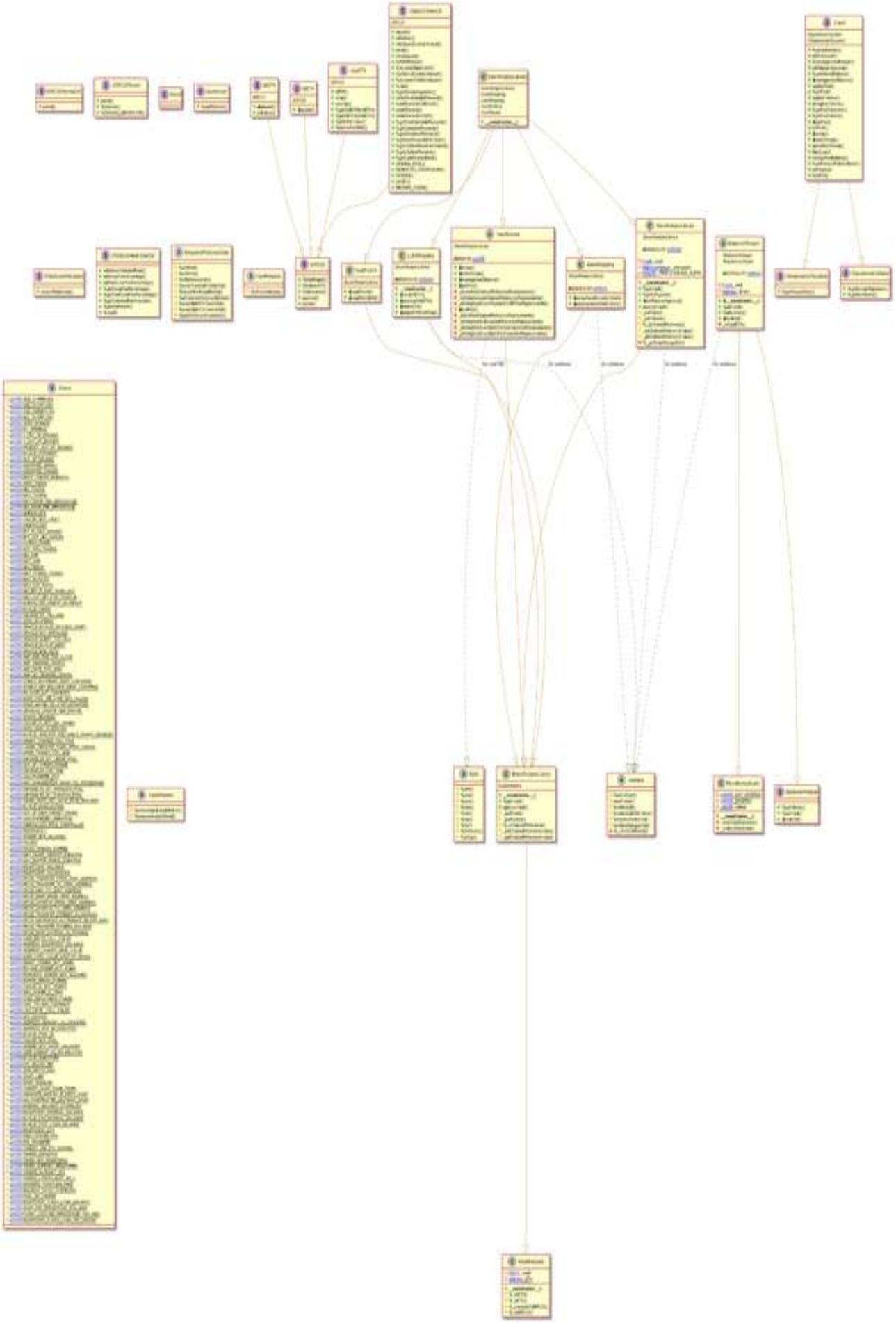
WeightedPool2TokenFactory Diagram



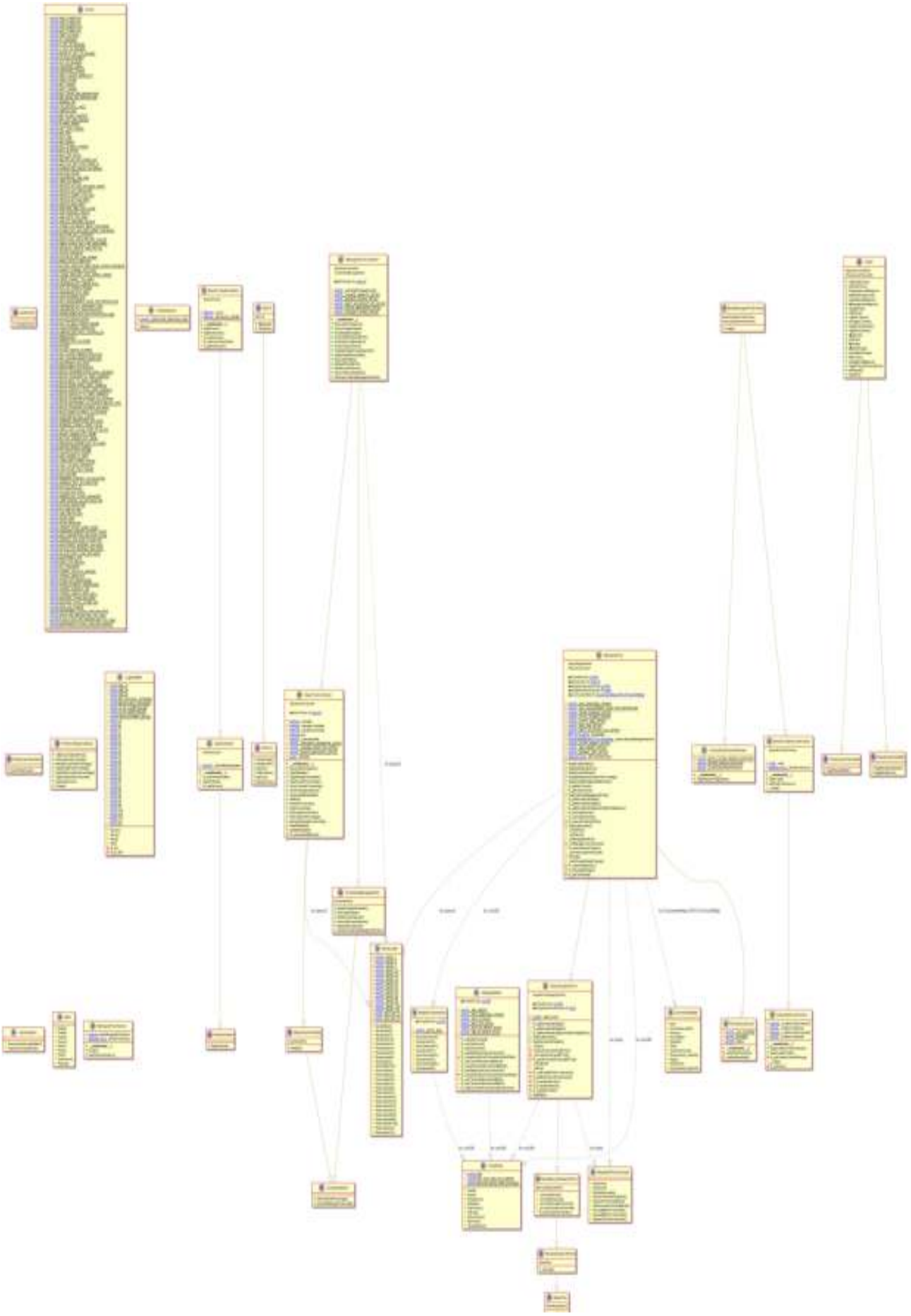
BalancerHelpers Diagram



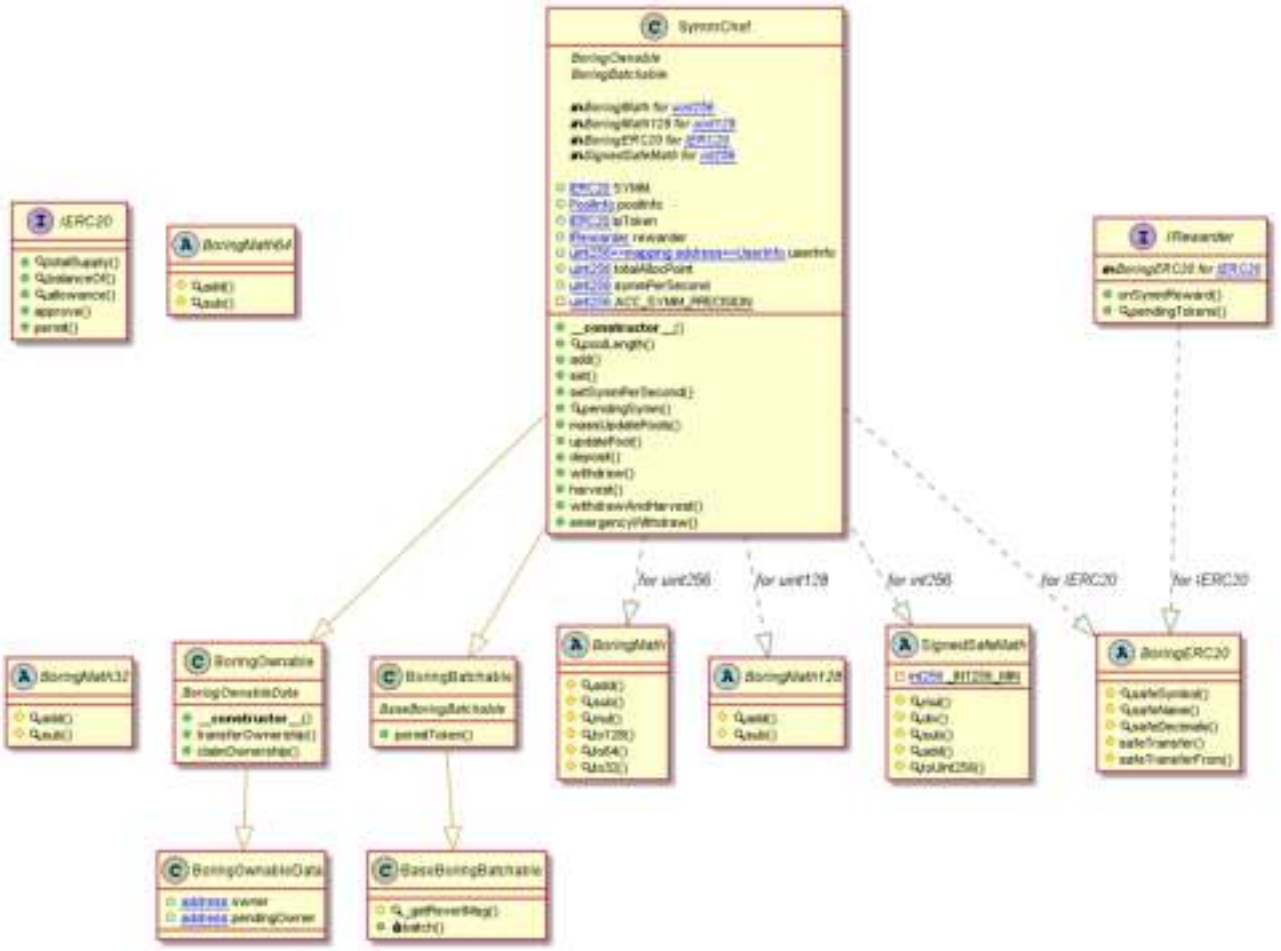
BatchRelayerLibrary Diagram



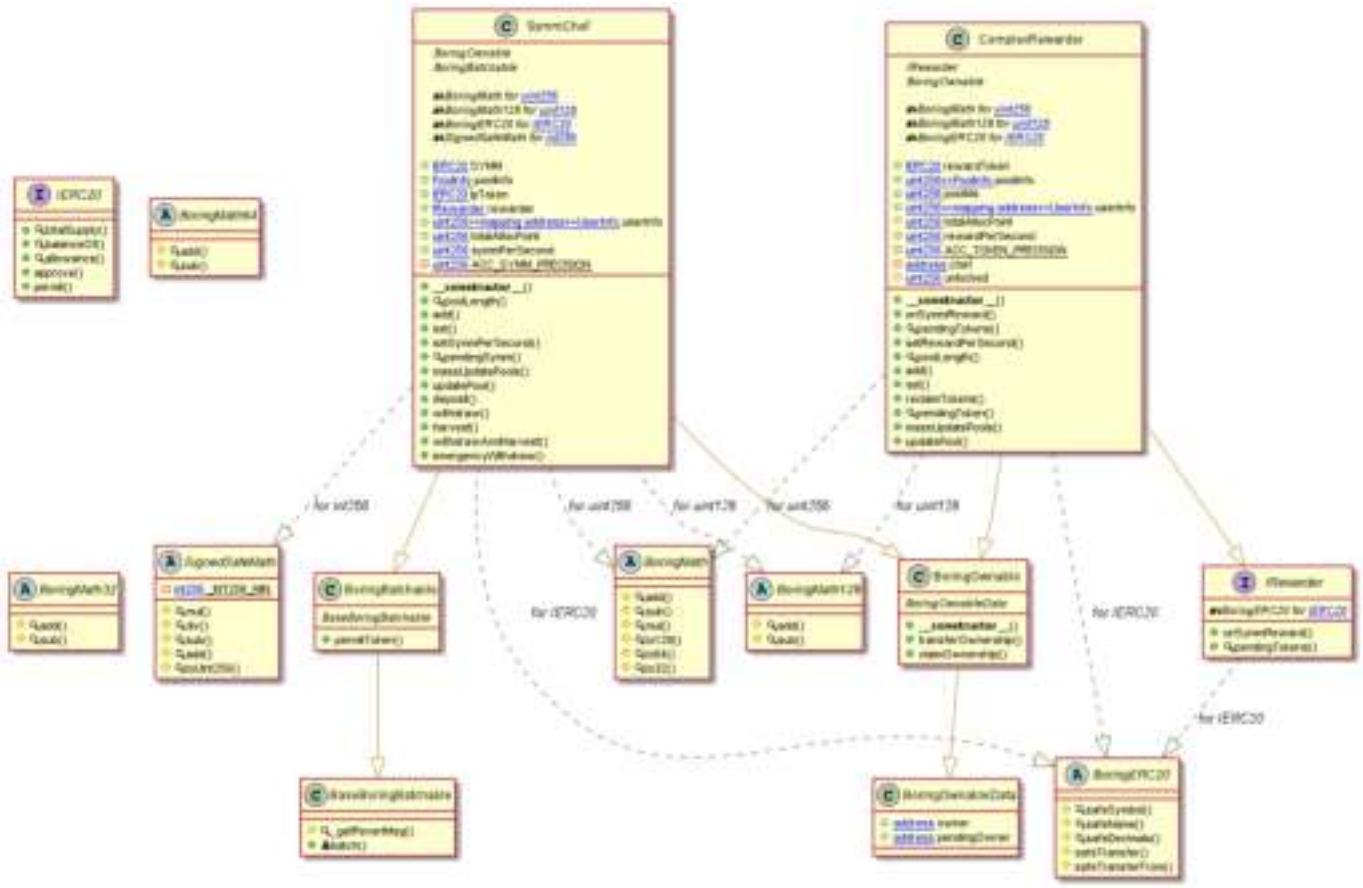
ManagedPoolFactory Diagram



SymmChef Diagram



ComplexRewarder Diagram

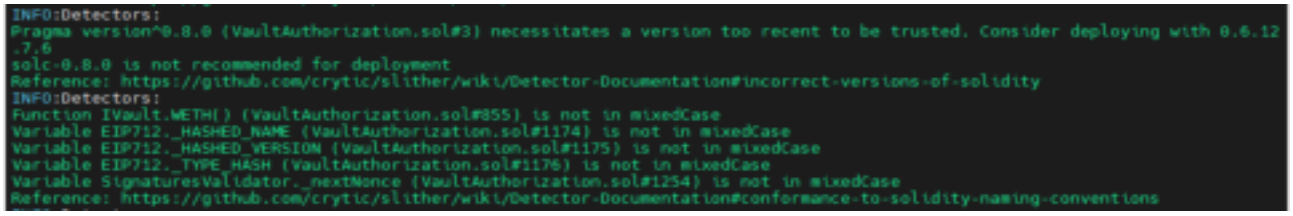


Slither Results Log

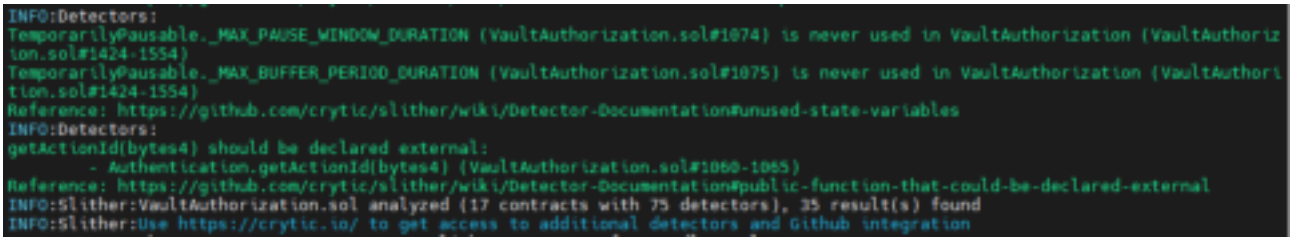
+Slither log >> Authorizer.sol

```
INFO:Detectors:
getRoleGlobalMemberCount(bytes32) should be declared external:
        - AccessControl.getRoleGlobalMemberCount(bytes32) (Authorizer.sol#306-308)
getRoleGlobalMember(bytes32,uint256) should be declared external:
        - AccessControl.getRoleGlobalMember(bytes32,uint256) (Authorizer.sol#310-312)
getRoleMemberCountByContract(bytes32,address) should be declared external:
        - AccessControl.getRoleMemberCountByContract(bytes32,address) (Authorizer.sol#314-316)
getRoleMemberByContract(bytes32,uint256,address) should be declared external:
        - AccessControl.getRoleMemberByContract(bytes32,uint256,address) (Authorizer.sol#318-320)
getRoleAdmin(bytes32) should be declared external:
        - AccessControl.getRoleAdmin(bytes32) (Authorizer.sol#322-324)
renounceRole(bytes32,address,address[]) should be declared external:
        - AccessControl.renounceRole(bytes32,address,address[]) (Authorizer.sol#348-351)
renounceRoleGlobally(bytes32,address) should be declared external:
        - AccessControl.renounceRoleGlobally(bytes32,address) (Authorizer.sol#353-356)
canPerform(bytes32,address,address) should be declared external:
        - Authorizer.canPerform(bytes32,address,address) (Authorizer.sol#470-476)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:Authorizer.sol analyzed (7 contracts with 75 detectors), 27 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> VaultAuthorization.sol

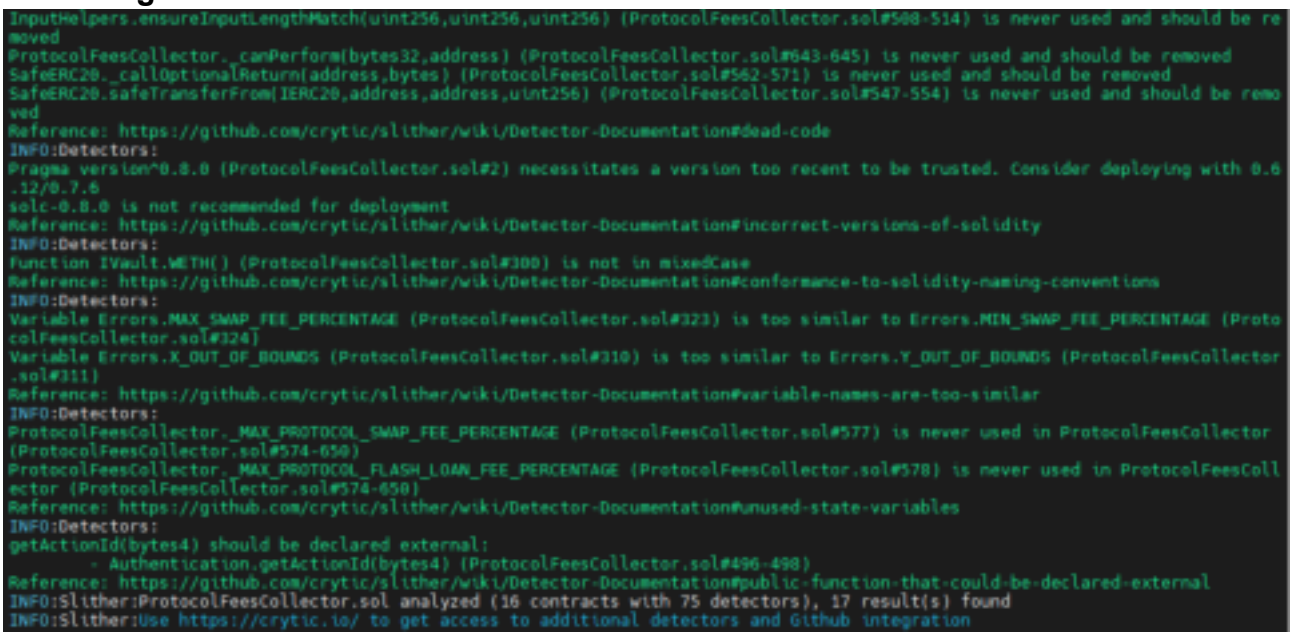


```
INFO:Detectors:
Pragma version^0.8.0 (VaultAuthorization.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.12
.7.6
solc-0.8.0 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Function IVault.WETH() (VaultAuthorization.sol#855) is not in mixedCase
Variable EIP712._HASHED_NAME (VaultAuthorization.sol#1174) is not in mixedCase
Variable EIP712._HASHED_VERSION (VaultAuthorization.sol#1175) is not in mixedCase
Variable EIP712._TYPE_HASH (VaultAuthorization.sol#1176) is not in mixedCase
Variable SignaturesValidator._nextNonce (VaultAuthorization.sol#1254) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```



```
INFO:Detectors:
TemporarilyPausable._MAX_PAUSE_WINDOW_DURATION (VaultAuthorization.sol#1074) is never used in VaultAuthorization (VaultAuthoriz
ion.sol#1424-1554)
TemporarilyPausable._MAX_BUFFER_PERIOD_DURATION (VaultAuthorization.sol#1075) is never used in VaultAuthorization (VaultAuthori
tion.sol#1424-1554)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variables
INFO:Detectors:
getActionId(bytes4) should be declared external:
        - Authentication.getActionId(bytes4) (VaultAuthorization.sol#1060-1065)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:VaultAuthorization.sol analyzed (17 contracts with 75 detectors), 35 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> ProtocolFeesCollector.sol



```
InputHelpers.ensureInputLengthMatch(uint256,uint256,uint256) (ProtocolFeesCollector.sol#508-514) is never used and should be re
moved
ProtocolFeesCollector._canPerform(bytes32,address) (ProtocolFeesCollector.sol#643-645) is never used and should be removed
SafeERC20._callOptionalReturn(address,bytes) (ProtocolFeesCollector.sol#562-571) is never used and should be removed
SafeERC20.safeTransferFrom(IERC20,address,address,uint256) (ProtocolFeesCollector.sol#547-554) is never used and should be remo
ved
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.0 (ProtocolFeesCollector.sol#2) necessitates a version too recent to be trusted. Consider deploying with 0.6
.12/0.7.6
solc-0.8.0 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Function IVault.WETH() (ProtocolFeesCollector.sol#300) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable Errors.MAX_SWAP_FEE_PERCENTAGE (ProtocolFeesCollector.sol#323) is too similar to Errors.MIN_SWAP_FEE_PERCENTAGE (Proto
colFeesCollector.sol#324)
Variable Errors.X_OUT_OF_BOUNDS (ProtocolFeesCollector.sol#310) is too similar to Errors.Y_OUT_OF_BOUNDS (ProtocolFeesCollector
.sol#311)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
ProtocolFeesCollector._MAX_PROTOCOL_SWAP_FEE_PERCENTAGE (ProtocolFeesCollector.sol#577) is never used in ProtocolFeesCollector
(ProtocolFeesCollector.sol#574-650)
ProtocolFeesCollector._MAX_PROTOCOL_FLASH_LOAN_FEE_PERCENTAGE (ProtocolFeesCollector.sol#578) is never used in ProtocolFeesColl
ector (ProtocolFeesCollector.sol#574-650)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variables
INFO:Detectors:
getActionId(bytes4) should be declared external:
        - Authentication.getActionId(bytes4) (ProtocolFeesCollector.sol#496-498)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:ProtocolFeesCollector.sol analyzed (16 contracts with 75 detectors), 17 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> PoolRegistry.sol

```
INFO:Detectors:
PoolRegistry._nextPoolNonce (PoolRegistry.sol#1543) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
getActionId(bytes4) should be declared external:
        - Authentication.getActionId(bytes4) (PoolRegistry.sol#1390-1395)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:PoolRegistry.sol analyzed (18 contracts with 75 detectors), 40 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> SymmChef.sol

```
INFO:Detectors:
transferOwnership(address,bool,bool) should be declared external:
        - BoringOwnable.transferOwnership(address,bool,bool) (SymmChef.sol#139-151)
claimOwnership() should be declared external:
        - BoringOwnable.claimOwnership() (SymmChef.sol#154-163)
permitToken(IERC20,address,address,uint256,uint256,uint8,bytes32,bytes32) should be declared external:
        - BoringBatchable.permitToken(IERC20,address,address,uint256,uint256,uint8,bytes32,bytes32) (SymmChef.sol#206-210)
poolLength() should be declared external:
        - SymmChef.poolLength() (SymmChef.sol#301-303)
add(uint256,IERC20,IRewarder) should be declared external:
        - SymmChef.add(uint256,IERC20,IRewarder) (SymmChef.sol#310-321)
set(uint256,uint256,IRewarder,bool) should be declared external:
        - SymmChef.set(uint256,uint256,IRewarder,bool) (SymmChef.sol#328-333)
setSymmPerSecond(uint256) should be declared external:
        - SymmChef.setSymmPerSecond(uint256) (SymmChef.sol#337-340)
deposit(uint256,uint256,address) should be declared external:
        - SymmChef.deposit(uint256,uint256,address) (SymmChef.sol#391-408)
withdraw(uint256,uint256,address) should be declared external:
        - SymmChef.withdraw(uint256,uint256,address) (SymmChef.sol#414-431)
harvest(uint256,address) should be declared external:
        - SymmChef.harvest(uint256,address) (SymmChef.sol#436-456)
withdrawAndHarvest(uint256,uint256,address) should be declared external:
        - SymmChef.withdrawAndHarvest(uint256,uint256,address) (SymmChef.sol#462-484)
emergencyWithdraw(uint256,address) should be declared external:
        - SymmChef.emergencyWithdraw(uint256,address) (SymmChef.sol#489-503)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:SymmChef.sol analyzed (13 contracts with 75 detectors), 49 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> ComplexRewarder.sol

```
INFO:Detectors:
transferOwnership(address,bool,bool) should be declared external:
        - BoringOwnable.transferOwnership(address,bool,bool) (ComplexRewarder.sol#138-150)
claimOwnership() should be declared external:
        - BoringOwnable.claimOwnership() (ComplexRewarder.sol#153-162)
permitToken(IERC20,address,address,uint256,uint256,uint8,bytes32,bytes32) should be declared external:
        - BoringBatchable.permitToken(IERC20,address,address,uint256,uint256,uint8,bytes32,bytes32) (ComplexRewarder.sol#206-21
0)
poolLength() should be declared external:
        - SymmChef.poolLength() (ComplexRewarder.sol#301-303)
add(uint256,IERC20,IRewarder) should be declared external:
        - SymmChef.add(uint256,IERC20,IRewarder) (ComplexRewarder.sol#310-321)
set(uint256,uint256,IRewarder,bool) should be declared external:
        - SymmChef.set(uint256,uint256,IRewarder,bool) (ComplexRewarder.sol#328-333)
setSymmPerSecond(uint256) should be declared external:
        - SymmChef.setSymmPerSecond(uint256) (ComplexRewarder.sol#337-340)
deposit(uint256,uint256,address) should be declared external:
        - SymmChef.deposit(uint256,uint256,address) (ComplexRewarder.sol#391-408)
withdraw(uint256,uint256,address) should be declared external:
        - SymmChef.withdraw(uint256,uint256,address) (ComplexRewarder.sol#414-431)
harvest(uint256,address) should be declared external:
        - SymmChef.harvest(uint256,address) (ComplexRewarder.sol#436-456)
withdrawAndHarvest(uint256,uint256,address) should be declared external:
        - SymmChef.withdrawAndHarvest(uint256,uint256,address) (ComplexRewarder.sol#462-484)
emergencyWithdraw(uint256,address) should be declared external:
        - SymmChef.emergencyWithdraw(uint256,address) (ComplexRewarder.sol#489-503)
setRewardPerSecond(uint256) should be declared external:
        - ComplexRewarder.setRewardPerSecond(uint256) (ComplexRewarder.sol#603-606)
poolLength() should be declared external:
        - ComplexRewarder.poolLength() (ComplexRewarder.sol#617-619)
add(uint256,uint256) should be declared external:
        - ComplexRewarder.add(uint256,uint256) (ComplexRewarder.sol#625-637)
set(uint256,uint256) should be declared external:
        - ComplexRewarder.set(uint256,uint256) (ComplexRewarder.sol#642-646)
```

```
set(uint256,uint256) should be declared external:
        - ComplexRewarder.set(uint256,uint256) (ComplexRewarder.sol#642-646)
reclaimTokens(address,uint256,address) should be declared external:
        - ComplexRewarder.reclaimTokens(address,uint256,address) (ComplexRewarder.sol#654-660)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:ComplexRewarder.sol analyzed (14 contracts with 75 detectors), 73 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Vault.sol

```
INFO:Detectors:
ProtocolFeesCollector._MAX_PROTOCOL_SWAP_FEE_PERCENTAGE (Vault.sol#1441) is never used in ProtocolFeesCollector (Vault.sol#1438-
5)
ProtocolFeesCollector._MAX_PROTOCOL_FLASH_LOAN_FEE_PERCENTAGE (Vault.sol#1442) is never used in ProtocolFeesCollector (Vault.sol
38-1505)
TemporarilyPausable._MAX_PAUSE_WINDOW_DURATION (Vault.sol#462) is never used in Vault (Vault.sol#2853-2864)
TemporarilyPausable._MAX_BUFFER_PERIOD_DURATION (Vault.sol#463) is never used in Vault (Vault.sol#2853-2864)
PoolRegistry._isPoolRegistered (Vault.sol#927) is never used in Vault (Vault.sol#2853-2864)
PoolRegistry._nextPoolNonce (Vault.sol#929) is never used in Vault (Vault.sol#2853-2864)
MinimalSwapInfoPoolsBalance._minimalSwapInfoPoolsTokens (Vault.sol#1752) is never used in Vault (Vault.sol#2853-2864)
TwoTokenPoolsBalance._twoTokenPoolTokens (Vault.sol#2026) is never used in Vault (Vault.sol#2853-2864)
AssetManagers._poolAssetManagers (Vault.sol#2195) is never used in Vault (Vault.sol#2853-2864)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variables
INFO:Detectors:
PoolRegistry._nextPoolNonce (Vault.sol#929) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Slither:Vault.sol analyzed (44 contracts with 75 detectors), 170 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> AssetTransfersHandler.sol

```
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.0 (AssetTransfersHandler.sol#4) necessitates a version too recent to be trusted. Consider deploying with 0.6.
12/0.7.6
solc-0.8.0 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Function IVault.WETH() (AssetTransfersHandler.sol#586) is not in mixedCase
Function AssetHelpers._WETH() (AssetTransfersHandler.sol#578-580) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Variable Errors.MAX_SWAP_FEE_PERCENTAGE (AssetTransfersHandler.sol#28) is too similar to Errors.MIN_SWAP_FEE_PERCENTAGE (AssetTr
ansfersHandler.sol#29)
Variable Errors.X_OUT_OF_BOUNDS (AssetTransfersHandler.sol#15) is too similar to Errors.Y_OUT_OF_BOUNDS (AssetTransfersHandler.s
ol#16)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Detectors:
AssetTransfersHandler (AssetTransfersHandler.sol#603-685) does not implement functions:
        - AssetHelpers._asIERC20(IAsset) (AssetTransfersHandler.sol#598-600)
        - AssetTransfersHandler._decreaseInternalBalance(address,IERC20,uint256,bool) (AssetTransfersHandler.sol#679-684)
        - AssetTransfersHandler._increaseInternalBalance(address,IERC20,uint256) (AssetTransfersHandler.sol#673-677)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unimplemented-functions
INFO:Slither:AssetTransfersHandler.sol analyzed (15 contracts with 75 detectors), 34 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> Swaps.sol

```
INFO:Detectors:
ProtocolFeesCollector._MAX_PROTOCOL_SWAP_FEE_PERCENTAGE (Swaps.sol#1441) is never used in ProtocolFeesCollector (Swaps.sol#1438-
1505)
ProtocolFeesCollector._MAX_PROTOCOL_FLASH_LOAN_FEE_PERCENTAGE (Swaps.sol#1442) is never used in ProtocolFeesCollector (Swaps.sol
#1438-1505)
TemporarilyPausable._MAX_PAUSE_WINDOW_DURATION (Swaps.sol#462) is never used in Vault (Swaps.sol#2853-2864)
TemporarilyPausable._MAX_BUFFER_PERIOD_DURATION (Swaps.sol#463) is never used in Vault (Swaps.sol#2853-2864)
TemporarilyPausable._MAX_PAUSE_WINDOW_DURATION (Swaps.sol#462) is never used in Swaps (Swaps.sol#2577-2848)
TemporarilyPausable._MAX_BUFFER_PERIOD_DURATION (Swaps.sol#463) is never used in Swaps (Swaps.sol#2577-2848)
PoolRegistry._isPoolRegistered (Swaps.sol#927) is never used in Swaps (Swaps.sol#2577-2848)
PoolRegistry._nextPoolNonce (Swaps.sol#929) is never used in Swaps (Swaps.sol#2577-2848)
MinimalSwapInfoPoolsBalance._minimalSwapInfoPoolsTokens (Swaps.sol#1505) is never used in Swaps (Swaps.sol#2577-2848)
TwoTokenPoolsBalance._twoTokenPoolTokens (Swaps.sol#1808) is never used in Swaps (Swaps.sol#2577-2848)
AssetManagers._poolAssetManagers (Swaps.sol#1977) is never used in Swaps (Swaps.sol#2577-2848)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variables
INFO:Detectors:
PoolRegistry._nextPoolNonce (Swaps.sol#929) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Slither:Swaps.sol analyzed (48 contracts with 75 detectors), 176 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> WeightedPoolFactory.sol

```
INFO:Detectors:
InputHelpers.ensureArrayIsSorted(IERC20[]) (WeightedPoolFactory.sol#246-253) uses assembly
        - INLINE ASM (WeightedPoolFactory.sol#249-251)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
InputHelpers.ensureArrayIsSorted(IERC20[]) (WeightedPoolFactory.sol#246-253) is never used and should be removed
InputHelpers.ensureArrayIsSorted(address[]) (WeightedPoolFactory.sol#255-266) is never used and should be removed
InputHelpers.ensureInputLengthMatch(uint256,uint256) (WeightedPoolFactory.sol#234-236) is never used and should be removed
InputHelpers.ensureInputLengthMatch(uint256,uint256,uint256) (WeightedPoolFactory.sol#238-244) is never used and should be rem
oved
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.0 (WeightedPoolFactory.sol#3) necessitates a version too recent to be trusted. Consider deploying with 0.6.
12/0.7.6
solc-0.8.0 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Variable Errors.MAX_SWAP_FEE_PERCENTAGE (WeightedPoolFactory.sol#103) is too similar to Errors.MIN_SWAP_FEE_PERCENTAGE (Weight
edPoolFactory.sol#104)
Variable Errors.X_OUT_OF_BOUNDS (WeightedPoolFactory.sol#88) is too similar to Errors.Y_OUT_OF_BOUNDS (WeightedPoolFactory.sol
#89)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#variable-names-are-too-similar
INFO:Slither:WeightedPoolFactory.sol analyzed (3 contracts with 75 detectors), 9 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> WeightedPool2TokenFactory.sol

```
INFO:Detectors:
getPastAccumulator(mapping(uint256 => bytes32),IPriceOracle.Variable,uint256,uint256) should be declared external:
        - QueryProcessor.getPastAccumulator(mapping(uint256 => bytes32),IPriceOracle.Variable,uint256,uint256) (WeightedPool2T
okenFactory.sol#521-571)
name() should be declared external:
        - ERC20.name() (WeightedPool2TokenFactory.sol#1154-1156)
symbol() should be declared external:
        - ERC20.symbol() (WeightedPool2TokenFactory.sol#1158-1160)
decimals() should be declared external:
        - ERC20.decimals() (WeightedPool2TokenFactory.sol#1162-1164)
totalSupply() should be declared external:
        - ERC20.totalSupply() (WeightedPool2TokenFactory.sol#1166-1168)
balanceOf(address) should be declared external:
        - ERC20.balanceOf(address) (WeightedPool2TokenFactory.sol#1170-1172)
transfer(address,uint256) should be declared external:
        - ERC20.transfer(address,uint256) (WeightedPool2TokenFactory.sol#1174-1177)
allowance(address,address) should be declared external:
        - ERC20.allowance(address,address) (WeightedPool2TokenFactory.sol#1179-1181)
approve(address,uint256) should be declared external:
        - ERC20.approve(address,uint256) (WeightedPool2TokenFactory.sol#1183-1186)
increaseAllowance(address,uint256) should be declared external:
        - ERC20.increaseAllowance(address,uint256) (WeightedPool2TokenFactory.sol#1189-1192)
nonces(address) should be declared external:
        - ERC20Permit.nonces(address) (WeightedPool2TokenFactory.sol#1254-1256)
getVault() should be declared external:
        - BalancerPoolToken.getVault() (WeightedPool2TokenFactory.sol#1274-1276)
getActionId(bytes4) should be declared external:
        - Authentication.getActionId(bytes4) (WeightedPool2TokenFactory.sol#1308-1310)
getPoolId() should be declared external:
        - WeightedPool2Tokens.getPoolId() (WeightedPool2TokenFactory.sol#1505-1507)
getSwapFeePercentage() should be declared external:
        - WeightedPool2Tokens.getSwapFeePercentage() (WeightedPool2TokenFactory.sol#1510-1512)
setSwapFeePercentage(uint256) should be declared external:
        - WeightedPool2Tokens.setSwapFeePercentage(uint256) (WeightedPool2TokenFactory.sol#1514-1516)
```

```
setSwapFeePercentage(uint256) should be declared external:
        - WeightedPool2Tokens.setSwapFeePercentage(uint256) (WeightedPool2TokenFactory.sol#1514-1516)
getRate() should be declared external:
        - WeightedPool2Tokens.getRate() (WeightedPool2TokenFactory.sol#1592-1594)
getCreationCodeContracts() should be declared external:
        - BaseSplitCodeFactory.getCreationCodeContracts() (WeightedPool2TokenFactory.sol#1785-1787)
getCreationCode() should be declared external:
        - BaseSplitCodeFactory.getCreationCode() (WeightedPool2TokenFactory.sol#1789-1791)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:WeightedPool2TokenFactory.sol analyzed (38 contracts with 75 detectors), 177 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> BatchRelayerLibrary.sol

```
INFO:Detectors:
BaseRelayerLibrary._TEMP_STORAGE_SUFFIX (BatchRelayerLibrary.sol#1441) is never used in BatchRelayerLibrary (BatchRelayerLibra
ry.sol#1450-1454)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variables
INFO:Detectors:
BaseRelayerLibrary._TEMP_STORAGE_SUFFIX (BatchRelayerLibrary.sol#1441) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
approveVault(IERC20,uint256) should be declared external:
        - BaseRelayerLibrary.approveVault(IERC20,uint256) (BatchRelayerLibrary.sol#1381-1383)
        - IBaseRelayerLibrary.approveVault(IERC20,uint256) (BatchRelayerLibrary.sol#871)
vaultPermit(IERC20Permit,address,uint256,uint256,uint8,bytes32,bytes32) should be declared external:
        - VaultPermit.vaultPermit(IERC20Permit,address,uint256,uint256,uint8,bytes32,bytes32) (BatchRelayerLibrary.sol#893-903
)
vaultPermitDAI(IERC20PermitDAI,address,uint256,uint256,bool,uint8,bytes32,bytes32) should be declared external:
        - VaultPermit.vaultPermitDAI(IERC20PermitDAI,address,uint256,uint256,bool,uint8,bytes32,bytes32) (BatchRelayerLibrary.
sol#905-916)
getEntrypoint() should be declared external:
        - BaseRelayerLibrary.getEntrypoint() (BatchRelayerLibrary.sol#1363-1365)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:BatchRelayerLibrary.sol analyzed (31 contracts with 75 detectors), 85 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> BalancerHelpers.sol

```
INFO:Detectors:
BasePool (BalancerHelpers.sol#1072-1346) does not implement functions:
        - BasePool._getMaxTokens() (BalancerHelpers.sol#1097)
        - BasePool._getTotalTokens() (BalancerHelpers.sol#1095)
        - BasePool._onExitPool(bytes32,address,address,uint256[],uint256,uint256,uint256[],bytes) (BalancerHelpers.sol#1217-12
33)
        - BasePool._onInitializePool(bytes32,address,address,uint256[],bytes) (BalancerHelpers.sol#1191-1197)
        - BasePool._onJoinPool(bytes32,address,address,uint256[],uint256,uint256,uint256[],bytes) (BalancerHelpers.sol#1199-12
15)
        - BasePool._scalingFactor(IERC20) (BalancerHelpers.sol#1245)
        - BasePool._scalingFactors() (BalancerHelpers.sol#1247)
        - IBasePool.onJoinPool(bytes32,address,address,uint256[],uint256,uint256,bytes) (BalancerHelpers.sol#708-716)
        - IERC20.permit(address,address,uint256,uint256,uint8,bytes32,bytes32) (BalancerHelpers.sol#158)
BalancerHelpers (BalancerHelpers.sol#1349-1379) does not implement functions:
        - AssetHelpers._asIERC20(IAsset) (BalancerHelpers.sol#196-198)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unimplemented-functions
INFO:Detectors:
ERC20Permit._PERMIT_TYPEHASH (BalancerHelpers.sol#978-979) is never used in BasePool (BalancerHelpers.sol#1072-1346)
TemporarilyPausable._MAX_PAUSE_WINDOW_DURATION (BalancerHelpers.sol#904) is never used in BasePool (BalancerHelpers.sol#1072-1
346)
TemporarilyPausable._MAX_BUFFER_PERIOD_DURATION (BalancerHelpers.sol#905) is never used in BasePool (BalancerHelpers.sol#1072-
1346)
BasePool._MIN_TOKENS (BalancerHelpers.sol#1076) is never used in BasePool (BalancerHelpers.sol#1072-1346)
BasePool._MIN_SWAP_FEE_PERCENTAGE (BalancerHelpers.sol#1080) is never used in BasePool (BalancerHelpers.sol#1072-1346)
BasePool._MAX_SWAP_FEE_PERCENTAGE (BalancerHelpers.sol#1081) is never used in BasePool (BalancerHelpers.sol#1072-1346)
BasePool._SWAP_FEE_PERCENTAGE_OFFSET (BalancerHelpers.sol#1084) is never used in BasePool (BalancerHelpers.sol#1072-1346)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-state-variables
INFO:Detectors:
BasePool._miscData (BalancerHelpers.sol#1083) should be constant
ERC20Permit._PERMIT_TYPEHASH (BalancerHelpers.sol#978-979) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Slither:BalancerHelpers.sol analyzed (28 contracts with 75 detectors), 99 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Slither log >> ManagedPoolFactory.sol

```
INFO:Detectors:
BaseWeightedPool._lastInvariant (ManagedPoolFactory.sol#1117) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
getActionId(bytes4) should be declared external:
        - Authentication.getActionId(bytes4) (ManagedPoolFactory.sol#629-631)
getManager() should be declared external:
        - BasePoolController.getManager() (ManagedPoolFactory.sol#714-716)
canTransferOwnership() should be declared external:
        - BasePoolController.canTransferOwnership() (ManagedPoolFactory.sol#726-728)
canChangeSwapFee() should be declared external:
        - BasePoolController.canChangeSwapFee() (ManagedPoolFactory.sol#730-732)
canUpdateMetadata() should be declared external:
        - BasePoolController.canUpdateMetadata() (ManagedPoolFactory.sol#734-736)
getMetadata() should be declared external:
        - BasePoolController.getMetadata() (ManagedPoolFactory.sol#781-783)
canChangeWeights() should be declared external:
        - ManagedPoolController.canChangeWeights() (ManagedPoolFactory.sol#836-838)
canDisableSwaps() should be declared external:
        - ManagedPoolController.canDisableSwaps() (ManagedPoolFactory.sol#840-842)
canSetMustAllowlistLPs() should be declared external:
        - ManagedPoolController.canSetMustAllowlistLPs() (ManagedPoolFactory.sol#844-846)
canSetCircuitBreakers() should be declared external:
        - ManagedPoolController.canSetCircuitBreakers() (ManagedPoolFactory.sol#848-850)
canChangeTokens() should be declared external:
        - ManagedPoolController.canChangeTokens() (ManagedPoolFactory.sol#852-854)
getCreationCodeContracts() should be declared external:
        - BaseSplitCodeFactory.getCreationCodeContracts() (ManagedPoolFactory.sol#969-971)
getCreationCode() should be declared external:
        - BaseSplitCodeFactory.getCreationCode() (ManagedPoolFactory.sol#973-975)
getLastInvariant() should be declared external:
        - BaseWeightedPool.getLastInvariant() (ManagedPoolFactory.sol#1127-1129)
        - ManagedPool.getLastInvariant() (ManagedPoolFactory.sol#1332-1334)
getRate() should be declared external:
        - BaseWeightedPool.getRate() (ManagedPoolFactory.sol#1145-1147)
```

```
getRate() should be declared external:
        - BaseWeightedPool.getRate() (ManagedPoolFactory.sol#1145-1147)
getSwapEnabled() should be declared external:
        - ManagedPool.getSwapEnabled() (ManagedPoolFactory.sol#1238-1240)
isAllowedAddress(address) should be declared external:
        - ManagedPool.isAllowedAddress(address) (ManagedPoolFactory.sol#1246-1248)
getManagementSwapFeePercentage() should be declared external:
        - ManagedPool.getManagementSwapFeePercentage() (ManagedPoolFactory.sol#1250-1252)
getCollectedManagementFees() should be declared external:
        - ManagedPool.getCollectedManagementFees() (ManagedPoolFactory.sol#1275-1286)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:ManagedPoolFactory.sol analyzed (31 contracts with 75 detectors), 107 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

Email: audit@EtherAuthority.io