auditace.tech

AuditAce

FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

HALLOWEEN

DATED : 28 October 23'

AUDIT SUMMARY

Project name – HALLOWEEN

Date: 28 October 2023

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

Audit Status: Passed

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	0	0	0	0
Acknowledged	0	0	0	0	0
Resolved	0	2	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/0x77dE11a0c82cdFa9cBE00b5BEe7B7D6105DABF8d

Token Information

Token Address :
0x3153B9E8525A5Bc9BC9514579C93EF02F204e162

Name: HALLOWEEN

Symbol: $HLW

Decimals: 18

Network: Binance smart chain

Token Type: BEP20

Owner: 0x0FFEEBB4ce58bB07CEc258086Fb50B2E155392D7

Deployer:
0x0FFEEBB4ce58bB07CEc258086Fb50B2E155392D7

Token Supply: 1,000,000,000

Checksum:
af747e29e250fa2181f56bced993ee804a62665c

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/0x77dE11a0c82cdFa9cBE00b5BEe7B7D6105DABF8d

TOKEN OVERVIEW

buy fee: 0-5%

Sell fee: 0-5%

transfer fee: 0-5%

Fee Privilege: Owner

Ownership: Owned

Minting: None

Max Tx: No

Blacklist: No

Other Privileges:

- Initial distribution of the tokens

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

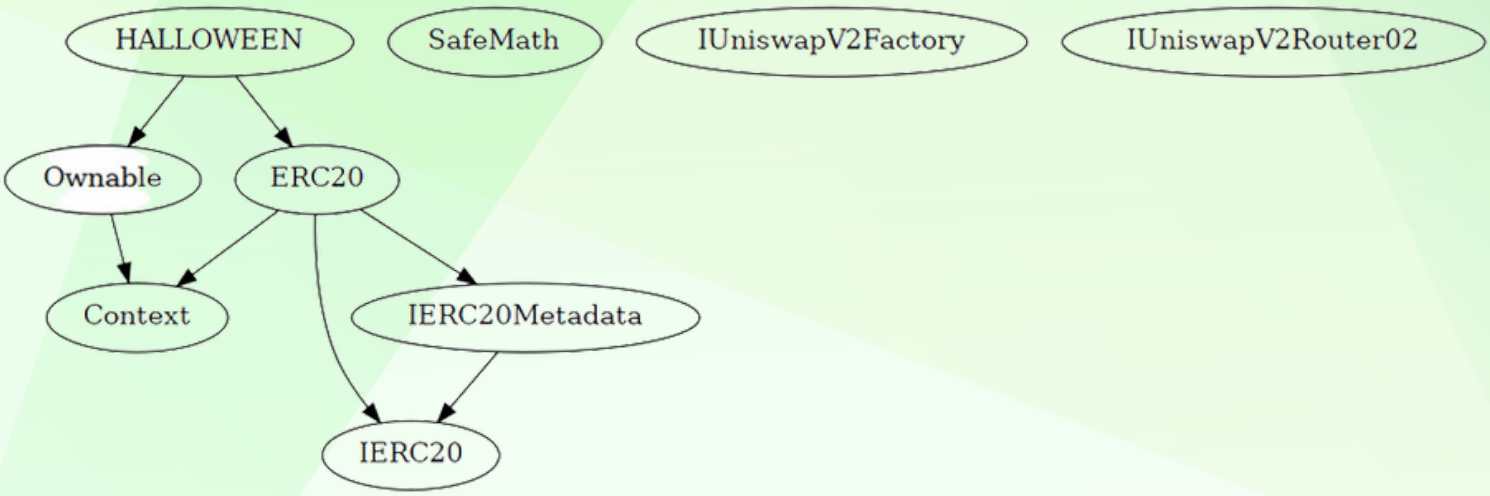
CLASSIFICATION OF RISK

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	2
Medium-Risk	0
Low-Risk	0
Gas Optimization / Suggestions	0

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to adjust current fees (5% buy and 5% sell)

- Owner is not able to blacklist an arbitrary wallet
- Owner is not able to disable trades
- Owner is not able to mint new tokens
- Owner is not able to set maximum wallet and maximum buy/sell/transfer limits

STATIC ANALYSIS

```
Reentrancy in HALLOWEEN.swapBack() (contracts/Token.sol#695-739):
        External calls:
        - swapTokensForEth(amountToSwapForETH) (contracts/Token.sol#716)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (contracts/Token.sol#671-677)
        - addLiquidity(liquidityTokens,ethForLiquidity) (contracts/Token.sol#730)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,devWallet,block.timestamp) (contracts/Token.sol#685-692)
        External calls sending eth:
        - addLiquidity(liquidityTokens,ethForLiquidity) (contracts/Token.sol#730)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,devWallet,block.timestamp) (contracts/Token.sol#685-692)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (contracts/Token.sol#269)
                - addLiquidity(liquidityTokens,ethForLiquidity) (contracts/Token.sol#730)
        - SwapAndLiquify(amountToSwapForETH,ethForLiquidity,tokensForLiquidity) (contracts/Token.sol#731-735)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
HALLOWEEN._transfer(address,address,uint256) (contracts/Token.sol#550-660) has a high cyclomatic complexity (13).
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#cyclomatic-complexity
INFO:Detectors:
Context._msgData() (contracts/Token.sol#18-20) is never used and should be removed
ERC20._burn(address,uint256) (contracts/Token.sol#243-258) is never used and should be removed
SafeMath.add(uint256,uint256) (contracts/Token.sol#286-288) is never used and should be removed
SafeMath.div(uint256,uint256,string) (contracts/Token.sol#317-326) is never used and should be removed
SafeMath.mod(uint256,uint256) (contracts/Token.sol#302-304) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (contracts/Token.sol#328-337) is never used and should be removed
SafeMath.sub(uint256,uint256,string) (contracts/Token.sol#306-315) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
Pragma version^0.8.17 (contracts/Token.sol#7) allows old versions
solc-0.8.17 is not recommended for deployment
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity
INFO:Detectors:
Low level call in HALLOWEEN.swapBack() (contracts/Token.sol#695-739):
        - (success,None) = address(devWallet).call{value: address(this).balance}() (contracts/Token.sol#738)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Router02.WETH() (contracts/Token.sol#350) is not in mixedCase
Parameter HALLOWEEN.updateFees(uint256,uint256,uint256,uint256)._buyLiquidityFee (contracts/Token.sol#476) is not in mixedCase
Parameter HALLOWEEN.updateFees(uint256,uint256,uint256,uint256)._buyMarketingFee (contracts/Token.sol#477) is not in mixedCase
Parameter HALLOWEEN.updateFees(uint256,uint256,uint256,uint256)._sellLiquidityFee (contracts/Token.sol#478) is not in mixedCase
Parameter HALLOWEEN.updateFees(uint256,uint256,uint256,uint256)._sellMarketingFee (contracts/Token.sol#479) is not in mixedCase
Variable HALLOWEEN._isExcludedMaxTransactionAmount (contracts/Token.sol#404) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
HALLOWEEN.constructor() (contracts/Token.sol#420-465) uses literals with too many digits:
        - swapTokensAtAmount = (totalSupply * 1000000000) / 1 (contracts/Token.sol#443)
HALLOWEEN.updateSwapTokensAtAmount(uint256) (contracts/Token.sol#496-509) uses literals with too many digits:
        - require(bool,string)(newAmount >= (totalSupply() * 1) / 100000,Swap amount cannot be lower than 0.001% total supply.) (contracts/Token.sol#499-502)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
HALLOWEEN.devWallet (contracts/Token.sol#381) should be immutable
HALLOWEEN.maxTransactionAmount (contracts/Token.sol#383) should be immutable
HALLOWEEN.maxWallet (contracts/Token.sol#385) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
INFO:Slither:./contracts/Token.sol analyzed (9 contracts with 88 detectors), 35 result(s) found
```

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

No major issues were found in the output

CONTRACT ASSESMENT

Contract	Type	Bases		
└	**Function Name**	**Visibility**	**Mutability**	**Modifiers**
Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO ❗
└	owner	Public ❗		NO ❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_transferOwnership	Internal 🔒	🛑	
IERC20	Interface			
└	totalSupply	External ❗		NO ❗
└	balanceOf	External ❗		NO ❗
└	transfer	External ❗	🛑	NO ❗
└	allowance	External ❗		NO ❗
└	approve	External ❗	🛑	NO ❗
└	transferFrom	External ❗	🛑	NO ❗
IERC20Metadata	Interface	IERC20		
└	name	External ❗		NO ❗
└	symbol	External ❗		NO ❗
└	decimals	External ❗		NO ❗
ERC20	Implementation	Context, IERC20, IERC20Metadata		
└	<Constructor>	Public ❗	🛑	NO ❗
└	name	Public ❗		NO ❗
└	symbol	Public ❗		NO ❗
└	decimals	Public ❗		NO ❗
└	totalSupply	Public ❗		NO ❗
└	balanceOf	Public ❗		NO ❗
└	transfer	Public ❗	🛑	NO ❗
└	allowance	Public ❗		NO ❗
└	approve	Public ❗	🛑	NO ❗

CONTRACT ASSESMENT

└	transferFrom	Public ❗	🛑	NO ❗
└	increaseAllowance	Public ❗	🛑	NO ❗
└	decreaseAllowance	Public ❗	🛑	NO ❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Internal 🔒	🛑	
└	_approve	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	
SafeMath	Library				
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	sub	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
IUniswapV2Factory	Interface				
└	createPair	External ❗	🛑	NO ❗
IUniswapV2Router02	Interface				
└	factory	External ❗		NO ❗
└	WETH	External ❗		NO ❗
└	addLiquidityETH	External ❗	💵	NO ❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO ❗

CONTRACT ASSESMENT

HALLOWEEN	Implementation	ERC20, Ownable		
└	<Constructor>	Public ❗	🛑	ERC20
└	<Receive Ether>	External ❗	💵	NO❗
└	enableTrading	External ❗	🛑	onlyOwner
└	updateFees	External ❗	🛑	onlyOwner
└	removeLimits	External ❗	🛑	onlyOwner
└	updateSwapTokensAtAmount	External ❗	🛑	onlyOwner
└	excludeFromMaxTransaction	Public ❗	🛑	onlyOwner
└	updateSwapEnabled	External ❗	🛑	onlyOwner
└	excludeFromFees	Public ❗	🛑	onlyOwner
└	setAutomatedMarketMakerPair	Public ❗	🛑	onlyOwner
└	_setAutomatedMarketMakerPair	Private 🔐	🛑	
└	isExcludedFromFees	Public ❗		NO❗
└	_transfer	Internal 🔒	🛑	
└	swapTokensForEth	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	
└	swapBack	Private 🔐	🛑	

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

FUNCTIONAL TESTING

1- Adding liquidity (passed):
https://testnet.bscscan.com/tx/0x39906540368236c457de12d38f16bfbfac7e933d01454f27bd2191d32f372653

2- Buying when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0x670dc1e35f05adb9948569502b0c12ddc69a9ffb9bd3c20cb4d3c73f3c13c5af

3- Selling when excluded (0% tax) (passed):
https://testnet.bscscan.com/tx/0xfd9b590890e6e3c9d73928c27b55f5651b9e14856163890913868a85bb6f214c

4- Transferring when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0xf18a97d544cacb7e7a2b6daf521bd42d6ebb4fc0ccc98cc5543084c740ff99a6

5- Buying when not excluded from fees (tax 0-100%) (passed):
https://testnet.bscscan.com/tx/0x9ff0ede588761f0524481dddf80639f5292b5b932c8a90fac39cbb972920e3d0

6- Selling when not excluded from fees (tax 0-100%) (passed):
https://testnet.bscscan.com/tx/0x83f0096dd46e17a72a1fb981e8cce350061c52dc97a13e85162ca014e997f24c

7- Transferring when not excluded from fees (0-100% tax)(passed):
https://testnet.bscscan.com/tx/0x071f5d69a5c937bf0b118cfcb7830e19cf37b1462d65766f0cec19bd2b7f423f

8- Internal swap (BNB set to dev wallet + Auto-liquidity) (passed):
https://testnet.bscscan.com/tx/0x83f0096dd46e17a72a1fb981e8cce350061c52dc97a13e85162ca014e997f24c

MANUAL TESTING

Centralization – Enabling Trades
Severity: High
function: enableTrading
Status: Resolved (ownership renounced and trades enabled)

Overview:
The enableTrading function permits only the contract owner to activate trading capabilities. Until this function is executed, no investors can buy, sell, or transfer their tokens. This places a high degree of control and centralization in the hands of the contract owner.

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

Suggestion
To reduce centralization and potential manipulation, consider one of the following approaches:
1.Automatically enable trading after a specified condition, such as the completion of a presale, is met.
2.If manual activation is still desired, consider transferring the ownership of the contract to a trustworthy, third-party entity like a certified "PinkSale Safu" developer. This can provide investors with more confidence in the eventual activation of trading capabilities, mitigating concerns of potential bad faith actions by the original owner

MANUAL TESTING

Centralization – Updating fees
Severity: High
function: updateFees
Status: Resolved (ownership renounced)

Overview:
Owner is able to adjust buy/sell/transfer fees within 0-100%. there are no upper bounds for maximum amount of fees

```
function updateFees(
    uint256 _buyLiquidityFee,
    uint256 _buyMarketingFee,
    uint256 _sellLiquidityFee,
    uint256 _sellMarketingFee
) external onlyOwner {
    buyLiquidityFee = _buyLiquidityFee;
    buyMarketingFee = _buyMarketingFee;
    buyTotalFees = buyLiquidityFee + buyMarketingFee;
    sellLiquidityFee = _sellLiquidityFee;
    sellMarketingFee = _sellMarketingFee;
    sellTotalFees = sellLiquidityFee + sellMarketingFee;
}
```

Suggestion
set an upper bound for maximum amount of fees

```
function updateFees(
    uint256 _buyLiquidityFee,
    uint256 _buyMarketingFee,
    uint256 _sellLiquidityFee,
    uint256 _sellMarketingFee
) external onlyOwner {
    buyLiquidityFee = _buyLiquidityFee;
    buyMarketingFee = _buyMarketingFee;
    buyTotalFees = buyLiquidityFee + buyMarketingFee;
    sellLiquidityFee = _sellLiquidityFee;
    sellMarketingFee = _sellMarketingFee;
    sellTotalFees = sellLiquidityFee + sellMarketingFee;
    require(buyTotalFees <= 10 && sellTotalFees <= 10, "Can't set fees more than 10%");
}
```

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