SAFUAUDIT
SMART CONTRACT AUDITING

WORLDUNITY

SMART CONTRACT AUDIT

March 7, 2022

INTRODUCTION

Client	World Unity Token (WUT2)
Language	Solidity
Contract address	0xf7ecfa5852253bC035a5c9FC5FFCA5d156De740e
Decimals	18
Supply	10,000,000,000,000
Platform	Binance Smart Chain
Compiler	v0.8.0+commit.c7dfd78e
Optimization	Yes, with 200 runs
Website	https://bryce3757014.wixsite.com/worldunitytoken
Telegram	https://t.me/WorldunityToken
Twitter	-

Description

World Unity Token is a project created by crypto participants and enthusiasts who are willing to make a difference to the lives of civilians affected by what is happening in Ukraine in 2022. The project is powered by the donations and trading frequency of the crypto community.

TABLE OF CONTENTS

01 INTRODUCTION

02 ABSTRACT

03 VULNERABILITIES TEST

04 MANUAL ANALYSIS

05 WEBSITE

06 CONCLUSIONS

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: 16% Sell Fees: 20%	Owned	No mint function found

Owner can set fees	Max Tx amount	Pause
Owner can set fees up to 100%	Owner can set max tx amount	Owner can't 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SWC-102	Outdated Compiler Version	Passed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	Passed
SWC-105	Unprotected Ether Withdrawal	Passed
SWC-106	Unprotected SELF-DESTRUCT Instruction	Passed
SWC-107	Re-entrancy	Passed
SWC-108	State Variable Default Visibility	Minor
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	**N/A**
Blacklist	Yes	**N/A**
Lock	Yes	**N/A**
Max Transaction	Yes	**Passed**
Transfer Ownership	Yes	**Passed**
Renounce Ownership	Yes	**Passed**

MANUAL AUDIT

CONTRACT INSPECTION

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❗
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❗

SignedSafeMath	Library			
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	sub	Internal 🔒		
└	add	Internal 🔒		
SafeMath	Library			
└	tryAdd	Internal 🔒		
└	trySub	Internal 🔒		
└	tryMul	Internal 🔒		
└	tryDiv	Internal 🔒		
└	tryMod	Internal 🔒		
└	add	Internal 🔒		
└	sub	Internal 🔒		
└	mul	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
└	sub	Internal 🔒		
└	div	Internal 🔒		
└	mod	Internal 🔒		
SafeCast	Library			
└	toUint224	Internal 🔒		
└	toUint128	Internal 🔒		
└	toUint96	Internal 🔒		
└	toUint64	Internal 🔒		
└	toUint32	Internal 🔒		
└	toUint16	Internal 🔒		
└	toUint8	Internal 🔒		
└	toUint256	Internal 🔒		
└	toInt128	Internal 🔒		
└	toInt64	Internal 🔒		
└	toInt32	Internal 🔒		
└	toInt16	Internal 🔒		
└	toInt8	Internal 🔒		
└	toInt256	Internal 🔒		

Context	Implementation			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
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❗
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	Public ❗	🛑	NO❗
└	_approve	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
└	_afterTokenTransfer	Internal 🔒	🛑	

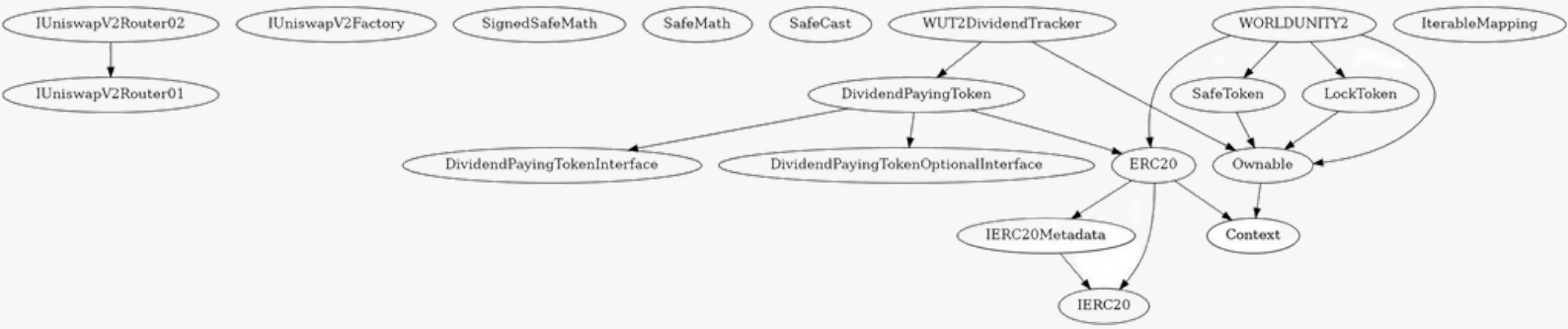
Ownable	Implementation	Context		
└	<Constructor>	Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	_setOwner	Private 🔐	🛑	
IterableMapping	Library			
└	get	Public ❗		NO❗
└	getIndexOfKey	Public ❗		NO❗
└	getKeyAtIndex	Public ❗		NO❗
└	size	Public ❗		NO❗
└	set	Public ❗	🛑	NO❗
└	remove	Public ❗	🛑	NO❗
DividendPayingTokenOptionalInterface	Interface			
└	withdrawableDividendOf	External ❗		NO❗
└	withdrawnDividendOf	External ❗		NO❗
└	accumulativeDividendOf	External ❗		NO❗
DividendPayingTokenInterface	Interface			
└	dividendOf	External ❗		NO❗
└	distributeDividends	External ❗	💸	NO❗
└	withdrawDividend	External ❗	🛑	NO❗
DividendPayingToken	Implementation	ERC20, DividendPayingTokenInterface, DividendPayingToken
└	<Constructor>	Public ❗	🛑	ERC20
└	<Receive Ether>	External ❗	💸	NO❗
└	distributeDividends	Public ❗	💸	NO❗
└	withdrawDividend	Public ❗	🛑	NO❗
└	_withdrawDividendOfUser	Internal 🔒	🛑	
└	dividendOf	Public ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	withdrawnDividendOf	Public ❗		NO❗
└	accumulativeDividendOf	Public ❗		NO❗
└	_transfer	Internal 🔒	🛑	
└	_mint	Internal 🔒	🛑	
└	_burn	Public ❗	🛑	NO❗
└	_setBalance	Internal 🔒	🛑	

WUT2DividendTracker	Implementation	DividendPayingToken, Ownable		
└	<Constructor>	Public ❗	🛑	DividendPayingToken
└	_transfer	Internal 🔒		
└	withdrawDividend	Public ❗		NO❗
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner
└	getLastProcessedIndex	External ❗		NO❗
└	getNumberOfTokenHolders	External ❗		NO❗
└	getAccount	Public ❗		NO❗
└	getAccountAtIndex	Public ❗		NO❗
└	canAutoClaim	Private 🔐		
└	setBalance	External ❗	🛑	onlyOwner
└	process	Public ❗	🛑	NO❗
└	processAccount	Public ❗	🛑	onlyOwner
SafeToken	Implementation	Ownable		
└	<Constructor>	Public ❗	🛑	NO❗
└	setSafeManager	Public ❗	🛑	onlyOwner
└	withdraw	External ❗	🛑	NO❗
└	withdrawBNB	External ❗	🛑	NO❗
LockToken	Implementation	Ownable		
└	<Constructor>	Public ❗	🛑	NO❗
└	openTrade	External ❗	🛑	onlyOwner
└	includeToWhiteList	External ❗	🛑	onlyOwner
WORLDUNITY2	Implementation	ERC20, Ownable, SafeToken, LockToken		
└	setFee	Public ❗	🛑	onlyOwner
└	setExtraFeeOnSell	Public ❗	🛑	onlyOwner
└	setMaxSelltx	Public ❗	🛑	onlyOwner
└	setMarketingWallet	Public ❗	🛑	onlyOwner
└	setpeopleWallet	Public ❗	🛑	onlyOwner
└	<Constructor>	Public ❗	🛑	ERC20
└	<Receive Ether>	External ❗	💶	NO❗
└	updateUniswapV2Router	Public ❗	🛑	onlyOwner
└	excludeFromFees	Public ❗	🛑	onlyOwner

└	setExcludeFromMaxTx	Public ❗	🛑	onlyOwner
└	setExcludeFromAll	Public ❗	🛑	onlyOwner
└	excludeMultipleAccountsFromFees	Public ❗	🛑	onlyOwner
└	setAutomatedMarketMakerPair	Public ❗	🛑	onlyOwner
└	setSWapTokensAtAmount	Public ❗	🛑	onlyOwner
└	_setAutomatedMarketMakerPair	Private 🔐	🛑	
└	updateGasForProcessing	Public ❗	🛑	onlyOwner
└	updateClaimWait	External ❗	🛑	onlyOwner
└	getClaimWait	External ❗		NO❗
└	getTotalDividendsDistributed	External ❗		NO❗
└	isExcludedFromFees	Public ❗		NO❗
└	isExcludedFromMaxTx	Public ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	dividendTokenBalanceOf	Public ❗		NO❗
└	getAccountDividendsInfo	External ❗		NO❗
└	getAccountDividendsInfoAtIndex	External ❗		NO❗
└	processDividendTracker	External ❗	🛑	NO❗
└	claim	External ❗	🛑	NO❗
└	getLastProcessedIndex	External ❗		NO❗
└	getNumberOfDividendTokenHolders	External ❗		NO❗
└	excludeFromDividends	External ❗	🛑	onlyOwner
└	setSwapAndLiquifyEnabled	Public ❗	🛑	onlyOwner
└	_transfer	Internal 🔒	🛑	open
└	swapAndLiquify	Private 🔐	🛑	lockTheSwap
└	swapTokensForBnb	Private 🔐	🛑	
└	swapAndSendBNBToMarketing	Private 🔐	🛑	
└	swapAndSendBNBTopeople	Private 🔐	🛑	
└	addLiquidity	Private 🔐	🛑	

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 exclude from fees

```
function excludeFromFees(address account, bool excluded) public onlyOwner {
    require(_isExcludedFromFees[account] != excluded, "WUT2: Account is already the value of 'excluded'");
    _isExcludedFromFees[account] = excluded;

    emit ExcludeFromFees(account, excluded);
}
```

Owner can set max transaction amount

```
function setMaxSelltx(uint256 _maxSellTxAmount) public onlyOwner {
    maxSellTransactionAmount = _maxSellTxAmount;
}
```

Owner can set fees up to 100%

```
function setFee(uint256 _bnbRewardFee, uint256 _liquidityFee, uint256 _marketingFee,
uint256 _peoplefee) public onlyOwner {
    BNBRewardsFee = _bnbRewardFee;
    liquidityFee = _liquidityFee;
    marketingFee = _marketingFee;
    peoplefee = _peoplefee;

    totalFees = BNBRewardsFee.add(liquidityFee).add(marketingFee).add(peoplefee);
}

function setExtraFeeOnSell(uint256 _extraFeeOnSell) public onlyOwner {
    extraFeeOnSell = _extraFeeOnSell; // extra fee on sell
}
```

GOOD PRACTICES

- The owner cannot stop or pause the smart contract
- The owner cannot mint tokens after initial deployment
- The smart contract utilizes "SafeMath" to prevent overflows

```
library SafeMath {
    function tryAdd(uint256 a, uint256 b) internal pure returns (bool, uint2  , {
        unchecked {
            uint256 c = a + b;
            if (c < a) return (false, 0);
            return (true, c);
        }
    }

    function trySub(uint256 a, uint256 b) internal pure returns (bool, uint256) {
        unchecked {
            if (b > a) return (false, 0);
            return (true, a - b);
        }
    }

    function tryMul(uint256 a, uint256 b) internal pure returns (bool, uint256) {
        unchecked {
            // Gas optimization: this is cheaper than requiring 'a' not being zero, but
            // benefit is lost if 'b' is also tested.
            // See: https://github.com/OpenZeppelin/openzeppelin-contracts/pull/522
            if (a == 0) return (true, 0);
            uint256 c = a * b;
            if (c / a != b) return (false, 0);
            return (true, c);
        }
    }

    function tryDiv(uint256 a, uint256 b) internal pure returns (bool, uint256) {
        unchecked {
            if (b == 0) return (false, 0);
            return (true, a / b);
        }
    }

    function tryMod(uint256 a, uint256 b) internal pure returns (bool, uint256) {
        unchecked {
            if (b == 0) return (false, 0);
            return (true, a % b);
        }
    }
```

WEBSITE

Website	https://bryce3757014.wixsite.com/worldunitytoken
Domain Registry	-
Domain Expiry Date	-
Response Code	200
SSL Checker and HTTPS Test	Passed
Deprecated HTML tags	Passed
Robots.txt	Informational
Sitemap Test	Informational
SEO Friendly URL	Passed
Responsive Test	Passed
JS Error Test	Passed
Console Errors Test	Informational
Site Loading Speed Test	1.13 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 to 100%
- Owner can set maxTx amount, if set to 0, transfers are blocked

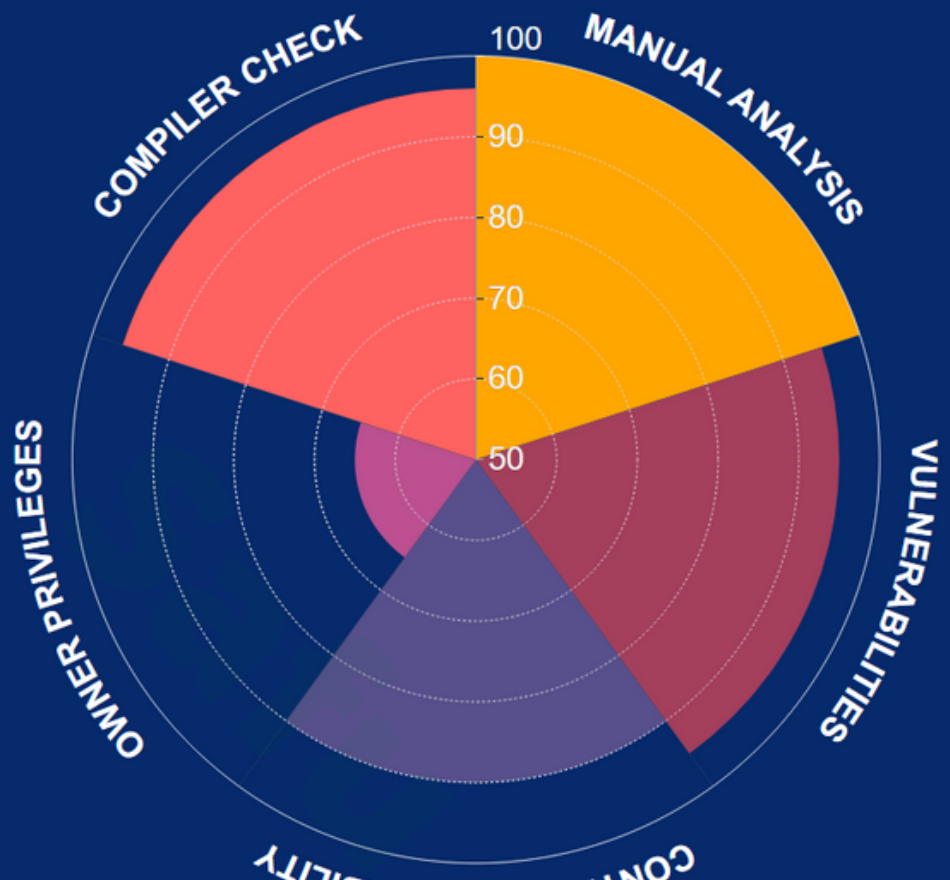
MINOR

- 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.
- State variable visibility is not set. It is best practice to set the visibility of state variables explicitly. The default visibility for "safeManager" is internal.
- Using "tx.origin" as a security control can lead to authorization bypass vulnerabilities. Consider using "msg.sender" .

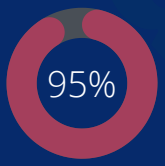
INFORMATIONAL

The standard audit model does not offer suggestions and consulting for improvements of efficacity.

SAFUSCORE

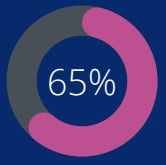


Manual Analysis



Vulnerabilities

Contract Readability



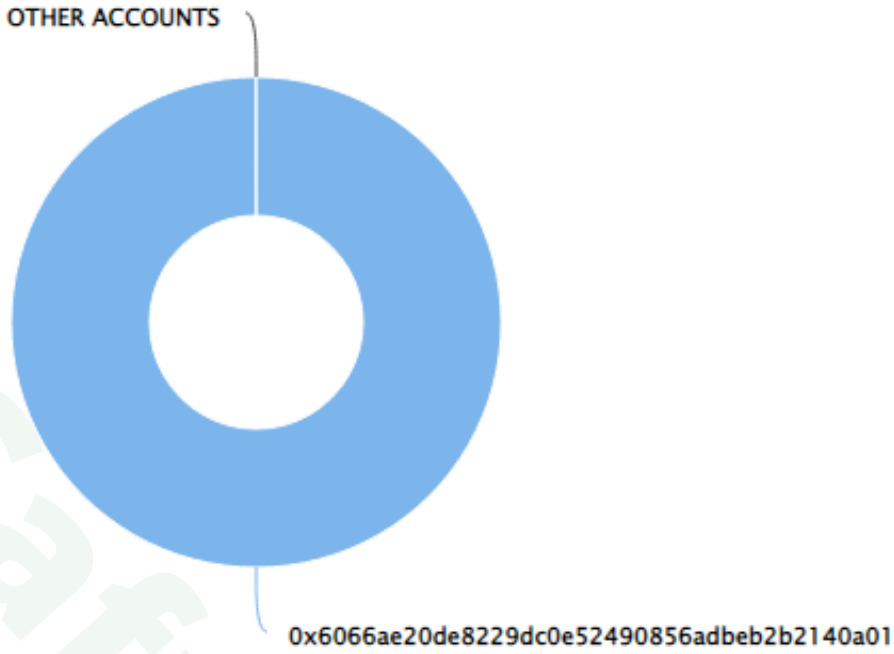
Owner Privileges

Compiler Check

Final Score:89.2

SUMMARY

Top 10 holders



Rank	Address	Quantity (Token)	Percentage
1	0x6066ae20de8229dc0e52490856adbeb2b2140a01	10,000,000,000,000	100.0000%

Conclusion

Project World Unity does not contain any severe issues, the owner can set fees to 100% and maxTx amount. 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