AuditAce
FROM INCEPTION TO SUCCESS

# Smart Contract Audit

FOR

# Asian Pepe

DATED : 6 July 23'

# MANUAL TESTING

## Centralization – Enabling Trades

**Severity**: High

**function**: enableTrading

**Status:** Not Resolved

**Overview:**

Owner of the contract must enable trades manually for investors, otherwise no one would be able to buy/sell/transfer their tokens.

```
function enableTrading() external onlyOwner {
    tradingEnabled = true;
}
```

**Suggestion**

Its suggested to either enable trades prior to presale, or transfer ownership of the contract to a certified pinsksale safu developer to guearanee enabling of trades.

# AUDIT SUMMARY

**Project name** – Asian Pepe

**Date**: 6 July, 2023

**Scope of Audit-** Audit Ace was consulted to conduct the smart contract audit of the solidity source codes.

**Audit Status:** **Passed With High Risk**

## Issues Found

| Status | Critical | High | Medium | Low | Suggestion |
|---|---|---|---|---|---|
| Open | 0 | 1 | 2 | 0 | 0 |
| Acknowledged | 0 | 0 | 0 | 0 | 0 |
| Resolved | 0 | 0 | 0 | 0 | 0 |

# USED TOOLS

## Tools:

**1- Manual Review:**
A line by line code review has been performed by audit ace team.

**2- BSC Test Network:** All tests were conducted on the BSC Test network, and each test has a corresponding transaction attached to it. These tests can be found in the "Functional Tests" section of the report.

**3- Slither :**
The code has undergone static analysis using Slither.

**Testnet version:**
The tests were performed using the contract deployed on the BSC Testnet, which can be found at the following address:
https://testnet.bscscan.com/token/0xe69d73d7d8edd288e7b899412cde1376d45bf078

# Token Information

**Token Name** : Asian Pepe

**Token Symbol**: $Asian Pepe

**Decimals:** 18

**Token Supply**: 100,000,000

**Token Address:**
--

**Checksum:**
aebbe21646d49e34c96296a62339ac485bc6a716

**Owner:**
--
**(at time of writing the audit)**

**Deployer:**
--

# TOKEN OVERVIEW

**Fees:**

Buy Fees: 0-15%

Sell Fees: 0-15%

Transfer Fees: 0-15%

**Fees Privilege:** owner

**Ownership**: owned

**Minting:** No mint function

**Max Tx Amount/ Max Wallet Amount: Yes**

**Blacklist:** No

**Other Privileges**: Initial distribution of the tokens

- modifying fees
- enabling trades

# AUDIT METHODOLOGY

The auditing process will follow a routine as special considerations by Auditace:

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.
- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.
- Test coverage analysis determines whether the test cases are covering the code and how much code isexercised when we run the test cases.
- Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.
- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

# VULNERABILITY CHECKLIST

- [x] Return values of low-level calls
- [x] Private modifier
- [x] Multiple Sends
- [x] Using Suicide
- [x] Gas Limitand Loops
- [x] Address hardcoded
- [x] Exception Disorder
- [x] Using inline assembly
- [x] Divide before multiply
- [x] Missing Zero Address Validation
- [x] Compiler version not fixed
- [x] **Gasless Send**
- [x] Using block.timestamp
- [x] Re-entrancy
- [x] Tautology or contradiction
- [x] Timestamp Dependence
- [x] Revert/require functions
- [x] Use of tx.origin
- [x] Integer overflow/underflow
- [x] Dangerous strict equalities
- [x] Using SHA3
- [x] Using throw

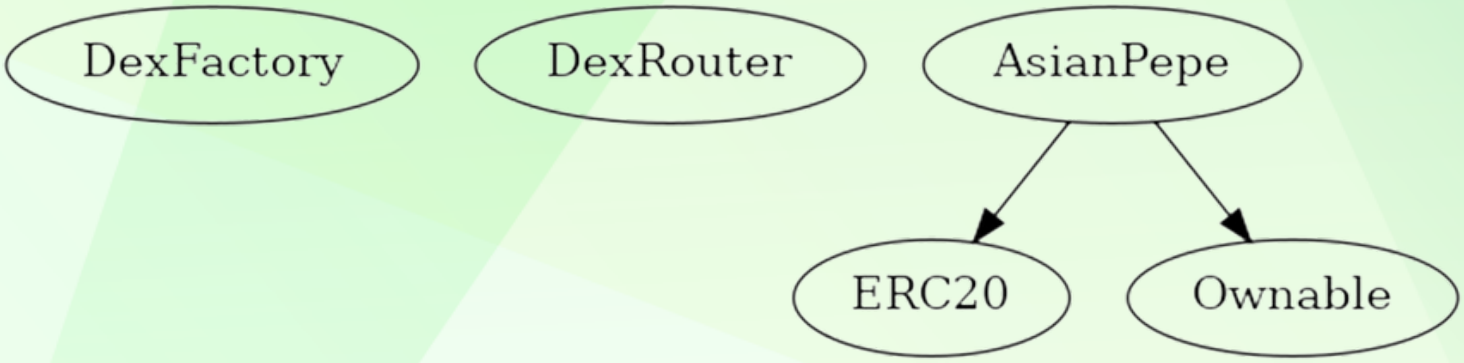
# CLASSIFICATION OF RISK

| Severity | Description |
| --- | --- |
| Critical | These vulnerabilities could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away. |
| High-Risk | A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way. |
| Medium-Risk | A vulnerability that could affect the desired outcome of executing the contract in a specific scenario. |
| Low-Risk | A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective. |
| Gas Optimization /Suggestion | A vulnerability that has an informational character but is not affecting any of the code. |

# Findings

| Severity | Found |
| --- | --- |
| Critical | 0 |
| High-Risk | 1 |
| Medium-Risk | 2 |
| Low-Risk | 0 |
| Gas Optimization / Suggestions | 0 |

# INHERITANCE TREE



# CONTRACT ASSESMENT

| Contract | Type | Bases | | |
|:---|:---|:---|:---|:---|
| └ | **Function Name** | **Visibility** | **Mutability** | **Modifiers** |
| | | | | |
| **DexFactory** | Interface | | | |
| └ | createPair | External ❗ | 🛑 | NO ❗ |
| | | | | |
| **DexRouter** | Interface | | | |
| └ | factory | External ❗ | | NO ❗ |
| └ | WETH | External ❗ | | NO ❗ |
| └ | addLiquidityETH | External ❗ | 💵 | NO ❗ |
| └ | swapExactTokensForETHSupportingFeeOnTransferTokens | External ❗ | 🛑 | NO ❗ |
| | | | | |
| **AsianPepe** | Implementation | ERC20, Ownable | | |
| └ | <Constructor> | Public ❗ | 🛑 | ERC20 |
| └ | initialize | External ❗ | 🛑 | onlyOwner |
| └ | setMarketingWallet | External ❗ | 🛑 | onlyOwner |
| └ | setSwapTokensAtAmount | External ❗ | 🛑 | onlyOwner |
| └ | toggleSwapping | External ❗ | 🛑 | onlyOwner |
| └ | setWhitelistStatus | External ❗ | 🛑 | onlyOwner |
| └ | checkWhitelist | External ❗ | | NO ❗ |
| └ | updateBuyFees | Public ❗ | 🛑 | onlyOwner |
| └ | updateSellFees | Public ❗ | 🛑 | onlyOwner |
| └ | updateTransferFees | Public ❗ | 🛑 | onlyOwner |
| └ | updateMaxBuy | Public ❗ | 🛑 | onlyOwner |
| └ | updateMaxSell | Public ❗ | 🛑 | onlyOwner |
| └ | _takeTax | Internal 🔒 | 🛑 | |
| └ | enableTrading | External ❗ | 🛑 | onlyOwner |
| └ | _transfer | Internal 🔒 | 🛑 | |
| └ | internalSwap | Internal 🔒 | 🛑 | |
| └ | swapAndLiquify | Internal 🔒 | 🛑 | |
| └ | swapToETH | Internal 🔒 | 🛑 | |
| └ | addLiquidity | Private 🔐 | 🛑 | |
| └ | withdrawStuckETH | External ❗ | 🛑 | onlyOwner |
| └ | withdrawStuckTokens | External ❗ | 🛑 | onlyOwner |
| └ | <Receive Ether> | External ❗ | 💵 | NO ❗ |

| Symbol | Meaning |
|:---|:---|
| 🛑 | Function can modify state |
| 💵 | Function is payable |

# POINTS TO NOTE

- **Owner is able to update buy/sell/transfer fees (0-15%)**
- Owner is not able to blacklist an address
- Owner is not able to disable buy/sell/transfers
- Owner is not able to set max wallet limit and minimum wallet limits
- Owner is not able to mint new tokens
- **Owner must enable trades manually**

# STATIC ANALYSIS

```
Context._msgData() (contracts/Token.sol#25-27) is never used and should be removed
ERC20._burn(address,uint256) (contracts/Token.sol#735-751) is never used and should be removed
SafeMath.add(uint256,uint256) (contracts/Token.sol#242-244) is never used and should be removed
SafeMath.div(uint256,uint256) (contracts/Token.sol#284-286) is never used and should be removed
SafeMath.div(uint256,uint256,string) (contracts/Token.sol#336-341) is never used and should be removed
SafeMath.mod(uint256,uint256) (contracts/Token.sol#300-302) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (contracts/Token.sol#358-363) is never used and should be removed
SafeMath.mul(uint256,uint256) (contracts/Token.sol#270-272) is never used and should be removed
SafeMath.sub(uint256,uint256) (contracts/Token.sol#256-258) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (contracts/Token.sol#317-322) is never used and should be removed
SafeMath.tryAdd(uint256,uint256) (contracts/Token.sol#171-177) is never used and should be removed
SafeMath.tryDiv(uint256,uint256) (contracts/Token.sol#213-218) is never used and should be removed
SafeMath.tryMod(uint256,uint256) (contracts/Token.sol#225-230) is never used and should be removed
SafeMath.tryMul(uint256,uint256) (contracts/Token.sol#196-206) is never used and should be removed
SafeMath.trySub(uint256,uint256) (contracts/Token.sol#184-189) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (contracts/Token.sol#8) allows old versions
Pragma version^0.8.0 (contracts/Token.sol#37) allows old versions
Pragma version^0.8.0 (contracts/Token.sol#120) allows old versions
Pragma version^0.8.0 (contracts/Token.sol#153) allows old versions
Pragma version^0.8.0 (contracts/Token.sol#373) allows old versions
Pragma version^0.8.0 (contracts/Token.sol#461) allows old versions
Pragma version^0.8.17 (contracts/Token.sol#836) necessitates a version too recent to be trusted. Consider deploying with 0.6.12/0.7.6/0.8.16
solc-0.8.20 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in AsianPepe.internalSwap() (contracts/Token.sol#1027-1056):
        - (success) = address(marketingWallet).call{value: address(this).balance}() (contracts/Token.sol#1054)
Low level call in AsianPepe.withdrawStuckETH() (contracts/Token.sol#1086-1089):
        - (success) = address(msg.sender).call{value: address(this).balance}() (contracts/Token.sol#1087)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function DexRouter.WETH() (contracts/Token.sol#845) is not in mixedCase
Parameter AsianPepe.setMarketingWallet(address)._newmarketing (contracts/Token.sol#924) is not in mixedCase
Parameter AsianPepe.setSwapTokensAtAmount(uint256)._newAmount (contracts/Token.sol#929) is not in mixedCase
Parameter AsianPepe.setWhitelistStatus(address,bool)._wallet (contracts/Token.sol#942) is not in mixedCase
Parameter AsianPepe.setWhitelistStatus(address,bool)._status (contracts/Token.sol#942) is not in mixedCase
Parameter AsianPepe.checkWhitelist(address)._wallet (contracts/Token.sol#947) is not in mixedCase
Parameter AsianPepe.swapAndLiquify(uint256)._amount (contracts/Token.sol#1058) is not in mixedCase
Parameter AsianPepe.swapToETH(uint256)._amount (contracts/Token.sol#1069) is not in mixedCase
Parameter AsianPepe.addLiquidity(uint256,uint256).ETHAmount (contracts/Token.sol#1079) is not in mixedCase
Parameter AsianPepe.withdrawStuckTokens(address).erc20_token (contracts/Token.sol#1091) is not in mixedCase
Constant AsianPepe._totalSupply (contracts/Token.sol#871) is not in UPPER_CASE_WITH_UNDERSCORES
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
```

**Result => A static analysis of contract's source code has been performed using slither,**

**No major issues were found in the output**

# FUNCTIONAL TESTING

**1- Adding liquidity (passed):**

https://testnet.bscscan.com/tx/0xc2a15ac3cc8437c1cc930014ab4a03ba7e5cf9180e35e703b7f35b4b98c96914

**2- Buying when excluded from fees (0% tax) (passed):**

https://testnet.bscscan.com/tx/0xd978315113178a0724ee1b48d68294ab32ebc793a352557dea66bef1996a0864

**3- Selling when excluded from fees (0% tax) (passed):**

https://testnet.bscscan.com/tx/0x857b80b2948e9a42e16cdd21532895f79c0e004cca4fc69d8e112cb92e66f6e6

**4- Transferring when excluded from fees (0% tax) (passed):**

https://testnet.bscscan.com/tx/0x1512bfdc89167244c15d3858df14782394511593036 4ffe3d958ce7e2768ef76

**5- Buying(0-15% tax) (passed):**

https://testnet.bscscan.com/tx/0x6b4a80e807d3a1e1289ac08ce5c1428aeecf1b1fe5bfed4204544ef312f351b3

**6- Selling (0-15% tax) (passed):**

https://testnet.bscscan.com/tx/0x51d3c5c78563ebcc12d8eb18e6e077207cfe620f5414229c0048da49bc666ff3

# FUNCTIONAL TESTING

**4- Transferring (0-15% tax) (passed):**

https://testnet.bscscan.com/tx/0xc65d25c42216a75ec15e8a000543c78ad1d4bf4457658f2656f9e0ee2fb4cbfa

**4- Internal swap (auto liquidity | marketing BNB) (passed):**

https://testnet.bscscan.com/tx/0x51d3c5c78563ebcc12d8eb18e6e077207cfe620f5414229c0048da49bc666ff3

# MANUAL TESTING

## Centralization – Enabling Trades

**Severity**: High

**function**: enableTrading

**Status:** Not Resolved

**Overview:**

Owner of the contract must enable trades manually for investors, otherwise no one would be able to buy/sell/transfer their tokens.

```
function enableTrading() external onlyOwner {
    tradingEnabled = true;
}
```

**Suggestion**

Its suggested to either enable trades prior to presale, or transfer ownership of the contract to a certified pinsksale safu developer to guearanee enabling of trades.

# MANUAL TESTING

## Centralization – Excessive Fees

**Severity**: Medium

**function**: updateBuyFees – updateSellFees - updateTransferFees

**Status:** Not Resolved

**Overview:**

Owner is able to change buy/sell/transfer fees within 0-15% each.

```
function updateBuyFees(uint256 marketing, uint256 liquidity) public onlyOwner {
    buyTaxes.liquidityTax = liquidity;
    buyTaxes.marketingTax = marketing;
    require(marketing + liquidity <= 15, "Can't set fees higher than 15%");
    totalBuyFees = marketing + liquidity;
}

function updateSellFees(uint256 marketing, uint256 liquidity) public onlyOwner {
    sellTaxes.liquidityTax = liquidity;
    sellTaxes.marketingTax = marketing;
    require(marketing + liquidity <= 15, "Can't set fees higher than 15%");
    totalSellFees = marketing + liquidity;
}

function updateTransferFees(uint256 marketing, uint256 liquidity) public onlyOwner {
    transferTaxes.liquidityTax = liquidity;
    transferTaxes.marketingTax = marketing;
    require(marketing + liquidity <= 15, "Can't set fees higher than 15%");
    totalTransferFees = marketing + liquidity;
}
```

### Suggestion

Its suggested to keep fees within 0-10%

0% <= buy fees <= 10%

0% <= sell fees <= 10%

0% <= transfer fees <= 10%

# MANUAL TESTING

## Centralization – Trade limits

**Severity**: Medium

**function**: updateMaxBuy - updateMaxSell

**Status:** Not Resolved

**Overview:**

Owner is able to set maximum buy and sell limits. This limits can not be set to 0 and are always more than 1% of supply

```
    function updateMaxBuy(uint256 maxBuyAmount) public onlyOwner {
        maxBuy = maxBuyAmount;
        require(maxBuyAmount >= _totalSupply / 100, "Can't update max buy to lower than 1% of
supply");
    }

    function updateMaxSell(uint256 maxSellAmount) public onlyOwner {
        maxSell = maxSellAmount;
        require(maxSellAmount >= _totalSupply / 100, "Can't update max sell to lower than 1% of
supply");
    }
```

# DISCLAIMER

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed. The Auditace team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Auditace receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token. The Auditace team disclaims any liability for the resulting losses.

# ABOUT AUDITACE

We specializes in providing thorough and reliable audits for Web3 projects. With a team of experienced professionals, we use cutting-edge technology and rigorous methodologies to evaluate the security and integrity of blockchain systems. We are committed to helping our clients ensure the safety and transparency of their digital assets and transactions.


**https://auditace.tech/**

**https://t.me/Audit_Ace**

**https://twitter.com/auditace_**

**https://github.com/Audit-Ace**