



# SMART CONTRACT AUDIT



interfinetwork



hello@interfi.network



<https://interfi.network>



PREPARED FOR

**LUCKYTOAD CONTRACTS**



# INTRODUCTION

Auditing Firm	InterFi Network
Client Firm	LuckyToad
Methodology	Automated Analysis, Manual Code Review
Language	Solidity
Token contract	0xBfB2b6870501a6Ff17121D676A0A45a38c9eeD1e
Router contract	0x410cDF454e85197F9771A5a73b8F0E96421bc945
Blockchain	Ethereum Chain
Centralization	Active ownership
Commit	0ebd9fe9ff018d3113d7e2f6a25fe88cdfa35c11
Website	<a href="http://LuckyToad.dev">http://LuckyToad.dev</a>
Telegram	<a href="http://T.me/luckytoad">http://T.me/luckytoad</a>
Twitter	<a href="http://Twitter.com/luckytoadeth">http://Twitter.com/luckytoadeth</a>
Report Date	March 03, 2023


 Verify the authenticity of this report on our website: <https://www.github.com/interfinetwork>




## EXECUTIVE SUMMARY

InterFi has performed the automated and manual analysis of solidity codes. Solidity codes were reviewed for common contract vulnerabilities and centralized exploits. Here's a quick audit summary:

Status	Critical <span style="color: red;">●</span>	Major <span style="color: orange;">●</span>	Medium <span style="color: yellow;">●</span>	Minor <span style="color: green;">●</span>	Unknown <span style="color: brown;">●</span>
Open	0	0	0	3	1
Acknowledged	0	0	1	4	0
Resolved	0	1	0	1	0
Noteworthy Token Functions	Set Bot, Remove Limits, Set Buy Sell and Transfer Taxes, Airdrop, Set Transaction and Wallet Limits				
Noteworthy Router Functions	Add / Remove Trusted Bot				

 Please note that smart contracts deployed on blockchains aren't resistant to exploits, vulnerabilities and/or hacks. Blockchain and cryptography assets utilize new and emerging technologies. These technologies present a high level of ongoing risks. For a detailed understanding of risk severity, source code vulnerability, and audit limitations, kindly review the audit report thoroughly.

 Please note that centralization privileges regardless of their inherited risk status – constitute an elevated impact on smart contract safety and security.



# TABLE OF CONTENTS

TABLE OF CONTENTS ..... 4

SCOPE OF WORK ..... 5

AUDIT METHODOLOGY ..... 6

RISK CATEGORIES..... 8

CENTRALIZED PRIVILEGES..... 9

AUTOMATED ANALYSIS .....10

INHERITANCE GRAPH.....17

MANUAL REVIEW .....18

DISCLAIMERS.....33

ABOUT INTERFI NETWORK.....36

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI  
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL




## SCOPE OF WORK

InterFi was consulted by LuckyToad to conduct the smart contract audit of their solidity source codes.

The audit scope of work is strictly limited to mentioned solidity file(s) only:

- LuckyToadv3.sol
- ToadRouter03.sol

 If source codes are not deployed on the main net, they can be modified or altered before main-net deployment. Verify the contract's deployment status below:

Public Contract Link	
<a href="https://etherscan.io/address/0xbfb2b6870501a6ff17121d676a0a45a38c9eed1e#code">https://etherscan.io/address/0xbfb2b6870501a6ff17121d676a0a45a38c9eed1e#code</a>	
Contract Name	LuckyToadv3
Compiler Version	0.8.15
License	Unlicensed

Public Contract Link	
<a href="https://etherscan.io/address/0x410cDF454e85197F9771A5a73b8F0E96421bc945#code">https://etherscan.io/address/0x410cDF454e85197F9771A5a73b8F0E96421bc945#code</a>	
Contract Name	ToadRouter03
Compiler Version	0.8.17
License	MIT



# AUDIT METHODOLOGY

Smart contract audits are conducted using a set of standards and procedures. Mutual collaboration is essential to performing an effective smart contract audit. Here's a brief overview of InterFi's auditing process and methodology:

## CONNECT

- The onboarding team gathers source codes, and specifications to make sure we understand the size, and scope of the smart contract audit.

## AUDIT

- Automated analysis is performed to identify common contract vulnerabilities. We may use the following third-party frameworks and dependencies to perform the automated analysis:
  - Remix IDE Developer Tool
  - Open Zeppelin Code Analyzer
  - SWC Vulnerabilities Registry
  - DEX Dependencies, e.g., Pancakeswap, Uniswap
- Simulations are performed to identify centralized exploits causing contract and/or trade locks.
- A manual line-by-line analysis is performed to identify contract issues and centralized privileges.

We may inspect below mentioned common contract vulnerabilities, and centralized exploits:

Centralized Exploits	<ul style="list-style-type: none"><li>○ Token Supply Manipulation</li><li>○ Access Control and Authorization</li><li>○ Assets Manipulation</li><li>○ Ownership Control</li><li>○ Liquidity Access</li><li>○ Stop and Pause Trading</li><li>○ Ownable Library Verification</li></ul>
----------------------	---



## Common Contract Vulnerabilities

- Integer Overflow
- Lack of Arbitrary limits
- Incorrect Inheritance Order
- Typographical Errors
- Requirement Violation
- Gas Optimization
- Coding Style Violations
- Re-entrancy
- Third-Party Dependencies
- Potential Sandwich Attacks
- Irrelevant Codes
- Divide before multiply
- Conformance to Solidity Naming Guides
- Compiler Specific Warnings
- Language Specific Warnings

**REPORT**

- The auditing team provides a preliminary report specifying all the checks which have been performed and the findings thereof.
- The client's development team reviews the report and makes amendments to solidity codes.
- The auditing team provides the final comprehensive report with open and unresolved issues.

**PUBLISH**

- The client may use the audit report internally or disclose it publicly.

 It is important to note that there is no pass or fail in the audit, it is recommended to view the audit as an unbiased assessment of the safety of solidity codes.



## RISK CATEGORIES

Smart contracts are generally designed to hold, approve, and transfer tokens. This makes them very tempting attack targets. A successful external attack may allow the external attacker to directly exploit. A successful centralization-related exploit may allow the privileged role to directly exploit. All risks which are identified in the audit report are categorized here for the reader to review:

Risk Type	Definition
Critical 	These risks could be exploited easily and can lead to asset loss, data loss, asset, or data manipulation. They should be fixed right away.
Major 	These risks are hard to exploit but very important to fix, they carry an elevated risk of smart contract manipulation, which can lead to high-risk severity.
Medium 	These risks should be fixed, as they carry an inherent risk of future exploits, and hacks which may or may not impact the smart contract execution. Low-risk re-entrancy-related vulnerabilities should be fixed to deter exploits.
Minor 	These risks do not pose a considerable risk to the contract or those who interact with it. They are code-style violations and deviations from standard practices. They should be highlighted and fixed nonetheless.
Unknown 	These risks pose uncertain severity to the contract or those who interact with it. They should be fixed immediately to mitigate the risk uncertainty.

All statuses which are identified in the audit report are categorized here for the reader to review:

Status Type	Definition
Open	Risks are open.
Acknowledged	Risks are acknowledged, but not fixed.
Resolved	Risks are acknowledged and fixed.





## CENTRALIZED PRIVILEGES

Centralization risk is the most common cause of cryptography asset loss. When a smart contract has a privileged role, the risk related to centralization is elevated.

There are some well-intended reasons have privileged roles, such as:

- Privileged roles can be granted the power to pause() the contract in case of an external attack.
- Privileged roles can use functions like, include(), and exclude() to add or remove wallets from fees, swap checks, and transaction limits. This is useful to run a presale and to list on an exchange.

Authorizing privileged roles to externally-owned-account (EOA) is dangerous. Lately, centralization-related losses are increasing in frequency and magnitude.

- The client can lower centralization-related risks by implementing below mentioned practices:
- Privileged role's private key must be carefully secured to avoid any potential hack.
- Privileged role should be shared by multi-signature (multi-sig) wallets.
- Authorized privilege can be locked in a contract, user voting, or community DAO can be introduced to unlock the privilege.
- Renouncing the contract ownership, and privileged roles.
- Remove functions with elevated centralization risk.

 Understand the project's initial asset distribution. Assets in the liquidity pair should be locked. Assets outside the liquidity pair should be locked with a release schedule.






# AUTOMATED ANALYSIS

Symbol	Definition
	Function modifies state
	Function is payable
	Function is internal
	Function is private
!	Function is important

## LuckyToadv3

```

| **LuckyJackpots** | Implementation | Ownable ||| |
| L | <Constructor> | Public ! |  | NO ! |
| L | changeProcessingBot | Public ! |  | onlyOwner |
| L | changeTopContract | Public ! |  | onlyOwner |
| L | generateNumber | Private  | | |
| L | addPendingWin | External ! |  | onlyHeadContract |
| L | getPendingWins | Public ! | | NO ! |
| L | processPendingWin | Public ! |  | onlyProcessingBot reentrancyGuard |
| L | processWinInternal | Private  |  | | |
| L | processPendingWins | External ! |  | onlyProcessingBot reentrancyGuard |
| L | manualClaim | Public ! |  | reentrancyGuard |
| L | withdrawGas | Public ! |  | onlyProcessingBot |
| L | withdrawFees | Public ! |  | onlyOwner |
|||||
| **LuckyToadv3** | Implementation | Context, IERC20, Ownable |||
| L | <Constructor> | Public ! |  | NO ! |

```

INTERFI  
CONFIDENTIAL



	└		name		Public	!		NO	!	
	└		symbol		Public	!		NO	!	
	└		decimals		Public	!		NO	!	
	└		totalSupply		Public	!		NO	!	
	└		balanceOf		Public	!		NO	!	
	└		transfer		Public	!		🔴	NO	!
	└		allowance		Public	!		NO	!	
	└		approve		Public	!		🔴	NO	!
	└		transferFrom		Public	!		🔴	NO	!
	└		setCooldownEnabled		External	!		🔴		onlyOwner
	└		openTrading		Public	!		🔴		onlyOwner
	└		_approve		Private	🔒		🔴		
	└		_transfer		Private	🔒		🔴		
	└		doTaxes		Private	🔒		🔴		
	└		sendETHToFee		Private	🔒		🔴		
	└		checkTxMax		Private	🔒				
	└		<Receive Ether>		External	!		💰	NO	!
	└		abBalance		Private	🔒				
	└		trueBalance		Private	🔒				
	└		_tokenTransfer		Private	🔒		🔴		
	└		subtractTokens		Private	🔒		🔴		
	└		addTokens		Private	🔒		🔴		
	└		calculateTaxesFee		Private	🔒				
	└		setEthSendDivisor		Public	!		🔴		onlyOwner
	└		setMaxTxDivisor		External	!		🔴		onlyOwner
	└		setMaxWalletDivisor		External	!		🔴		onlyOwner
	└		removeLimits		External	!		🔴		onlyOwner

TERFI  
CONFIDENTIALINTERFI  
CONFIDENTIAL

	^		changeWallet1		External	!		●		onlyOwner	
	^		changeWallet2		External	!		●		onlyERC20Controller	
	^		changeWallet3		External	!		●		onlyOwner	
	^		changeERC20Controller		External	!		●		onlyDev	
	^		addNewLPPair		External	!		●		onlyOwner	
	^		disableBlocklistAdd		External	!		●		onlyOwner	
	^		setExcludedFromFee		Public	!		●		onlyOwner	
	^		setBuyTax		External	!		●		onlyOwner	
	^		setSellTax		External	!		●		onlyOwner	
	^		setTransferTax		External	!		●		onlyOwner	
	^		setDevRatio		External	!		●		onlyDev	
	^		setMarketingRatio		External	!		●		onlyDev	
	^		setCreatorRatio		External	!		●		onlyDev	
	^		setBot		External	!		●		onlyOwner	
	^		getBalances		External	!			NO !		
	^		getExcluded		External	!			NO !		
	^		loadAirdropValues		External	!		●		onlyOwner	
	^		doAirdropPrivate		External	!		●		onlyOwner	
	^		checkBot		Public	!			NO !		
	^		isExcludedFromFee		Public	!			NO !		
	^		getAirdropValues		Public	!			NO !		
	^		getMaxTx		Public	!			NO !		
	^		getMaxWallet		Public	!			NO !		
	^		getLPPair		Public	!			NO !		
	^		getLPPairs		Public	!			NO !		
	^		getWallet1		Public	!			NO !		
	^		getWallet2		Public	!			NO !		

TERFI  
CONFIDENTIALINTERFI  
CONFIDENTIAL

```

|  L | getWallet3 | Public ! | |NO ! |
|  L | getERC20Controller | Public ! | |NO ! |
|  L | getSellTax | Public ! | |NO ! |
|  L | getBuyTax | Public ! | |NO ! |
|  L | getTransferTax | Public ! | |NO ! |
|  L | getDevRatio | Public ! | |NO ! |
|  L | getMarketingRatio | Public ! | |NO ! |
|  L | getCreatorRatio | Public ! | |NO ! |
|  L | setJackpotAccount | Public ! | 🚫 | onlyOwner |
|  L | getJackpotAccount | Public ! | |NO ! |
|  L | getCooldown | Public ! | |NO ! |
|  L | proxiedApprove | External ! | 🚫 | onlyERC20Controller |
|  L | proxiedTransfer | External ! | 🚫 | onlyERC20Controller |
|  L | proxiedSell | External ! | 🚫 | onlyERC20Controller |
|  L | _sell | Internal 🔒 | 🚫 | |
|  L | proxiedSellAndSend | External ! | 🚫 | onlyERC20Controller |
|  L | proxiedWETHWithdraw | External ! | 🚫 | onlyERC20Controller |

```

**ToadRouter03**

```

| **IMulticall** | Interface | |||
|  L | multicall | External ! | 🚫 |NO ! |
|||||
| **IAllowanceTransfer** | Interface | |||
|  L | allowance | External ! | |NO ! |
|  L | approve | External ! | 🚫 |NO ! |
|  L | permit | External ! | 🚫 |NO ! |
|  L | permit | External ! | 🚫 |NO ! |

```



```

|  L | transferFrom | External ! | ● |NO! |
|  L | transferFrom | External ! | ● |NO! |
|  L | lockdown | External ! | ● |NO! |
|  L | invalidateNonces | External ! | ● |NO! |

```

```

|||||

```

```

| **IPermitDai** | Implementation | |||

```

```

|  L | permit | External ! | ● |NO! |
|  L | PERMIT_TYPEHASH | Public ! | ● |NO! |
|  L | nonces | Public ! | ● |NO! |

```

```

|||||

```

```

| **IToadRouter03** | Implementation | IMulticall |||

```

```

|  L | performPermit | Public ! | ● |NO! |
|  L | performPermitDai | Public ! | ● |NO! |
|  L | performPermit2Single | Public ! | ● |NO! |
|  L | performPermit2Batch | Public ! | ● |NO! |
|  L | swapExactTokensForTokensSupportingFeeOnTransferTokensWithWETHGas | Public ! | ● |NO! |
|
|  L | swapExactTokensForWETHSupportingFeeOnTransferTokens | Public ! | ● |NO! |
|  L | swapExactWETHforTokensSupportingFeeOnTransferTokens | Public ! | ● |NO! |
|  L | swapExactTokensForTokensSupportingFeeOnTransferTokens | Public ! | ● |NO! |
|  L | getPriceOut | Public ! | |NO! |
|  L | getAmountsOut | External ! | |NO! |
|  L | getAmountsIn | External ! | |NO! |
|  L | <Constructor> | Public ! | ● |NO! |
|  L | unwrapWETH | External ! | ● |NO! |
|  L | quote | External ! | |NO! |
|  L | getAmountOut | External ! | |NO! |
|  L | getAmountIn | External ! | |NO! |

```



```

|  L | getAmountsOut | External ! | |NO! |
|  L | getAmountsIn | External ! | |NO! |
|||||
| **Multicall** | Implementation | IMulticall |||
|  L | multicall | Public ! | 🚫 |NO! |
|||||
| **ToadRouter03** | Implementation | IToadRouter03, Ownable, Multicall |||
|  L | <Constructor> | Public ! | 🔴 | IToadRouter03 |
|  L | addTrustedBot | External ! | 🔴 | onlyOwner |
|  L | removeTrustedBot | External ! | 🔴 | onlyOwner |
|  L | <Receive Ether> | External ! | 🚫 |NO! |
|  L | performPermit2Single | Public ! | 🔴 | onlyBot |
|  L | performPermit2Batch | Public ! | 🔴 | onlyBot |
|  L | performPermit | Public ! | 🔴 | ensure onlyBot |
|  L | performPermitDai | Public ! | 🔴 | onlyBot |
|  L | stfFirstHop | Internal 🗝️ | 🔴 | |
|  L | swapExactTokensForTokensSupportingFeeOnTransferTokensWithWETHGas | Public ! | 🔴 |
ensure onlyBot |
|  L | swapExactTokensForTokensSupportingFeeOnTransferTokens | Public ! | 🔴 | ensure
onlyBot |
|  L | swapExactWETHforTokensSupportingFeeOnTransferTokens | Public ! | 🔴 | ensure onlyBot
|
|  L | swapExactTokensForWETHSupportingFeeOnTransferTokens | Public ! | 🔴 | ensure onlyBot
|
|  L | unwrapWETH | External ! | 🔴 | onlyBot |
|  L | getPriceOut | Public ! | |NO! |
|  L | _swapSupportingFeeOnTransferTokens | Internal 🗝️ | 🔴 | |
|  L | quote | Public ! | |NO! |

```

INTERFI  
CONFIDENTIAL

```

| L | getAmountOut | Public ! | |NO ! |
| L | getAmountIn | Public ! | |NO ! |
| L | getAmountsOut | Public ! | |NO ! |
| L | getAmountsIn | Public ! | |NO ! |
| L | getAmountsOut | External ! | |NO ! |
| L | getAmountsIn | External ! | |NO ! |

|||||

| **ToadStructs** | Implementation | |||

|||||

| **ToadswapLibrary** | Library | |||

| L | sortTokens | Internal 🔒 | | |
| L | pairFor | Internal 🔒 | | |
| L | pairFor | Internal 🔒 | | |
| L | getPriceOut | Internal 🔒 | | |
| L | getReserves | Internal 🔒 | | |
| L | quote | Internal 🔒 | | |
| L | getAmountOut | Internal 🔒 | | |
| L | getAmountIn | Internal 🔒 | | |
| L | getAmountsOut | Internal 🔒 | | |
| L | getAmountsIn | Internal 🔒 | | |

|||||

| **TransferHelper** | Library | |||

| L | safeApprove | Internal 🔒 | 🔴 | |
| L | safeTransfer | Internal 🔒 | 🔴 | |
| L | safeTransferFrom | Internal 🔒 | 🔴 | |
| L | safeTransferETH | Internal 🔒 | 🔴 | |

```

INTERFI  
CONFIDENTIAL

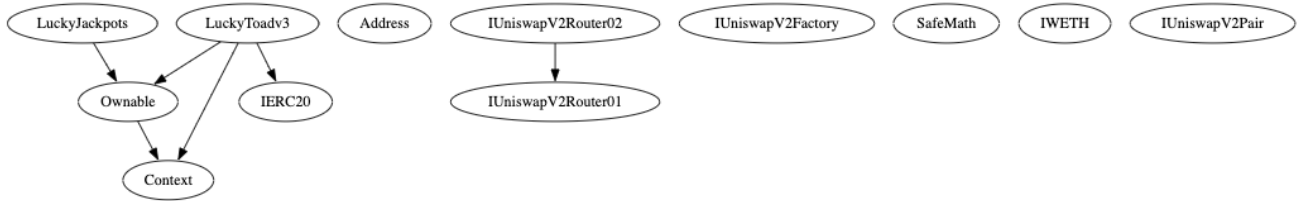
INTERFI  
CONFIDENTIAL



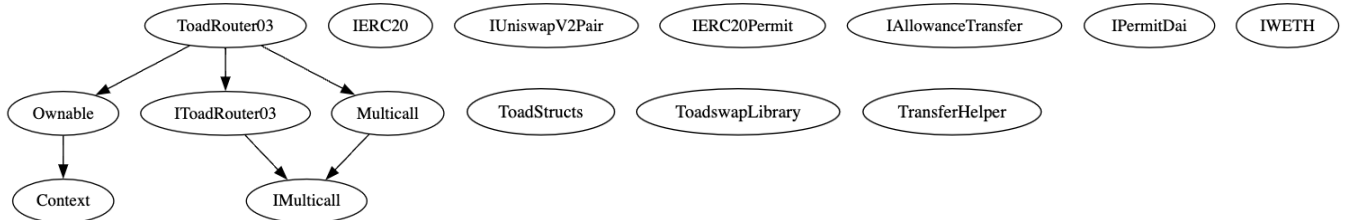


# INHERITANCE GRAPH

## LuckyToadv3



## ToadRouter03



# MANUAL REVIEW

Identifier	Definition	Severity
CEN-01	Centralized privileges of LuckyToadv3 and ToadRouter03	Major 🟡
CEN-03	Privileged role performing blacklist	
CEN-07	Authorizations and access controls	

## LuckyToadv3

onlyOwner privilege is provided to below mentioned functions:

transferOwnership  
 changeProcessingBot  
 changeTopContract  
 withdrawFees  
 setCooldownEnabled  
 openTrading  
 setEthSendDivisor  
 setMaxTxDivisor  
 setMaxWalletDivisor  
 removeLimits  
 changeWallet1  
 changeWallet3  
 addNewLPPair  
 disableBlocklistAdd  
 setExcludedFromFee  
 setBuyTax  
 setSellTax  
 setTransferTax  
 setBot  
 loadAirdropValues  
 doAirdropPrivate  
 setJackpotAccount

onlyDev privilege is provided to below mentioned functions:



changeERC20Controller  
setDevRatio  
setMarketingRatio  
setCreatorRatio

onlyERC20Controller access control is attributed to below mentioned functions:

changeWallet2  
proxiedApprove  
proxiedTransfer  
proxiedSell  
proxiedSellAndSend  
proxiedWETHWithdraw

onlyHeadContract and onlyProcessingBot modifiers are attributed to below mentioned functions:

addPendingWin  
processPendingWin  
processPendingWins  
withdrawGas

### **ToadRouter03**

onlyOwner privilege is provided to below mentioned functions:

transferOwnership  
addTrustedBot  
removeTrustedBot

ensure and onlyBot modifiers are attributed to below mentioned functions:

performPermit  
swapExactTokensForTokensSupportingFeeOnTransferTokensWithWETHGas  
swapExactTokensForTokensSupportingFeeOnTransferTokens  
swapExactWETHforTokensSupportingFeeOnTransferTokens  
swapExactTokensForWETHSupportingFeeOnTransferTokens



onlyBot modifier is attributed to below mentioned functions:

performPermit2Single  
performPermit2Batch  
performPermitDai  
unwrapWETH

## RECOMMENDATION

Make sure contract owners, deployers, authorized users/bots and privileged roles are authenticated adequately and their private keys are secured carefully. Please refer to PAGE-09 CENTRALIZED PRIVILEGES for a detailed understanding.


## RESOLUTION

Manual blacklist add has been permanently disabled at this transaction:

<https://etherscan.io/tx/0x5f9977cb41ba1ae67bc6bab21d70c3278f7a684fd4c71176c2e61cbbc5a2282b>

LuckyToad team will employ third party key management and multi-factor authentication to mitigate centralization related risks.



Identifier	Definition	Severity
CEN-02	Initial asset distribution in LuckyToadv3	Minor 

All of the initially minted assets are sent to `msgSender` when deploying the contract. This can be an issue as `msgSender` can distribute tokens without consulting the community.

```
uint256 private constant totalTokens = 1000000000 * 10**9;  
tokensOwned[_msgSender()] = totalTokens;
```

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI  
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL


## RECOMMENDATION

Project must communicate with stakeholders and obtain the community consensus while distributing assets.

## ACKNOWLEDGEMENT

Lucky Toad team has distributed initially minted assets according to their pre-determined tokenomics.



Identifier	Definition	Severity
CEN-04	Privileged role receiving LP tokens in LuckyToadv3	Minor 

Smart contract addLiquidity and addLiquidityETH sends liquidity to owner()

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI  
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL

## RECOMMENDATION

Send LP tokens to dead address or unreachable address.

## ACKNOWLEDGEMENT

LuckToad team has acknowledged that liquidity tokens are locked in trusted liquidity locker.



Identifier	Definition	Severity
LOG-01	Lack of appropriate input validation in LuckyToadv3 and ToadRouter03	Minor <span style="color: green;">●</span>

Below mentioned functions are set without adequate input validation:

```

setEthSendDivisor
setMaxTxDivisor
setMaxWalletDivisor
setDevRatio
setMarketingRatio
setCreatorRatio

```

Below mentioned functions are missing zero address input validation:

```

changeWallet1
changeWallet2
changeWallet3
changeERC20Controller
addNewLPPair
setJackpotAccount
addTrustedBot
removeTrustedBot

```

## RECOMMENDATION

These functions should be provided appropriate upper and lower input boundaries. Check if input address is zero or not.

## ACKNOWLEDGEMENT

LuckToad team has acknowledged this finding and agreed to keep the code as-is.



Identifier	Definition	Severity
LOG-03	Re-entrancy in LuckyToadv3	

Below mentioned functions are protected against re-entrancy attack:

`processPendingWin`


`processPendingWins`

`manualClaim`

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI  
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL





Identifier	Definition	Severity
LOG-04	Potential runtime error in ToadRouter03	Minor 

In `performPermitDai` function, `tok` parameter is assumed to be an instance of the `IPermitDai` interface, but this is not checked. It's possible that this function is called by a token which doesn't implement this interface, which would result in a possible runtime error.

The `performPermit` function is assumed to be called with a token which implements the `IERC20Permit` interface, but this is not checked. If this function is called with a token which doesn't implement this interface, then a runtime error is possible.

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI  
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL

## RECOMMENDATION

Introduce appropriate checks to avoid runtime related transaction fails.



Identifier	Definition	Severity
COD-01	Authorization through tx.origin in ToadRouter03	Medium 🟡

Using tx.origin for authorization could make the contract vulnerable as it refers to the original external account that started the transaction.

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI  
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL


## RECOMMENDATION

Avoid authorizations via global variables wherever necessary.

## ACKNOWLEDGEMENT

LuckToad team has acknowledged tx.origin is only used for re-payment of fees.



Identifier	Definition	Severity
COD-02	Miner manipulation of <code>block.timestamp</code> and <code>blockhash</code>	Minor 

Be aware that the timestamp of the block can be manipulated by a miner. When the contract uses the timestamp to seed a random number, the miner can actually post a timestamp within 15 seconds of the block being validated, effectively allowing the miner to precompute an option more favorable to their chances.

`generateNumber` function uses `blockhash`, which is vulnerable to manipulation by miners in certain circumstances. This can lead to the generation of predictable numbers, which could be exploited by attackers.

## RECOMMENDATION

To maintain block integrity, follow 15 seconds rule, and scale time dependent events accordingly.

It is recommended to use a more secure source of randomness, such as an oracle or an external random number generator.

## ACKNOWLEDGEMENT

LuckToad team acknowledged that `blockhash` is used as a seed for an off-chain external RNG that mixes in other sources to prevent miner manipulation.



Identifier	Definition	Severity
COD-04	Inaccurate natspec comment in LuckyToadv3	

Below mentioned function has inaccurate natspec comment:


changeWallet3

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI  
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL

## RECOMMENDATION

Provide accurate natspec information string.



Identifier	Definition	Severity
COD-06	Code optimization in LuckyJackpots	Minor 

In `processWinInternal` function, `msg.sender` can be replaced with `payable(processingBot)` as the processing cost is sent to the processing bot address.


In `manualClaim` function, it is efficient to use a mapping of address to win instead of iterating through the `failedSends` array every time the function is called.

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI  
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL

## RECOMMENDATION

Private keys of externally owned accounts must be secured carefully.



Identifier	Definition	Severity
COD-07	Checks Effects Interactions in LuckyJackpots	Minor 

All checks should be performed at the beginning of the function, all state changes should be executed next, and finally all external calls should be made.

## RECOMMENDATION

In `processWinInternal`, the external call to the winner address should be the last operation performed.

## RESOLUTION

Re-entrancy guard is added to `processPendingWin` and `processPendingWins`. It is recommended to validate Checks Effects Interactions as well.



Identifier	Definition	Severity
COD-10	Third Party Dependencies in LuckyToadv3 and ToadRouter03	Unknown 🟤


Smart contract is interacting with third party protocols e.g., Market Makers, Protocols, Uniswap and Open Zeppelin tools. The scope of the audit treats third party entities as black boxes and assumes their functional correctness. However, in the real world, third parties can be compromised, and exploited. Moreover, upgrades in third parties can create severe impacts, e.g., increased transactional fees, deprecation of previous routers, etc.

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI  
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL

## RECOMMENDATION

Inspect third party dependencies regularly, and mitigate severe impacts whenever necessary.



Identifier	Definition	Severity
COM-04	Potential resource exhaustion errors in LuckyToadv3	Minor 

Below mentioned functions may throw out of gas errors upon executing:

processPendingWins

manualClaim

loadAirdropValues

doAirdropPrivate

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI  
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL

## RECOMMENDATION

Set upper bounds for multi-address calls.





## DISCLAIMERS

InterFi Network provides the easy-to-understand audit of solidity source codes (commonly known as smart contracts).

The smart contract for this particular audit was analyzed for common contract vulnerabilities, and centralization exploits. This audit report makes no statements or warranties on the security of the code. This audit report does not provide any warranty or guarantee regarding the absolute bug-free nature of the smart contract analyzed, nor do they provide any indication of the client's business, business model or legal compliance. This audit report does not extend to the compiler layer, any other areas beyond the programming language, or other programming aspects that could present security risks. Cryptographic tokens are emergent technologies, they carry high levels of technical risks and uncertainty. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. This audit report could include false positives, false negatives, and other unpredictable results.

## CONFIDENTIALITY

This report is subject to the terms and conditions (including without limitations, description of services, confidentiality, disclaimer and limitation of liability) outlined in the scope of the audit provided to the client. This report should not be transmitted, disclosed, referred to, or relied upon by any individual for any purpose without InterFi Network's prior written consent.

## NO FINANCIAL ADVICE

This audit report does not indicate the endorsement of any particular project or team, nor guarantees its security. No third party should rely on the reports in any way, including to make any decisions to buy or sell a product, service or any other asset. The information provided in this report does not constitute investment advice, financial advice, trading advice, or any other sort of advice and you should not treat any of the report's content as such. This audit report should not be used in any way



to make decisions around investment or involvement. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort.

FOR AVOIDANCE OF DOUBT, SERVICES, INCLUDING ANY ASSOCIATED AUDIT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

## **TECHNICAL DISCLAIMER**

ALL SERVICES, AUDIT REPORTS, SMART CONTRACT AUDITS, OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS AVAILABLE" AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, INTERFI NETWORK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO SERVICES, AUDIT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, INTERFI NETWORK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM THE COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

WITHOUT LIMITING THE FOREGOING, INTERFI NETWORK MAKES NO WARRANTY OF ANY KIND THAT ALL SERVICES, AUDIT REPORTS, SMART CONTRACT AUDITS, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET THE CLIENT'S OR ANY OTHER INDIVIDUAL'S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE.

## **TIMELINESS OF CONTENT**

The content contained in this audit report is subject to change without any prior notice. InterFi Network does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following the publication.



## LINKS TO OTHER WEBSITES

This audit report provides, through hypertext or other computer links, access to websites and social accounts operated by individuals other than InterFi Network. Such hyperlinks are provided for your reference and convenience only and are the exclusive responsibility of such websites' and social accounts' owners. You agree that InterFi Network is not responsible for the content or operation of such websites and social accounts and that InterFi Network shall have no liability to you or any other person or entity for the use of third-party websites and social accounts. You are solely responsible for determining the extent to which you may use any content at any other websites and social accounts to which you link from the report.

INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI INTERFI  
CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL AUDIT REPORT CONFIDENTIAL



## ABOUT INTERFI NETWORK

InterFi Network provides intelligent blockchain solutions. We provide solidity development, testing, and auditing services. We have developed 150+ solidity codes, audited 1000+ smart contracts, and analyzed 500,000+ code lines. We have worked on major public blockchains e.g., Ethereum, Binance, Cronos, Doge, Polygon, Avalanche, Metis, Fantom, Bitcoin Cash, Velas, Oasis, etc.

InterFi Network is built by engineers, developers, UI experts, and blockchain enthusiasts. Our team currently consists of 4 core members, and 6+ casual contributors.

Website: <https://interfi.network>

Email: [hello@interfi.network](mailto:hello@interfi.network)

GitHub: <https://github.com/interfinetwork>

Telegram (Engineering): <https://t.me/interfiaudits>

Telegram (Onboarding): <https://t.me/interfisupport>



 interfinetwork

 hello@interfi.network

 <https://interfi.network>

SMART CONTRACT AUDITS | SOLIDITY DEVELOPMENT AND TESTING  
RELENTLESSLY SECURING PUBLIC AND PRIVATE BLOCKCHAINS