Smart Contract Security Audit Report

[2021]

Table Of Contents

1 Executive Summary

2 Audit Methodology

3 Project Overview

3.1 Project Introduction

3.2 Vulnerability Information

4 Code Overview

4.1 Contracts Description

4.2 Visibility Description

4.3 Vulnerability Summary

5 Audit Result

6 Statement

1 Executive Summary

On 2021.09.03, the SlowMist security team received the ENSURO team's security audit application for Ensuro, developed the audit plan according to the agreement of both parties and the characteristics of the project, and finally issued the security audit report.

The SlowMist security team adopts the strategy of "white box lead, black, grey box assists" to conduct a complete security test on the project in the way closest to the real attack.

The test method information:

Test method	Description
Black box testing	Conduct security tests from an attacker's perspective externally.
Grey box testing	Conduct security testing on code modules through the scripting tool, observing the internal running status, mining weaknesses.
White box testing	Based on the open source code, non-open source code, to detect whether there are vulnerabilities in programs such as nodes, SDK, etc.

The vulnerability severity level information:

Level	Description
Critical	Critical severity vulnerabilities will have a significant impact on the security of the DeFi project, and it is strongly recommended to fix the critical vulnerabilities.
High	High severity vulnerabilities will affect the normal operation of the DeFi project. It is strongly recommended to fix high-risk vulnerabilities.
Medium	Medium severity vulnerability will affect the operation of the DeFi project. It is recommended to fix medium-risk vulnerabilities.
Low	Low severity vulnerabilities may affect the operation of the DeFi project in certain scenarios. It is suggested that the project team should evaluate and consider whether these vulnerabilities need to be fixed.
Weakness	There are safety risks theoretically, but it is extremely difficult to reproduce in engineering.

Level	Description
Suggestion	There are better practices for coding or architecture.

2 Audit Methodology

The security audit process of SlowMist security team for smart contract includes two steps:

Smart contract codes are scanned/tested for commonly known and more specific vulnerabilities using automated analysis tools.

Manual audit of the codes for security issues. The contracts are manually analyzed to look for any potential problems.

Following is the list of commonly known vulnerabilities that was considered during the audit of the smart contract:

- Reentrancy Vulnerability
- Replay Vulnerability
- Reordering Vulnerability
- Short Address Vulnerability
- Denial of Service Vulnerability
- Transaction Ordering Dependence Vulnerability
- Race Conditions Vulnerability
- Authority Control Vulnerability
- Integer Overflow and Underflow Vulnerability
- TimeStamp Dependence Vulnerability
- Uninitialized Storage Pointers Vulnerability
- Arithmetic Accuracy Deviation Vulnerability
- tx.origin Authentication Vulnerability

- "False top-up" Vulnerability
- Variable Coverage Vulnerability
- Gas Optimization Audit
- Malicious Event Log Audit
- Redundant Fallback Function Audit
- Unsafe External Call Audit
- Explicit Visibility of Functions State Variables Aduit
- Design Logic Audit
- Scoping and Declarations Audit

3 Project Overview

3.1 Project Introduction

Audit version code:

https://github.com/ensuro/ensuro

commit: 6711b17dffd79a2835e6690c681fb2b45c275564

Fixed version code:

https://github.com/ensuro/ensuro/pull/16/files

3.2 Vulnerability Information

The following is the status of the vulnerabilities found in this audit:

NO	Title	Category	Level	Status
N1	Risk of excessive authority	Authority Control Vulnerability	Medium	Ignored

NO	Title	Category	Level	Status
N2	Gas Consumption	Gas Optimization Audit	Suggestion	Ignored
N3	Unsafe External Call Audit	Unsafe External Call Audit	Low	Fixed
N4	Business design defects	Design Logic Audit	Medium	Ignored
N5	Slippage Detection Issue	Others	High	Fixed

4 Code Overview

4.1 Contracts Description

The main network address of the contract is as follows:

The code was not deployed to the mainnet.

4.2 Visibility Description

The SlowMist Security team analyzed the visibility of major contracts during the audit, the result as follows:

AaveAssetManager			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
__AaveAssetManager_init	Internal	Can Modify State	initializer
getInvestmentValue	Public	-	-
lendingPool	Public	-	-

AaveAssetManager			
_aaveDataProvider	Internal	-	-
aToken	Public	-	-
rewardToken	Public	-	-
rewardAToken	Public	-	-
_exchangePath	Internal	-	-
_rewardToCurrency	Internal	-	-
reinvestRewardToken	Public	Can Modify State	-
_swapRewards	Internal	Can Modify State	-
swapRewards	External	Can Modify State	onlyPoolRole
rebalance	Public	Can Modify State	whenNotPaused
_invest	Internal	Can Modify State	-
_deinvest	Internal	Can Modify State	-

BaseAssetManager			
Function Name	Visibility	Mutability	Modifiers
__BaseAssetManager_init	Public	Can Modify State	initializer
__BaseAssetManager_init_unchained	Public	Can Modify State	initializer
_validateParameters	Internal	-	-
totalInvestable	External	-	-
_totalInvestable	Internal	-	-

BaseAssetManager			
distibuteEarnings	Public	Can Modify State	whenNotPaused
getInvestmentValue	Public	-	-
rebalance	Public	Can Modify State	whenNotPaused
checkpoint	External	Can Modify State	-
refillWallet	External	Can Modify State	onlyPolicyPool
_invest	Internal	Can Modify State	-
_deinvest	Internal	Can Modify State	-
deinvestAll	External	Can Modify State	onlyPolicyPool
liquidityMin	External	-	-
liquidityMiddle	External	-	-
liquidityMax	External	-	-
setLiquidityMin	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setLiquidityMiddle	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setLiquidityMax	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange

PolicyPoolComponent			
Function Name	Visibility	Mutability	Modifiers
__PolicyPoolComponent_init	Internal	Can Modify State	initializer

PolicyPoolComponent			
__PolicyPoolComponent_init_unchained	Internal	Can Modify State	initializer
_authorizeUpgrade	Internal	Can Modify State	onlyPoolRole2
pause	Public	Can Modify State	onlyPoolRole
unpause	Public	Can Modify State	onlyPoolRole2
policyPool	Public	-	-
currency	Public	-	-
hasPoolRole	Internal	-	-
_isTweakRay	Internal	-	-
_isTweakWad	Internal	-	-
_parameterChanged	Internal	Can Modify State	-
lastTweak	External	-	-
_registerTweak	Internal	Can Modify State	-

EToken			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
__EToken_init_unchained	Public	Can Modify State	initializer
_validateParameters	Internal	-	-
name	Public	-	-
symbol	Public	-	-

EToken			
decimals	Public	-	-
totalSupply	Public	-	-
balanceOf	Public	-	-
transfer	Public	Can Modify State	-
allowance	Public	-	-
approve	Public	Can Modify State	-
transferFrom	Public	Can Modify State	-
increaseAllowance	Public	Can Modify State	-
decreaseAllowance	Public	Can Modify State	-
_scaleAmount	Internal	-	-
_transfer	Internal	Can Modify State	-
_mint	Internal	Can Modify State	-
_burn	Internal	Can Modify State	-
_approve	Internal	Can Modify State	-
_beforeTokenTransfer	Internal	Can Modify State	-
_updateCurrentScale	Internal	Can Modify State	-
_updateTokenInterestRate	Internal	Can Modify State	-
_calculateCurrentScale	Internal	-	-

EToken			
getCurrentScale	Public	-	-
ocean	Public	-	-
oceanForNewScr	Public	-	-
scr	Public	-	-
scrInterestRate	Public	-	-
tokenInterestRate	Public	-	-
liquidityRequirement	Public	-	-
maxUtilizationRate	Public	-	-
lockScr	External	Can Modify State	onlyPolicyPool
unlockScr	External	Can Modify State	onlyPolicyPool
_discreteChange	Internal	Can Modify State	-
discreteEarning	External	Can Modify State	onlyPolicyPool
assetEarnings	External	Can Modify State	onlyAssetManager
deposit	External	Can Modify State	onlyPolicyPool whenNotPaused
totalWithdrawable	Public	-	-
withdraw	External	Can Modify State	onlyPolicyPool whenNotPaused
accepts	Public	-	-
_updatePoolLoanScale	Internal	Can Modify State	-

EToken			
_maxNegativeAdjustment	Internal	-	-
lendToPool	External	Can Modify State	onlyPolicyPool
repayPoolLoan	External	Can Modify State	onlyPolicyPool
_getPoolLoanScale	Internal	-	-
getPoolLoan	Public	-	-
poolLoanInterestRate	Public	-	-
setPoolLoanInterestRate	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setLiquidityRequirement	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setMaxUtilizationRate	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
getInvestable	Public	-	-

LPManualWhitelist			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
whitelistAddress	External	Can Modify State	onlyPoolRole
acceptsDeposit	External	-	-
acceptsTransfer	External	-	-

PolicyNFT			
Function Name	Visibility	Mutability	Modifiers

PolicyNFT			
initialize	Public	Can Modify State	initializer
__PolicyNFT_init_unchained	Internal	Can Modify State	initializer
connect	External	Can Modify State	-
safeMint	External	Can Modify State	onlyPolicyPool whenNotPaused
_beforeTokenTransfer	Internal	Can Modify State	whenNotPaused

PolicyPool			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
__PolicyPool_init_unchained	Internal	Can Modify State	initializer
_authorizeUpgrade	Internal	Can Modify State	onlyRole2
pause	Public	Can Modify State	onlyRole
unpause	Public	Can Modify State	onlyRole2
config	External	-	-
currency	External	-	-
policyNFT	External	-	-
purePremiums	External	-	-
activePremiums	External	-	-
activePurePremiums	External	-	-

PolicyPool			
wonPurePremiums	External	-	-
borrowedActivePP	External	-	-
addEToken	External	Can Modify State	onlyRole
removeEToken	External	Can Modify State	onlyRole
changeETokenStatus	External	Can Modify State	onlyRole2
setAssetManager	External	Can Modify State	-
deposit	External	Can Modify State	whenNotPaused
withdraw	External	Can Modify State	whenNotPaused
newPolicy	External	Can Modify State	whenNotPaused
_lockScr	Internal	Can Modify State	-
_distributeScr	Internal	Can Modify State	-
_balance	Internal	-	-
_transferTo	Internal	Can Modify State	-
_payFromPool	Internal	Can Modify State	-
_storePurePremiumWon	Internal	Can Modify State	-
_processResolution	Internal	Can Modify State	-
expirePolicy	External	Can Modify State	whenNotPaused

PolicyPool			
resolvePolicy	External	Can Modify State	whenNotPaused
resolvePolicy	External	Can Modify State	whenNotPaused
_resolvePolicy	Internal	Can Modify State	-
_interestAdjustment	Internal	-	-
_updatePolicyFundsCustWon	Internal	Can Modify State	-
_updatePolicyFundsCustLost	Internal	Can Modify State	-
_takeLoanFromAnyEtk	Internal	Can Modify State	-
repayETokenLoan	External	Can Modify State	whenNotPaused
receiveGrant	External	Can Modify State	-
withdrawWonPremiums	External	Can Modify State	onlyRole
rebalancePolicy	External	Can Modify State	onlyRole whenNotPaused
getInvestable	External	-	-
totalETokenSupply	Public	-	-
assetEarnings	External	Can Modify State	onlyAssetManager whenNotPaused
getPolicy	External	-	-
getPolicyFundCount	External	-	-
getPolicyFundAt	External	-	-
getPolicyFund	External	-	-

PolicyPool			
getETokenCount	External	-	-
getETokenAt	External	-	-

PolicyPoolConfig			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
__PolicyPoolConfig_init_unchained	Internal	Can Modify State	initializer
connect	External	Can Modify State	-
_authorizeUpgrade	Internal	Can Modify State	onlyRole2
checkRole	External	-	-
checkRole2	External	-	-
setAssetManager	External	Can Modify State	onlyRole
assetManager	External	-	-
setTreasury	External	Can Modify State	onlyRole
treasury	External	-	-
setInsolvencyHook	External	Can Modify State	onlyRole2
insolvencyHook	External	-	-
setLPWhitelist	External	Can Modify State	onlyRole2
lpWhitelist	External	-	-
addRiskModule	External	Can Modify State	onlyRole2

PolicyPoolConfig			
removeRiskModule	External	Can Modify State	onlyRole
changeRiskModuleStatus	External	Can Modify State	onlyRole3
checkAcceptsNewPolicy	External	-	-
checkAcceptsResolvePolicy	External	-	-

RiskModule			
Function Name	Visibility	Mutability	Modifiers
__RiskModule_init	Internal	Can Modify State	initializer
__RiskModule_init_unchained	Internal	Can Modify State	initializer
_validateParameters	Internal	-	-
name	Public	-	-
scrPercentage	Public	-	-
moc	Public	-	-
ensuroFee	Public	-	-
scrInterestRate	Public	-	-
maxScrPerPolicy	Public	-	-
scrLimit	Public	-	-
totalScr	Public	-	-
sharedCoverageMinPercentage	Public	-	-
sharedCoveragePercentage	Public	-	-

RiskModule			
sharedCoverageScr	Public	-	-
wallet	Public	-	-
setScrPercentage	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setMoc	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setScrInterestRate	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setEnsuroFee	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setMaxScrPerPolicy	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setScrLimit	External	Can Modify State	onlyPoolRole3 validateParamsAfterChange
setSharedCoverageMinPercentage	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setSharedCoveragePercentage	External	Can Modify State	onlyRole validateParamsAfterChange
setWallet	External	Can Modify State	onlyRole validateParamsAfterChange
_newPolicy	Internal	Can Modify State	whenNotPaused

TrustfulRiskModule			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
newPolicy	External	Can Modify State	onlyRole
resolvePolicy	External	Can Modify State	onlyRole whenNotPaused

TrustfulRiskModule			
resolvePolicy	External	Can Modify State	onlyRole whenNotPaused

AaveAssetManager			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
__AaveAssetManager_init	Internal	Can Modify State	initializer
getInvestmentValue	Public	-	-
lendingPool	Public	-	-
_aaveDataProvider	Internal	-	-
aToken	Public	-	-
rewardToken	Public	-	-
rewardAToken	Public	-	-
_exchangePath	Internal	-	-
_rewardToCurrency	Internal	-	-
reinvestRewardToken	Public	Can Modify State	-
_swapRewards	Internal	Can Modify State	-
swapRewards	External	Can Modify State	onlyPoolRole
rebalance	Public	Can Modify State	whenNotPaused
_invest	Internal	Can Modify State	-
_deinvest	Internal	Can Modify State	-

BaseAssetManager			
Function Name	Visibility	Mutability	Modifiers
__BaseAssetManager_init	Public	Can Modify State	initializer
__BaseAssetManager_init_unchained	Public	Can Modify State	initializer
_validateParameters	Internal	-	-
totalInvestable	External	-	-
_totalInvestable	Internal	-	-
distibuteEarnings	Public	Can Modify State	whenNotPaused
getInvestmentValue	Public	-	-
rebalance	Public	Can Modify State	whenNotPaused
checkpoint	External	Can Modify State	-
refillWallet	External	Can Modify State	onlyPolicyPool
_invest	Internal	Can Modify State	-
_deinvest	Internal	Can Modify State	-
deinvestAll	External	Can Modify State	onlyPolicyPool
liquidityMin	External	-	-
liquidityMiddle	External	-	-
liquidityMax	External	-	-
setLiquidityMin	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange

BaseAssetManager			
setLiquidityMiddle	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setLiquidityMax	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange

PolicyPoolComponent			
Function Name	Visibility	Mutability	Modifiers
__PolicyPoolComponent_init	Internal	Can Modify State	initializer
__PolicyPoolComponent_init_unchained	Internal	Can Modify State	initializer
_authorizeUpgrade	Internal	Can Modify State	onlyPoolRole2
pause	Public	Can Modify State	onlyPoolRole
unpause	Public	Can Modify State	onlyPoolRole2
policyPool	Public	-	-
currency	Public	-	-
hasPoolRole	Internal	-	-
_isTweakRay	Internal	-	-
_isTweakWad	Internal	-	-
_parameterChanged	Internal	Can Modify State	-
lastTweak	External	-	-
_registerTweak	Internal	Can Modify State	-

EToken

EToken			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
__EToken_init_unchained	Public	Can Modify State	initializer
_validateParameters	Internal	-	-
name	Public	-	-
symbol	Public	-	-
decimals	Public	-	-
totalSupply	Public	-	-
balanceOf	Public	-	-
transfer	Public	Can Modify State	-
allowance	Public	-	-
approve	Public	Can Modify State	-
transferFrom	Public	Can Modify State	-
increaseAllowance	Public	Can Modify State	-
decreaseAllowance	Public	Can Modify State	-
_scaleAmount	Internal	-	-
_transfer	Internal	Can Modify State	-
_mint	Internal	Can Modify State	-

EToken			
_burn	Internal	Can Modify State	-
_approve	Internal	Can Modify State	-
_beforeTokenTransfer	Internal	Can Modify State	-
_updateCurrentScale	Internal	Can Modify State	-
_updateTokenInterestRate	Internal	Can Modify State	-
_calculateCurrentScale	Internal	-	-
getCurrentScale	Public	-	-
ocean	Public	-	-
oceanForNewScr	Public	-	-
scr	Public	-	-
scrInterestRate	Public	-	-
tokenInterestRate	Public	-	-
liquidityRequirement	Public	-	-
maxUtilizationRate	Public	-	-
lockScr	External	Can Modify State	onlyPolicyPool
unlockScr	External	Can Modify State	onlyPolicyPool
_discreteChange	Internal	Can Modify State	-
discreteEarning	External	Can Modify State	onlyPolicyPool

EToken			
assetEarnings	External	Can Modify State	onlyAssetManager
deposit	External	Can Modify State	onlyPolicyPool whenNotPaused
totalWithdrawable	Public	-	-
withdraw	External	Can Modify State	onlyPolicyPool whenNotPaused
accepts	Public	-	-
_updatePoolLoanScale	Internal	Can Modify State	-
_maxNegativeAdjustment	Internal	-	-
lendToPool	External	Can Modify State	onlyPolicyPool
repayPoolLoan	External	Can Modify State	onlyPolicyPool
_getPoolLoanScale	Internal	-	-
getPoolLoan	Public	-	-
poolLoanInterestRate	Public	-	-
setPoolLoanInterestRate	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setLiquidityRequirement	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setMaxUtilizationRate	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
getInvestable	Public	-	-

LPManualWhitelist

LPManualWhitelist			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
whitelistAddress	External	Can Modify State	onlyPoolRole
acceptsDeposit	External	-	-
acceptsTransfer	External	-	-

PolicyNFT			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
__PolicyNFT_init_unchained	Internal	Can Modify State	initializer
connect	External	Can Modify State	-
safeMint	External	Can Modify State	onlyPolicyPool whenNotPaused
_beforeTokenTransfer	Internal	Can Modify State	whenNotPaused

PolicyPool			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
__PolicyPool_init_unchained	Internal	Can Modify State	initializer
_authorizeUpgrade	Internal	Can Modify State	onlyRole2
pause	Public	Can Modify State	onlyRole

PolicyPool			
unpause	Public	Can Modify State	onlyRole2
config	External	-	-
currency	External	-	-
policyNFT	External	-	-
purePremiums	External	-	-
activePremiums	External	-	-
activePurePremiums	External	-	-
wonPurePremiums	External	-	-
borrowedActivePP	External	-	-
addEToken	External	Can Modify State	onlyRole
removeEToken	External	Can Modify State	onlyRole
changeETokenStatus	External	Can Modify State	onlyRole2
setAssetManager	External	Can Modify State	-
deposit	External	Can Modify State	whenNotPaused
withdraw	External	Can Modify State	whenNotPaused
newPolicy	External	Can Modify State	whenNotPaused
_lockScr	Internal	Can Modify State	-
_distributeScr	Internal	Can Modify State	-

PolicyPool			
_balance	Internal	-	-
_transferTo	Internal	Can Modify State	-
_payFromPool	Internal	Can Modify State	-
_storePurePremiumWon	Internal	Can Modify State	-
_processResolution	Internal	Can Modify State	-
expirePolicy	External	Can Modify State	whenNotPaused
resolvePolicy	External	Can Modify State	whenNotPaused
resolvePolicy	External	Can Modify State	whenNotPaused
_resolvePolicy	Internal	Can Modify State	-
_interestAdjustment	Internal	-	-
_updatePolicyFundsCustWon	Internal	Can Modify State	-
_updatePolicyFundsCustLost	Internal	Can Modify State	-
_takeLoanFromAnyEtk	Internal	Can Modify State	-
repayETokenLoan	External	Can Modify State	whenNotPaused
receiveGrant	External	Can Modify State	-
withdrawWonPremiums	External	Can Modify State	onlyRole
rebalancePolicy	External	Can Modify State	onlyRole whenNotPaused

PolicyPool			
getInvestable	External	-	-
totalETokenSupply	Public	-	-
assetEarnings	External	Can Modify State	onlyAssetManager whenNotPaused
getPolicy	External	-	-
getPolicyFundCount	External	-	-
getPolicyFundAt	External	-	-
getPolicyFund	External	-	-
getETokenCount	External	-	-
getETokenAt	External	-	-

PolicyPoolConfig			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
__PolicyPoolConfig_init_unchained	Internal	Can Modify State	initializer
connect	External	Can Modify State	-
_authorizeUpgrade	Internal	Can Modify State	onlyRole2
checkRole	External	-	-
checkRole2	External	-	-
setAssetManager	External	Can Modify State	onlyRole
assetManager	External	-	-

PolicyPoolConfig			
setTreasury	External	Can Modify State	onlyRole
treasury	External	-	-
setInsolvencyHook	External	Can Modify State	onlyRole2
insolvencyHook	External	-	-
setLPWhitelist	External	Can Modify State	onlyRole2
lpWhitelist	External	-	-
addRiskModule	External	Can Modify State	onlyRole2
removeRiskModule	External	Can Modify State	onlyRole
changeRiskModuleStatus	External	Can Modify State	onlyRole3
checkAcceptsNewPolicy	External	-	-
checkAcceptsResolvePolicy	External	-	-

RiskModule			
Function Name	Visibility	Mutability	Modifiers
__RiskModule_init	Internal	Can Modify State	initializer
__RiskModule_init_unchained	Internal	Can Modify State	initializer
_validateParameters	Internal	-	-
name	Public	-	-
scrPercentage	Public	-	-
moc	Public	-	-

RiskModule			
ensuroFee	Public	-	-
scrInterestRate	Public	-	-
maxScrPerPolicy	Public	-	-
scrLimit	Public	-	-
totalScr	Public	-	-
sharedCoverageMinPercentage	Public	-	-
sharedCoveragePercentage	Public	-	-
sharedCoverageScr	Public	-	-
wallet	Public	-	-
setScrPercentage	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setMoc	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setScrInterestRate	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setEnsuroFee	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setMaxScrPerPolicy	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setScrLimit	External	Can Modify State	onlyPoolRole3 validateParamsAfterChange
setSharedCoverageMinPercentage	External	Can Modify State	onlyPoolRole2 validateParamsAfterChange
setSharedCoveragePercentage	External	Can Modify State	onlyRole validateParamsAfterChange
setWallet	External	Can Modify State	onlyRole validateParamsAfterChange

RiskModule			
_newPolicy	Internal	Can Modify State	whenNotPaused

TrustfulRiskModule			
Function Name	Visibility	Mutability	Modifiers
initialize	Public	Can Modify State	initializer
newPolicy	External	Can Modify State	onlyRole
resolvePolicy	External	Can Modify State	onlyRole whenNotPaused
resolvePolicy	External	Can Modify State	onlyRole whenNotPaused

4.3 Vulnerability Summary

[N1] [Medium] Risk of excessive authority

Category: Authority Control Vulnerability

Content

The permission of the RM_PROVIDER_ROLE is too large.

RM_PROVIDER_ROLE can change any wallet_.

Code location:ENSURO-MAIN/contracts/RiskModule.sol#L325-L336

```
function setWallet(address wallet_)
    external
    onlyRole(RM_PROVIDER_ROLE)
    validateParamsAfterChange
  {
    _wallet = wallet_;
    _parameterChanged(
      IPolicyPoolConfig.GovernanceActions.setWallet,
      uint256(uint160(wallet_)),
      false
```

```
    );
  }
```

PRICER_ROLE role has excessive permissions.

The PRICER_ROLE role has the right to change the funds in the PolicyPool.sol contract by calling the

RiskModule.sol.newPolicy function.

Please confirm whether the PRICER_ROLE role has the right to do so.

Code location:ENSURO-MAIN/contracts/PolicyPool.sol#L246-L265

```
function newPolicy(Policy.PolicyData memory policy_, address customer)
    external
    override
    whenNotPaused
    returns (uint256)
  {
    IRiskModule rm = policy_.riskModule;
    require(address(rm) == msg.sender, "Only the RM can create new policies");
    _config.checkAcceptsNewPolicy(rm);
    _currency.safeTransferFrom(customer, address(this), policy_.premium);
    uint256 policyId = _policyNFT.safeMint(customer);
    Policy.PolicyData storage policy = _policies[policyId] = policy_;
    policy.id = policyId;
    if (policy.rmScr() > 0) _currency.safeTransferFrom(rm.wallet(), address(this),
policy.rmScr());
    _activePurePremiums += policy.purePremium;
    _activePremiums += policy.premium;
    _lockScr(policy);
    emit NewPolicy(rm, policy.id);
    return policy.id;
  }
```

Code location:ENSURO-MAIN/contracts/RiskModule.sol#L338-L366

```
function _newPolicy(
    uint256 payout,
    uint256 premium,
    uint256 lossProb,
    uint40 expiration,
    address customer
```

```
  ) internal whenNotPaused returns (uint256) {
    require(premium < payout, "Premium must be less than payout");
    require(expiration > uint40(block.timestamp), "Expiration must be in the
future");
    require(customer != address(0), "Customer can't be zero address");
    require(
      _policyPool.currency().allowance(customer, address(_policyPool)) >= premium,
      "You must allow ENSURO to transfer the premium"
    );
    Policy.PolicyData memory policy = Policy.initialize(
      this,
      premium,
      payout,
      lossProb,
      expiration
    );
    require(policy.scr <= _maxScrPerPolicy, "RiskModule: SCR is more than maximum per
policy");
    _totalScr += policy.scr;
    require(_totalScr <= _scrLimit, "RiskModule: SCR limit exceeded");
    _sharedCoverageScr += policy.rmCoverage;
    uint256 policyId = _policyPool.newPolicy(policy, customer);
    return policyId;
  }
}
```

Code location:ENSURO-MAIN/contracts/RiskModule.sol#L55-L63

```
function newPolicy(
    uint256 payout,
    uint256 premium,
    uint256 lossProb,
    uint40 expiration,
    address customer
  ) external onlyRole(PRICER_ROLE) returns (uint256) {
    return _newPolicy(payout, premium, lossProb, expiration, customer);
  }
```

Solution

It is recommended to transfer the permission to the timelock contract, and the function that modifies the contract parameters adopts event logging.

Status

Ignored

[N2] [Suggestion] Gas Consumption

Category: Gas Optimization Audit

Content

The following functions have the logic of first assigning and then verifying. If the verification fails, it will consume Gas in vain.

Code location:ENSURO-MAIN/contracts/EToken.sol#L64-L132

```
 /**
  * @dev Initializes the eToken
  * @param policyPool_ The address of the Ensuro PolicyPool where this eToken will be used
  * @param expirationPeriod Maximum expirationPeriod (from block.timestamp) of policies to be accepted
  * @param liquidityRequirement_ Liquidity requirement to allow withdrawal (in Ray - default=1 Ray)
  * @param maxUtilizationRate_ Max utilization rate (scr/totalSupply) (in Ray - default=1 Ray)
  * @param poolLoanInterestRate_ Rate of loans givencrto the policy pool (in Ray)
  * @param name_ Name of the eToken
  * @param symbol_ Symbol of the eToken
  */
 function initialize(
   string memory name_,
   string memory symbol_,
   IPolicyPool policyPool_,
   uint40 expirationPeriod,
   uint256 liquidityRequirement_,
   uint256 maxUtilizationRate_,
   uint256 poolLoanInterestRate_
 ) public initializer {
   __PolicyPoolComponent_init(policyPool_);
   __EToken_init_unchained(
     name_,
     symbol_,
     expirationPeriod,
```

```
    liquidityRequirement_,
    maxUtilizationRate_,
    poolLoanInterestRate_
  );
}

// solhint-disable-next-line func-name-mixedcase
function __EToken_init_unchained(
  string memory name_,
  string memory symbol_,
  uint40 expirationPeriod,
  uint256 liquidityRequirement_,
  uint256 maxUtilizationRate_,
  uint256 poolLoanInterestRate_
) public initializer {
  _name = name_;
  _symbol = symbol_;
  _expirationPeriod = expirationPeriod;
  _scaleFactor = WadRayMath.ray();
  _lastScaleUpdate = uint40(block.timestamp);
  _scr = 0;
  _scrInterestRate = 0;
  _tokenInterestRate = 0;
  _liquidityRequirement = liquidityRequirement_;
  _maxUtilizationRate = maxUtilizationRate_;

  _poolLoan = 0;
  _poolLoanInterestRate = poolLoanInterestRate_;
  _poolLoanScale = WadRayMath.ray();
  _poolLoanLastUpdate = uint40(block.timestamp);
  _validateParameters();
}

// runs validation on EToken parameters
function _validateParameters() internal view {
  require(
    _liquidityRequirement >= 8e26 && _liquidityRequirement <= 13e26,
    "Validation: liquidityRequirement must be [0.8, 1.3]"
  );
  require(
    _maxUtilizationRate >= 5e26 && _maxUtilizationRate <= WadRayMath.RAY,
    "Validation: maxUtilizationRate must be [0.5, 1]"
  );
  require(_poolLoanInterestRate <= 5e26, "Validation: poolLoanInterestRate must be
```

```
<= 50%");
 }
```

Code location:ENSURO-MAIN/contracts/BaseAssetManager.sol#L56-L78

```
// solhint-disable-next-line func-name-mixedcase
  function __BaseAssetManager_init_unchained(
    uint256 liquidityMin_,
    uint256 liquidityMiddle_,
    uint256 liquidityMax_
  ) public initializer {
    /*
    _cashBalance = 0;
    _lastInvestmentValue = 0;
    */
    _liquidityMin = liquidityMin_;
    _liquidityMiddle = liquidityMiddle_;
    _liquidityMax = liquidityMax_;
    _validateParameters();
  }

  // runs validation on EToken parameters
  function _validateParameters() internal view {
    require(
      _liquidityMin <= _liquidityMiddle && _liquidityMiddle <= _liquidityMax,
      "Validation: Liquidity limits are invalid"
    );
  }
```

Solution

It is recommended to check before assignment.

Status

Ignored

[N3] [Low] Unsafe External Call Audit

Category: Unsafe External Call Audit

Content

IPolicyPool has no whitelist restrictions and allows any address to access the contract, which may be attacked by malicious _policyPool. And a malicious _policyPool can preemptively call after PolicyPoolConfig, PolicyNFT were deployed. Much of the logic is performed by invoking the _policyPool contract.

Need to deduct the next scene:

Code location:ENSURO-MAIN/contracts/PolicyPoolConfig.sol #L75-L80

```
function connect() external override {
    require(address(_policyPool) == address(0), "PolicyPool already connected");
    _policyPool = IPolicyPool(_msgSender());
    // Not possible to do this validation because connect is called in _policyPool
initialize :'(
    // require(_policyPool.config() == this, "PolicyPool not connected to this
config");
  }
```

Code location:ENSURO-MAIN/contracts/PolicyNFT.sol#L27-L32

```
function connect() external override {
    require(address(_policyPool) == address(0), "PolicyPool already connected");
    _policyPool = IPolicyPool(_msgSender());
    // Not possible to do this validation because connect is called in _policyPool
initialize :'(
    // require(_policyPool.config() == this, "PolicyPool not connected to this
config");
  }
```

Solution

It is recommended to add permission check.

Status

Fixed

[N4] [Medium] Business design defects

Category: Design Logic Audit

Content

This function exists to judge totalsupply_ >= Locked, this may cause customers who withdraw money later to lose money.

Code location:ENSURO-MAIN/contracts/EToken.sol （#L561-L570）

```
function totalWithdrawable() public view virtual override returns (uint256) {
    uint256 locked = _scr
    .wadToRay()
    .rayMul(WadRayMath.ray() + _scrInterestRate)
    .rayMul(_liquidityRequirement)
    .rayToWad();
    uint256 totalSupply_ = totalSupply();
    if (totalSupply_ >= locked) return totalSupply_ - locked;
    else return 0;
  }
```

If the connect function is called repeatedly, the current contract cannot be used.

Code location:ENSURO-MAIN/contracts/PolicyPoolConfig.sol （#L75 - L80）

```
  function connect() external override {
    require(address(_policyPool) == address(0), "PolicyPool already connected");
    _policyPool = IPolicyPool(_msgSender());
    // Not possible to do this validation because connect is called in _policyPool
initialize :'(
    // require(_policyPool.config() == this, "PolicyPool not connected to this
config");
  }
```

Solution

The fix method needs to be discussed with the project team.

Status

Ignored

[N5] [High] Slippage Detection Issue

Category: Others

Content

The _swapRewards and _deinvest functions use the GetAmountOut function in uniswap as the amountOutMin to detect slippage, it is don't work.

Code location:ENSURO-MAIN/contracts/AaveAssetManager.sol#L111-L135

```
function _swapRewards(uint256 amount, address outAddr) internal returns (uint256,
uint256) {
    address[] memory path = _exchangePath();
    uint256 swapIn = IERC20Metadata(path[0]).balanceOf(address(this));
    if (swapIn < amount) {
      uint256 toWithdraw = amount - swapIn;
      if (rewardAToken().balanceOf(address(this)) < toWithdraw) {
        toWithdraw = type(uint256).max; // if not enought withdraw all
      }
      swapIn += lendingPool().withdraw(path[0], toWithdraw, address(this));
    } else {
      swapIn = amount;
    }
    // TODO Check: this is safe? Or I should use IPriceOracle instead
    uint256 swapOutMin = _swapRouter.getAmountsOut(swapIn, path)[1];
    IERC20Metadata(path[0]).approve(address(_swapRouter), swapIn);
    uint256[] memory amounts = _swapRouter.swapExactTokensForTokens(
      swapIn,
      swapOutMin,
      path,
      outAddr,
      block.timestamp
    );
    emit RewardSwapped(swapIn, amounts[1]);
    return (swapIn, amounts[1]);
  }
```

Code location:ENSURO-MAIN/contracts/AaveAssetManager.sol#L164 - L187

```
function _deinvest(uint256 amount) internal override {
    uint256 remainingAmount = amount;
    uint256 toWithdraw = amount;
    if (aToken().balanceOf(address(this)) < toWithdraw) {
      toWithdraw = type(uint256).max;
    }
    remainingAmount -= lendingPool().withdraw(
      address(currency()),
      toWithdraw,
      address(_policyPool) // Withdraw directly to _policyPool
    );
    if (remainingAmount > 0) {
      // TODO Check: this is safe? Or I should use IPriceOracle instead
      uint256 requiredRewards = _swapRouter.getAmountsIn(remainingAmount,
_exchangePath())[0];
      (, uint256 currencyOut) = _swapRewards(requiredRewards, address(_policyPool));
      if (currencyOut < remainingAmount) {
        remainingAmount -= currencyOut;
      } else {
        remainingAmount = 0;
      }
    }
    super._deinvest(amount - remainingAmount);
  }
}
```

Solution

Need to refer to the oracle to detect slippage, and can't simply rely on the data on the chain to detect slippage.

Status

Fixed

5 Audit Result

Audit Number	Audit Team	Audit Date	Audit Result

Audit Number	Audit Team	Audit Date	Audit Result
0X002109250001	SlowMist Security Team	2021.09.03 - 2021.09.24	Passed

Summary conclusion: The SlowMist security team uses a manual and SlowMist team's analysis tool to audit the project, during the audit work we found 1 high risk, 2 medium risk, 1 low risk, 1 suggestion vulnerabilities. Among them, N3 and N5 has been fixed, N1,N2 and N4 have been ignored. The code is not deployed to the mainnet.

6 Statement

SlowMist issues this report with reference to the facts that have occurred or existed before the issuance of this report, and only assumes corresponding responsibility based on these.

For the facts that occurred or existed after the issuance, SlowMist is not able to judge the security status of this project, and is not responsible for them. The security audit analysis and other contents of this report are based on the documents and materials provided to SlowMist by the information provider till the date of the insurance report (referred to as "provided information"). SlowMist assumes: The information provided is not missing, tampered with, deleted or concealed. If the information provided is missing, tampered with, deleted, concealed, or inconsistent with the actual situation, the SlowMist shall not be liable for any loss or adverse effect resulting therefrom. SlowMist only conducts the agreed security audit on the security situation of the project and issues this report. SlowMist is not responsible for the background and other conditions of the project.

Official Website

www.slowmist.com

E-mail

team@slowmist.com

Twitter

@SlowMist_Team

Github

https://github.com/slowmist