RugFreeCoins Audit

ThunderETH Token Audit

Smart Contract Security Audit

August 29, 2021

Contents

Audit details

Audited project

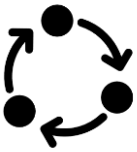
ThunderETH Token

Contract Address

0x73c499099BFc964D3CBcd7425026b074869748cE

Client contact

ThunderETH Token Team



Blockchain

Binance smart chain

Project website

https://thundereth.app/

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the disclaimer below – please make sure to read it in full.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and Rugfreecoins and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (Rugfreecoins) owe no duty of care towards you or any other person, nor does Rugfreecoins make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and Rugfreecoins hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, Rugfreecoins hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against Rugfreecoins, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. 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.

Background

Rugfreecoins was commissioned by ThunderETH to perform an audit of the smart contract.

https://bscscan.com/token/0x73c499099BFc964D3CBcd7425026b074869748cE

The focus of this audit is to verify that the smart contract is secure, resilient and working according to the specifications.

The information in this report should be used to understand the risk exposure of the smart contract, project feasibility, long term sustainability and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

About the project

ThunderETH is a token built on the Binance Smart Chain. Each transaction, purchase incur a 18% fee, and sales incur a 27% fee.

The first token of its type to provide its users with double reflection rewards, at a ratio of 80% given in a token which will provide immediate benefits to the user short term, and 20% provided in a token specifically built to provide profit for long term investors.

- **80% ETH distribution.**
- **20% THOREUM distribution.**

Features

- The **reward** system is what ThunderETH's entire marketing strategy is based around, especially in the early stages of the life of the coin. The notion of 'passive income' can be leveraged in ThunderETH's marketing approach and facilitate the next steps in the development and growth of the community. The **fee of 10% when buying and the fee of 15% when selling** from every trade will be divided to distribute **8%, 10% of ETH and 2%, 5% of THOREUM respectively** everyone hour among every holder proportional to how many tokens each individual holds.

- The **sustainability fee of 4% marketing fee when buying and selling** is what allows ThunderETH to hold the aforementioned promise. Tokens will be swapped into BNBs and will be sent to a marketing wallet per transaction. This way, ThunderETH will have enough funds to promote the coin and spend for future development without selling tokens as the traditional way.

- The 2% fee is used to buy back & burn ThunderADA, which is affiliated with the ThunderETH token.

- 2% auto liquidity pool and buy back & burn ThunderETH. Based on volume & liquidity analysis, 2% of every transaction is transformed either into liquidity for Pancakeswap or into ThunderETH automatic Buyback & burn fund. Once the system detects a sale, it will automatically buy back a certain value of tokens and burn. It's automatic and helps create either a higher price floor for stability or pump the price and rarify token.

- Anti-dump: Sales are restricted to less than 0.1% of the total supply. Buys have a 0.5% limitation. So only big buys and never see a big dump.

- Anti-bot Exploit: The Buyback and Burn will only cover the same sell value under a certain threshold. So massive bot-triggered micro-sells will not dry out the buyback fund.

Tokenomics

18% fee when buying

- 8% of trade goes to holders pockets in ETH.
- 2% of trade goes to holders pockets in **THOREUM**.
- 4% of trade will be allocated for marketing.
- 2% of trade will be allocated to buyback and burn ThunderADA.
- 2% of trade will be allocated for liquidity pool or buyback and burn ThunderETH.

27% fee when selling

- 12% of trade goes to holders pockets in ETH.
- 3% of trade goes to holders pockets in THOREUM.
- 6% of trade will be allocated for marketing.
- 3% of trade will be allocated to buyback and burn ThunderADA.
- 3% of trade will be allocated for liquidity pool or buyback and burn ThunderETH

Roadmap

Phase 1

- ❖ Website Launch
- ❖ +70k Telegram Members
- ❖ Presale Launch & Liquidity locked
- ❖ Launch on PancakeSwap
- ❖ $5M Marketcap
- ❖ Coingecko listing
- ❖ Blockfolio Application
- ❖ Influencer marketing push
- ❖ Poocoin and other big website banner ads
- ❖ Meme contests

Phase 2

- ❖ +10k followers in Twitter
- ❖ 10,000 holders
- ❖ Partnership with influencer Youtube, Twitter, & TikTok
- ❖ ThunderETH Dashboard (earnings tracking)
- ❖ CoinmarketCap listing
- ❖ Massive Marketing
- ❖ $25M Market Cap
- ❖ $50M Market Cap

Phase 3

- ❖ $100M Market Cap
- ❖ ThunderETHSwap
- ❖ ThunderETH Merch Shop
- ❖ Marketing: ThunderETH goes to mainstream
- ❖ Listing on CEX
- ❖ $500M MarketCap
- ❖ Charity donation
- ❖ Research donation
- ❖ More to be announced

Target market and the concept

Target market

- Anyone who's interested in Crypto space with long term investment plans.
- Anyone who's ready to earn a passive income in ETH or THOREUM by holding tokens.
- Anyone who's interested in trading tokens.
- All ETH or THOREUM and fans out there.
- Anyone who's interested in taking part with the future plans of the project.
- Anyone who's interested in making financial transactions with any other party using ThunderETH as the currency.

Core concept

The reward system

10% of each transaction when buying and 15% when selling gets converted to ETH and THOREUM in the power of 80/20, and is split amongst all holders. The rewards are sent to holders that have at least 200,000 ThunderBNB tokens, holders will be eligible to receive tokens every one hour and rewards are proportional to how many tokens each individual holds.

ETH - 80% for now and mid-term:

ETH is now about $3,100 but expected to be $6,200 at the end of this year and potential to $15,500 (x5) in mid-term.

8% of transaction tax in ETH makes sure that ThunderETH holders benefit immediately from the big rewards coming from every transaction, and this can be equal to 40% when ETH price x5 in the near future.

THOREUM - 20% for long-term future:

Thoreum is expected to be the next Safemoon 2.0 with many innovative features that is the first in the crypto world, such as Thor Thunder to buy back & burn with 4,900+ BNB fund & increasing ~100 BNB every day, or Thunder Boost to earn staking rewards while still getting reflection rewards, etc. With a hyper deflationary nature, Thoreum price is expected to be x100 in this year and next.

2% of transaction tax for a token like Thoreum is like a long-term investment and it can be equal to 200% in the long-term future.

Sustainable mechanism

The **fee of 4% marketing when buying and 6% when selling** is what allows ThunderETH to promote the token and use funds to further development of the platform. Tokens will be swapped into BNB and will be sent to a marketing wallet per transaction. This way, ThunderETH will have access to the funds without selling tokens as the traditional way, which will enable them to consume funds without hurting the project.

The liquidity fee of 3%, which is a redistribution mechanism that ensures the trading pool always has sufficient liquidity.

The 2% buy back & burn ThunderADA is to promote the affiliated token of ThunderETH, which will enable ThunderETH to be more popular and be strong in terms of marketing as ThunderADA success will directly affect the ThunderETH and vice versa.

2% auto liquidity pool and buy back & burn ThunderETH

Based on volume & liquidity analysis, 2% of every transaction is transformed either into liquidity for Pancakeswap or into ThunderETH automatic Buyback & burn fund. Liquidity pool is the redistribution mechanism that ensures the trading pool always has sufficient liquidity. Once the system detects a sale, it will automatically buy back a certain value of tokens and burn. It’s automatic and helps create either a higher price floor for stability or pump the price and rarify token.

Potential to grow with score points

1.	Project efficiency	9/10
2.	Project uniqueness	9/10
3	Information quality	9/10
4	Service quality	9/10
5	System quality	8/10
6	Impact on the community	9/10
7	Impact on the business	9/10
8	Preparing for the future	8/10
Total Points		**8.75/10**

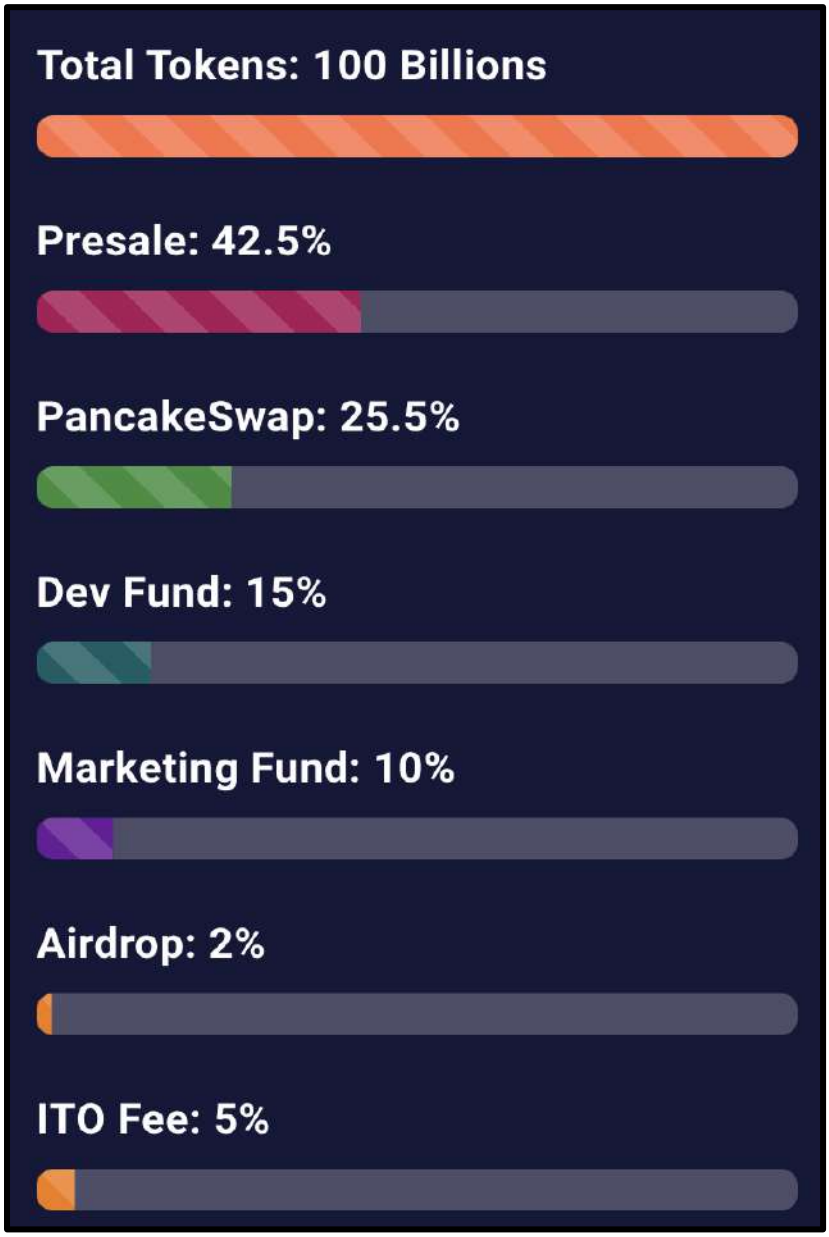
Contract details

Token contract details for 29th August 2021

Contract name	ThunderETH
Contract address	0x73c499099BFc964D3CBcd7425026b074869748cE
Token supply	100,000,000,000
Token ticker	THUNDERETH
Decimals	18
Token holders	1
Transaction count	1
Marketing wallet address	0x6708792796a0bee9493584bb26d207972debb9bf
Team wallet address:	0x27b81cf8a4ce177e7f9b0850984e06954d0090c2
ThunderADA burn address	0x0a7bb2bdda1c0ea02d98a7b048f4bf809f40277b
Contract deployer address	0x8cA23deA974781c94D961950230FB1Ffdb61bB60
Contract's current owner address	0x8ca23dea974781c94d961950230fb1ffdb61bb60

Token distribution

Tokens are distributed as follows:



Contract code function details

No	Category	Item	Result
1	Coding conventions	BRC20 Token standards	pass
		compile errors	pass
		Compiler version security	pass
		visibility specifiers	pass
		Gas consumption	pass
		SafeMath features	pass
		Fallback usage	pass
		tx.origin usage	pass
		deprecated items	pass
		Redundant code	pass
		Overriding variables	pass
2	Function call audit	Authorization of function call	pass
		Low level function (call/delegate call) security	pass
		Returned value security	pass
		Selfdestruct function security	pass
3	Business security	Access control of owners	informational
		Business logics	pass
		Business implementations	pass
4	Integer overflow/underflow		pass
5	Reentrancy		pass
6	Exceptional reachable state		pass
7	Transaction ordering dependence		pass
8	Block properties dependence		pass
9	Pseudo random number generator (PRNG)		pass
10	DoS (Denial of Service)		pass
11	Token vesting implementation		pass
12	Fake deposit		pass
13	Event security		pass

Contract description table

Below table represents the summary of the contracts and methods in the token contract. We scanned the whole contract and listed down all the Interfaces, functions and implementations with its visibility and mutability.

Contract	**Type**	**Bases**		
└	Function Name	Visibility	Mutability	Modifiers
Context	**Implementation**			
└	_msgSender	Internal 🔒		
└	_msgData	Internal 🔒		
Ownable	**Implementation**	**Context**		
└		Public ❗	🛑	NO❗
└	owner	Public ❗		NO❗
└	renounceOwnership	Public ❗	🛑	onlyOwner
└	transferOwnership	Public ❗	🛑	onlyOwner
└	lock	Public ❗	🛑	onlyOwner
IERC20	**Interface**			
└	totalSupply	External ❗		NO❗
└	balanceOf	External ❗		NO❗
└	transfer	External ❗	🛑	NO❗
└	allowance	External ❗		NO❗
└	approve	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
ERC20	**Implementation**	**Context, IERC20**		
└		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 🔒	🛑	
└	_setupDecimals	Internal 🔒	🛑	
└	_beforeTokenTransfer	Internal 🔒	🛑	
IDividendPayingToken	**Interface**			
└	dividendOf	External ❗		NO❗
└	withdrawDividend	External ❗	🛑	NO❗
IDividendPayingTokenOptional	**Interface**			
└	withdrawableDividendOf	External ❗		NO❗
└	withdrawnDividendOf	External ❗		NO❗
└	accumulativeDividendOf	External ❗		NO❗
DividendPayingToken	**Implementation**	**ERC20, IDividendPayingToken, IDividendPayingTokenOptional**		

└		Public ❗	🔘	ERC20
└		External ❗	💸	NO❗
└	distributeDividends	Public ❗	🔘	NO❗
└	withdrawDividend	Public ❗	🔘	NO❗
└	setOnlyCaller	External ❗	🔘	NO❗
└	setDividendTokenAddress	External ❗	🔘	NO❗
└	setMinTokenBeforeSendDividend	External ❗	🔘	NO❗
└	_withdrawDividendOfUser	Internal 🔒	🔘	
└	dividendOf	Public ❗		NO❗
└	withdrawableDividendOf	Public ❗		NO❗
└	withdrawnDividendOf	Public ❗		NO❗
└	accumulativeDividendOf	Public ❗		NO❗
└	_transfer	Internal 🔒	🔘	
└	_mint	Internal 🔒	🔘	
└	_burn	Internal 🔒	🔘	
└	_setBalance	Internal 🔒	🔘	
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❗
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❗

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❗
└	swapExactETHForTokensSupportingF	External ❗	💸	NO❗

	eeOnTransferTokens			
└	swapExactTokensForETHSupportingFeeOnTransferTokens	External ❗	🔘	NO❗
IterableMapping	**Library**			
└	get	Public ❗		NO❗
└	getIndexOfKey	Public ❗		NO❗
└	getKeyAtIndex	Public ❗		NO❗
└	size	Public ❗		NO❗
└	set	Public ❗	🔘	NO❗
└	remove	Public ❗	🔘	NO❗
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 🔒		
SafeMathInt	**Library**			
└	mul	Internal 🔒		

└	div	Internal 🔒		
└	sub	Internal 🔒		
└	add	Internal 🔒		
└	toUint256Safe	Internal 🔒		
SafeMathUint	**Library**			
└	toInt256Safe	Internal 🔒		
ThunderEth	**Implementation**	**ERC20, Ownable**		
└		Public ❗	🛑	ERC20
└		External ❗	💳	NO❗
└	prepareForPartnerOrExchangeListing	External ❗	🛑	onlyOwner
└	setMaxBuyTransaction	External ❗	🛑	onlyOwner
└	setMaxSellTransaction	External ❗	🛑	onlyOwner
└	updateA80DividendToken	External ❗	🛑	onlyOwner
└	updateB20DividendToken	External ❗	🛑	onlyOwner
└	updateMinA80BeforeSendDividend	External ❗	🛑	onlyOwner
└	updateMinB20BeforeSendDividend	External ❗	🛑	onlyOwner
└	getMinA80BeforeSendDividend	External ❗		NO❗
└	getMinB20BeforeSendDividend	External ❗		NO❗
└	setSendA80InTx	External ❗	🛑	onlyOwner
└	setSendB20InTx	External ❗	🛑	onlyOwner
└	setA80DividendPriority	External ❗	🛑	onlyOwner
└	updateBurnThunderCakeAddress	External ❗	🛑	onlyOwner
└	updateTeamWallet	External ❗	🛑	onlyOwner
└	updateMarketingWallet	External ❗	🛑	onlyOwner
└	setToBurnThunderAdaFee	External ❗	🛑	onlyOwner
└	setMaxWalletToken	External ❗	🛑	onlyOwner

L	setSwapTokensAtAmount	External		onlyOwner
L	setSellTransactionMultiplier	External		onlyOwner
L	afterPreSale	External		onlyOwner
L	setTradingIsEnabled	External		onlyOwner
L	setBuyBackMode	External		onlyOwner
L	setMinimumBalanceRequired	Public		onlyOwner
L	setMinimumSellOrderAmount	Public		onlyOwner
L	setBuyBackUpperLimit	External		onlyOwner
L	_updateTotalFee	Internal		
L	setBuyBackAndLiquifyEnabled	External		onlyOwner
L	setA80DividendEnabled	External		onlyOwner
L	setB20DividendEnabled	External		onlyOwner
L	setMarketingEnabled	External		onlyOwner
L	updateA80DividendTracker	External		onlyOwner
L	updateB20DividendTracker	External		onlyOwner
L	updateA80DividendRewardFee	External		onlyOwner
L	updateB20DividendRewardFee	External		onlyOwner
L	updateMarketingFee	External		onlyOwner
L	updateBuyBackAndLiquidityFee	External		onlyOwner
L	updateUniswapV2Router	External		onlyOwner
L	excludeFromFees	Public		onlyOwner
L	excludeFromDividend	Public		onlyOwner
L	setAutomatedMarketMakerPair	Public		onlyOwner
L	_setAutomatedMarketMakerPair	Private		onlyOwner
L	updateGasForProcessing	External		onlyOwner
L	updateMinimumBalanceForDividends	External		onlyOwner
L	updateClaimWait	External		onlyOwner
L	getA80ClaimWait	External		NO

L	getB20ClaimWait	External		NO
L	getTotalA80DividendsDistributed	External		NO
L	getTotalB20DividendsDistributed	External		NO
L	getIsExcludedFromFees	Public		NO
L	withdrawableA80DividendOf	External		NO
L	withdrawableB20DividendOf	External		NO
L	a80DividendTokenBalanceOf	External		NO
L	b20DividendTokenBalanceOf	External		NO
L	getAccountA80DividendsInfo	External		NO
L	getAccountB20DividendsInfo	External		NO
L	getAccountA80DividendsInfoAtIndex	External		NO
L	getAccountB20DividendsInfoAtIndex	External		NO
L	processDividendTracker	External		onlyOwner
L	rand	Internal		
L	claim	External		NO
L	getLastA80DividendProcessedIndex	External		NO
L	getLastB20DividendProcessedIndex	External		NO
L	getNumberOfA80DividendTokenHolders	External		NO
L	getNumberOfB20DividendTokenHolders	External		NO
L	_transfer	Internal		
L	swapAndLiquify	Private		
L	addLiquidity	Private		
L	useBnbToBurnThunderBnb	Private		
L	buyBackThunderBnbAndBurn	Private		
L	buyBackAndBurn	Private		
L	manualBuyBackAndBurn	Public		onlyOwner
L	swapTokensForBNB	Private		

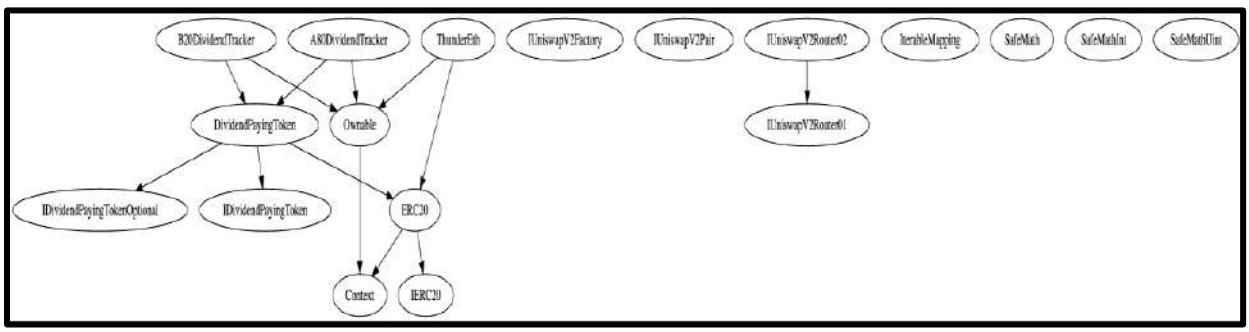
L	swapTokensForDividendToken	Private		
L	swapAndSendA80Dividends	Private		
L	swapAndSendB20Dividends	Private		
L	transferToWallet	Private		
A80DividendTracker	**Implementation**	**DividendPayingToken, Ownable**		
L		Public		DividendPayingToken
L	_transfer	Internal		
L	withdrawDividend	Public		NO
L	setDividendTokenAddress	External		onlyOwner
L	updateMinimumTokenBalanceForDividends	External		onlyOwner
L	excludeFromDividends	External		onlyOwner
L	updateClaimWait	External		onlyOwner
L	getLastProcessedIndex	External		NO
L	getNumberOfTokenHolders	External		NO
L	getAccount	Public		NO
L	getAccountAtIndex	Public		NO
L	canAutoClaim	Private		
L	setBalance	External		onlyOwner
L	process	Public		NO
L	processAccount	Public		onlyOwner
B20DividendTracker	**Implementation**	**DividendPayingToken, Ownable**		
L		Public		DividendPayingToken
L	_transfer	Internal		
L	withdrawDividend	Public		NO
L	setDividendTokenAddress	External		onlyOwner

L	updateMinimumTokenBalanceForDividends	External ❗	🛑	onlyOwner
L	excludeFromDividends	External ❗	🛑	onlyOwner
L	updateClaimWait	External ❗	🛑	onlyOwner
L	getLastProcessedIndex	External ❗		NO❗
L	getNumberOfTokenHolders	External ❗		NO❗
L	getAccount	Public ❗		NO❗
L	getAccountAtIndex	Public ❗		NO❗
L	canAutoClaim	Private 🔐		
L	setBalance	External ❗	🛑	onlyOwner
L	process	Public ❗	🛑	NO❗
L	processAccount	Public ❗	🛑	onlyOwner

Legend

Symbol	Meaning
🛑	**Function can modify state**
💵	**Function is payable**

Inheritance Hierarchy



Security issue checking status

❖ **High severity issues**

- **No high severity issues found.**

❖ **Medium severity issues**

- **No medium severity issues found.**

❖ **Low severity issues**

- **No low severity issues found.**

❖ **Informational**

- **The owner can enable and disable trading at any time.**

```
ftrace | funcSig
function setTradingIsEnabled(bool _enabled) external onlyOwner {
    tradingIsEnabled = _enabled;
}
```

- **The owner can increase the sell fee without any limitation.**

```
ftrace | funcSig
function setSellTransactionMultiplier(uint256 _multiplier)
    external
    onlyOwner
{
    sellFeeIncreaseFactor = _multiplier;
}
```

Owner privileges

- The owner can change max buy and sell transaction amount.

```
ftrace | funcSig
function setMaxBuyTransaction(uint256 _maxTxn⬆) external onlyOwner {
    maxBuyTransactionAmount = _maxTxn⬆ * (10**18);
}

ftrace | funcSig
function setMaxSellTransaction(uint256 _maxTxn⬆) external onlyOwner {
    maxSellTransactionAmount = _maxTxn⬆ * (10**18);
}
```

- The owner can change dividend tokens.

```
ftrace | funcSig
function updateA80DividendToken(address _newContract⬆) external onlyOwner {
    a80DividendToken = _newContract⬆;
    a80DividendTracker.setDividendTokenAddress(_newContract⬆);
}

ftrace | funcSig
function updateB20DividendToken(address _newContract⬆) external onlyOwner {
    b20DividendToken = _newContract⬆;
    b20DividendTracker.setDividendTokenAddress(_newContract⬆);
}
```

- The owner can change the minimum dividend amount that can be sent.

```
function updateMinA80BeforeSendDividend(uint256 _newAmount⬆)
    external
    onlyOwner
{
    a80DividendTracker.setMinTokenBeforeSendDividend(_newAmount⬆);
}

ftrace | funcSig
function updateMinB20BeforeSendDividend(uint256 _newAmount⬆)
    external
    onlyOwner
{
    b20DividendTracker.setMinTokenBeforeSendDividend(_newAmount⬆);
}
```

❖ The owner can enable/disable dividend sending.

```
ftrace | funcSig
function setSendA80InTx(bool _newStatus⬆) external onlyOwner {
    sendA80InTx = _newStatus⬆;
}

ftrace | funcSig
function setSendB20InTx(bool _newStatus⬆) external onlyOwner {
    sendB20InTx = _newStatus⬆;
}
```

❖ The owner can change max BNB sending percentage as dividends.

```
ftrace | funcSig
function setA80DividendPriority(uint256 _newAmount⬆) external onlyOwner {
    require(_newAmount⬆ >= 0 && _newAmount⬆ <= 100, "Error amount");
    a80DividendPriority = _newAmount⬆;
}
```

❖ The owner can change the ThunderADA address.

```
ftrace | funcSig
function updateBurnThunderCakeAddress(address _newAddress⬆) external onlyOwner {
    toBurnThunderAdaAddress = _newAddress⬆;
}
```

❖ The owner can change the Team wallet address.

```
ftrace | funcSig
function updateTeamWallet(address _newWallet⬆) external onlyOwner {
    excludeFromFees(_newWallet⬆, true);
    teamWallet = _newWallet⬆;
}
```

❖ The owner can change the marketing wallet address.

```
ftrace | funcSig
function updateMarketingWallet(address _newWallet⬆) external onlyOwner {
    excludeFromFees(_newWallet⬆, true);
    marketingWallet = _newWallet⬆;
}
```

❖ The owner can change ThunderADA burn fee maximum up to 25%.

```
ftrace | funcSig
function setToBurnThunderAdaFee(uint256 newFee⬆) external onlyOwner {
    require(newFee⬆ <= MAX_FEE_RATE, "wrong");
    toBurnThunderAdaFee = newFee⬆;
    _updateTotalFee();
}
```

❖ The owner can change max wallet token amount (the maximum token amount a wallet can hold)

```
ftrace | funcSig
function setMaxWalletToken(uint256 _maxToken⬆) external onlyOwner {
    maxWalletToken = _maxToken⬆ * (10**18);
}
```

❖ The owner can change the swap token amount.

```
ftrace | funcSig
function setSwapTokensAtAmount(uint256 _swapAmount⬆) external onlyOwner {
    swapTokensAtAmount = _swapAmount⬆ * (10**18);
}
```

❖ The owner can enable and disable trading.

```
ftrace | funcSig
function setTradingIsEnabled(bool _enabled⬆) external onlyOwner {
    tradingIsEnabled = _enabled⬆;
}
```

❖ The owner can change the minimum required BNB balance for buy back.

❖ The owner can change the minimum sell token amount to buy back.

❖ The owner can change the buy back upper limit.

❖ The owner can change all fees maximum up to 25%.

```
ftrace | funcSig
function updateA80DividendRewardFee(uint8 newFee⬆) external onlyOwner {
    require(newFee⬆ <= MAX_FEE_RATE, "wrong");
    a80DividendRewardsFee = newFee⬆;
    _updateTotalFee();
}

ftrace | funcSig
function updateB20DividendRewardFee(uint8 newFee⬆) external onlyOwner {
    require(newFee⬆ <= MAX_FEE_RATE, "wrong");
    b20DividendRewardsFee = newFee⬆;
    _updateTotalFee();
}

ftrace | funcSig
function updateMarketingFee(uint8 newFee⬆) external onlyOwner {
    require(newFee⬆ <= MAX_FEE_RATE, "wrong");
    marketingFee = newFee⬆;
    _updateTotalFee();
}

ftrace | funcSig
function updateBuyBackAndLiquidityFee(uint8 newFee⬆) external onlyOwner {
    require(newFee⬆ <= MAX_FEE_RATE, "wrong");
    buyBackAndLiquidityFee = newFee⬆;
    _updateTotalFee();
}
```

❖ The owner can change v2 router address.

```
ftrace | funcSig
function updateUniswapV2Router(address newAddress⬆) external onlyOwner {
    uniswapV2Router = IUniswapV2Router02(newAddress⬆);
}
```

❖ The owner can exclude wallet from fees and dividends.

```
ftrace | funcSig
function excludeFromFees(address account⬆, bool excluded⬆) public onlyOwner {
    require(isExcludedFromFees[account⬆] != excluded⬆, "Already excluded");
    isExcludedFromFees[account⬆] = excluded⬆;

    emit ExcludeFromFees(account⬆, excluded⬆);
}

ftrace | funcSig
function excludeFromDividend(address account⬆) public onlyOwner {
    a80DividendTracker.excludeFromDividends(address(account⬆));
    b20DividendTracker.excludeFromDividends(address(account⬆));
}
```

❖ The owner can change the claim wait.

```
ftrace | funcSig
function updateClaimWait(uint256 claimWait⬆) external onlyOwner {
    a80DividendTracker.updateClaimWait(claimWait⬆);
    b20DividendTracker.updateClaimWait(claimWait⬆);
}
```

❖ The owner can manually process dividends.

```
ftrace | funcSig
function processDividendTracker(uint256 gas⬆) external onlyOwner {
    (
        uint256 aIterations,
        uint256 aClaims,
        uint256 aLastProcessedIndex
    ) = a80DividendTracker.process(gas⬆);
    emit ProcessedA80DividendTracker(
        aIterations,
        aClaims,
        aLastProcessedIndex,
        false,
        gas⬆,
        tx.origin
    );

    (
        uint256 bIterations,
        uint256 bClaims,
        uint256 bLastProcessedIndex
    ) = b20DividendTracker.process(gas⬆);
    emit ProcessedB20DividendTracker(
        bIterations,
        bClaims,
        bLastProcessedIndex,
        false,
        gas⬆,
        tx.origin
    );
}
```

❖ The owner can manually trigger the buy back function.

```
ftrace | funcSig
function manualBuyBackAndBurn(uint256 _amount⬆) public onlyOwner {
    uint256 balance = address(this).balance;
    require(buyBackAndLiquifyEnabled, "not enabled");
    require(
        balance >= minimumBalanceRequired.add(_amount⬆),
        "amount is too big"
    );

    if (!swapping) {
        buyBackAndBurn(_amount⬆);
    }
}
```

Audit conclusion

While conducting the audit of the ThunderETH token smart contract, it was observed that there is nothing alarming with the code and it only contains informational concerns.