Ozys Corp

KLAYswap Security Audit

: Final Report

Oct 2021

Theori Korea

Table of Contents

Executive Summary

Starting on August 30, 2021, Theori assessed the Ozys' KLAYswap for five weeks. The purpose of this audit was to identify security issues and establish the appropriate measures for improvement. For this, we analyzed the relevant source code to examine the internal process, identify security issues, and give recommendations.

- Source code received: 8/30 (Mon)
- Static code analysis: 8/30 (Mon) ~ 10/06 (Wed)
- Project report: 10/06 (Wed)

We evaluated the attack surfaces by understanding the KLAYswap service and threat modeling and proceeded with a source code audit. Through this, we were able to identify issues based on threat scenarios that can occur in the KLAYswap service.

We identified one security issue where an attacker could obtain slight profit and two recommendations regarding improvement. Moreover, we provide design comments that consider further implementation in the future.

Project Overview

Engagement Summary

Dates	2021. 8. 30. - 2021. 10. 06. (5 weeks)
Methodology	Source code auditing

Scope

Name	KLAYswap
Version	2021. 8. 30. • Github Repository: https://github.com/KLAYswap/audit-theori • Commit Hash: b4f12ea2b61c63bca583ab8e3f0843ad24e9fdf8
Application Type	Smart contracts
Lang. / Platforms	Solidity / Klaytn network

Contracts Summary

Directory	Contract	Description
contracts/	BuybackFund.sol	This contract buys back tokens from token holders.
	KSStore.sol	This contract is used to prevent reentrant attacks between two or more different contracts using a shared variable that describes locked status.
	KlaytnExchange.impl.sol	Implementation of the KlaytnExchange contract.
	KlaytnExchange.sol	This contract expands KIP7 that is created with every liquidity pair.
	KlaytnFactory.impl.sol	Implementation of the KlaytnFactory contract.
	KlaytnFactory.sol	This contract oversees the full functionality of token pair registration and transactions in KLAYswap.
	KlaytnMiningView.sol	This contract is a viewer that allows sorted searching of various KLAYswap data.
contracts/kai/	Admin.impl.sol	These have yet to be implemented completely. (Out of the audit scope)
	Admin.sol	
	Kai.sol	
contracts/supporter/	Helper.sol	This contract supports single side deposit to the liquidity pool.
	Supporter.impl.sol	Implementation of the Supporter contract.
	Supporter.sol	This contract supports adding liquidity to the KLAY-sKLAY pool.
	Wallet.impl.sol	Implementation of the Wallet contract.

	Wallet.sol	This contract is a wallet to store each user's LP token that is created to every single user.
contracts/treasury/	AirdropOperator.sol	This contract operates a token distribution contract.
	Distribution.impl.sol	Implementation of the Distribution contract.
	Distribution.sol	This contract supports an airdrop distribution.
	Treasury.impl.sol	Implementation of the Treasury contract.
	Treasury.sol	This contract organizes airdrop operations.
contracts/vksp/	Governance.impl.sol	Implementation of the Governance contract.
	Governance.sol	This contract is the governance module of KLAYswap. Currently, Ozys manages the ecosystem using this contract for themselves.
	PoolVoting.impl.sol	Implementation of the PoolVoting contract.
	PoolVoting.sol	This contract supports a pool voting system that includes voting and managing rewards.
	Store.impl.sol	Implementation of the Store contract.
	Store.sol	This contract calculates Time-weighted average price(TWAP) and is forked from Uniswap Oracle.
	VotingKSP.impl.sol	Implementation of the VotingKSP contract.
	VotingKSP.sol	This contract manages voting power(votingKSP).

Severity Categories

Severity	Description
Critical	The attack cost is low (not requiring much time or effort to succeed in the actual attack), and the vulnerability causes a high-impact issue. (e.g. Effect on service availability, Attacker taking financial gain)
High	An attacker can succeed in an attack which clearly causes problems in the service's operation. Even when the attack cost is high, the severity of the issue is considered "high" if the impact of the attack is remarkably high.
Medium	An attacker may perform an unintended action in the service, and the action may impact service operation. However, there are some restrictions for the actual attack to succeed.
Low	An attacker can perform an unintended action in the service, but the action does not cause significant impact or the success rate of the attack is remarkably low.
Informational	The issue is not currently recognized as a vulnerability, but may develop into a potential security threat as the service is further developed.

Issue Breakdown by Severity

Category	Count	Issues
Critical	N/A	N/A
High	N/A	N/A
Medium	1	**Issue #1.** BuybackFund contract is vulnerable to front-running attack due to missing owner check.
Low	N/A	N/A
Informational	2	**Recommendation #1.** Reentrancy occurs when the createPool function transfers the token. **Recommendation #2.** Incorrect documentation

Findings

ID	Title	Summary	Severity	Status
BUYBACK-001	BuybackFund contract is vulnerable to front-running attack due to missing owner check.	Unauthorized users can call `BubackFund`'s buyback function. A front-running attack may occur in the `BuybackFund` contract. Due to this, an attacker can get illegitimate profits.	Medium	FIXED
Last Updated: 2021 - 10 - 07				

Recommendations

ID	Title	Summary	Severity	Status
FACTORY-001	Reentrancy occurs when the createPool function transfers the token	`createPool` function can be abused using reentrancy attacks, which leads to unintended behavior creating the same token pair.	Informational	FIXED
COMMON-001	Incorrect documentation	Incorrect content of official documentation can confuse the user, or monetary damage can occur.	Informational	FIXED (Documentation has not yet been updated.)
Last Updated: 2021 - 10 - 07