HashEx
Blockchain Security

ApeSwap IAZO

smart contracts preliminary audit report

November 2021


hashex.org


contact@hashex.org

Contents

1. Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and HashEx and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (HashEx) owe no duty of care towards you or any other person, nor does HashEx make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties, or other terms of any kind except as set out in this disclaimer, and HashEx hereby excludes all representations, warranties, conditions, and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, HashEx hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against HashEx, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of

money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of the use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed. HashEx owns all copyright rights to the text, images, photographs, and other content provided in the following document. When using or sharing partly or in full, third parties must provide a direct link to the original document mentioning the author (hashex.org).

2. Overview

HashEx was commissioned by the ApeSwap Finance team to perform an audit of their smart contracts .

The code located in the github repository @apeswapfinance/apeswap-iazo was audited after the commit 74a5d9f. The updated code was re-checked after the 2540152 commit in the same repository.

The IAZO project is an ERC20 token sale platform with an optional add liquidity function and LP tokens locking. Documentation was provided with the README.md and separate contract descriptions. The repository contains tests with ~75% coverage.

The purpose of this audit was to achieve the following:

- Identify potential security issues with smart contracts.
- Formally check the logic behind given smart contracts.

Information in this report should be used to understand the risk exposure of smart contracts, and as a guide to improving the security posture of smart contracts by remediating the issues that were identified.

2.1 Summary

Project name	ApeSwap IAZO
URL	https://apeswap.finance/
Platform	Binance Smart Chain
Language	Solidity

2.2 Contracts

Name	Address
IAZOFactory	https://github.com/ApeSwapFinance/apeswap-iazo/blob/2540152d05640334ea8b71c0c5e41b081ca8d190/contracts/IAZOFactory.sol
IAZO	https://github.com/ApeSwapFinance/apeswap-iazo/blob/2540152d05640334ea8b71c0c5e41b081ca8d190/contracts/IAZO.sol
IAZOLiquidityLocker	https://github.com/ApeSwapFinance/apeswap-iazo/blob/2540152d05640334ea8b71c0c5e41b081ca8d190/contracts/IAZOLiquidityLocker.sol
IAZOTokenTimelock	https://github.com/ApeSwapFinance/apeswap-iazo/blob/2540152d05640334ea8b71c0c5e41b081ca8d190/contracts/IAZOTokenTimelock.sol
IAZOExposer	https://github.com/ApeSwapFinance/apeswap-iazo/blob/2540152d05640334ea8b71c0c5e41b081ca8d190/contracts/IAZOExposer.sol
IAZOSettings	https://github.com/ApeSwapFinance/apeswap-iazo/blob/2540152d05640334ea8b71c0c5e41b081ca8d190/contracts/IAZOSettings.sol
IAZOUpgradeProxy	https://github.com/ApeSwapFinance/apeswap-iazo/blob/2540152d05640334ea8b71c0c5e41b081ca8d190/contracts/IAZOUpgradeProxy.sol
OwnableProxy	https://github.com/ApeSwapFinance/apeswap-iazo/blob/2540152d05640334ea8b71c0c5e41b081ca8d190/contracts/OwnableProxy.sol
Gas optimizations and general recommendations	

3. Found issues

High	5 (22%)
Medium	3 (13%)
Low	8 (35%)
Informational	7 (30%)

IAZOFactory

ID	Title	Severity	Status
01	setIAZOVersion() frontrun problem	High	Acknowledged
02	Reflect token protection doesn't work	Medium	Acknowledged
03	TOKEN_PRICE lack of documentation	Low	Resolved
04	Input data not checked	Low	Resolved
05	createIAZO lack of documentation	Low	Resolved

IAZO

ID	Title	Severity	Status
01	No restrictions on forceFailAdmin()	High	Resolved
02	No input data checks in updateStart()	High	Resolved
03	BNB transfer to fee address	High	Resolved
04	RFI and tokens with commissions	Informational	Acknowledged

IAZOLiquidityLocker

ID	Title	Severity	Status
01	Lack of documentation: revocable locks	Medium	Resolved
02	Unused parameter	Low	Resolved
03	apePairIsInitialised() not checked in lockLiquidity()	Low	Acknowledged

IAZOTokenTimelock

ID	Title	Severity	Status
01	Beneficiaries can't be removed	High	Resolved

IAZOExposer

ID	Title	Severity	Status
01	require() statement in view function	Low	Resolved
02	EnumerableSet is not used	Low	Resolved

IAZOSettings

ID	Title	Severity	Status
01	Burn address can be changed by admin	Medium	Resolved

Gas optimizations and general recommendations

ID	Title	Severity	Status
01	OwnableProxy	Low	Resolved
02	IAZOFactory	Informational	Resolved
03	IAZO	Informational	Partially fixed
04	IAZOLiquidityLocker	Informational	Resolved
05	IAZOTimelock	Informational	Resolved
06	IAZOExposer	Informational	Acknowledged
07	IAZOSettings	Informational	Resolved

4. Contracts

4.1 IAZOFactory

4.1.1 Overview

Factory contract for creation of sales contracts by Clones library.

4.1.2 Issues

01. setIAZOVersion() frontrun problem

- High ⓘ Acknowledged

setIAZOVersion() function could be used by the owner to frontrun new IAZO and potentially steal the sale's earnings.

Recommendation

We recommend transferring admin/owner privileges to a Timelock contract.

02. Reflect token protection doesn't work

- Medium ⓘ Acknowledged

Reflect tokens protection in L205 doesn’t work in general as transfers from the owner are usually free from taxes.

Recommendation

We recommend explicitly describing the risks of participating in malicious sales as Factory is meant to be used without constant admin intervention.

03. TOKEN_PRICE lack of documentation

- Low ✓ Resolved

TOKEN_PRICE should have extended description, i.g. BASE(or NATIVE) amount in wei for 1.0 IAZO (weis divided by 10^decimals). The current comment in L73 is insufficient and may confuse users.

04. Input data not checked

- Low ✓ Resolved

The New IAZO owner is not checked for zero in createIAZO() function.

05. createIAZO lack of documentation

- Low ✓ Resolved

In L219 min _amount is 1e4, but _tokenPrice isn't checked to be greater than decimals/ amount, i.e. hardcap could be calculated as 0.

4.2 IAZO

4.2.1 Overview

Token sale contract for chosen currency or BNB. Automatically adds percent of liquidity to ApeSwap after a successful sale and locks LP tokens.

4.2.2 Issues

01. No restrictions on forceFailAdmin()

- High ✓ Resolved

forceFailAdmin() function irreversibly changes IAZO state to FORCE_FAILED, causing changed conditions in userWithdraw() L247. The problem occurs if the sale has been successful and liquidity has already been added, leaving the contract without BASE tokens.

Recommendation

Deny forceFailAdmin() calls after adding liquidity.

02. No input data checks in updateStart()

- High ✓ Resolved

updateStart() function doesn't check new _activeTime for max limit, i.e. IAZO_SETTINGS.getMaxIAZOLength().

Recommendation

Limit updated value from above.

03. BNB transfer to fee address

- High ✓ Resolved

Transferring fees in native currency in addLiquidity() function in L365 may fail if FEE_ADDRESS is set to the contract without receiving functions. In that case, all withdrawals would be blocked and the only possibility would be the FORCE_FAILED option. Also need to mention that the recommended way of sending native currency is call() with a reentrancy guard.

Recommendation

Implement a separate external function for fee collecting or use try/catch with a limited gas sending method.

04. RFI and tokens with commissions

- Informational ⓘ Acknowledged

RFI tokens and tokens with ommissions aren't supported by the contract.

4.3 IAZOLiquidityLocker

4.3.1 Overview

Support contract to be called by IAZO contracts to add liquidity. Creates a new vault contract for each IAZO to lock their LP tokens.

4.3.2 Issues

01. Lack of documentation: revocable locks

- Medium
- ✓ Resolved

Admin can grant permission to withdraw tokens before releaseTime if the revocable variable on deploy is set to true, which it is by default. This must be described on the project's website and in its documentation, users should be aware of this feature.

02. Unused parameter

- Low
- ✓ Resolved

_iazoAddress is always equal to msg.sender in L156. Updates of IAZO could use this to register wrong Timelock information in the Exposer contract, see L184.

03. apePairIsInitialised() not checked in lockLiquidity()

- Low
- ! Acknowledged

apePairIsInitialised() is designed to be checked during lockLiquidity(), but it's called in the IAZO contract instead. The possible updates of the IAZO version may miss that part of the

code and break the safety guard.

4.4 IAZOTokenTimelock

4.4.1 Overview

Locking contract for storing LP tokens of successful sales. The minimum locking period is set in IAZOSettings by ApeSwap admin. By default, locks could be lifted with ApeSwap admin permission.

4.4.2 Issues

01. Beneficiaries can't be removed

- High ✓ Resolved

addBeneficiary() function is irreversible, wrong or compromised address can be stopped only with transaction race.

Recommendation

Remove can be implemented via onlyAdmin and checking that at least 1 beneficiary remains in the list.

4.5 IAZOExposer

4.5.1 Overview

Registry contract for tracking IAZOs. Stores all factory created contracts and their Timelock for successful sales.

4.5.2 Issues

01. require() statement in view function

- Low ✓ Resolved

require() statement in getTokenTimelock() view function L94 may lead to wrong reads in explorers.

02. EnumerableSet is not used

- Low ✓ Resolved

EnumerableSet.AddressSet IAZOs and IAZOAddressToIndex is redundant. The enumerable set already contains mapping address -> index. But single mapping address -> bool should be sufficient.

4.6 IAZOSettings

4.6.1 Overview

Default parameters and limits for new IAZOs. Parameters, updatable for admins, are read-only at the moment of IAZO creation.

4.6.2 Issues

01. Burn address can be changed by admin

- Medium
- ✓ Resolved

Admin can set the burn address to his own account and collect all leftovers of IAZO_TOKEN (in case of burning leftovers).

Recommendation

Burn address should be constant 0x00 or 0xdead.

4.7 IAZOUpgradeProxy

4.7.1 Overview

TransparentUpgradeableProxy by OpenZeppelin.

4.8 OwnableProxy

4.8.1 Overview

Modification of Ownable contract from OpenZeppelin repository with a removed constructor to work with initializable contracts.

4.9 Gas optimizations and general recommendations

4.9.1 Overview

These issues are combined into one section with Informational severity. We recommend avoiding using modified OpenZeppelin contracts but inherit from originals if custom functionality is needed. We also recommend following Solidity naming conventions.

4.9.2 Issues

01. OwnableProxy

- Low ✓ Resolved

OwnableUpgradeable contract from OpenZeppelin could be used with initializable contracts.

02. IAZOFactory

- Informational ✓ Resolved

Excessive computations in L250: should be amount*percent*tokenPrice/1000/listingPrice.

Variable declaration with zero assignment L59.

03. IAZO

- Informational ◯ Partially fixed

Excessive or unnecessary reads in L132, (208, 217, 220), (209, 222), (207, 226,230), (261, 262, 263), (271, 272, 273), (301, 304, 309), 317, (334, 338), (328, 345, 349, 351), 322, (365, 367, 369, 370). IAZO_INFO.BASE_TOKEN is read 3 to 5 times, IAZO_INFO.IAZO_TOKEN is read 6 to 7 times in the addLiquidity() function.

In the function userDepositPrivate() calculation of the amount_in variable is unnecessary, the _amount variable can always be used.

Checking input amount for zero in userDepositPrivate() may reduce gas consumption in certain cases.

In the function userDepositPrivate() there is an external call to IAZO_TOKEN to get the decimals value. It is more gas-efficient to store decimals in the global variable.

The variable is declared with zero assignment in L105.

getIAZOState() could use enum constants for better readability.

Structures in L49-89 could be modified and/or rearranged to save gas on storage by packing multiple variables into 256bit slots.

No checks on input data in the initialize() function, although they are mostly performed in the Factory contract.

Inconsistent comment in L54, should describe TOKEN_PRICE with mentioning decimals.

Typos in L71, 320, 327, 340.

04. IAZOLiquidityLocker

- Informational ✓ Resolved

Contracts IAZOTokenTimelock can be deployed using Clones by OpenZeppelin.

No need to import full interfaces besides the needed functions.

Excessive read in L164, createPair() returns new address.

Typos in L122.

05. IAZOTimelock

- Informational ✓ Resolved

Variables releaseTime, IAZO_SETTINGS and revocable can be marked as immutable. Also, the variable isIAZOTokenTimelock can be marked as constant.

06. IAZOExposer

- Informational ! Acknowledged

Variable declaration with zero assignment L38.

07. IAZOSettings

- Informational ✓ Resolved

Excessive reads in L129, 132,135, 149, 156.

5. Conclusion

5 high severity issues were found. The contracts are highly dependent on the owner’s account. Most of the issues were fixed with the update, including all the High severity ones. Users using the project have to trust the owner and that the owner's account is properly secured.

This audit includes recommendations on the code improving and preventing potential attacks.

Open for all createIAZO() function of IAZOFactory contract allows creating fraud sales. Deny of responsibility should be mentioned on the project website and docs section.

Appendix A. Issues' severity classification

Critical. Issues that may cause an unlimited loss of funds or entirely break the contract workflow. Malicious code (including malicious modification of libraries) is also treated as a critical severity issue. These issues must be fixed before deployments or fixed in already running projects as soon as possible.

High. Issues that may lead to a limited loss of funds, break interaction with users, or other contracts under specific conditions. Also, issues in a smart contract, that allow a privileged account the ability to steal or block other users' funds.

Medium. Issues that do not lead to a loss of funds directly, but break the contract logic. May lead to failures in contracts operation.

Low. Issues that are of a non-optimal code character, for instance, gas optimization tips, unused variables, errors in messages.

Informational. Issues that do not impact the contract operation. Usually, informational severity issues are related to code best practices, e.g. style guide.

Appendix B. List of examined issue types

- Business logic overview
- Functionality checks
- Following best practices
- Access control and authorization
- Reentrancy attacks
- Front-run attacks
- DoS with (unexpected) revert
- DoS with block gas limit
- Transaction-ordering dependence
- ERC/BEP and other standards violation
- Unchecked math
- Implicit visibility levels
- Excessive gas usage
- Timestamp dependence
- Forcibly sending ether to a contract
- Weak sources of randomness
- Shadowing state variables
- Usage of deprecated code

HashEx
Blockchain Security