AuditAce
FROM INCEPTION TO SUCCESS

Smart Contract Audit

FOR

SRG

DATED : 3 MAY 23'

AUDIT SUMMARY

Project name – SRG

Date: 3 May, 2023

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

Audit Status: Passed

Issues Found

Status	Critical	High	Medium	Low	Suggestion
Open	0	0	0	0	1
Acknowledged	0	0	0	0	0
Resolved	0	0	0	0	0

USED TOOLS

Tools:

1.Manual Review: The code has undergone a line-by-line review by the Ace team.

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

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

Token Information

Name : Scrooge Inu

Symbol : SRG

Decimals: 9

Network: Binance smart chain

Token Type: BEP20

Token Address :
0xf9A1f359b4D2e822F25200adD1E6B5ee716083F3

Owner:
0x811Bf4f7695e56b1c1C5926103CA023c82a919dc
(at time of audit)

Deployer:
0x811Bf4f7695e56b1c1C5926103CA023c82a919dc

Token Information

Fees:

Buy Fees: up to 8%

Sell Fees: up to 8%

Transfer Fees: 8%

Fees Privilige: Owner

Ownership : Owned

Minting: None

Max Tx Amount/ Max Wallet Amount: Yes

Blacklist: No

Other Priviliges: Toggling internal swap - excluding wallets from fee - including wallets in fee - modifying fee

AUDIT METHODOLOGY

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

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.

- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.

- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.

- Test coverage analysis determines whether the test cases are covering the code and how much code isexercised when we run the test cases.

- Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.

- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

VULNERABILITY CHECKLIST

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

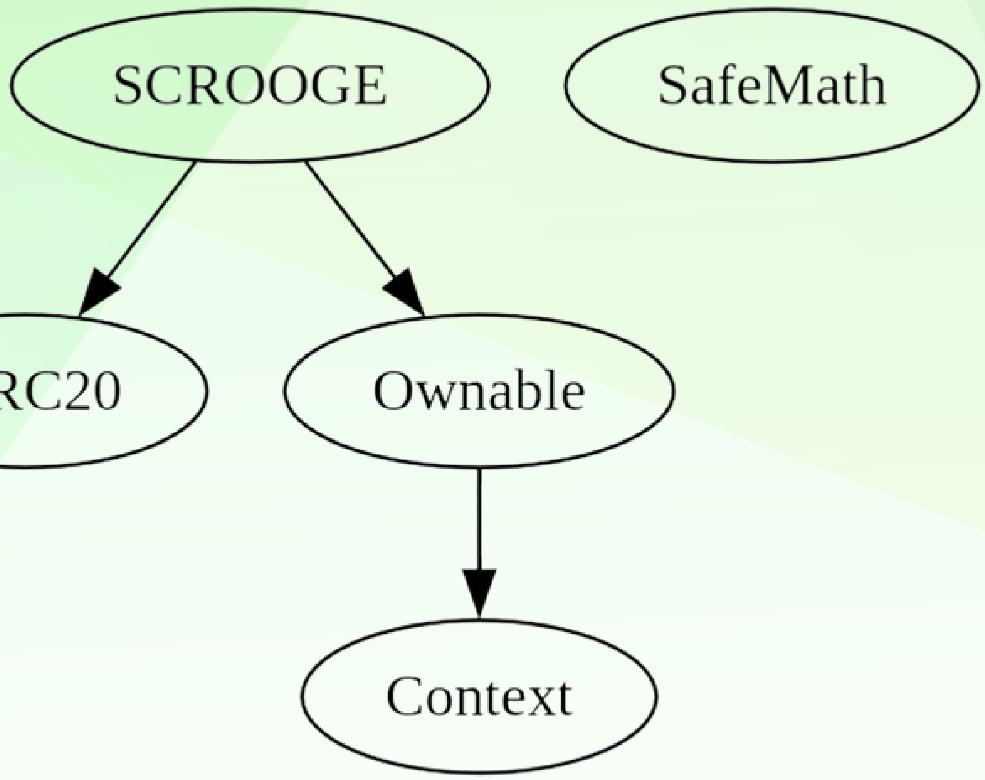
CLASSIFICATION OF RISK

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

Findings

Severity	Found
◆ **Critical**	0
◆ **High-Risk**	0
◆ **Medium-Risk**	0
◆ **Low-Risk**	0
◆ **Gas Optimization / Suggestions**	1

INHERITANCE TREE



POINTS TO NOTE

- Owner is not able to set buy/sell fees more than 10% (20% max fee)
- Owner is not able to set transfer fees (0% always)
- Owner is not able to set max buy/sell/transfer/hold amount
- Owner is not able to blacklist an arbitrary wallet
- Owner is not able to disable trades
- Owner is not able to mint new tokens
- **Owner must enable trading for investors**

CONTRACT ASSESMENT

```
| Contract |        Type        |       Bases      |                  |                 |
|:----------:|:-------------------:|:----------------:|:----------------:|:---------------:|
|     └      | **Function Name** | **Visibility** | **Mutability** | **Modifiers** |
| **IERC20** | Interface |  |||
| └ | totalSupply | External ❗ |   |NO ❗ |
| └ | balanceOf | External ❗ |   |NO ❗ |
| └ | transfer | External ❗ | 🛑  |NO ❗ |
| └ | allowance | External ❗ |   |NO ❗ |
| └ | approve | External ❗ | 🛑  |NO ❗ |
| └ | transferFrom | External ❗ | 🛑  |NO ❗ |
||||||
| **SafeMath** | Library |  |||
| └ | add | Internal 🔒 |   | |
| └ | sub | Internal 🔒 |   | |
| └ | sub | Internal 🔒 |   | |
| └ | mul | Internal 🔒 |   | |
| └ | div | Internal 🔒 |   | |
| └ | div | Internal 🔒 |   | |
||||||
| **Context** | Implementation |  |||
| └ | _msgSender | Internal 🔒 |   | |
| └ | _msgData | Internal 🔒 |   | |
||||||
| **IDEXFactory** | Interface |  |||
| └ | createPair | External ❗ | 🛑  |NO ❗ |
||||||
| **IPancakePair** | Interface |  |||
| └ | sync | External ❗ | 🛑  |NO ❗ |
||||||
| **IDEXRouter** | Interface |  |||
| └ | factory | External ❗ |   |NO ❗ |
| └ | WETH | External ❗ |   |NO ❗ |
| └ | addLiquidityETH | External ❗ |  💵 |NO ❗ |
| └ | swapExactTokensForETHSupportingFeeOnTransferTokens | External ❗ | 🛑  |NO ❗ |
||||||
| **Ownable** | Implementation | Context |||
| └ | <Constructor> | Public ❗ | 🛑  |NO ❗ |
| └ | owner | Public ❗ |   |NO ❗ |
| └ | renounceOwnership | Public ❗ | 🛑  | onlyOwner |
| └ | transferOwnership | Public ❗ | 🛑  | onlyOwner |
||||||
| **SCROOGE** | Implementation | IERC20, Ownable |||
| └ | <Constructor> | Public ❗ | 🛑  |NO ❗ |
```

CONTRACT ASSESMENT

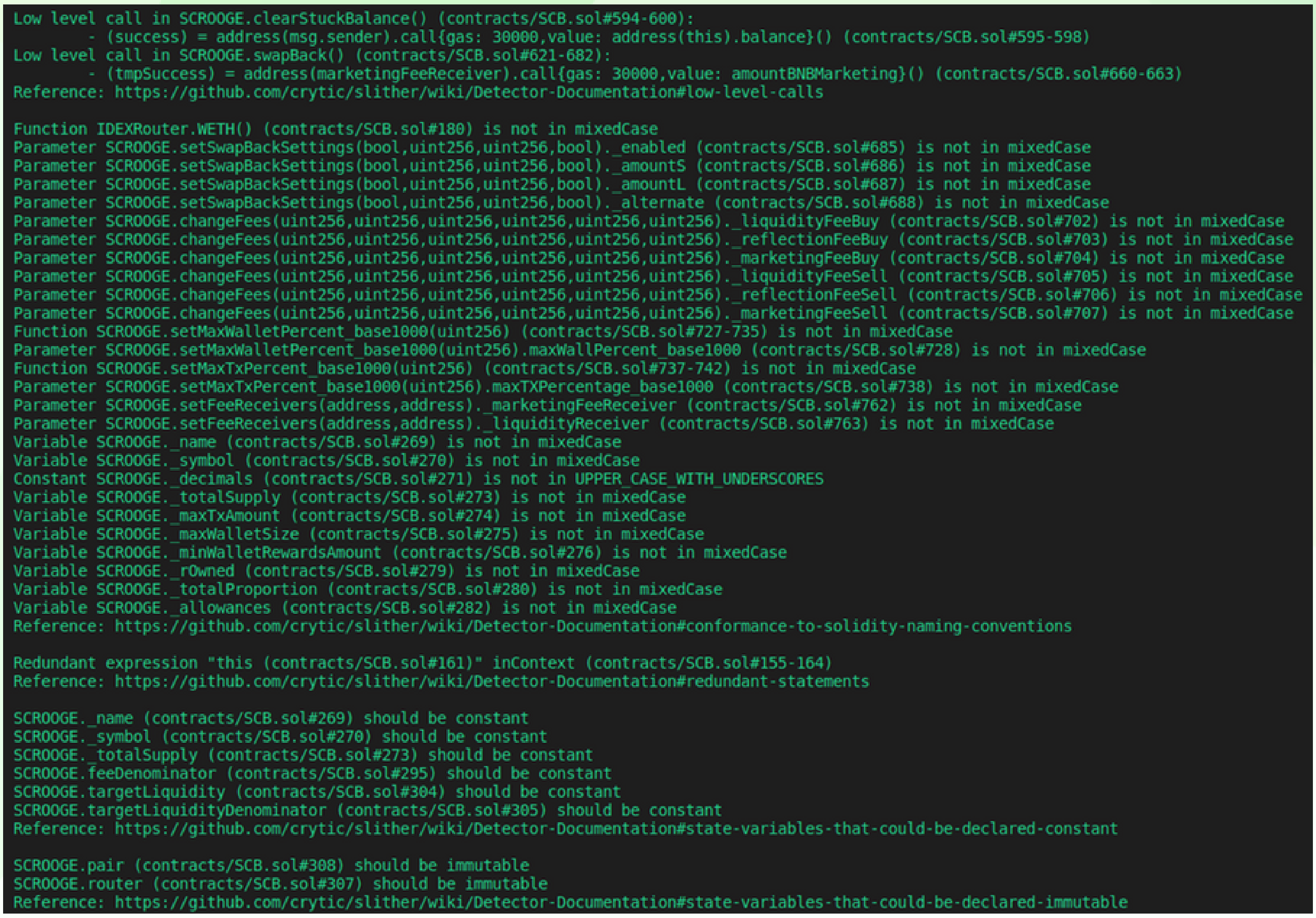
```
| └ | <Receive Ether> | External ❗ | 💵 |NO ❗ |
| └ | totalSupply | External ❗ | |NO ❗ |
| └ | decimals | External ❗ | |NO ❗ |
| └ | name | External ❗ | |NO ❗ |
| └ | symbol | External ❗ | |NO ❗ |
| └ | getOwner | External ❗ | |NO ❗ |
| └ | balanceOf | Public ❗ | |NO ❗ |
| └ | allowance | External ❗ | |NO ❗ |
| └ | viewFeesBuy | External ❗ | |NO ❗ |
| └ | viewFeesSell | External ❗ | |NO ❗ |
| └ | approve | Public ❗ | 🛑 |NO ❗ |
| └ | approveMax | External ❗ | 🛑 |NO ❗ |
| └ | transfer | External ❗ | 🛑 |NO ❗ |
| └ | transferFrom | External ❗ | 🛑 |NO ❗ |
| └ | _transferFrom | Internal 🔒 | 🛑 | |
| └ | tokensToProportion | Public ❗ | |NO ❗ |
| └ | tokenFromReflection | Public ❗ | |NO ❗ |
| └ | _basicTransfer | Internal 🔒 | 🛑 | |
| └ | shouldTakeFee | Internal 🔒 | | |
| └ | checkTxLimit | Internal 🔒 | | |
| └ | getTotalFeeBuy | Public ❗ | |NO ❗ |
| └ | getTotalFeeSell | Public ❗ | |NO ❗ |
| └ | takeFeeInProportions | Internal 🔒 | 🛑 | |
| └ | clearStuckBalance | External ❗ | 🛑 | onlyOwner |
| └ | clearForeignToken | Public ❗ | 🛑 |NO ❗ |
| └ | shouldSwapBack | Internal 🔒 | | |
| └ | swapBack | Internal 🔒 | 🛑 | swapping |
| └ | setSwapBackSettings | External ❗ | 🛑 | onlyOwner |
| └ | enableTrading | Public ❗ | 🛑 | onlyOwner |
| └ | changeFees | External ❗ | 🛑 | onlyOwner |
| └ | setMaxWalletPercent_base1000 | External ❗ | 🛑 | onlyOwner |
| └ | setMaxTxPercent_base1000 | External ❗ | 🛑 | onlyOwner |
| └ | setIsFeeExempt | External ❗ | 🛑 | onlyOwner |
| └ | setIsTxLimitExempt | External ❗ | 🛑 | onlyOwner |
| └ | setMinWalletRewardAmount | External ❗ | 🛑 | onlyOwner |
| └ | setFeeReceivers | External ❗ | 🛑 | onlyOwner |
| └ | getCirculatingSupply | Public ❗ | |NO ❗ |
| └ | getLiquidityBacking | Public ❗ | |NO ❗ |
| └ | isOverLiquified | Public ❗ | |NO ❗ |
```

CONTRACT ASSESMENT

Legend

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable

STATIC ANALYSIS



```
Low level call in SCROOGE.clearStuckBalance() (contracts/SCB.sol#594-600):
        - (success) = address(msg.sender).call{gas: 30000,value: address(this).balance}() (contracts/SCB.sol#595-598)
Low level call in SCROOGE.swapBack() (contracts/SCB.sol#621-682):
        - (tmpSuccess) = address(marketingFeeReceiver).call{gas: 30000,value: amountBNBMarketing}() (contracts/SCB.sol#660-663)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Function IDEXRouter.WETH() (contracts/SCB.sol#180) is not in mixedCase
Parameter SCROOGE.setSwapBackSettings(bool,uint256,uint256,bool)._enabled (contracts/SCB.sol#685) is not in mixedCase
Parameter SCROOGE.setSwapBackSettings(bool,uint256,uint256,bool)._amountS (contracts/SCB.sol#686) is not in mixedCase
Parameter SCROOGE.setSwapBackSettings(bool,uint256,uint256,bool)._amountL (contracts/SCB.sol#687) is not in mixedCase
Parameter SCROOGE.setSwapBackSettings(bool,uint256,uint256,bool)._alternate (contracts/SCB.sol#688) is not in mixedCase
Parameter SCROOGE.changeFees(uint256,uint256,uint256,uint256,uint256,uint256)._liquidityFeeBuy (contracts/SCB.sol#702) is not in mixedCase
Parameter SCROOGE.changeFees(uint256,uint256,uint256,uint256,uint256,uint256)._reflectionFeeBuy (contracts/SCB.sol#703) is not in mixedCase
Parameter SCROOGE.changeFees(uint256,uint256,uint256,uint256,uint256,uint256)._marketingFeeBuy (contracts/SCB.sol#704) is not in mixedCase
Parameter SCROOGE.changeFees(uint256,uint256,uint256,uint256,uint256,uint256)._liquidityFeeSell (contracts/SCB.sol#705) is not in mixedCase
Parameter SCROOGE.changeFees(uint256,uint256,uint256,uint256,uint256,uint256)._reflectionFeeSell (contracts/SCB.sol#706) is not in mixedCase
Parameter SCROOGE.changeFees(uint256,uint256,uint256,uint256,uint256,uint256)._marketingFeeSell (contracts/SCB.sol#707) is not in mixedCase
Function SCROOGE.setMaxWalletPercent_base1000(uint256) (contracts/SCB.sol#727-735) is not in mixedCase
Parameter SCROOGE.setMaxWalletPercent_base1000(uint256).maxWallPercent_base1000 (contracts/SCB.sol#728) is not in mixedCase
Function SCROOGE.setMaxTxPercent_base1000(uint256) (contracts/SCB.sol#737-742) is not in mixedCase
Parameter SCROOGE.setMaxTxPercent_base1000(uint256).maxTXPercentage_base1000 (contracts/SCB.sol#738) is not in mixedCase
Parameter SCROOGE.setFeeReceivers(address,address)._marketingFeeReceiver (contracts/SCB.sol#762) is not in mixedCase
Parameter SCROOGE.setFeeReceivers(address,address)._liquidityReceiver (contracts/SCB.sol#763) is not in mixedCase
Variable SCROOGE._name (contracts/SCB.sol#269) is not in mixedCase
Variable SCROOGE._symbol (contracts/SCB.sol#270) is not in mixedCase
Constant SCROOGE._decimals (contracts/SCB.sol#271) is not in UPPER_CASE_WITH_UNDERSCORES
Variable SCROOGE._totalSupply (contracts/SCB.sol#273) is not in mixedCase
Variable SCROOGE._maxTxAmount (contracts/SCB.sol#274) is not in mixedCase
Variable SCROOGE._maxWalletSize (contracts/SCB.sol#275) is not in mixedCase
Variable SCROOGE._minWalletRewardsAmount (contracts/SCB.sol#276) is not in mixedCase
Variable SCROOGE._rOwned (contracts/SCB.sol#279) is not in mixedCase
Variable SCROOGE._totalProportion (contracts/SCB.sol#280) is not in mixedCase
Variable SCROOGE._allowances (contracts/SCB.sol#282) is not in mixedCase
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Redundant expression "this (contracts/SCB.sol#161)" inContext (contracts/SCB.sol#155-164)
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#redundant-statements

SCROOGE._name (contracts/SCB.sol#269) should be constant
SCROOGE._symbol (contracts/SCB.sol#270) should be constant
SCROOGE._totalSupply (contracts/SCB.sol#273) should be constant
SCROOGE.feeDenominator (contracts/SCB.sol#295) should be constant
SCROOGE.targetLiquidity (contracts/SCB.sol#304) should be constant
SCROOGE.targetLiquidityDenominator (contracts/SCB.sol#305) should be constant
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

SCROOGE.pair (contracts/SCB.sol#308) should be immutable
SCROOGE.router (contracts/SCB.sol#307) should be immutable
Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-immutable
```

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

No issues found

FUNCTIONAL TESTING

1- Adding liquidity (passed):
https://testnet.bscscan.com/tx/0x528daa394eafe3c274369bc6fd1685e0fe6d7cd0416d76af65087ef10a213175

2- Buying when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x5dbdf3b07fc5f5a9c681761ad01a3c75203ea76160de25f150fb0370bc38ae13

3- Selling when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x949a3aab985523beeecc8ef76c7f071cfc89fdd4a8b9b1fcdf159443180d4145

4- Transferring when excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0xa38cc1f17ce16507c44d40c2c439c0f9e839f8a65dbce61bdd85212d77f33294

5- Buying when not excluded from fees (up to 12% tax) (passed):
https://testnet.bscscan.com/tx/0x6d81ddbd8771f60b40229afe1f66bad3996a00f8a7c1c306b74cf484399e67a5

6- Selling when not excluded from fees (up to 12% tax) (passed):
https://testnet.bscscan.com/tx/0x3280b6d5558ef6512ad6d3af3103e1f6b3ccba3257b738bc914f6df1a093f821

7- Transferring when not excluded from fees (0% tax) (passed):
https://testnet.bscscan.com/tx/0x8250e159bc31e3a474b11ddf70c91631a9f9086183c489d6d8c48bead5985863

FUNCTIONAL TESTING

8- Internal swap (passed):
As can seen in this transaction, marketing wallet received BNB
https://testnet.bscscan.com/token/0xbaf6001499915a03b5dd2e020e9e721df4e36955?a=0x322dab6325de6f5bc2ba8efecc2bcbecac4f89f3

9- Auto Liquidity (passed):
Auto liquidity generated tokens are burnt, as can be seen in this transaction
https://testnet.bscscan.com/token/0xbaf6001499915a03b5dd2e020e9e721df4e36955?a=0x322dab6325de6f5bc2ba8efecc2bcbecac4f89f3

MANUAL TESTING

Centralization - Owner must enable trading

Severity: Informational
Function: enableTrading
Status: Not Resolved
Overview:
The owner must activate trading for investors to buy, sell, or transfer tokens. If trading remains disabled, token holders will be unable to trade their tokens.

```
function enableTrading() public onlyOwner {
    tradingOpen = true;
}
```

DISCLAIMER

ABOUT AUDITACE

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

https://auditace.tech/

https://t.me/Audit_Ace

https://twitter.com/auditace_

https://github.com/Audit-Ace