

SKELETON ECOSYSTEM

SMART CONTRACT AUDIT



Elon Diablo

\$EDT

BEP20

0x37A7458443178d1C8C2887C9122261b715B6



Table of Contents

Table of Contents	1
Disclaimer	2
Overview	3
Creation/Audit Date	3
Verified Socials	3
Contract Functions Analysis	4
Contract Safety and Weakness	8
Detected Vulnerability Description	12
Contract Flow Graph	15
Contract Interaction Graph	16
Inheritance Graph	17
Contract Descriptions	18
Audit Scope	30

Global Disclaimer

This document serves as a disclaimer for the crypto smart contract audit conducted by Skeleton Ecosystem. The purpose of the audit was to review the codebase of the smart contracts for potential vulnerabilities and issues. It is important to note the following:

Limited Scope: The audit is based on the code and information available up to the audit completion date. It does not cover external factors, system interactions, or changes made after the audit. The audit itself can not guarantee 100% safety and can not detect common scam methods like farming and developer sell-out.

No Guarantee of Security: While we have taken reasonable steps to identify vulnerabilities, it is impossible to guarantee the complete absence of security risks or issues. The audit report provides an assessment of the contract's security as of the audit date.

Continued Development: Smart contracts and blockchain technology are evolving fields. Updates, forks, or changes to the contract post-audit may introduce new risks that were not present during the audit.

Third-party Code: If the smart contract relies on third-party libraries or code, those components were not thoroughly audited unless explicitly stated. Security of these dependencies is the responsibility of their respective developers.

Non-Exhaustive Testing: The audit involved automated analysis, manual review, and testing under controlled conditions. It is possible that certain vulnerabilities or issues may not have been identified.

Risk Evaluation: The audit report includes a risk assessment for identified vulnerabilities. It is recommended that the development team carefully reviews and addresses these risks to mitigate potential exploits.

Not Financial Advice: This audit report is not intended as financial or investment advice. Decisions regarding the use, deployment, or investment in the smart contract should be made based on a comprehensive assessment of the associated risks.

By accessing and using this audit report, you acknowledge and agree to the limitations outlined above. Skeleton Ecosystem and its auditors shall not be held liable for any direct or indirect damages resulting from the use of the audit report or the smart contract itself.

Please consult with legal, technical, and financial professionals before making any decisions related to the smart contract.

Overview

Contract Name	ElonDiablo
Ticker/Symbol	\$EDT
Blockchain	Binance Smart Chain BEP20
Contract Address	0x37A7458443178d1C8C2887C9122261b715B6f08B
Creator Address	0x8096955003F92B7882F586Eb7aaa224adbf7E6eF
Current Owner Address	0x00
Contract Explorer	https://bscscan.com/address/0x37a7458443178d1c8c2887c9122261b715b6f08b#code
Compiler Version	v0.8.19+commit.7dd6d404
License	MIT
Optimisation	Yes with 200 Runs
Total Supply	661,624.990287 \$EDT
Decimals	18

Creation/Audit

Contract Deployed	23.01.2024
Audit Created	13.02.2024
Audit Update	V 1.0

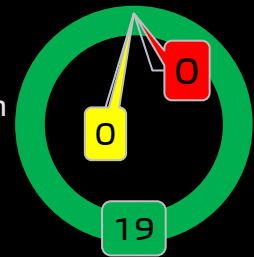
Verified Socials








Website	https://elondiablo.com/
Telegram	https://t.m/ElonDiablo
Twitter (X)	https://twitter.com/ElonDiablo











Contract Function Analysis

 Pass
  Attention Item
  Risky Item

■ Pass
 ■ Attention
 ■ Risk



Contract Verified		The contract source code is uploaded to blockchain explorer and is open source, so everybody can read it.
Contract Ownership		0x00 Sometimes referred to as the "zero address" or "dead address" and is not owned by anyone.
Buy Tax	3 %	Shows the taxes for purchase transactions. Above 10% may be considered a high tax rate. More than 50% tax rate means may not be tradable. Fee can be set!
Sell Tax	3 %	Shows the taxes for sell transactions. Above 10% may be considered a high tax rate. More than 50% tax rate means may not be tradable. Fee can be set!
Honeypot Analyse		Holder is able to buy and sell. If honeypot: The contract blocks sell transfer from holder wallet. Multiple events may cause honeypot. Trading disabled, extremely high tax
Liquidity Status		Liquidity status on 12.02.2024 99% Locked for 100170 Days on Gempad Locker
Trading Disable Functions		No Trading suspendable function found. If a suspendable code is included, the token maybe neither be bought or sold (honeypot risk). If contract is renounced this function can't be used
Set Fees function		Fee Setting function found. Contract renounced, function can not be triggered by owner. The contract owner may contain the authority to modify the transaction tax. If the transaction tax is increased to more than 49%, the tokens may not be able to be traded (honeypot risk).
Proxy Contract		Not a Proxy contract with authorisations.
Mint Function		No Mint Function detected Mint function is transparent or non-existent. Hidden mint functions may increase the amount of tokens in circulation and effect the price of the token. Owner can mint new tokens and sell.

Balance Modifier Function		<p>No Balance Modifier function found.</p> <p>If there is a function for this, the contract owner can have the authority to modify the balance of tokens at other addresses. For example revoke the bought tokens from the holders wallet. Common form of scam: You buy the token, but it's disappearing from your wallet.</p>
Blacklist Function		<p>No Blacklist Setting function found.</p> <p>If there is a blacklist, some addresses may not be able to trade normally. Example: you buy the token and right after your Wallet getting blacklisted. Like so you will be unable to sell. Honeypot Risk.</p>
Whitelist Function		<p>Whitelist Setting function found. Contract renounced, function can not be triggered by owner.</p> <p>If there is a function for this Developer can set zero fee or no max wallet size for addresses (for example team wallets can trade without fee. Can cause farming)</p>
Hidden Owner Analysis		<p>No Hidden or multi owner with authorisation</p> <p>For contract with a hidden owner, developer can still manipulate the contract even if the ownership has been abandoned.</p>
Retrieve Ownership Function		<p>No Functions found which can retrieve ownership of the contract.</p> <p>If this function exists, it is possible for the project owner to regain ownership even after relinquishing it. Also known as fake renounce.</p>
Self Destruct Function		<p>No Self Destruct function found.</p> <p>If this function exists and is triggered, the contract will be destroyed, all functions will be unavailable, and all related assets will be erased.</p>
Specific Tax Changing Function		<p>No Specific Tax Changing Functions found.</p> <p>If it exists, the contract owner may set a very outrageous tax rate for assigned address to block it from trading. Can assign all wallets at once!</p>
Trading Cooldown Function	 max 300s	<p>Trading Cooldown Function found. Contract renounced, function can not be triggered by owner.</p> <p>If there is a trading cooldown function, the user will not be able to sell the token within a certain time or block after buying. Like a temporary honeypot.</p>
Max Transaction and Holding Modify Function	 min 0.05%	<p>Max Transaction and Holding Modify function found. Contract renounced, function can not be triggered by owner.</p> <p>If there is a function for this, the maximum trading amount or maximum position can be modified. Can cause honeypot</p>
Transaction Limiting Function		<p>No Transaction Limiter Function Found.</p> <p>The number of overall token transactions may be limited (honeypot risk)</p>

Details of Risk - Attention Items

Removing Risk of contract function based on renounced ownership

Transaction Receipt Event Logs

106

Address

0x37a7458443178d1c8c2887c9122261b715b6f08b

🔍

Name

OwnershipTransferred (index_topic_1 address previousOwner, index_topic_2 address newOwner) [View Source](#)

Topics

0

0x8Be0079c531659141344cd1fd0a4f28419497f9722a3daafe3b4186f6b6457e0

1: previousOwner

Dec

→ 0x8096955003F92B7882F586Eb7aaa224adbf7E6eF

2: newOwner

Dec

→ 0x00

Data

0x

Following detected contract functions serve as informational purposes about the contract. The owner has no more authorisation to trigger the following functions.

⚠️ Set Fee

Contract renounced, function can not be triggered by owner.

The contract owner may contain the authority to modify the transaction tax. If the transaction tax is increased to more than 49%, the tokens may not be able to be traded (honeypot risk).

```

ftrace | funcSig
1289 function updateFees(
1290     uint256 deadBuy!,
1291     uint256 deadSell!,
1292     uint256 marketingBuy!,
1293     uint256 marketingSell!,
1294     uint256 liquidityBuy!,
1295     uint256 liquiditySell!,
1296     uint256 RewardsBuy!,
1297     uint256 RewardsSell!,
1298     uint256 devBuy!,
1299     uint256 devSell!
1300 ) public onlyOwner {
1301     buyDeadFees = deadBuy!;
1302     buyMarketingFees = marketingBuy!;
1303     buyLiquidityFee = liquidityBuy!;
1304     buyRewardsFee = RewardsBuy!;
1305     sellDeadFees = deadSell!;
1306     sellMarketingFees = marketingSell!;
1307     sellLiquidityFee = liquiditySell!;
1308     sellRewardsFee = RewardsSell!;
1309     buyDevFee = devBuy!;
1310     sellDevFee = devSell!;
1311 }
  
```

⚠ Whitelist

Contract renounced, function can not be triggered by owner.

If there is a function for this Developer can set zero fee or no max wallet size for addresses (for example team wallets can trade without fee. Can cause farming)

```

1162
1163 // exclude a wallet from fees
      ftrace | funcSig
1164 function setExcludeFees(address account!, bool excluded!) public onlyOwner {
1165     | isExcludedFromFees[account!] = excluded!;
1166     | emit ExcludeFromFees(account!, excluded!);
1167 }
1168
  
```

⚠ Max Transaction and Holding Modify Function (Min. 0.05%)

Contract renounced, function can not be triggered by owner.

If there is a function for this, the maximum trading amount or maximum position can be modified. Can cause honeypot

```

1205 // set max wallet, can not be lower than 0.05% of supply
      ftrace | funcSig
1206 function setmaxWallet(uint256 value!) external onlyOwner {
1207     | value! = value! * (10**18);
1208     | require(value! >= _totalSupply / 2000, "max wallet cannot be set to less than 0.05%");
1209     | maxWallet = value!;
1210 }
1211
  
```

⚠ Trading cooldown (max 300 sec.)

Contract renounced, function can not be triggered by owner.

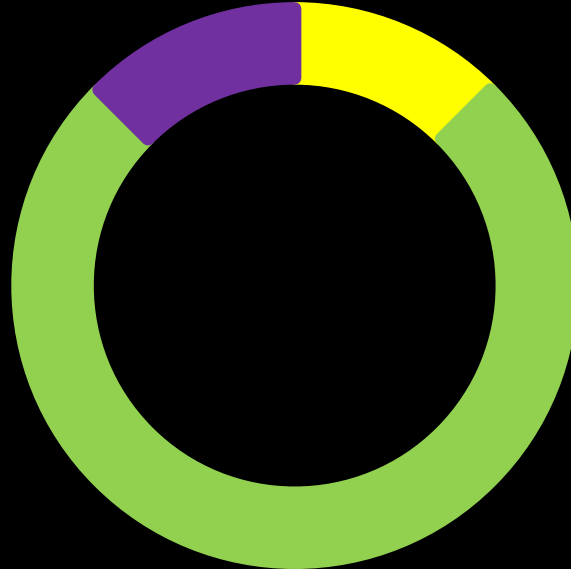
If there is a trading cooldown function, the user will not be able to sell the token within a certain time or block after buying. Like a temporary honeypot.

```

1198
1199 // set cooldown timer, can only be between 0 and 300 seconds (5 mins max)
      ftrace | funcSig
1200 function setcooldowntimer(uint256 value!) external onlyOwner {
1201     | require(value! <= 300, "cooldown timer cannot exceed 5 minutes");
1202     | cooldowntimer = value!;
1203 }
1204
  
```


Contract Security

Total Findings: 8



■ **High Severity Issues:** High possibility to cause problems, need to be resolved.

■ **Medium Severity Issue:** Will likely cause problems, recommended to resolve.

■ **Low Severity Issues:** Won't cause problems, but for improvement purposes could be adjusted.

■ **Informational Severity Issues:** Not harmful in any way, information for the developer team.

Contract Security

List of Found Issues

High severity Issues: (0)

Medium severity issues: (1)

- Authorization through tx.origin

Low severity issues: (6)

- Missing Events
- Long number literals
- Low level calls
- Outdated Compiler Version
- Floating Pragma
- Approve of front running attack (Sandwich bots)

Informational severity issues: (1)

- Public Functions Should be Declared External

Contract Weakness Classification

THE SMART CONTRACT WEAKNESS CLASSIFICATION REGISTRY (SWC REGISTRY) IS AN IMPLEMENTATION OF THE WEAKNESS CLASSIFICATION SCHEME PROPOSED IN EIP-1470. IT IS LOOSELY ALIGNED TO THE TERMINOLOGIES AND STRUCTURE USED IN THE COMMON WEAKNESS ENUMERATION (CWE) WHILE OVERLAYING A WIDE RANGE OF WEAKNESS VARIANTS THAT ARE

ID	Description	AI	Manual	Result
SWC-100	Function Default Visibility	Passed	Passed	Passed
SWC-101	Integer Overflow and Underflow	Passed	Passed	Passed
SWC-102	Outdated Compiler Version	low	Passed	Passed
SWC-103	Floating Pragma	low	Passed	Passed
SWC-104	Unchecked Call Return Value	Passed	Passed	Passed
SWC-105	Unprotected Ether Withdrawal	Passed	Passed	Passed
SWC-106	Unprotected SELFDESTRUCT Instruction	Passed	Passed	Passed
SWC-107	Reentrancy	Passed	Passed	Passed
SWC-108	State Variable Default Visibility	Passed	Passed	Passed
SWC-109	Uninitialized Storage Pointer	Passed	Passed	Passed
SWC-110	Assert Violation	Passed	Passed	Passed
SWC-111	Use of Deprecated Solidity Functions	Passed	Passed	Passed
SWC-112	Delegatecall to Untrusted Callee	Passed	Passed	Passed
SWC-113	DoS with Failed Call	Passed	Passed	Passed
SWC-114	Transaction Order Dependence	Passed	Passed	Passed
SWC-115	Authorization through tx.origin	High	Medium	Medium
SWC-116	Block values as a proxy for time	Passed	Passed	Passed
SWC-117	Signature Malleability	Passed	Passed	Passed
SWC-118	Incorrect Constructor Name	Passed	Passed	Passed
SWC-119	Shadowing State Variables	Passed	Passed	Passed
SWC-120	Weak Sources of Randomness from Chain Attributes	Passed	Passed	Passed

SWC-121	Missing Protection against Signature Replay Attacks	Passed	Passed	Passed
SWC-122	Lack of Proper Signature Verification	Passed	Passed	Passed
SWC-123	Requirement Violation	Passed	Passed	Passed
SWC-124	Write to Arbitrary Storage Location	Passed	Passed	Passed
SWC-125	Incorrect Inheritance Order	Passed	Passed	Passed
SWC-126	Insufficient Gas Griefing	Passed	Passed	Passed
SWC-127	Arbitrary Jump with Function Type Variable	Passed	Passed	Passed
SWC-128	DoS With Block Gas Limit	Passed	Passed	Passed
SWC-129	Typographical Error	low	Passed	Passed
SWC-130	Right-To-Left-Override control character (U+202E)	Passed	Passed	Passed
SWC-131	Presence of unused variables	Passed	Passed	Passed
SWC-132	Unexpected Ether balance	Passed	Passed	Passed
SWC-133	Hash Collisions With Multiple Variable Length Arguments	Passed	Passed	Passed
SWC-134	Message call with hardcoded gas amount	Passed	Passed	Passed
SWC-135	Code With No Effects	Passed	Passed	Passed
SWC-136	Unencrypted Private Data On-Chain	Passed	Passed	Passed

Detected High and Medium Severity Vulnerability Description.

⚠️ Authorization through tx.origin (6 Items)

Item: 1	Location:	Line 1412	Severity:	Medium
Item: 1	Location:	Line 1506	Severity:	Medium
Item: 1	Location:	Line 1508	Severity:	Medium
Item: 1	Location:	Line 1522	Severity:	Medium
Item: 1	Location:	Line 1523	Severity:	Medium
Item: 1	Location:	Line 1603	Severity:	Medium

Function	In Solidity, tx.origin is a global variable that returns the address of the account that sent the transaction. Using the variable for authorization could make a contract vulnerable. For example, if an authorized account calls a malicious contract which triggers it to call the vulnerable contract that passes an authorization check since tx.origin returns the original sender of the transaction which in this case is the authorized account.
Remediation	tx.origin should not be used for authorization in smart contracts. It does have some legitimate use cases, for example, To prevent external contracts from calling the current contract, you can implement a require of the form require(tx.origin == msg.sender). This prevents intermediate contracts from calling the current contract, thus limiting the contract to regular codeless addresses.

⚠️ Approve of front running attack (2 Items)

Item: 1	Location:	Line 277-285	Severity: ■ Low
---------	-----------	--------------	--

Function	<p>The approve() method overrides current allowance regardless of whether the spender already used it or not, so there is no way to increase or decrease allowance by a certain value atomically unless the token owner is a smart contract, not an account. This can be abused by a token receiver when they try to withdraw certain tokens from the sender's account.</p> <p>Meanwhile, if the sender decides to change the amount and sends another approve transaction, the receiver can notice this transaction before it's mined and can extract tokens from both the transactions, therefore, ending up with tokens from both the transactions. This is a front-running attack affecting the ERC20 Approve function.</p> <p>The function approve can be front-run by abusing the <code>_approve</code> function.</p>
Remedation	<ol style="list-style-type: none"> 1. Introduce mechanisms that limit the maximum acceptable gas price for transactions. This can help prevent front-runners from drastically increasing the gas fees to prioritize their transactions. 2. Use transaction taxes to prevent against front-run attack

```

ftrace | funcSig
277 function approve(address spender!, uint256 amount!)
278     public
279     virtual
280     override
281     returns (bool)
282 {
283     _approve(_msgSender(), spender!, amount!);
284     return true;
285 }
286
  
```

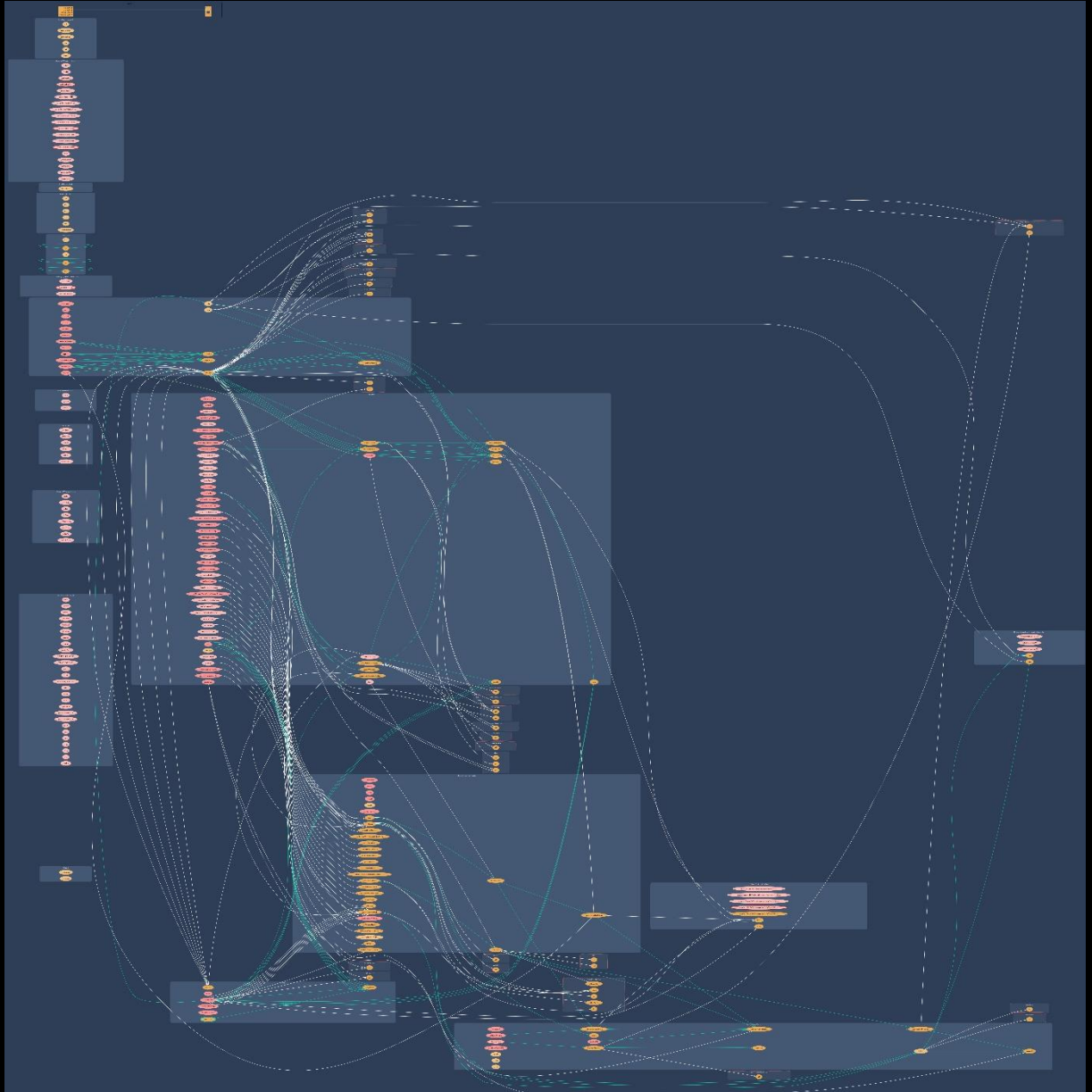
Item: 2	Location:	Line 287-302	Severity:	■ Low
---------	-----------	--------------	-----------	--

Function	<p>The <code>_transferfrom()</code> method overrides current allowance regardless of whether the spender already used it or not, so there is no way to increase or decrease allowance by a certain value atomically unless the token owner is a smart contract, not an account.</p> <p>This can be abused by a token receiver when they try to withdraw certain tokens from the sender's account.</p> <p>Meanwhile, if the sender decides to change the amount and sends another approve transaction, the receiver can notice this transaction before it's mined and can extract tokens from both the transactions, therefore, ending up with tokens from both the transactions. This is a front-running attack affecting the ERC20 Approve function.</p> <p>The function approve can be front-run by abusing the <code>_approve</code> function.</p>
Remedation	<ol style="list-style-type: none"> 1. Introduce mechanisms that limit the maximum acceptable gas price for transactions. This can help prevent front-runners from drastically increasing the gas fees to prioritize their transactions. 2. Use transaction taxes to prevent against front-run attack

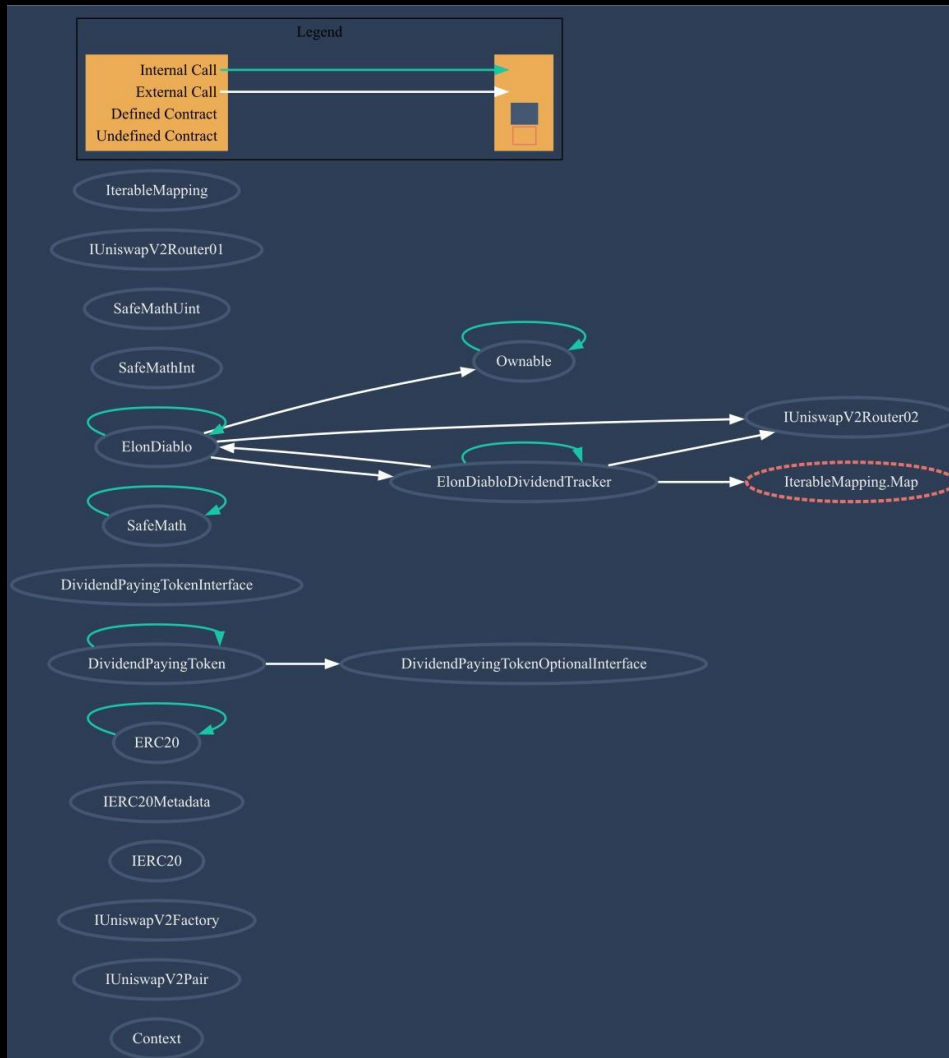
```

ftrace | funcSig
287 function transferFrom(
288     address sender!,
289     address recipient!,
290     uint256 amount!
291 ) public virtual override returns (bool) {
292     _transfer(sender!, recipient!, amount!);
293     _approve(
294         sender!,
295         _msgSender(),
296         allowances[sender!][_msgSender()].sub(
297             amount!,
298             "ERC20: transfer amount exceeds allowance"
299         )
300     );
301     return true;
302 }
303
  
```

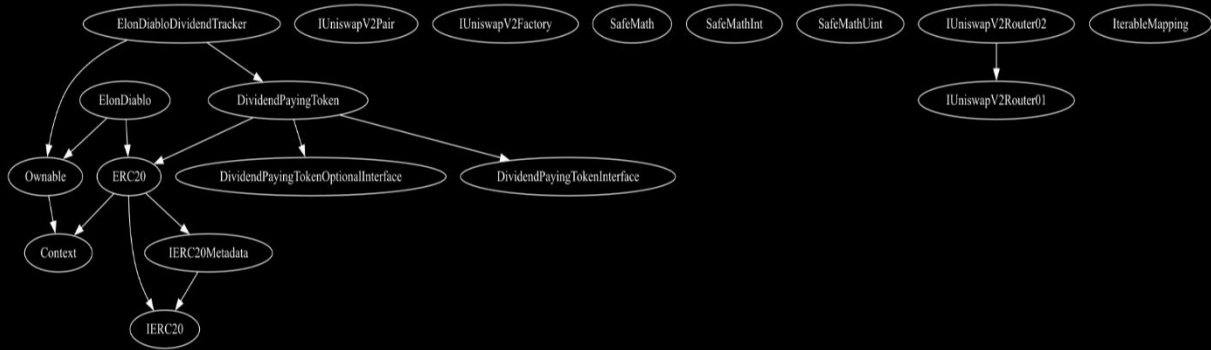
Contract Flow Graph







































Contract Interaction Graph



Inheritance Graph


































Contract Functions



Contract	Type	Bases		
L	Function Name	Visibility	Mutability	Modifiers
Context	Implementation			
L	_msgSender	Internal 		
L	_msgData	Internal 		
IUniswapV2Pair	Interface			
L	name	External 		NO 
L	symbol	External 		NO 
L	decimals	External 		NO 
L	totalSupply	External 		NO 
L	balanceOf	External 		NO 
L	allowance	External 		NO 
L	approve	External 		NO 
L	transfer	External 		NO 
L	transferFrom	External 		NO 
L	DOMAIN_SEPARATOR	External 		NO 
L	PERMIT_TYPEHASH	External 		NO 
L	nonces	External 		NO 
L	permit	External 		NO 
L	MINIMUM_LIQUIDITY	External 		NO 
L	factory	External 		NO 




































L	token0	External ¶		NO ¶
L	token1	External ¶		NO ¶
L	getReserves	External ¶		NO ¶
L	price0CumulativeLast	External ¶		NO ¶
L	price1CumulativeLast	External ¶		NO ¶
L	kLast	External ¶		NO ¶
L	mint	External ¶	⦿	NO ¶
L	burn	External ¶	⦿	NO ¶
L	swap	External ¶	⦿	NO ¶
L	skim	External ¶	⦿	NO ¶
L	sync	External ¶	⦿	NO ¶
L	initialize	External ¶	⦿	NO ¶
IUniswapV2Factory	Interface			
L	feeTo	External ¶		NO ¶
L	feeToSetter	External ¶		NO ¶
L	getPair	External ¶		NO ¶
L	allPairs	External ¶		NO ¶
L	allPairsLength	External ¶		NO ¶
L	createPair	External ¶	⦿	NO ¶
L	setFeeTo	External ¶	⦿	NO ¶
L	setFeeToSetter	External ¶	⦿	NO ¶
IERC20	Interface			
L	totalSupply	External ¶		NO ¶









































L	balanceOf	External ¶		NO ¶
L	transfer	External ¶	⦿	NO ¶
L	allowance	External ¶		NO ¶
L	approve	External ¶	⦿	NO ¶
L	transferFrom	External ¶	⦿	NO ¶
IERC20Metadata	Interface	IERC20		
L	name	External ¶		NO ¶
L	symbol	External ¶		NO ¶
L	decimals	External ¶		NO ¶
ERC20	Implementation	Context, IERC20, IERC20Metadata		
L		Public ¶	⦿	NO ¶
L	name	Public ¶		NO ¶
L	symbol	Public ¶		NO ¶
L	decimals	Public ¶		NO ¶
L	totalSupply	Public ¶		NO ¶
L	balanceOf	Public ¶		NO ¶
L	transfer	Public ¶	⦿	NO ¶
L	allowance	Public ¶		NO ¶
L	approve	Public ¶	⦿	NO ¶
L	transferFrom	Public ¶	⦿	NO ¶
L	increaseAllowance	Public ¶	⦿	NO ¶
L	decreaseAllowance	Public ¶	⦿	NO ¶






































L	_transfer	Internal 🔒	🔒	
L	_mint	Internal 🔒	🔒	
L	_burn	Internal 🔒	🔒	
L	_approve	Internal 🔒	🔒	
L	_beforeTokenTransfer	Internal 🔒	🔒	
DividendPayingTokenOptionalInterface				
L	withdrawableDividendOf	External ⚠️		NO ⚠️
L	withdrawnDividendOf	External ⚠️		NO ⚠️
L	accumulativeDividendOf	External ⚠️		NO ⚠️
DividendPayingTokenInterface				
L	dividendOf	External ⚠️		NO ⚠️
L	distributeDividends	External ⚠️	🔒	NO ⚠️
L	withdrawDividend	External ⚠️	🔒	NO ⚠️
SafeMath				
L	add	Internal 🔒		
L	sub	Internal 🔒		
L	sub	Internal 🔒		
L	mul	Internal 🔒		
L	div	Internal 🔒		
L	div	Internal 🔒		





















L	mod	Internal 		
L	mod	Internal 		
Ownable	Implementation	Context		
L		Public 		NO 
L	owner	Public 		NO 
L	renounceOwnership	Public 		onlyOwner
L	transferOwnership	Public 		onlyOwner
SafeMathInt	Library			
L	mul	Internal 		
L	div	Internal 		
L	sub	Internal 		
L	add	Internal 		
L	abs	Internal 		
L	toUint256Safe	Internal 		
SafeMathUint	Library			
L	toInt256Safe	Internal 		
IUniswapV2Router01	Interface			
L	factory	External 		NO 
L	WETH	External 		NO 
L	addLiquidity	External 		NO 
L	addLiquidityETH	External 		NO 
L	removeLiquidity	External 		NO 


L	removeLiquidity ETH	External ¶		NO ¶
L	removeLiquidity WithPermit	External ¶		NO ¶
L	removeLiquidity ETHWithPermit	External ¶		NO ¶
L	swapExactTokes nsForTokens	External ¶		NO ¶
L	swapTokensFor ExactTokens	External ¶		NO ¶
L	swapExactETHF orTokens	External ¶		NO ¶
L	swapTokensFor ExactETH	External ¶		NO ¶
L	swapExactTokes nsForETH	External ¶		NO ¶
L	swapETHForExa ctTokens	External ¶		NO ¶
L	quote	External ¶		NO ¶
L	getAmountOut	External ¶		NO ¶
L	getAmountIn	External ¶		NO ¶
L	getAmountsOut	External ¶		NO ¶
L	getAmountsIn	External ¶		NO ¶
IUniswapV2Router02	Interface	IUniswapV2Router01		
L	removeLiquidity ETHSupportingFeeOnTransferTokens	External ¶		NO ¶
L	removeLiquidity ETHWithPermit SupportingFeeOnTransferTokens	External ¶		NO ¶
L	swapExactTokes nsForTokensSup	External ¶		NO ¶



























	portingFeeOnTransferTokens			
L	swapExactETHForTokensSupportingFeeOnTransferTokens	External 		NO 
L	swapExactTokensForETHSupportingFeeOnTransferTokens	External 		NO 
DividendPaying Token	Implementation	ERC20, DividendPayingTokenInterface, DividendPayingTokenOptionalInterface		
L		Public 		ERC20
L		External 		NO 
L	distributeDividends	Public 		NO 
L	withdrawDividend	Public 		NO 
L	_withdrawDividendOfUser	Internal 		
L	dividendOf	Public 		NO 
L	withdrawableDividendOf	Public 		NO 
L	withdrawnDividendOf	Public 		NO 
L	accumulativeDividendOf	Public 		NO 
L	_transfer	Internal 		
L	_mint	Internal 		
L	_burn	Internal 		
L	_setBalance	Internal 		

ElonDiablo	Implementation	ERC20, Ownable		
L		Public 		ERC20
L	decimals	Public 		NO 
L		External 		NO 
L	updateStakingAmounts	Public 		onlyOwner
L	enableTrading	External 		onlyOwner
L	setPresaleWallet	External 		onlyOwner
L	setExcludeFees	Public 		onlyOwner
L	setExcludeDividends	Public 		onlyOwner
L	setIncludeDividends	Public 		onlyOwner
L	setCanTransferBefore	External 		onlyOwner
L	setLimitsInEffect	External 		onlyOwner
L	setGasPriceLimit	External 		onlyOwner
L	setcooldowntimer	External 		onlyOwner
L	setMaxWallet	External 		onlyOwner
L	enableStaking	Public 		onlyOwner
L	stake	Public 		NO 
L	setSwapTriggerAmount	Public 		onlyOwner
L	enableSwapAndLiquify	Public 		onlyOwner
L	setAutomatedMarketMakerPair	Public 		onlyOwner

L	setAllowCustomTokens	Public 		onlyOwner
L	setAllowAutoReinvest	Public 		onlyOwner
L	_setAutomatedMarketMakerPair	Private 		
L	updateGasForProcessing	Public 		onlyOwner
L	transferAdmin	Public 		onlyOwner
L	updateTransferFee	Public 		onlyOwner
L	updateFees	Public 		onlyOwner
L	getStakingInfo	External 		NO 
L	getTotalDividendsDistributed	External 		NO 
L	isExcludedFromFees	Public 		NO 
L	withdrawableDividendOf	Public 		NO 
L	dividendTokenBalanceOf	Public 		NO 
L	getAccountDividendsInfo	External 		NO 
L	getAccountDividendsInfoAtIndex	External 		NO 
L	processDividendTracker	External 		NO 
L	claim	External 		NO 
L	getLastProcessedIndex	External 		NO 
L	getNumberOfDividendTokenHolders	External 		NO 

L	setAutoClaim	External !		NO!
L	setReinvest	External !		NO!
L	setDividendsPaused	External !		onlyOwner
L	isExcludedFromAutoClaim	External !		NO!
L	isReinvest	External !		NO!
L	_transfer	Internal 		
L	getStakingBalance	Private 		
L	swapAndLiquify	Private 		
L	swapTokensForEth	Private 		
L	updatePayoutToken	Public !		onlyOwner
L	getPayoutToken	Public !		NO!
L	setMinimumTokenBalanceForAutoDividends	Public !		onlyOwner
L	setMinimumTokenBalanceForDividends	Public !		onlyOwner
L	addLiquidity	Private 		
L	forceSwapAndSendDividends	Public !		onlyOwner
L	swapAndSendDividends	Private 		
L	multiSend	Public !		onlyOwner
L	airdropToWallets	External !		onlyOwner
ElonDiabloDividendTracker	Implementation	DividendPaying Token, Ownable		

L		Public ⓘ		DividendPayingToken
L	decimals	Public ⓘ		NO ⓘ
L	name	Public ⓘ		NO ⓘ
L	symbol	Public ⓘ		NO ⓘ
L	_transfer	Internal 🔒		
L	withdrawDividend	Public ⓘ		NO ⓘ
L	isExcludedFromAutoClaim	External ⓘ		onlyOwner
L	isReinvest	External ⓘ		onlyOwner
L	setAllowCustomTokens	External ⓘ		onlyOwner
L	setAllowAutoReinvest	External ⓘ		onlyOwner
L	excludeFromDividends	External ⓘ		onlyOwner
L	includeFromDividends	External ⓘ		onlyOwner
L	setAutoClaim	External ⓘ		onlyOwner
L	setReinvest	External ⓘ		onlyOwner
L	setMinimumTokenBalanceForAutoDividends	External ⓘ		onlyOwner
L	setMinimumTokenBalanceForDividends	External ⓘ		onlyOwner
L	setDividendsPaused	External ⓘ		onlyOwner
L	getLastProcessedIndex	External ⓘ		NO ⓘ
L	getNumberOfTokenHolders	External ⓘ		NO ⓘ

L	getAccount	Public 		NO 
L	getAccountAtIndex	Public 		NO 
L	setBalance	External 		onlyOwner
L	process	Public 		NO 
L	processAccount	Public 		onlyOwner
L	updateUniswapV2Router	Public 		onlyOwner
L	updatePayoutToken	Public 		onlyOwner
L	getPayoutToken	Public 		NO 
L	_reinvestDividendOfUser	Private 		
L	_withdrawDividendOfUser	Internal 		
IterableMapping	Library			
L	get	Internal 		
L	getIndexOfKey	Internal 		
L	getKeyAtIndex	Internal 		
L	size	Internal 		
L	set	Internal 		
L	remove	Internal 		



Function
can modify
state



Function
is payable

Audit Scope

Audit Method.

Our smart contract audit is an extensive methodical examination and analysis of the smart contract's code that is used to interact with the blockchain. Goal: discover errors, issues and security vulnerabilities in the code. Findings getting reported and improvements getting suggested.

Automatic and Manual Review

We are using automated tools to scan functions and weaknesses of the contract. Transfers, integer over-undeflow checks such as all CWE events.

Tools we use:

Visual Studio Code

CWE

SWC

Solidity Scan

SVD

In manual code review our auditor looking at source code and performing line by line examination. This method helps to clarify developer's coding decisions and business logic.

Skeleton Ecosystem

<https://skeletonecosystem.com>

<https://github.com/SkeletonEcosystem/Audits>

