SAFUAUDIT
SMART CONTRACT AUDITING

FAZEINU (FINU)

SMART CONTRACT AUDIT

March 19, 2022

INTRODUCTION

Client	Fazelnu (FINU)
Language	Solidity
Contract address	0xDFd1E9e3Be422d2c1FE9740fC241d08d99b31512
Decimals	18
Supply	1,000,000,000
Platform	Binance Smart Chain
Compiler	v0.8.10+commit.fc410830
Optimization	Yes, with 200 runs
Website	https://fazeinu.com/
Telegram	https://t.me/fazeinu
Twitter	https://twitter.com/Fazelnu

Description

Fazelnu is an innovative Binance Smart Chain Token that maximizes profit with DeFI Yield Generation and Crypto Earning Systems. Fazelnu Token has been created with a progressive, automatic burn mechanism.

Fazelnu source code which itself generates the Self Executable database shows how diverse the Intra-Capability of the Web3 version it is capable of.

TABLE OF CONTENTS

Approach

Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.

Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch, that lead to scams and rugpulls.

,

Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
- Back-doors
- Vulnerability
- Accuracy
- Readability

Tools

- Remix IDE
- MythX, Mythril
- SWC Registry
- Open Zeppelin Code Analyzer
- Solidity Code Complier

RISK CLASSIFICATION

CRITICAL

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

MEDIUM

Issues on this level could potentially bring problems and should eventually be fixed.

MINOR

Issues on this level are minor details and warning that can remain unfixed but would be better fixed at some point in the future

INFORMATIONAL

Information level is to offer suggestions for improvement of efficacity or security for features with a risk free factor.

ABSTRACT

Fees	Ownership	Mint function
Buy Fees: 7% Sell Fees: 7% *at audit time	Owned	No mint function found

Owner can set fees	Max Tx amount	Pause
Owner can set fees up to 100%	Owner can't set max Tx amount	Owner can pause trading

Vulnerabilities Test

SWC ID	Description	
SWC-100	Function Default Visibility	Passed
SWC-101	Integer Overflow and Underflow	Passed
SWC-102	Outdated Compiler Version	Passed
SWC-103	Floating Pragma	Minor
SWC-104	Unchecked Call Return Value	Medium
SWC-105	Unprotected Ether Withdrawal	Passed
SWC-106	Unprotected SELF-DESTRUCT Instruction	Passed
SWC-107	Re-entrancy	Passed
SWC-108	State Variable Default Visibility	Passed
SWC-109	Uninitialized Storage Pointer	Passed
SWC-110	Assert Violation	Passed
SWC-111	Use of Deprecated Solidity Functions	Passed
SWC-112	Delegate Call to Untrusted Callee	Passed
SWC-113	DoS with Failed Call	Passed
SWC-114	Transaction Order Dependence	Passed
SWC-115	Authorization through tx.origin	Minor

SWC-116	Block values as a proxy for time	**Passed**
SWC-117	Signature Malleability	**Passed**
SWC-118	Incorrect Constructor Name	**Passed**
SWC-119	Shadowing State Variables	**Passed**
SWC-120	Weak Sources of Randomness from Chain Attributes	**Passed**
SWC-121	Missing Protection against Signature Replay Attacks	**Passed**
SWC-122	Lack of Proper Signature Verification	**Passed**
SWC-123	Requirement Violation	**Passed**
SWC-124	Write to Arbitrary Storage Location	**Passed**
SWC-125	Incorrect Inheritance Order	**Passed**
SWC-126	Insufficient Gas Griefing	**Passed**
SWC-127	Arbitrary Jump with Function Type Variable	**Passed**
SWC-128	DoS With Block Gas Limit	**Passed**
SWC-129	Typographical Error	**Passed**
SWC-130	Right-To-Left-Override control character (U+202E)	**Passed**
SWC-131	Presence of unused variables	**Passed**
SWC-132	Unexpected Ether balance	**Passed**
SWC-133	Hash Collisions With Multiple Variable Length Arguments	**Passed**
SWC-134	Message call with the hardcoded gas amount	**Passed**
SWC-135	Code With No Effects (Irrelevant/Dead Code)	**Passed**
SWC-136	Unencrypted Private Data On-Chain	**Passed**

MANUAL ANALYSIS

The contract is verified to check if functions do and work as they should and malicious code is not inserted.

	Tested	Result
Transfer	Yes	**Passed**
Total Supply	Yes	**Passed**
Buy Back	Yes	**N/A**
Burn	Yes	**Passed**
Mint	Yes	**N/A**
Rebase	Yes	**N/A**
Pause	Yes	**Passed**
Blacklist	Yes	**Passed**
Lock	Yes	**N/A**
Max Transaction	Yes	**N/A**
Transfer Ownership	Yes	**Passed**
Renounce Ownership	Yes	**Passed**

MANUAL AUDIT

CONTRACT INSPECTION

IERC20	Interface			
└	totalSupply	External ❗		|NO❗
└	balanceOf	External ❗		|NO❗
└	transfer	External ❗	🛑	|NO❗
└	allowance	External ❗		|NO❗
└	approve	External ❗	🛑	|NO❗
└	transferFrom	External ❗	🛑	|NO❗
IERC20Metadata	Interface	IERC20		
└	name	External ❗		|NO❗
└	symbol	External ❗		|NO❗
└	decimals	External ❗		|NO❗
Context	Implementation			
└	_msgSender	Internal 🔒			
└	_msgData	Internal 🔒			
ERC20	Implementation	Context, IERC20, IERC20Metadata		
└	<Constructor>	Public ❗	🛑	|NO❗
└	name	Public ❗		|NO❗
└	symbol	Public ❗		|NO❗
└	decimals	Public ❗		|NO❗
└	totalSupply	Public ❗		|NO❗
└	balanceOf	Public ❗		|NO❗
└	transfer	Public ❗	🛑	|NO❗
└	allowance	Public ❗		|NO❗
└	approve	Public ❗	🛑	|NO❗
└	transferFrom	Public ❗	🛑	|NO❗
└	increaseAllowance	Public ❗	🛑	|NO❗
└	decreaseAllowance	Public ❗	🛑	|NO❗
└	_transfer	Internal 🔒	🛑		
└	_mint	Internal 🔒	🛑		
└	_burn	Internal 🔒	🛑		
└	_approve	Internal 🔒	🛑		
└	_beforeTokenTransfer	Internal 🔒	🛑		
└	_afterTokenTransfer	Internal 🔒	🛑		

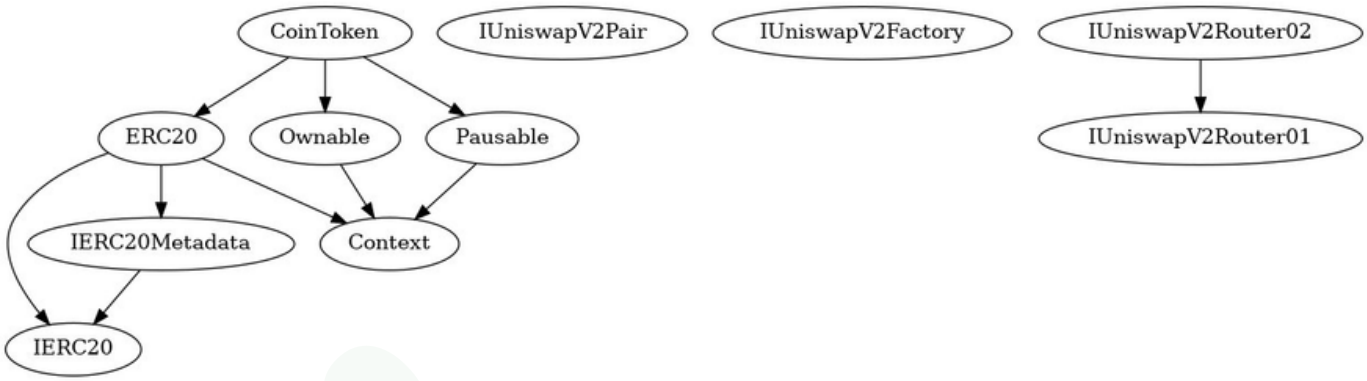
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_setOwner	Internal 🔒	🛑		
Pausable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	paused	Public ❗		NO❗
└	_pause	Internal 🔒	🛑	whenNotPaused
└	_unpause	Internal 🔒	🛑	whenPaused
IUniswapV2Pair	Interface			
└	name	External ❗		NO❗
└	symbol	External ❗		NO❗
└	decimals	External ❗		NO❗
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transfer	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
└	DOMAIN_SEPARATOR	External ❗		NO❗
└	PERMIT_TYPEHASH	External ❗		NO❗
└	nonces	External ❗		NO❗
└	permit	External ❗	🛑	NO❗
└	MINIMUM_LIQUIDITY	External ❗		NO❗
└	factory	External ❗		NO❗
└	token0	External ❗		NO❗
└	token1	External ❗		NO❗
└	getReserves	External ❗		NO❗
└	price0CumulativeLast	External ❗		NO❗
└	price1CumulativeLast	External ❗		NO❗
└	kLast	External ❗		NO❗
└	mint	External ❗	🛑	NO❗
└	burn	External ❗	🛑	NO❗
└	swap	External ❗	🛑	NO❗

└	skim	External ❗	🛑	NO❗
└	sync	External ❗	🛑	NO❗
└	initialize	External ❗	🛑	NO❗
IUniswapV2Factory	Interface				
└	feeTo	External ❗		NO❗
└	feeToSetter	External ❗		NO❗
└	getPair	External ❗		NO❗
└	allPairs	External ❗		NO❗
└	allPairsLength	External ❗		NO❗
└	createPair	External ❗	🛑	NO❗
└	setFeeTo	External ❗	🛑	NO❗
└	setFeeToSetter	External ❗	🛑	NO❗
IUniswapV2Router01	Interface				
└	factory	External ❗		NO❗
└	WETH	External ❗		NO❗
└	addLiquidity	External ❗	🛑	NO❗
└	addLiquidityETH	External ❗	💵	NO❗
└	removeLiquidity	External ❗	🛑	NO❗
└	removeLiquidityETH	External ❗	🛑	NO❗
└	removeLiquidityWithPermit	External ❗	🛑	NO❗
└	removeLiquidityETHWithPermit	External ❗	🛑	NO❗
└	swapExactTokensForTokens	External ❗	🛑	NO❗
└	swapTokensForExactTokens	External ❗	🛑	NO❗
└	swapExactETHForTokens	External ❗	💵	NO❗
└	swapTokensForExactETH	External ❗	🛑	NO❗
└	swapExactTokensForETH	External ❗	🛑	NO❗
└	swapETHForExactTokens	External ❗	💵	NO❗
└	quote	External ❗		NO❗
└	getAmountOut	External ❗		NO❗
└	getAmountIn	External ❗		NO❗
└	getAmountsOut	External ❗		NO❗
└	getAmountsIn	External ❗		NO❗
IUniswapV2Router02	Interface	IUniswapV2Router01			
└	removeLiquidityETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	removeLiquidityETHWithPermitSupportingFeeOnTransferTokens	External ❗	🛑	NO❗

└	swapExactTokensForTokensSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
└	swapExactETHForTokensSupportingFeeOnTransferTokens	External ❗	💵	NO❗
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🛑	NO❗
CoinToken	Implementation	ERC20, Ownable, Pausable		
└	<Constructor>	Public ❗	💵	ERC20
└	handleTax	Private 🔐	🛑	
└	_transfer	Internal 🔒	🛑	
└	triggerTax	Public ❗	🛑	onlyOwner
└	pause	Public ❗	🛑	onlyOwner
└	unpause	Public ❗	🛑	onlyOwner
└	burn	Public ❗	🛑	onlyOwner
└	enableBlacklist	Public ❗	🛑	onlyOwner
└	disableBlacklist	Public ❗	🛑	onlyOwner
└	exclude	Public ❗	🛑	onlyOwner
└	removeExclude	Public ❗	🛑	onlyOwner
└	setBuyTax	Public ❗	🛑	onlyOwner
└	setSellTax	Public ❗	🛑	onlyOwner
└	setTaxWallets	Public ❗	🛑	onlyOwner
└	enableTax	Public ❗	🛑	onlyOwner
└	disableTax	Public ❗	🛑	onlyOwner
└	isBlacklisted	Public ❗		NO❗
└	isExcluded	Public ❗		NO❗
└	<Receive Ether>	External ❗	💵	NO❗

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable
🔐	Private function
🔒	Internal function
NO❗	Function has no modifier

INHERITANCE TREE



Inheritance is a feature of the object-oriented programming language. It is a way of extending the functionality of a program, used to separate the code, reduces the dependency, and increases the re-usability of the existing code. Solidity supports inheritance between smart contracts, where multiple contracts can be inherited into a single contract.

Important Snippets

Owner can pause/unpause contract

```
function pause() public onlyOwner {
    require(!paused(), "CoinToken: Contract is already paused");
    _pause();
}

function unpause() public onlyOwner {
    require(paused(), "CoinToken: Contract is not paused");
    _unpause();
}
```

Owner can blacklist / remove blacklisted wallets

```
function enableBlacklist(address account) public onlyOwner {
    require(!blacklist[account], "CoinToken: Account is already blacklisted");
    blacklist[account] = true;
}

/**
 * @dev Remove the specified account from the blacklist.
 */
function disableBlacklist(address account) public onlyOwner {
    require(blacklist[account], "CoinToken: Account is not blacklisted");
    blacklist[account] = false;
}
```

Owner can set buy/sell fees up to 100%

```
function setBuyTax(uint256 dev, uint256 marketing, uint256 liquidity, uint256 charity) public onlyOwner {
    buyTaxes["dev"] = dev;
    buyTaxes["marketing"] = marketing;
    buyTaxes["liquidity"] = liquidity;
    buyTaxes["charity"] = charity;
}

/**
 * @dev Sets tax for sells.
 */
function setSellTax(uint256 dev, uint256 marketing, uint256 liquidity, uint256 charity) public onlyOwner {

    sellTaxes["dev"] = dev;
    sellTaxes["marketing"] = marketing;
    sellTaxes["liquidity"] = liquidity;
    sellTaxes["charity"] = charity;
}
```

GOOD PRACTICES

- The owner cannot mint new tokens
- The owner cannot set max Tx

WEBSITE

Website	https://fazeinu.com/
Domain Registry	https://www.godaddy.com
Domain Expiry Date	2023-02-25
Response Code	200
SSL Checker and HTTPS Test	Passed
Deprecated HTML tags	Passed
Robots.txt	Passed
Sitemap Test	Passed
SEO Friendly URL	Passed
Responsive Test	Passed
JS Error Test	Passed
Console Errors Test	Passed
Site Loading Speed Test	2.06 seconds - Passed
HTTP2 Test	Passed
Safe Browsing Test	Passed

DISCLAIMER

SafuAudit.com is not a financial institution and the information provided on this website does not constitute investment advice, financial advice, trading advice or any other sort of advice. You should not treat any of the website's content as such. Investing in crypto assets carries a high level of risk and does not hold guarantees for not sustaining financial loss due to their volatility.

Accuracy of Information

SafuAudit will strive to ensure accuracy of information listed on this website although it will not hold any responsibility for any missing or wrong information. SafuAudit provides all information as is. You understand that you are using any and all information available here at your own risk. Any use or reliance on our content and services is solely at your own risk and discretion.

The purpose of the audit is to analyse the on-chain smart contract source code, and to provide basic overview of the project.

While we have used all the information available to us for this straightforward investigation, you should not rely on this report only — we recommend proceeding with several independent audits Be aware that smart contracts deployed on a blockchain aren't secured enough against external vulnerability, or a hack. Be aware that active smart contract owner privileges constitute an elevated impact to smart contract's safety and security. Therefore, SafuAudit does not guarantee the explicit security of the audited smart contract. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

AUDIT RESULTS

CRITICAL

No critical severity issues have been found.

MEDIUM

- Owner can set fees up to 100%
- Owner can enable/disable trading
- Low level call in CoinToken.handleTax(address,address,uint256)(#894-994):
  - taxWallets[marketing].call{value: marketingETH}()(#978)
  - taxWallets[dev].call{value: devETH}()(#979)
  - taxWallets[charity].call{value: charityETH}()(#980)
  - taxWallets[marketing].call{value: ethGained - (marketingETH + devETH + liquidityETH + charityETH)}()(#979)

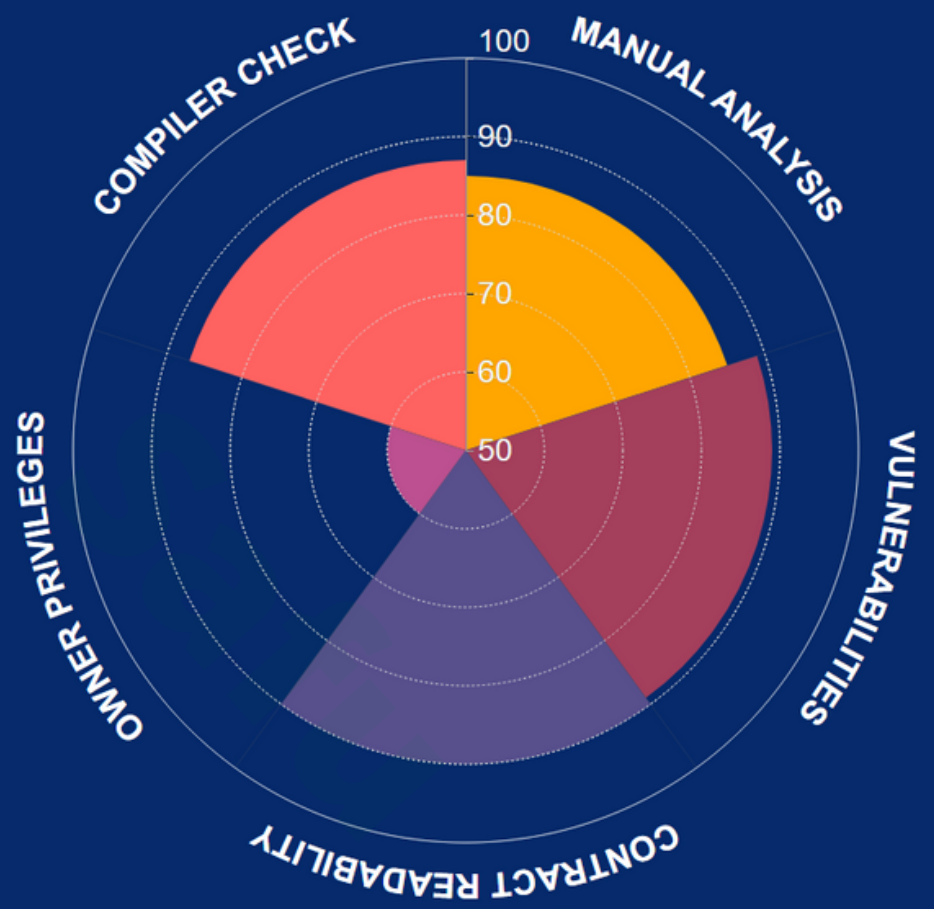
Avoid low-level calls. If the call is meant for a contract, check for code existence

MINOR

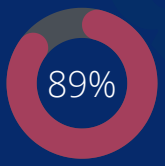
- A floating pragma is set. It is recommended to specify a fixed compiler version to ensure that the bytecode produced does not vary between builds. This is especially important if you rely on bytecode-level verification of the code.
- tx.origin - Using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" instead.

INFORMATIONAL

SCORE

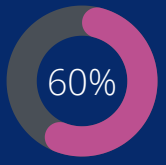


Manual Analysis

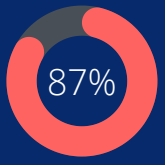


Vulnerabilities

Contract Readability



Owner Privileges

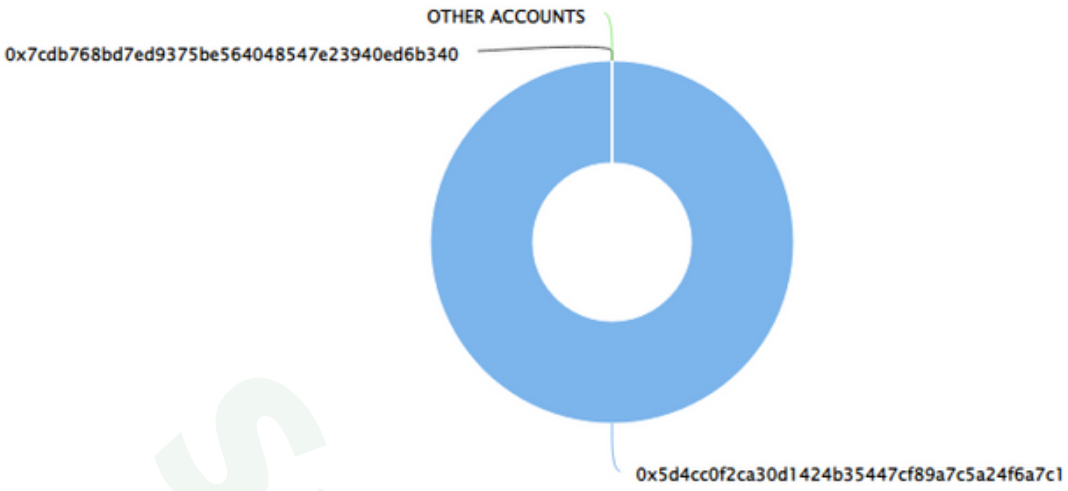


Compiler Check

Final Score:82.2

SUMMARY

Top 10 holders



Rank	Address	Quantity (Token)	Percentage
1	0x5d4cc0f2ca30d1424b35447cf89a7c5a24f6a7c1	999,999,999	100.0000%
2	0x7cdb768bd7ed9375be564048547e23940ed6b340	1	0.0000%

Conclusion

Project Fazelnu does not contain any severe issues. Owner can set fees up to 100% and pause transactions. Please verify that developers have no malicious intents or have no control over the contract in the future.

SafuAudit has tested the security based on manual and automated tests. Please note that we don't offer any warranties for business model.

SafuAudit.com