Lido on Polygon

Shard Labs Internal Audit

CRITICAL:

1. An operator can be overwritten.

Description:
An operator could be overwritten during the removal process. When we add a new operator the operator id is calculated using the totalNodeOperators link. This totalNodeOperators variable decreases when we remove an operator from the system link.

Steps:

1. Call addOperator: the operator id = 1 because the totalNodeOperators is equal to 1.
2. Call addOperator: the operator id = 2 because the totalNodeOperators is equal to 2.
3. The operators stake.
4. Call stopOperator: stop operator 1.
5. Call removeOperator 1: this will update the totalNodeOperators = 1
6. Call addOperator: the operator id = 2 because the totalNodeOperators is equal to 2, the previous operator with id 2 was overwritten

Result: An operator can be overwritten

Recommendation: Add a nodeOperatorCounter and increment it each time we add a new operator.

PR: https://github.com/Shard-Labs/PoLido/pull/60

Status: Fixed

MAJOR:

1. Withdraw totalDelegated may lead to unbalancing of the system.

Description:
When a validator is stopped, the system automatically withdraws the total delegated. This amount is hidden until we claim it. If we call getTotalPooled the result is not correct StMatic.

Steps:

1. Delegate 10 Matic (totalPooled is 10 Matic)
2. Stop an operator and withdraw the total delegated from the validator.

3. As we don’t burn stMatic when withdrawing total delegated the result is:
   totalPooled = 0 (Matic)
   totalSupply = 10 (stMatic)
4. A user submits 1 Matic he will receive 10 stMatic.

Result: Invalid total pooled Matic.
Recommendation: Include the total NFTs owned by the StMatic contract in the total Pooled.
PR: https://github.com/Shard-Labs/PoLido/pull/67
Status: Fixed

2. Submit threshold does include delegated tokens.

Description:
The submit threshold check does not include delegated tokens, but only buffered. So, when those tokens are delegated the submit threshold is reset and the contract can accept new submissions.
Steps:

1. setSubmitThreshold 100
2. User1 submit 100
3. Delegate
4. User1 submit 100 works (but it should fail)

Result:
When we call the delegate function, the threshold is reset and users are able to submit again.
Recommendation: Include the total delegated amount.
PR: https://github.com/Shard-Labs/PoLido/pull/62
Status: Fixed

3. Validator(s) may not receive rewards

Description:
Validator(s) could be ignored during the reward distribution. The condition inside the getOperatorInfos checks if the validator accumulated enough rewards NodeOperatorRegistry.sol.
When we do a call to a validator share contract inside StMATIC to requestWithdraw, claimTokens, or delegate, the validator share contract transfer the rewards to the msg.sender ValidatorShare.sol
Steps:

1. Call delegate, requestWithdraw or claimTokens automatically transfer the rewards to the StMATIC contract.
2. Right after step 1, we call the distributeRewards. In this case, the getOperatorInfos function will return an empty array because we check if they accumulated enough rewards and after step 1 the rewards are almost zero.
3. Result: Validator(s) will not receive rewards even if they generate rewards.
4. Recommendation: move the logic inside the StMATIC contract, reverse this condition to (if (stMaticReward > rewardThreshold)) then move this inside the if statement. Then remove the _withdrawRewards param.

Result: Validator(s) may not receive rewards.
Recommendation: get status from the stakeManager.
PR: https://github.com/Shard-Labs/PoLido/pull/56
Status: Fixed

WARNING:

1. DAO can not update the commission rate on stakeManager (Active operator)

Description:
Validator(s) can not update operator commission. The condition that checks if the operator is active uses the local status not the actual stakeManager status. So, when we check if the validator is ACTIVE it returns INACTIVE here. The code that interacts with the stakeManager will never execute.
Steps:

1. Call updateOperatorCommissionRate The local commission rate is updated but there is no call to the stakeManager.

Result: DAO can not update the commission rate on stakeManager.
Recommendation: get status from the stakeManager.
PR: https://github.com/Shard-Labs/PoLido/pull/61
Status: Fixed

2. Validator(s) can not update the signer key

Description:

Validator(s) can not update the signer key. The condition that checks if the operator is active uses the local status not the actual stakeManager status. So, when we check if the validator is ACTIVE it returns INACTIVE here. The code that interacts with the stakeManager will never execute.

Steps:

1. Call updateSigner. The local signer is updated but there is no call to the stakeManager.

Result: Validator(s) can not update signer on stakeManager.
Recommendation: get status from the stakeManager
PR: https://github.com/Shard-Labs/PoLido/pull/61
Status: Fixed

3. getMaticFromTokenId revert if the token validatorShare address is ZERO.)

Description:
The getMaticFromTokenId returns only the amount if it was requested from a validator share and reverts if it was requested from the stMatic contract.
Result: Revert
Recommendation: Check if the validator share address is zero.
PR: https://github.com/Shard-Labs/PoLido/pull/67
Status: Fixed