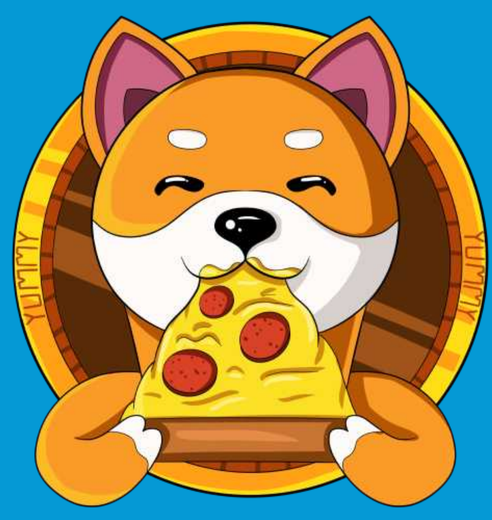
QuillAudits

Audit Report
June, 2021



Contents

Scope of Audit

The scope of this audit was to analyze and document the YUMMY smart contract codebase for quality, security, and correctness.

Check Vulnerabilities

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- DoS with Block Gas Limit
- Transaction-Ordering Dependence
- Use of tx.origin
- Exception disorder
- Gasless send
- Balance equality
- Byte array
- Transfer forwards all gas
- ERC20 API violation
- Malicious libraries
- Compiler version not fixed
- Redundant fallback function
- Send instead of transfer
- Style guide violation
- Unchecked external call
- Unchecked math
- Unsafe type inference
- Implicit visibility level

Techniques and Methods

Throughout the audit of smart contracts care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of ERC-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods and tools were used to review all the smart contracts.

Structural Analysis
In this step we have analyzed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.
SmartCheck.

Static Analysis
Static Analysis of Smart Contracts was done to identify contract vulnerabilities. In this step a series of automated tools are used to test security of smart contracts.

Code Review / Manual Analysis
Manual Analysis or review of code was done to identify new vulnerability or verify the vulnerabilities found during the static analysis. Contracts were completely manually analyzed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of automated analysis were manually verified.

Gas Consumption
In this step we have checked the behaviour of smart contracts in production. Checks were done to know how much gas gets consumed and possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit
Remix IDE, Truffle, Truffle Team, Ganache, Solhint, Mythril, Slither, SmartCheck.

Issue Categories

Every issue in this report has been assigned with a severity level. There are four levels of severity and each of them has been explained below.

High severity issues

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality and we recommend these issues to be fixed before moving to a live environment.

Medium level severity issues

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems and they should still be fixed.

Low level severity issues

Low level severity issues can cause minor impact and or are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational

These are severity four issues which indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Number of issues per severity

	High	Medium	Low	Informational
Open	0	0	2	1
Closed	0	0	1	0

Introduction

During the period of MAY 31, 2021 to June 02, 2021 - QuillAudits Team performed a security audit for YUMMY smart contracts.

The code for the audit was taken from following the official link: https://bscscan.com/address/0x05f2df7b3d612a23fe12162a6c996447dce728a5

Issues Found – Code Review / Manual Testing

High severity issues

No Issues found.

Medium severity issues

No Issues found.

Low level severity issues

1. Transfer() function is utilized

Line	Function Names

987-990	function sendBNBToCharity(uint256 amount) private
	{
	swapTokensForEth(amount);
	_charityWalletAddress.transfer(address(this).balance);
	}

Description:

.transfer() and **.send()** forward exactly 2,300 gas to the recipient. The goal of this hardcoded gas stipend was to prevent reentrancy vulnerabilities, but this only makes sense under the assumption that gas costs are constant. Recently EIP 1884 was included in the Istanbul hard fork. One of the changes included in EIP 1884 is an increase in the gas cost of the SLOAD operation, causing a contract's fallback function to cost more than 2300 gas.

```
// bad
contract Vulnerable {
    function withdraw(uint256 amount) external {
        // This forwards 2300 gas, which may not be enough if the recipient
        // is a contract and gas costs change.
        msg.sender.transfer(amount);
    }
}

// good
contract Fixed {
    function withdraw(uint256 amount) external {
        // This forwards all available gas. Be sure to check the return value!
        (bool success, ) = msg.sender.call.value(amount)("");
        require(success, "Transfer failed.");
    }
}
```

Recommendation:
It's recommended to stop using .transfer() and .send() and instead use .call().

References:
External Calls

Status: Open

The auditee confirmed that it is impossible to update the contract since the ownership has been renounced.

However, the auditee has addressed this issue by ensuring that the msg.sender parameter is always a standard wallet address rather than the contract address.

2. Missing zero address validation

Line	Function Names
741-742	constructor (address payable charityWalletAddress) public { _charityWalletAddress = charityWalletAddress;

Description:
Missing zero address validation has been found.

Recommendation:
Check that the address is not zero.

References:
Hundreds of Millions of Dollars Locked at Ethereum 0x0 Address and Smart Contracts' Addresses.

Status: Closed

The auditee confirmed that it is impossible to update the contract since the ownership has been renounced.

As a result, the charityWalletAddress cannot be modified, and the problem has been resolved.

3. Block timestamp

Line	Function Names
1089	uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount, 0, // accept any amount of ETH path, address(this), block.timestamp);

1104	function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
	// approve token transfer to cover all possible scenarios
	_approve(address(this), address(uniswapV2Router), tokenAmount);
	
	// add the liquidity
	uniswapV2Router.addLiquidityETH{value: ethAmount}(
	address(this),
	tokenAmount,
	0, // slippage is unavoidable
	0, // slippage is unavoidable
	owner(),

```
            block.timestamp
        );
    }
```

Description:
Do not rely on **block.timestamp**, **now** and **blockhash** as a source of randomness, unless you know what you are doing.

Both the timestamp and the block hash can be influenced by miners to some degree. Bad actors in the mining community can, for example, run a casino payout function on a chosen hash and just retry a different hash if they did not receive any money.

The current block timestamp must be strictly larger than the timestamp of the last block. Still, the only guarantee is that it will be somewhere between the timestamps of two consecutive blocks in the canonical chain.

Remediation:
Avoid relying on block.timestamp. The miner could cheat in the timestamp by a tolerance of 900 seconds; therefore, if the contract checks outside this interval, the contract is safe.

References:
Block-Protocol-2.0
Block and Transaction Properties

Status: Open

The auditee confirmed that it is impossible to update the contract since the ownership has been renounced.

However, the auditee is aware of the problem and has confirmed that the block.timestamp in this contract will not be used as a source of randomness.

Informational

1. State variables that could be declared constant

Line	Function Names
699	uint256 private _tTotal = 100000000000 * 10**1 * 10**9;

703-707	string private _name = "YUMMY";
	string private _symbol = "YUMMY";
	uint8 private _decimals = 9;
	uint256 public _taxFee = 3;

710	uint256 public _liquidityFee = 6;
720-721	uint256 public _maxTxAmount = 1000000000 * 10**1 * 10**9; uint256 private numTokensSellToAddToLiquidity = 300000000 * 10**1 * 10**9;

Description:
Compared to regular state variables, the gas costs of constant and immutable variables are much lower. Constant state variables should be declared constant to save gas.

Remediation:
Add the constant attributes to state variables that never change.

References:
Constant and Immutable State Variables

Status: Open

The auditee confirmed that it is impossible to update the contract since the ownership has been renounced.

However, the auditee decided not to proceed with the code optimization remediation because the gas cost for the current setup is acceptable.

Functional test

Function Names	Testing results
includeInReward	Passed
excludeFromReward	Passed
tokenFromReflection	Passed
reflectionFromToken	Passed
deliver	Passed
transferFrom	Passed
transfer	Passed
lock	Passed
approve	Passed

Automated Testing

Slither

```
INFO:Detectors:
YUMMYToken.sendBNBToCharity(uint256) (yummy.sol#990-993) sends eth to arbitrary user
        Dangerous calls:
        - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#functions-that-send-ether-to-arbitrary-destinations
INFO:Detectors:
Reentrancy in YUMMYToken._transfer(address,address,uint256) (yummy.sol#1007-1051):
        External calls:
        - swapAndLiquify(contractTokenBalance) (yummy.sol#1038)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (yummy.sol#1087-1093
)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (yummy.sol#1038)
                - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (yummy.sol#954)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (yummy.sol#1134)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (yummy.sol#1143)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (yummy.sol#1154)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (yummy.sol#870)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (yummy.sol#1135)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (yummy.sol#1145)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (yummy.sol#1156)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (yummy.sol#872)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _rTotal = _rTotal.sub(rFee) (yummy.sol#909)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (yummy.sol#956)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (yummy.sol#869)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (yummy.sol#1153)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (yummy.sol#1144)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (yummy.sol#871)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities
INFO:Detectors:
YUMMYToken.addLiquidity(uint256,uint256) (yummy.sol#1096-1109) ignores return value by uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),toke
nAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return
INFO:Detectors:
YUMMYToken.allowance(address,address).owner (yummy.sol#786) shadows:
        - Ownable.owner() (yummy.sol#420-422) (function)
YUMMYToken._approve(address,address,uint256).owner (yummy.sol#999) shadows:
        - Ownable.owner() (yummy.sol#420-422) (function)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing
INFO:Detectors:
YUMMYToken.constructor(address).charityWalletAddress (yummy.sol#741) lacks a zero-check on :
                - _charityWalletAddress = charityWalletAddress (yummy.sol#742)
YUMMYToken._setCharityWallet(address).charityWalletAddress (yummy.sol#995) lacks a zero-check on :
                - _charityWalletAddress = charityWalletAddress (yummy.sol#996)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation
INFO:Detectors:
Reentrancy in YUMMYToken._transfer(address,address,uint256) (yummy.sol#1007-1051):
        External calls:
        - swapAndLiquify(contractTokenBalance) (yummy.sol#1038)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (yummy.sol#1087-1093
)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (yummy.sol#1038)
                - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _liquidityFee = _previousLiquidityFee (yummy.sol#983)
                - _liquidityFee = 0 (yummy.sol#978)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _previousLiquidityFee = _liquidityFee (yummy.sol#975)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _previousTaxFee = _taxFee (yummy.sol#974)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _tFeeTotal = _tFeeTotal.add(tFee) (yummy.sol#910)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _taxFee = _previousTaxFee (yummy.sol#982)
                - _taxFee = 0 (yummy.sol#977)
```

```
allowance(address,address) should be declared external:
        - YUMMYToken.allowance(address,address) (yummy.sol#786-788)
approve(address,uint256) should be declared external:
        - YUMMYToken.approve(address,uint256) (yummy.sol#790-793)
transferFrom(address,address,uint256) should be declared external:
        - YUMMYToken.transferFrom(address,address,uint256) (yummy.sol#795-799)
increaseAllowance(address,uint256) should be declared external:
        - YUMMYToken.increaseAllowance(address,uint256) (yummy.sol#801-804)
decreaseAllowance(address,uint256) should be declared external:
        - YUMMYToken.decreaseAllowance(address,uint256) (yummy.sol#806-809)
isExcludedFromReward(address) should be declared external:
        - YUMMYToken.isExcludedFromReward(address) (yummy.sol#811-813)
totalFees() should be declared external:
        - YUMMYToken.totalFees() (yummy.sol#815-817)
deliver(uint256) should be declared external:
        - YUMMYToken.deliver(uint256) (yummy.sol#819-826)
reflectionFromToken(uint256,bool) should be declared external:
        - YUMMYToken.reflectionFromToken(uint256,bool) (yummy.sol#828-837)
excludeFromReward(address) should be declared external:
        - YUMMYToken.excludeFromReward(address) (yummy.sol#845-853)
excludeFromFee(address) should be declared external:
        - YUMMYToken.excludeFromFee(address) (yummy.sol#878-880)
includeInFee(address) should be declared external:
        - YUMMYToken.includeInFee(address) (yummy.sol#882-884)
setSwapAndLiquifyEnabled(bool) should be declared external:
        - YUMMYToken.setSwapAndLiquifyEnabled(bool) (yummy.sol#900-903)
isExcludedFromFee(address) should be declared external:
        - YUMMYToken.isExcludedFromFee(address) (yummy.sol#906-908)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external
INFO:Slither:yummy.sol analyzed (10 contracts with 75 detectors), 126 result(s) found
INFO:Slither:Use https://crytic.io/ to get access to additional detectors and Github integration
```

```
INFO:Detectors:
YUMMYToken.slitherConstructorVariables() (yummy.sol#689-1165) uses literals with too many digits:
        - _tTotal = 100000000000 * 10 ** 1 * 10 ** 9 (yummy.sol#703)
YUMMYToken.slitherConstructorVariables() (yummy.sol#689-1165) uses literals with too many digits:
        - _maxTxAmount = 1000000000 * 10 ** 1 * 10 ** 9 (yummy.sol#724)
YUMMYToken.slitherConstructorVariables() (yummy.sol#689-1165) uses literals with too many digits:
        - numTokensSellToAddToLiquidity = 300000000 * 10 ** 1 * 10 ** 9 (yummy.sol#725)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#too-many-digits
INFO:Detectors:
YUMMYToken._decimals (yummy.sol#709) should be constant
YUMMYToken._name (yummy.sol#707) should be constant
YUMMYToken._symbol (yummy.sol#708) should be constant
YUMMYToken._tTotal (yummy.sol#703) should be constant
YUMMYToken.numTokensSellToAddToLiquidity (yummy.sol#725) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant
INFO:Detectors:
renounceOwnership() should be declared external:
        - Ownable.renounceOwnership() (yummy.sol#439-442)
transferOwnership(address) should be declared external:
        - Ownable.transferOwnership(address) (yummy.sol#448-452)
getUnlockTime() should be declared external:
        - Ownable.getUnlockTime() (yummy.sol#454-456)
lock(uint256) should be declared external:
        - Ownable.lock(uint256) (yummy.sol#459-464)
unlock() should be declared external:
        - Ownable.unlock() (yummy.sol#467-472)
name() should be declared external:
        - YUMMYToken.name() (yummy.sol#760-762)
symbol() should be declared external:
        - YUMMYToken.symbol() (yummy.sol#764-766)
decimals() should be declared external:
        - YUMMYToken.decimals() (yummy.sol#768-770)
totalSupply() should be declared external:
        - YUMMYToken.totalSupply() (yummy.sol#772-774)
transfer(address,uint256) should be declared external:
        - YUMMYToken.transfer(address,uint256) (yummy.sol#781-784)
```

```
        - sendBNBToCharity(portionForFees) (yummy.sol#1073)
                - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
        External calls sending eth:
        - addLiquidity(otherHalfOfLiquify,newBalance) (yummy.sol#1072)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        - sendBNBToCharity(portionForFees) (yummy.sol#1073)
                - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
        Event emitted after the call(s):
        - SwapAndLiquify(halfOfLiquify,newBalance,otherHalfOfLiquify) (yummy.sol#1076)
Reentrancy in YUMMYToken.transferFrom(address,address,uint256) (yummy.sol#795-799):
        External calls:
        - _transfer(sender,recipient,amount) (yummy.sol#796)
                - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (yummy.sol#796)
                - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (yummy.sol#797)
                - _allowances[owner][spender] = amount (yummy.sol#1003)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (yummy.sol#1004)
                - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (yummy.sol#797)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-4
INFO:Detectors:
Variable IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountADesired (yummy.sol#556) is too similar to
IUniswapV2Router01.addLiquidity(address,address,uint256,uint256,uint256,uint256,address,uint256).amountBDesired (yummy.sol#557)
Variable YUMMYToken._transferBothExcluded(address,address,uint256).rTransferAmount (yummy.sol#868) is too similar to YUMMYToken._getValues(uint256).tTransf
erAmount (yummy.sol#914)
Variable YUMMYToken._transferStandard(address,address,uint256).rTransferAmount (yummy.sol#1133) is too similar to YUMMYToken._transferStandard(address,addr
ess,uint256).tTransferAmount (yummy.sol#1133)
Variable YUMMYToken.reflectionFromToken(uint256,bool).rTransferAmount (yummy.sol#834) is too similar to YUMMYToken._transferToExcluded(address,address,uint
256).tTransferAmount (yummy.sol#1142)
Variable YUMMYToken._transferBothExcluded(address,address,uint256).rTransferAmount (yummy.sol#868) is too similar to YUMMYToken._getTValues(uint256).tTrans
ferAmount (yummy.sol#922)
Variable YUMMYToken._getRValues(uint256,uint256,uint256,uint256).rTransferAmount (yummy.sol#930) is too similar to YUMMYToken._transferFromExcluded(address
,address,uint256).tTransferAmount (yummy.sol#1152)
Variable YUMMYToken._transferBothExcluded(address,address,uint256).rTransferAmount (yummy.sol#868) is too similar to YUMMYToken._transferToExcluded(address
,address,uint256).tTransferAmount (yummy.sol#1142)
```

```
                - _liquidityFee = 0 (yummy.sol#978)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _previousLiquidityFee = _liquidityFee (yummy.sol#975)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _previousTaxFee = _taxFee (yummy.sol#974)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _rOwned[address(this)] = _rOwned[address(this)].add(rLiquidity) (yummy.sol#954)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (yummy.sol#1134)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (yummy.sol#1143)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (yummy.sol#1154)
                - _rOwned[sender] = _rOwned[sender].sub(rAmount) (yummy.sol#870)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (yummy.sol#1135)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (yummy.sol#1145)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (yummy.sol#1155)
                - _rOwned[recipient] = _rOwned[recipient].add(rTransferAmount) (yummy.sol#872)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _rTotal = _rTotal.sub(rFee) (yummy.sol#909)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _tFeeTotal = _tFeeTotal.add(tFee) (yummy.sol#910)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _tOwned[address(this)] = _tOwned[address(this)].add(tLiquidity) (yummy.sol#956)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (yummy.sol#869)
                - _tOwned[sender] = _tOwned[sender].sub(tAmount) (yummy.sol#1153)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (yummy.sol#1144)
                - _tOwned[recipient] = _tOwned[recipient].add(tTransferAmount) (yummy.sol#871)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _taxFee = _previousTaxFee (yummy.sol#982)
                - _taxFee = 0 (yummy.sol#977)
        Event emitted after the call(s):
        - Transfer(sender,recipient,tTransferAmount) (yummy.sol#1138)
                - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
        - Transfer(sender,recipient,tTransferAmount) (yummy.sol#1148)
                - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
        - Transfer(sender,recipient,tTransferAmount) (yummy.sol#1158)
                - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
        - Transfer(sender,recipient,tTransferAmount) (yummy.sol#875)
                - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
Reentrancy in YUMMYToken.swapAndLiquify(uint256) (yummy.sol#1053-1076):
        External calls:
        - sendBNBToCharity(portionForFees) (yummy.sol#1073)
```

```
INFO:Detectors:
Low level call in Address.sendValue(address,uint256) (yummy.sol#299-305):
        - (success) = recipient.call{value: amount}() (yummy.sol#303)
Low level call in Address._functionCallWithValue(address,bytes,uint256,string) (yummy.sol#365-386):
        - (success,returndata) = target.call{value: weiValue}(data) (yummy.sol#369)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls
INFO:Detectors:
Function IUniswapV2Pair.DOMAIN_SEPARATOR() (yummy.sol#511) is not in mixedCase
Function IUniswapV2Pair.PERMIT_TYPEHASH() (yummy.sol#512) is not in mixedCase
Function IUniswapV2Pair.MINIMUM_LIQUIDITY() (yummy.sol#529) is not in mixedCase
Function IUniswapV2Router01.WETH() (yummy.sol#551) is not in mixedCase
Parameter YUMMYToken.setSwapAndLiquifyEnabled(bool)._enabled (yummy.sol#900) is not in mixedCase
Parameter YUMMYToken.calculateTaxFee(uint256)._amount (yummy.sol#959) is not in mixedCase
Parameter YUMMYToken.calculateLiquidityFee(uint256)._amount (yummy.sol#965) is not in mixedCase
Function YUMMYToken._setCharityWallet(address) (yummy.sol#995-997) is not in mixedCase
Variable YUMMYToken._taxFee (yummy.sol#711) is not in mixedCase
Variable YUMMYToken._liquidityFee (yummy.sol#714) is not in mixedCase
Variable YUMMYToken._charityWalletAddress (yummy.sol#719) is not in mixedCase
Variable YUMMYToken._maxTxAmount (yummy.sol#724) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions
INFO:Detectors:
Redundant expression "this (yummy.sol#245)" inContext (yummy.sol#239-248)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements
INFO:Detectors:
Reentrancy in YUMMYToken._transfer(address,address,uint256) (yummy.sol#1007-1051):
        External calls:
        - swapAndLiquify(contractTokenBalance) (yummy.sol#1038)
                - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (yummy.sol#1038)
                - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        State variables written after the call(s):
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _liquidityFee = _previousLiquidityFee (yummy.sol#983)
                - _liquidityFee = 0 (yummy.sol#970)
        - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
                - _previousLiquidityFee = _liquidityFee (yummy.sol#976)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (yummy.sol#796)
                - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (yummy.sol#1004)
                - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (yummy.sol#797)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3
INFO:Detectors:
Ownable.unlock() (yummy.sol#467-472) uses timestamp for comparisons
        Dangerous comparisons:
        - require(bool,string)(now > _lockTime,Contract is locked until 7 days) (yummy.sol#469)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#block-timestamp
INFO:Detectors:
Address.isContract(address) (yummy.sol#272-281) uses assembly
        - INLINE ASM (yummy.sol#279)
Address._functionCallWithValue(address,bytes,uint256,string) (yummy.sol#365-386) uses assembly
        - INLINE ASM (yummy.sol#378-381)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage
INFO:Detectors:
Address._functionCallWithValue(address,bytes,uint256,string) (yummy.sol#365-386) is never used and should be removed
Address.functionCall(address,bytes) (yummy.sol#325-327) is never used and should be removed
Address.functionCall(address,bytes,string) (yummy.sol#335-337) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256) (yummy.sol#350-352) is never used and should be removed
Address.functionCallWithValue(address,bytes,uint256,string) (yummy.sol#360-363) is never used and should be removed
Address.isContract(address) (yummy.sol#272-281) is never used and should be removed
Address.sendValue(address,uint256) (yummy.sol#299-305) is never used and should be removed
Context._msgData() (yummy.sol#244-247) is never used and should be removed
SafeMath.mod(uint256,uint256) (yummy.sol#217-219) is never used and should be removed
SafeMath.mod(uint256,uint256,string) (yummy.sol#233-236) is never used and should be removed
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code
INFO:Detectors:
YUMMYToken._rTotal (yummy.sol#704) is set pre-construction with a non-constant function or state variable:
        - (MAX - (MAX % _tTotal))
YUMMYToken._previousTaxFee (yummy.sol#712) is set pre-construction with a non-constant function or state variable:
        - _taxFee
YUMMYToken._previousLiquidityFee (yummy.sol#715) is set pre-construction with a non-constant function or state variable:
        - _liquidityFee
```

```
        - Transfer(address(0),_msgSender(),_tTotal) (yummy.sol#757)
Reentrancy in YUMMYToken.swapAndLiquify(uint256) (yummy.sol#1053-1076):
        External calls:
        - swapTokensForEth(halfOfLiquify) (yummy.sol#1066)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (yummy.sol#1087-1093
)
        - addLiquidity(otherHalfOfLiquify,newBalance) (yummy.sol#1072)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        External calls sending eth:
        - addLiquidity(otherHalfOfLiquify,newBalance) (yummy.sol#1072)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (yummy.sol#1004)
                - addLiquidity(otherHalfOfLiquify,newBalance) (yummy.sol#1072)
Reentrancy in YUMMYToken.swapAndLiquify(uint256) (yummy.sol#1053-1076):
        External calls:
        - swapTokensForEth(halfOfLiquify) (yummy.sol#1066)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (yummy.sol#1087-1093
)
        - addLiquidity(otherHalfOfLiquify,newBalance) (yummy.sol#1072)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        - sendBNBToCharity(portionForFees) (yummy.sol#1073)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (yummy.sol#1087-1093
)
        External calls sending eth:
        - addLiquidity(otherHalfOfLiquify,newBalance) (yummy.sol#1072)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        - sendBNBToCharity(portionForFees) (yummy.sol#1073)
                - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
        Event emitted after the call(s):
        - Approval(owner,spender,amount) (yummy.sol#1004)
                - sendBNBToCharity(portionForFees) (yummy.sol#1073)
        - SwapAndLiquify(halfOfLiquify,newBalance,otherHalfOfLiquify) (yummy.sol#1075)
Reentrancy in YUMMYToken.transferFrom(address,address,uint256) (yummy.sol#795-799):
        External calls:
        - _transfer(sender,recipient,amount) (yummy.sol#796)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (yummy.sol#1087-1093
)
```

```
Reentrancy in YUMMYToken.transferFrom(address,address,uint256) (yummy.sol#795-799):
        External calls:
        - _transfer(sender,recipient,amount) (yummy.sol#796)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (yummy.sol#1087-1093
)
        External calls sending eth:
        - _transfer(sender,recipient,amount) (yummy.sol#796)
                - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        State variables written after the call(s):
        - _approve(sender,_msgSender(),_allowances[sender][_msgSender()].sub(amount,ERC20: transfer amount exceeds allowance)) (yummy.sol#797)
                - _allowances[owner][spender] = amount (yummy.sol#1003)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-2
INFO:Detectors:
Reentrancy in YUMMYToken._transfer(address,address,uint256) (yummy.sol#1007-1051):
        External calls:
        - swapAndLiquify(contractTokenBalance) (yummy.sol#1038)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (yummy.sol#1087-1093
)
        External calls sending eth:
        - swapAndLiquify(contractTokenBalance) (yummy.sol#1038)
                - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        Event emitted after the call(s):
        - Transfer(sender,recipient,tTransferAmount) (yummy.sol#1138)
                - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
        - Transfer(sender,recipient,tTransferAmount) (yummy.sol#1158)
                - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
        - Transfer(sender,recipient,tTransferAmount) (yummy.sol#1148)
                - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
        - Transfer(sender,recipient,tTransferAmount) (yummy.sol#875)
                - _tokenTransfer(from,to,amount,takeFee) (yummy.sol#1050)
Reentrancy in YUMMYToken.constructor(address) (yummy.sol#741-758):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (yummy.sol#747-748)
        Event emitted after the call(s):
```

```
Reentrancy in YUMMYToken.constructor(address) (yummy.sol#741-768):
        External calls:
        - uniswapV2Pair = IUniswapV2Factory(_uniswapV2Router.factory()).createPair(address(this),_uniswapV2Router.WETH()) (yummy.sol#747-748)
        State variables written after the call(s):
        - _isExcludedFromFee[owner()] = true (yummy.sol#754)
        - _isExcludedFromFee[address(this)] = true (yummy.sol#755)
        - uniswapV2Router = _uniswapV2Router (yummy.sol#751)
Reentrancy in YUMMYToken.swapAndLiquify(uint256) (yummy.sol#1053-1076):
        External calls:
        - swapTokensForEth(halfOfLiquify) (yummy.sol#1066)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (yummy.sol#1087-1093
)
        - addLiquidity(otherHalfOfLiquify,newBalance) (yummy.sol#1072)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        External calls sending eth:
        - addLiquidity(otherHalfOfLiquify,newBalance) (yummy.sol#1072)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        State variables written after the call(s):
        - addLiquidity(otherHalfOfLiquify,newBalance) (yummy.sol#1072)
                - _allowances[owner][spender] = amount (yummy.sol#1003)
Reentrancy in YUMMYToken.swapAndLiquify(uint256) (yummy.sol#1053-1076):
        External calls:
        - swapTokensForEth(halfOfLiquify) (yummy.sol#1066)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (yummy.sol#1087-1093
)
        - addLiquidity(otherHalfOfLiquify,newBalance) (yummy.sol#1072)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        - sendBNBToCharity(portionForFees) (yummy.sol#1073)
                - uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(tokenAmount,0,path,address(this),block.timestamp) (yummy.sol#1087-1093
)
        External calls sending eth:
        - addLiquidity(otherHalfOfLiquify,newBalance) (yummy.sol#1072)
                - uniswapV2Router.addLiquidityETH{value: ethAmount}(address(this),tokenAmount,0,0,owner(),block.timestamp) (yummy.sol#1101-1108)
        - sendBNBToCharity(portionForFees) (yummy.sol#1073)
                - _charityWalletAddress.transfer(address(this).balance) (yummy.sol#992)
        State variables written after the call(s):
        - sendBNBToCharity(portionForFees) (yummy.sol#1073)
                - _allowances[owner][spender] = amount (yummy.sol#1003)
Reentrancy in YUMMYToken.transferFrom(address,address,uint256) (yummy.sol#795-799):
        External calls:
```

Mythril

```
enderphan@enderphan Yummy % myth a yummy.sol
The analysis was completed successfully. No issues were detected.
```

Manticore

```
enderphan@enderphan Yummy % manticore yummy.sol
2021-06-01 10:47:36,170: [12955] m.c.manticore:WARNING: Manticore is only supported on Linux. Proceed at your own risk!
2021-06-01 10:47:36,361: [12955] m.main:INFO: Registered plugins: <class 'manticore.core.plugin.IntrospectionAPIPlugin'>, DetectInvalid, DetectIntegerOverf
low, DetectUninitializedStorage, DetectUninitializedMemory, DetectReentrancySimple, DetectReentrancyAdvanced, DetectUnusedRetVal, DetectSuicidal, DetectDel
egatecall, DetectExternalCallAndLeak, DetectEnvInstruction, DetectManipulableBalance
2021-06-01 10:47:36,362: [12955] m.main:INFO: Beginning analysis
2021-06-01 10:47:36,365: [12955] m.e.manticore:INFO: Starting symbolic create contract
2021-06-01 10:47:36,932: [12955] m.e.manticore:INFO: Failed to build contract None Solidity file must contain exactly one contract or you must select one.
Contracts found: IUniswapV2Router01, YUMMYToken, IERC20, Address, Ownable, IUniswapV2Pair, IUniswapV2Factory, SafeMath, IUniswapV2Router02, Context
2021-06-01 10:47:37,267: [12960] m.c.manticore:INFO: Generated testcase No. 0 - NO STATE RESULT (?)(0 txs)
2021-06-01 10:47:37,268: [12960] m.c.plugin:WARNING: Caught will_solve in state None, but failed to capture its initialization
2021-06-01 10:47:37,300: [12955] m.c.manticore:INFO: Results in /Users/enderphan/Quilhash/Yummy/mcore_h82yh9ut
enderphan@enderphan Yummy % manticore yummy.sol --contract YUMMYToken
2021-06-01 10:47:44,749: [12978] m.c.manticore:WARNING: Manticore is only supported on Linux. Proceed at your own risk!
2021-06-01 10:47:44,926: [12978] m.main:INFO: Registered plugins: <class 'manticore.core.plugin.IntrospectionAPIPlugin'>, DetectInvalid, DetectIntegerOverf
low, DetectUninitializedStorage, DetectUninitializedMemory, DetectReentrancySimple, DetectReentrancyAdvanced, DetectUnusedRetVal, DetectSuicidal, DetectDel
egatecall, DetectExternalCallAndLeak, DetectEnvInstruction, DetectManipulableBalance
2021-06-01 10:47:44,926: [12978] m.main:INFO: Beginning analysis
2021-06-01 10:47:44,930: [12978] m.e.manticore:INFO: Starting symbolic create contract
2021-06-01 15:40:22,359: [12978] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0c' -> 10105199358484917891513682139526534617786838482358775498407859366794786738605 5
2021-06-01 15:42:02,987: [12978] m.e.manticore:INFO: Found a concrete globalsha3 b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\r' -> 97569884405916225051403212656556507955018248777258318895762758824193532385077
2021-06-01 15:43:43,595: [12978] m.e.detectors:WARNING: Potentially reading uninitialized memory at instruction (address: 1084433955801361527881657660549945782571 57187927, offset <BitVecAdd at 7fb8487f82c8-T>)
2021-06-01 15:43:44,727: [12978] m.e.detectors:WARNING: Potentially reading uninitialized memory at instruction (address: 1084433955801361527881657660549945782571 57187927, offset <BitVecAdd at 7fb84f66e728-T>)
2021-06-01 15:43:45,877: [12978] m.e.detectors:WARNING: Potentially reading uninitialized memory at instruction (address: 1084433955801361527881657660549945782571 57187927, offset <BitVecAdd at 7fb84dd0ba98-T>)
2021-06-01 15:43:46,942: [12978] m.e.detectors:WARNING: Potentially reading uninitialized memory at instruction (address: 1084433955801361527881657660549945782571 57187927, offset <BitVecAdd at 7fb84dd0bb88-T>)
2021-06-01 15:43:48,120: [12978] m.e.detectors:WARNING: Potentially reading uninitialized memory at instruction (address: 1084433955801361527881657660549945782571 57187927, offset <BitVecAdd at 7fb84ce53638-T>)
2021-06-01 15:43:49,485: [12978] m.e.detectors:WARNING: Potentially reading uninitialized memory at instruction (address: 1084433955801361527881657660549945782571 57187927, offset <BitVecAdd at 7fb84c9c96d8-T>)
2021-06-01 15:43:50,706: [12978] m.e.detectors:WARNING: Potentially reading uninitialized memory at instruction (address: 1084433955801361527881657660549945782571 57187927, offset <BitVecAdd at 7fb8485dae08-T>)
```

Solhint Linter

Potential violation of Checks-Effects-Interaction pattern in Address._functionCallWithValue(address,bytes,uint256,string): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 361:4:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in YUMMYToken.(address payable): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 737:4:

Check-effects-interaction:

Potential violation of Checks-Effects-Interaction pattern in YUMMYToken.swapTokensForEth(uint256): Could potentially lead to re-entrancy vulnerability. Note: Modifiers are currently not considered by this static analysis.
more
Pos: 1075:4:

Inline assembly:

The Contract uses inline assembly, this is only advised in rare cases. Additionally static analysis modules do not parse inline Assembly, this can lead to wrong analysis results.
more
Pos: 275:8:

Inline assembly:

The Contract uses inline assembly, this is only advised in rare cases. Additionally static analysis modules do not parse inline Assembly, this can lead to wrong analysis results.
more
Pos: 374:16:

Block timestamp:

Use of "now": "now" does not mean current time. "now" is an alias for "block.timestamp". "block.timestamp" can be influenced by miners to a certain degree, be careful.
more
Pos: 458:20:

Block timestamp:

Use of "now": "now" does not mean current time. "now" is an alias for "block.timestamp". "block.timestamp" can be influenced by miners to a certain degree, be careful.

more

Pos: 458:20:

Block timestamp:

Use of "now": "now" does not mean current time. "now" is an alias for "block.timestamp". "block.timestamp" can be influenced by miners to a certain degree, be careful.

more

Pos: 465:16:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.

more

Pos: 1089:12:

Block timestamp:

Use of "block.timestamp": "block.timestamp" can be influenced by miners to a certain degree. That means that a miner can "choose" the block.timestamp, to a certain degree, to change the outcome of a transaction in the mined block.

more

Pos: 1104:12:

Low level calls:

Use of "call": should be avoided whenever possible. It can lead to unexpected behavior if return value is not handled properly. Please use Direct Calls via specifying the called contract's interface.

more

Pos: 299:27:

Low level calls:

Use of "call": should be avoided whenever possible. It can lead to unexpected behavior if return value is not handled properly. Please use Direct Calls via specifying the called contract's interface.

more

Pos: 365:50:

Solidity Static Analysis

```
yummy.sol:5:1: Error: Compiler version ^0.6.12 does not satisfy the r
semver requirement

yummy.sol:458:21: Error: Avoid to make time-based decisions in your
business logic

yummy.sol:465:17: Error: Avoid to make time-based decisions in your
business logic

yummy.sol:507:5: Error: Function name must be in mixedCase

yummy.sol:508:5: Error: Function name must be in mixedCase

yummy.sol:525:5: Error: Function name must be in mixedCase

yummy.sol:547:5: Error: Function name must be in mixedCase

yummy.sol:685:1: Error: Contract has 21 states declarations but allowed
no more than 15

yummy.sol:717:5: Error: Explicitly mark visibility of state

yummy.sol:903:32: Error: Code contains empty blocks

yummy.sol:1089:13: Error: Avoid to make time-based decisions in your
business logic

yummy.sol:1104:13: Error: Avoid to make time-based decisions in your
business logic
```

Closing Summary

Overall, smart contracts are very well written and adhere to guidelines.

No instances of Re-entrancy or Back-Door Entry were found in the contract.

A few issues at various severity levels have been reported during the initial audit, and appropriate comments have been made during the final audit. It is recommended to kindly go through the above-mentioned details.

Disclaimer

Quillhash audit is not a security warranty, investment advice, or an endorsement of the YUMMY platform. This audit does not provide a security or correctness guarantee of the audited smart contracts. The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them. Securing smart contracts is a multistep process. One audit cannot be considered enough. We recommend that the YUMMY Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

QuillAudits

Canada, India, Singapore and United Kingdom

audits.quillhash.com

audits@quillhash.com