



# SMART CONTRACT AUDIT



interfinetwork



hello@interfi.network



<https://interfi.network>

PREPARED FOR

**CZ GOAT**



# INTRODUCTION

Auditing Firm	InterFi Network
Client Firm	CZ Goat
Methodology	Automated Analysis, Manual Code Review
Language	Solidity
Contract	0x0fcfB6014bFd2b1E53b016ed93f4c5AF29a4266F
Blockchain	Binance Smart Chain
Centralization	Active Ownership
Commit	eea5494129b95df8d3ebb4a9d977adb8b2878157
Website	<a href="https://czgoatofficial.com/">https://czgoatofficial.com/</a>
Telegram	<a href="https://t.me/czgoatofficial">https://t.me/czgoatofficial</a>
X (Twitter)	<a href="https://x.com/czgoatofficial">https://x.com/czgoatofficial</a>
Report Date	April 25, 2025

 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	0
Acknowledged	0	1	1	2	1
Resolved	0	0	0	0	0
Noteworthy Privileges	clearStuckToken, enableTrading, manage_FeeExempt, setMultipliers, setSwapBackSettings				

**i** 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.

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

**i** Please note that the absence of public KYC verification of the project owners, team members, or deployers associated with CZ Goat. Typically, third-party KYC processes are instrumental in ensuring the transparency and accountability of a project's leadership, thereby enhancing user trust and regulatory compliance.



# TABLE OF CONTENTS

TABLE OF CONTENTS .....	3
SCOPE OF WORK.....	5
AUDIT METHODOLOGY .....	6
RISK CATEGORIES .....	8
CENTRALIZED PRIVILEGES .....	9
AUTOMATED ANALYSIS.....	10
INHERITANCE GRAPH .....	13
MANUAL REVIEW .....	14
DISCLAIMERS .....	23
ABOUT INTERFI NETWORK .....	26

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 CZ Goat 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:

- CZGoat.sol

**i** 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://bscscan.com/address/0x0fcfb6014bfd2b1e53b016ed93f4c5af29a4266f#code">https://bscscan.com/address/0x0fcfb6014bfd2b1e53b016ed93f4c5af29a4266f#code</a>	
Contract Name	CZGoat
Compiler Version	0.8.22
License	UNLICENSED



# 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	<ul style="list-style-type: none"> <li>○ Integer Overflow</li> <li>○ Lack of Arbitrary limits</li> <li>○ Incorrect Inheritance Order</li> <li>○ Typographical Errors</li> <li>○ Requirement Violation</li> <li>○ Gas Optimization</li> <li>○ Coding Style Violations</li> <li>○ Re-entrancy</li> <li>○ Third-Party Dependencies</li> <li>○ Potential Sandwich Attacks</li> <li>○ Irrelevant Codes</li> <li>○ Divide before multiply</li> <li>○ Conformance to Solidity Naming Guides</li> <li>○ Compiler Specific Warnings</li> <li>○ Language Specific Warnings</li> </ul>
---------------------------------	---

## 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

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:

Risk Type	Definition
Critical 🚫	These risks pose immediate and severe threats, such as asset theft, data manipulation, or complete loss of contract functionality. They are often easy to exploit and can lead to significant, irreparable damage. Immediate fix is required.
Major 🟡	These risks can significantly impact code performance and security, and they may indirectly lead to asset theft and data loss. They can allow unauthorized access or manipulation of sensitive functions if exploited. Fixing these risks are important.
Medium 🟡	These risks may create attack vectors under certain conditions. They may enable minor unauthorized actions or lead to inefficiencies that can be exploited indirectly to escalate privileges or impact functionality over time.
Minor 🟢	These risks may include inefficiencies, lack of optimizations, code-style violations. These should be addressed to enhance overall code quality and maintainability.
Unknown 🟤	These risks pose uncertain severity to the contract or those who interact with it. Immediate fix is required to mitigate risk uncertainty.

All statuses which are identified in the audit report are categorized here:

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

```

| **SafeMath** | Library |   | | |
| L | add | Internal  |   |
| L | sub | Internal  |   |
| L | sub | Internal  |   |
| L | mul | Internal  |   |
| L | div | Internal  |   |
| L | div | Internal  |   |
|||||
| **BEP20** | Interface |   |
| L | getOwner | External  | |NO  |
| L | balanceOf | External  | |NO  |
| L | transfer | External  |  |NO  |
| L | allowance | External  | |NO  |
| L | approve | External  |  |NO  |
| L | transferFrom | External  |  |NO  |
|||||
| **Auth** | Implementation |   |
| L | <Constructor> | Public  |  |NO  |

```



```

|  L | isOwner | Public ! | |NO ! |
|  L | renounceOwnership | External ! | 🚫 | onlyOwner |
|||||
| **IDEXFactory** | Interface | |||
|  L | createPair | External ! | 🚫 |NO ! |
|||||
| **IDEXRouter** | Interface | |||
|  L | factory | External ! | |NO ! |
|  L | WETH | External ! | |NO ! |
|  L | swapExactTokensForETHSupportingFeeOnTransferTokens | External ! | 🚫 |NO ! |
|||||
| **CZGoat** | Implementation | BEP20, Auth |||
|  L | <Constructor> | Public ! | 🚫 | Auth |
|  L | <Receive Ether> | External ! | 🚫 |NO ! |
|  L | getOwner | External ! | |NO ! |
|  L | allowance | External ! | |NO ! |
|  L | approve | Public ! | 🚫 |NO ! |
|  L | approveMax | External ! | 🚫 |NO ! |
|  L | transfer | External ! | 🚫 |NO ! |
|  L | transferFrom | External ! | 🚫 |NO ! |
|  L | _transferFrom | Internal 🔒 | 🚫 | |
|  L | _basicTransfer | Internal 🔒 | 🚫 | |
|  L | takeFee | Internal 🔒 | 🚫 | |
|  L | shouldSwapBack | Internal 🔒 | | |
|  L | clearStuckBalance | External ! | 🚫 | onlyOwner |
|  L | clearStuckToken | External ! | 🚫 | onlyOwner |
|  L | enableTrading | External ! | 🚫 | onlyOwner |
|  L | swapBack | Internal 🔒 | 🚫 | swapping |

```

INTERFI  
CONFIDENTIAL

INTERFI  
CONFIDENTIAL



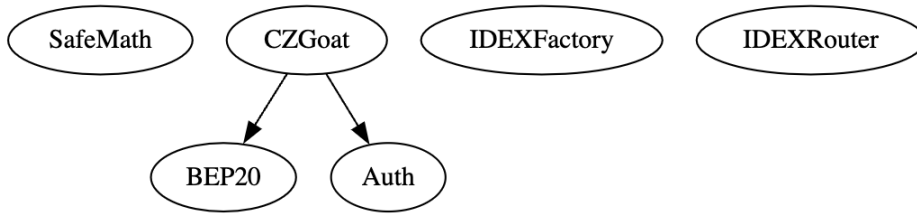
	└		manage_FeeExempt		External	!		🔴		onlyOwner	
	└		setMultipliers		External	!		🔴		onlyOwner	
	└		setFeeReceivers		External	!		🔴		onlyOwner	
	└		setSwapBackSettings		External	!		🔴		onlyOwner	
	└		getCirculatingSupply		Public	!				NO!	

INTERFI  
CONFIDENTIAL

INTERFI  
CONFIDENTIAL



## INHERITANCE GRAPH



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



## MANUAL REVIEW

Identifier	Definition	Severity
CEN-01	Centralized privileges	Major 🟡
CEN-01-03	Privileged role must enable trading to allow transactions	

Important on ly0wner centralized privileges are listed below:

renounce0wnership  
 clearStuckBalance  
 clearStuckToken  
 enableTrading  
 manage\_FeeExempt  
 setMultipliers  
 setFeeReceivers  
 setSwapBackSettings

Although setMultipliers enforces max fees of 20%, during initial deployment, buyFee, sellFee, and transferFee are all set at 25%, which violates the later limits. Owner can delay calling setMultipliers.

clearStuckToken(address tokenAddress, uint256 tokens) lets owner transfer any token from contract.

### RECOMMENDATION

Securing private keys or access credentials of deployers, contract owners, operators, and other roles with privileged access is crucial to prevent single points of failure that can compromise contract security.

- Consider implementing a multi-signature wallet for ownership actions to mitigate single-point-of-failure risks.
- Introduce time-locks for critical ownership changes (for example, transferring ownership).
- Clearly document the roles and responsibilities of authorized addresses.




## ACKNOWLEDGEMENT

CZ Goat team argued that centralized and controlled privileges are used as required.

CZ Goat team acknowledges that high fees deter snipers and bots, with a planned reduction to 5/5 post-LP stability. They note that no tokens are allocated to liquidity, so removing the native token should not be an issue.

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
CEN-02	Initial token allocation	Minor 

Upon deployment, all initially minted tokens are transferred to the contract deployer. It could be an issue as the deployer can distribute tokens without consulting the community.

```
uint256 public constant totalSupply = 4_000_000_000 * 10**decimals;
```

## RECOMMENDATION

Establish transparent tokenomics model that involves community input in the decision-making process regarding token allocation.

## ACKNOWLEDGEMENT

CZ Goat team has clarified that initial token allocation will adhere strictly to pre-determined tokenomics outlined in project documentation.





Identifier	Definition	Severity
LOG-01	Insufficient input restrictions	Medium 🟡

setMultipliers  
clearStuckToken

## RECOMMENDATION

All operational parameters must remain within safe and rational ranges.

- Total of all collected fees should be below 20%  
 $\text{buyFee} + \text{sellFee} + \text{transferFee} \leq 20$
- Contract owner should not be able to remove native token from contract.

## ACKNOWLEDGEMENT

CZ Goat team acknowledges that high fees deter snipers and bots, with a planned reduction to 5/5 post-LP stability. They note that no tokens are allocated to liquidity, so removing the native token should not be an issue.

- CZ Goat team commits to keeping the fees below 20%.
- Contract owner will avoid removing native tokens unless necessary.



Identifier	Definition	Severity
LOG-02	Potential front-running	Minor 

Potential front-running happens when an attacker observes a transaction swapping tokens or adding liquidity without setting restrictions on slippage or minimum output amount. The attacker can manipulate the exchange rate by front-running a transaction to purchase assets and make profits by back-running a transaction to sell assets. Below mentioned functions are called without setting restrictions on slippage or minimum output:

```

_transferFrom
_basicTransfer
swapExactTokensForETHSupportingFeeOnTransferTokens

```

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


## RECOMMENDATION

Functions that execute critical state changes should enforce minimum output thresholds. Setting these minimums above zero can deter malicious actors by reducing the predictability and profitability of front-running strategies.

## ACKNOWLEDGEMENT

Front-running is not avoidable on public blockchains. CZ Goat team commented that, most EVM chains are prone to some sort of front-running and external manipulation.



Identifier	Definition	Severity
COD-06	Unknown externally owned account	Minor 

An externally owned account (EOA) has no code, and one can send messages from an externally owned account by creating and signing a transaction.

```
marketingWallet = 0xCd3759b76e5461cf779949Cb6BF753b464c6ec92;
```

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-10	Direct and indirect dependencies	Unknown 🟤

Smart contract is interacting with third party protocols e.g., DEX router such as Pancakeswap, external contracts, web3 applications, *OpenZeppelin* upgradeable and ERC20 libraries. The scope of the audit treats these entities as black boxes and assumes their functional correctness. However, in the real world, all of them can be compromised, and exploited. Moreover, upgrades in these entities can create severe impacts, e.g., increased transactional fees, deprecation of previous routers, etc.

## RECOMMENDATION

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

## ACKNOWLEDGEMENT

CZ Goat team will inspect third party dependencies regularly, and push upgrades whenever required.



Identifier	Definition	Severity
COD-12	Lack of event-driven architecture	Minor ●

Smart contract uses function calls to update state, which can make it difficult to track and analyze changes to the contract over time.

contract Auth

contract CZGoat

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

## RECOMMENDATION

Use events to track state changes. Events improve transparency and provide a more granular view of contract activity.



Identifier	Definition	Severity
VOL-01	Irrelevant code	Minor ●

Redundant code found in:

SafeMath  
approveMax

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

## RECOMMENDATION

Remove redundant and dead code.



## 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