LIDO KSM
SMART CONTRACT AUDIT

February 08, 2022

MixBytes()

CONTENTS

1. INTRODUCTION

1.1 DISCLAIMER

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, investment advice, endorsement of the platform or its products, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only. The information presented in this report is confidential and privileged. If you are reading this report, you agree to keep it confidential, not to copy, disclose or disseminate without the agreement of Lido KSM. If you are not the intended recipient(s) of this document, please note that any disclosure, copying or dissemination of its content is strictly forbidden.

1.2 SECURITY ASSESSMENT METHODOLOGY

A group of auditors are involved in the work on the audit who check the provided source code independently of each other in accordance with the methodology described below:

01 Project architecture review:
> Reviewing project documentation
> General code review
> Reverse research and study of the architecture of the code based on the source code only
> Mockup prototyping

Stage goal:
Building an independent view of the project's architecture and identifying logical flaws in the code.

02 Checking the code against the checklist of known vulnerabilities:
> Manual code check for vulnerabilities from the company's internal checklist
> The company's checklist is constantly updated based on the analysis of hacks, research and audit of the clients' code
> Checking with static analyzers (i.e Slither, Mythril, etc.)

Stage goal:
Eliminate typical vulnerabilities (e.g. reentrancy, gas limit, flashloan attacks, etc.)

03 Checking the code for compliance with the desired security model:
> Detailed study of the project documentation
> Examining contracts tests
> Examining comments in code
> Comparison of the desired model obtained during the study with the reversed view obtained during the blind audit
> Exploits PoC development using Brownie

Stage goal:
Detection of inconsistencies with the desired model

04 Consolidation of interim auditor reports into a general one:
> Cross-check: each auditor reviews the reports of the others
> Discussion of the found issues by the auditors
> Formation of a general (merged) report

Stage goal:
Re-check all the problems for relevance and correctness of the threat level and provide the client with an interim report.

05 Bug fixing & re-check:
> Client fixes or comments on every issue
> Upon completion of the bug fixing, the auditors double-check each fix and set the statuses with a link to the fix

Stage goal:
Preparation of the final code version with all the fixes

06 Preparation of the final audit report and delivery to the customer.

Findings discovered during the audit are classified as follows:

FINDINGS SEVERITY BREAKDOWN

Level	Description	Required action
Critical	Bugs leading to assets theft, fund access locking, or any other loss funds to be transferred to any party	Immediate action to fix issue
Major	Bugs that can trigger a contract failure. Further recovery is possible only by manual modification of the contract state or replacement.	Implement fix as soon as possible
Warning	Bugs that can break the intended contract logic or expose it to DoS attacks	Take into consideration and implement fix in certain period
Comment	Other issues and recommendations reported to/acknowledged by the team	Take into consideration

Based on the feedback received from the Customer's team regarding the list of findings discovered by the Contractor, they are assigned the following statuses:

Status	Description
Fixed	Recommended fixes have been made to the project code and no longer affect its security.
Acknowledged	The project team is aware of this finding. Recommendations for this finding are planned to be resolved in the future. This finding does not affect the overall safety of the project.
No issue	Finding does not affect the overall safety of the project and does not violate the logic of its work.

1.3 PROJECT OVERVIEW

Lido KSM is a Liquid staking protocol on the Kusama network (Polkadot) deployed in the Moonriver parachain network. Its purpose is to let users receive income from KSM (DOT) staking without restrictions imposed by the Kusama network, such as blocking liquidity for a long time. Lido is a set of EVM-compatible smart contracts operating in the Moonriver/Moonbeam environment and relay-chain (Kusama/Polkadot) XCMP messages.`Lido.sol` contract is the core contract which acts as a liquid staking pool.
The contract is responsible for `xcKSM` deposits and withdrawals, minting and burning `stKSM`, delegating funds to node operators, applying fees, and accepting updates from the oracle contract.The smart contracts reviewed in this audit are designed wherein Lido also acts as an ERC20 token which represents staked `xcKSM`,`stKSM`. Tokens are minted upon deposit and burned when redeemed.`stKSM` tokens are pegged 1:1 to the `xcKSM` ones that are held by Lido. `stKSM` tokens balances are updated when the oracle reports change in total stake every era.

1.4 PROJECT DASHBOARD

Client	Lido KSM
Audit name	LIDO KSM
Initial version	76a10efa5f223c4c613f26794802b8fb9bb188e1 130bdc416933cb57ff5bf279e74d3f48decf224e 30b1f028f7e73075845c07f69c70c1cd0926055b
Final version	2f2725faa0bc371e4d1ddfceacd8c45d8f0905f8
Date	November 09, 2021 - February 08, 2022
Auditors engaged	3 auditors

FILES LISTING

AuthManager.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/contracts/AuthManager.sol
Controller.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/contracts/Controller.sol
Ledger.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/contracts/Ledger.sol

Lido.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/contracts/Lido.sol
OracleMaster.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/contracts/OracleMaster.sol
Oracle.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/contracts/Oracle.sol
stKSM.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/contracts/stKSM.sol
LedgerUtils.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/contracts/utils/LedgerUtils.sol
ReportUtils.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/contracts/utils/ReportUtils.sol
IAuthManager.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/interfaces/IAuthManager.sol
IController.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/interfaces/IController.sol
ILedger.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/interfaces/ILedger.sol
ILido.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/interfaces/ILido.sol
IOracleMaster.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/interfaces/IOracleMaster.sol
IOracle.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/interfaces/IOracle.sol
IRelayEncoder.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/interfaces/IRelayEncoder.sol
IXcmTransactor.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/interfaces/IXcmTransactor.sol
IxTokens.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/interfaces/IxTokens.sol
Types.sol	https://github.com/mixbytes/lido-dot-ksm/blob/76a10efa5f223c4c613f26794802b8fb9bb188e1/interfaces/Types.sol
LedgerFactory.sol	https://github.com/mixbytes/lido-dot-ksm/blob/da1accb85e028b0d5e1e5ed1c10622e852d9b43b/contracts/LedgerFactory.sol

Withdrawal.sol	https://github.com/mixbytes/lido-dot-ksm/blob/da1accb85e028b0d5e1e5ed1c10622e852d9b43b/contracts/Withdrawal.sol
wstKSM.sol	https://github.com/mixbytes/lido-dot-ksm/blob/da1accb85e028b0d5e1e5ed1c10622e852d9b43b/contracts/wstKSM.sol
LedgerBeacon.sol	https://github.com/mixbytes/lido-dot-ksm/blob/da1accb85e028b0d5e1e5ed1c10622e852d9b43b/contracts/proxy/LedgerBeacon.sol
LedgerProxy.sol	https://github.com/mixbytes/lido-dot-ksm/blob/da1accb85e028b0d5e1e5ed1c10622e852d9b43b/contracts/proxy/LedgerProxy.sol
WithdrawalQueue.sol	https://github.com/mixbytes/lido-dot-ksm/blob/da1accb85e028b0d5e1e5ed1c10622e852d9b43b/contracts/utils/WithdrawalQueue.sol
IWithdrawal.sol	https://github.com/mixbytes/lido-dot-ksm/blob/da1accb85e028b0d5e1e5ed1c10622e852d9b43b/interfaces/IWithdrawal.sol
ILedgerFactory.sol	https://github.com/mixbytes/lido-dot-ksm/blob/da1accb85e028b0d5e1e5ed1c10622e852d9b43b/interfaces/ILedgerFactory.sol
IvKSM.sol	https://github.com/mixbytes/lido-dot-ksm/blob/da1accb85e028b0d5e1e5ed1c10622e852d9b43b/interfaces/IvKSM.sol

FINDINGS SUMMARY

Level	Amount
Critical	2
Major	8
Warning	2
Comment	1

CONCLUSION

The smart contracts have been audited and several suspicious places were found. During the audit 2 critical and 7 major issues were identified. Several issues were marked as warnings. Havig worked on the audit report, all issues were fixed by the client. Thus, the contracts are assumed as secure to use according to our security criteria.Final commit identifier with all fixes: `2f2725faa0bc371e4d1ddfceacd8c45d8f0905f8`

2. FINDINGS REPORT

2.1 CRITICAL

CRT-1	Possible underflow
File	Lido.sol
Severity	Critical
Status	Fixed at 130bdc41

DESCRIPTION

If a ledger's stake drammaticaly decreases due to rebalance and after that the ledger receives a huge slash, then underflow can occur: Lido.sol#L608

RECOMMENDATION

We recommend distributing slashes across all the ledgers.

CLIENT'S COMMENTARY

Fixed

CRT-2	Possible overflow on cast to uint
File	Lido.sol
Severity	Critical
Status	Fixed at 130bdc41

DESCRIPTION

If `newStake` is a negative number, then overflow can occur: Lido.sol#L730

RECOMMENDATION

We recommend checking overall diff in order to exclude such scenarios.

CLIENT'S COMMENTARY

Fixed

2.2 MAJOR

MJR-1	Public access to all functions
File	Controller.sol
Severity	Major
Status	Fixed at 130bdc41

DESCRIPTION

In contract `Controller` all functions have public access which can be exploited: Controller.sol

RECOMMENDATION

We recommend adding access modificators.

CLIENT'S COMMENTARY

Fixed

MJR-2	Controller can be initialized several times
File	Controller.sol
Severity	Major
Status	Fixed at 130bdc41

DESCRIPTION

In contract `Controller` the `initialize` function can be called several times:
Controller.sol#L140

RECOMMENDATION

We recommend adding the `initializer` modifier.

CLIENT'S COMMENTARY

Fixed

MJR-3	Incorrect condition
File	Lido.sol
Severity	Major
Status	Fixed at 130bdc41

DESCRIPTION

The condition is incorrect here that can lead to an infinite loop: Lido.sol#L748

RECOMMENDATION

We recommend changing `||` into `&&`.

CLIENT'S COMMENTARY

Fixed

MJR-4	Possible burn of zero shares
File	Lido.sol
Severity	Major
Status	Fixed at 130bdc41

DESCRIPTION

Due to rounding errors a user can burn zero shares: Lido.sol#L522

RECOMMENDATION

We recommend adding a check so that a user couldn't burn zero shares.

CLIENT'S COMMENTARY

Fixed

MJR-5	Possible division by zero
File	Lido.sol
Severity	Major
Status	Fixed at 130bdc41

DESCRIPTION

In some cases division by zero can take place here:

- Lido.sol#L658
- Lido.sol#L708

RECOMMENDATION

We recommend to set a stake to zero if the overall shares amount is equal to zero.

CLIENT'S COMMENTARY

Fixed

MJR-6	Insufficient xcKSm balance on `Lido`
File	Lido.sol
Severity	Major
Status	Fixed at 130bdc41

DESCRIPTION

It is possible that `Lido` can have less than `_readyToClaim` : Lido.sol#L563

RECOMMENDATION

We recommend to add a requirement that `Lido` would have enough tokens to transfer.

CLIENT'S COMMENTARY

Fixed

MJR-7	Possible zero balance on `Lido`
File	Lido.sol
Severity	Major
Status	Fixed at 130bdc41

DESCRIPTION

It is possible that `Lido` can have zero balance on reward distribution: Lido.sol#L588

RECOMMENDATION

We recommend to add a check for the case when `Lido` has zero balance on reward distribution.

CLIENT'S COMMENTARY

Fixed

MJR-8	Possible underflow
File	Ledger.sol
Severity	Major
Status	Fixed at 130bdc41

DESCRIPTION

It is possible that free balance from the report can be less than free balance from the previous era: Ledger.sol#L297

RECOMMENDATION

We recommend to add a variable to control which amount should be bonded on the next era.

CLIENT'S COMMENTARY

Fixed

2.3 WARNING

WRN-1	Possible free tokens on Ledger
File	Ledger.sol
Severity	Warning
Status	Fixed at 130bdc41

DESCRIPTION

If someone sends `xcKSM` to Ledger: Ledger.sol#L282

RECOMMENDATION

We recommend sendig excess in funds to treasury.

CLIENT'S COMMENTARY

Fixed

WRN-2	Rewards can be lost
File	Lido.sol
Severity	Warning
Status	Fixed at 130bdc41

DESCRIPTION

If these addresses have been set to 0, then the rewards can be lost:
Lido.sol#L218
Lido.sol#L225
Lido.sol#L318
Lido.sol#L328

RECOMMENDATION

We recommend adding a zero address check.

CLIENT'S COMMENTARY

Fixed

2.4 COMMENT

CMT-1	Unusable variable
File	Lido.sol
Severity	Comment
Status	Fixed at 130bdc41

DESCRIPTION

The variable is defined and initialized, but not used in the smart contract:
Lido.sol#L201

RECOMMENDATION

We recommend removing this variable.

CLIENT'S COMMENTARY

Fixed

3. ABOUT MIXBYTES

MixBytes is a team of blockchain developers, auditors and analysts keen on decentralized systems. We build open-source solutions, smart contracts and blockchain protocols, perform security audits, work on benchmarking and software testing solutions, do research and tech consultancy.

BLOCKCHAINS

- Ethereum
- Cosmos
- EOS
- Substrate

TECH STACK

- Python
- Solidity
- Rust
- C++

CONTACTS

https://github.com/mixbytes/audits_public

https://mixbytes.io/

hello@mixbytes.io

https://t.me/MixBytes

https://twitter.com/mixbytes