



#### 0x59FB9a377004CC8974009F72d94b4d801C58499B





#### 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	6
Detected Vulnerability Description	11
Contract Flow Chart	18
Inheritance Graph	19
Contract Desciptions	26
Audit Scope	31



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% safaty 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	BABY LTC	
Ticker/Simbol	BLTC	
Blockchain	Binance Smart Chain Bep20	
Contract Address	0x59FB9a377004CC8974009F72d94b4d801C58499B	
Creator Address	0xB811D56ed0E71794869A2445cb159B2C235e46c9	
Current Owner Address	0xC3bc56601d96Faa84beE11EB587F4bF89F9Df529	
Contract Explorer	https://bscscan.com/token/0x59FB9a377004CC8974009 F72d94b4d801C58499B	
Compiler Version	v0.8.4+commit.c7e474f2	
License	MIT License	
Optimisation	Yes with 200 Runs	
Total Supply	1,000,000,000 BLTC	
Decimals	18	

## Creation/Audit

Contract Deployed	5 May 2023	
Audit Created	22-Aug-23 21:00:00 UTC	
Audit Update	V 0.1	

## Verified Socials

Website	https://babyltc.io	
Telegram	https://t.me/+rJXf3p1fjnljYWRk	
Twitter	https://twitter.com/babyltc_	



# Contract Function Analysis





Pass Attention Item A Risky Item





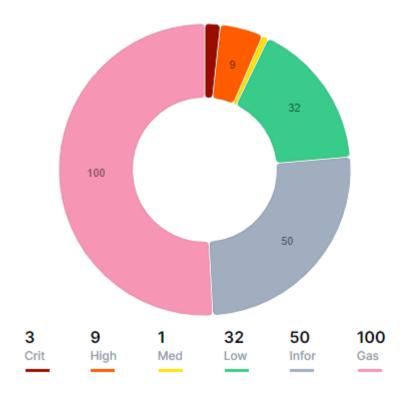
Contract Verified	<b>✓</b>	The contract source code is uploaded to blockchain explorer and is open source, so everybody can read it.
Contract Renounce	<u> </u>	Current Owner: 0xC3bc56601d96Faa84beE11EB587F4bF89F9Df529 Attention marked functions can be modified and used.
Buy Tax	14 %	Shows the taxes for purchase transactions. Above 10% may be considered a high tax rate. More than 50% tax rate means may not be tradable. <b>Setfee max found: 25</b> %
Sell Tax	14 %	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. Setfee max found: 25%
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
Liqudity Status	<b>✓</b>	Locked on 22.08.2023: 62% for 43331 days on Mudra locker Burned (locked forewer) 38% Note! Initial liqudity tokens scanned. For new LP Lockers allways re-check with skeleton scanner on telegram.
Trading Disable Functions	<b>~</b>	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.   If there is authorised hidden owner, or there is Retrieve Ownership Function, the trading disable function may be used!
Set Fees function	<u>^</u>	Fee Setting function found.  Setfee max found: 25%
Proxy Contract	~	The proxy contract means contract owner can modifiy the function of the token and possibly effect the price. The Owner is not the creator but the creator may have authorisation to change functions.
Mint Function	<b>~</b>	No mint function found.  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. If contract is renounced this function can't be used.



Balance	<b>✓</b>	No Balance Modifier function found.
Modifier Function		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.
		⚠ If contract is renounced this function still can be used as auto self Destruct
Whitelist Function	<u>^</u>	Whitelist Function Found.
		If there is a function for this Developer can set zero fee or no max wallet size for adresses (for example team wallets can trade without fee. Can cause farming)
		If there is a whitelist, some addresses may not be able to trade normally (honeypot risk
Hidden	<b>✓</b>	No authorised hidden owner found.
Owner Analysis		For contract with a hidden owner, developer can still manipulate the contract even if the ownership has been abandoned. Fake renounce.
Retrieve Ownership	<b>✓</b>	No functions found which can retrieve ownership of the contract.
Function		If this function exists, it is possible for the project owner to regain ownership even after relinquishing it. Also known as fake renounce.
Self	<b>✓</b>	No Self Destruct function found.
Destruct Function	•	If this function exists and is triggered, the contract will be destroyed, all functions will be unavailable, and all related assets will be erased.
Specific Tax	<u> </u>	Specific Tax Changing Functions found.
Changing Function		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!
Trading	<u> </u>	Trading Cooldown Function found.
Cooldown Function		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.
Max Transaction	<u> </u>	Max Transaction and Holding Modify function found.
and Holding Modify Function		If there is a function for this, the maximum trading amount or maximum position can be modified. Can cause honeypot
Transaction Limiting	<u> </u>	Transaction Limiter Function Found.
Function		The number of overall token transactions may be limited (honeypot risk)



# Contract Safety and Weakness





INCORRECT ACCESS CONTROL	3
USE OF TX.ORIGIN	2
UNCHECKED ARRAY LENGTH	1
UNCHECKED TRANSFER	3
APPROVE FRONT-RUNNING ATTACK	3
DANGEROUS UNARY EXPRESSIONS	1
USE OF MULTIPLE PRAGMA VERSIONS	1
USE OWNABLE2STEP	3
EMPTY TRY/CATCH BLOCK	3
OUTDATED COMPILER VERSION	1



MISSING EVENTS	21
LONG NUMBER LITERALS	3
VARIABLES SHOULD BE IMMUTABLE	4
MISSING UNDERSCORE IN NAMING VA	22
MISSING INDEXED KEYWORDS IN EVE	3
UNUSED RECEIVE FALLBACK	1
REQUIRE WITH EMPTY MESSAGE	11
USE CALL INSTEAD OF TRANSFER OR	1
BLOCK VALUES AS A PROXY FOR TIME	8



USE OF SAFEMATH LIBRARY	3
CONSTANT STATE VARIABLES	1
CODE OPTIMIZATION BY USING MAX	1
MAKE PUBLIC LIBRARY FUNCTIONS IN	6
FUNCTION SHOULD BE EXTERNAL	1
SPLITTING REQUIRE STATEMENTS	4
INTERNAL FUNCTIONS NEVER USED	1
GAS OPTIMIZATION IN INCREMENTS	1
UNNECESSARY CHECKED ARITHMETI	1
CHEAPER INEQUALITIES IN IF()	7



FUNCTION SHOULD RETURN STRUCT	4
ARRAY LENGTH CACHING	1
VARIABLES DECLARED BUT NEVER US	1
PUBLIC CONSTANTS CAN BE PRIVATE	1
LONG REQUIRE/REVERT STRINGS	11
CHEAPER INEQUALITIES IN REQUIRE()	10
CHEAPER CONDITIONAL OPERATORS	6
STORAGE VARIABLE CACHING IN MEM	40



# ⚠ Incorrect Acces Control (3 item)

```
319
320
              return dividendTracker.getAccountAtIndex(index);
321
322
323
          function processDividendTracker(uint256 gas) external {
324
325
                 uint256 iterations,
326
                 uint256 claims,
327
                 uint256 lastProcessedIndex
             ) = dividendTracker.process(gas);
328
329
             emit ProcessedDividendTracker(
330
                 iterations,
331
                 claims,
                lastProcessedIndex,
332
333
                 false,
334
                 gas,
335
                  tx.origin
336
              );
338
339
          function claim() external {
336
337
338
339
          function claim() external {
340
              dividendTracker.processAccount(payable(msg.sender), false);
341
342
343
          function getLastProcessedIndex() external view returns (uint256) {
344
              return dividendTracker.getLastProcessedIndex();
345
346
```

Function	Severity	Remedation
Access control plays an important role in segregation of privileges in smart contracts and other applications. If this is misconfigured or not properly validated on sensitive functions, it may lead to loss of funds, tokens and in some cases compromise of the smart contract.	Severity : Critical	It is recommended to go through the contract and observe the functions that are lacking an access control modifier. If they contain sensitive administrative actions, it is advised to add a suitable modifier to the same
The contract BABYTOKEN is importing an access control library @openzeppelin/contracts/access/Ownable.sol but the function processDividendTracker is missing the modifier onlyOwner.		



# ⚠ Unchecked Array Lenght (1 item)

```
169
              for (uint256 i = 0; i < accounts.length; i++) {</pre>
170
                  _isExcludedFromFees[accounts[i]] = true;
171
173
              emit ExcludeMultipleAccountsFromFees(accounts);
```

174 }		
Function	Severity	Remedation
Ethereum is a very resource-constrained environment. Prices per computational step are orders of magnitude higher than with centralized providers. Moreover, Ethereum miners impose a limit on the total number of Gas consumed in a block. If array.length is large enough, the function exceeds the block gas limit, and transactions calling it will never be confirmed.  for (uint256 i = 0; i < array.length; i++) { cosltyFunc(); }	Severity : High	Either explicitly or just due to normal operation, the number of iterations in a loop can grow beyond the block gas limit, which can cause the complete contract to be stalled at a certain point. Therefore, loops with a bigger or unknown number of steps should always be avoided.
This becomes a security issue if an external actor influences array.length.		
E.g., if an array enumerates all registered addresses, an adversary can register many addresses, causing the problem described above.		



#### ⚠ Use of TX. origin (2 Item)



Function	Severity	Remedation
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.	Severity : High	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.



#### ⚠ Unchecked Transfers (3 Item)

```
if (amount == 0) {
359
360
                  super._transfer(from, to, 0);
361
                  return;
362
363
364
               uint256 contractTokenBalance = balanceOf(address(this));
365
408
                 uint256 fees = amount.mul(totalFees).div(100);
410
411
                 super._transfer(from, address(this), fees);
412
414
              super._transfer(from, to, amount);
415
411
                  super._transfer(from, address(this), fees);
413
414
               super._transfer(from, to, amount);
415
417
                  \label{lem:dividendTracker.setBalance(payable(from), balanceOf(from))} \\
               () catch ()
```

Function	Severity	Remedation
Some tokens do not revert the transaction when the transfer or transferFrom fails and returns False. Hence we must check the return value after calling the transfer or transferFrom function.	Severity : High	Use OpenZeppelin SafeERC20's safetransfer and safetransferFrom functions.



#### ⚠ Approve Front Running Attack (3 Item)

```
476
          function swapTokensForEth(uint256 tokenAmount) private {
477
             // generate the uniswap pair path of token -> weth
478
              address[] memory path = new address[](2);
479
              path[0] = address(this);
             path[1] = uniswapV2Router.WETH();
480
481
482
             _approve(address(this), address(uniswapV2Router), tokenAmount);
483
484
             // make the swap
485
             uniswapV2Router.swapExactTokensForETHSupportingFeeOnTransferTokens(
486
                 tokenAmount,
487
                 0, // accept any amount of ETH
488
                 path,
489
                  address(this),
490
                  block.timestamp
491
492
           function swapTokensForCake(uint256 tokenAmount) private {
 495
              address[] memory path = new address[](3);
 496
              path[0] = address(this);
 497
              path[1] = uniswapV2Router.WETH();
 498
               path[2] = rewardToken;
 500
               _approve(address(this), address(uniswapV2Router), tokenAmount);
 502
               // make the swap
 503
              uniswapV2Router.swapExactTokensForTokensSupportingFeeOnTransferTokens(
 504
                  tokenAmount,
 505
                  0.
 506
                  path,
 507
                  address(this).
 508
                  block.timestamp
 509
              );
 510
           }
 512
          function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
512
         function addLiquidity(uint256 tokenAmount, uint256 ethAmount) private {
513
            // approve token transfer to cover all possible scenarios
514
             _approve(address(this), address(uniswapV2Router), tokenAmount);
515
516
             // add the liquidity
517
             uniswapV2Router.addLiquidityETH{value: ethAmount}(
518
                address(this),
519
                 tokenAmount,
520
                 0, // slippage is unavoidable
521
                 0, // slippage is unavoidable
522
                 address(0xdead).
523
                  block.timestamp
524
              );
526
527
          function swapAndSendDividends(uint256 tokens) private {
```



Function	Severity	Remedation
The swapTokensForEth() 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. 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. The function swapTokensForEth can be front-run by abusing the _approve function.	Severity : High	Only use the approve function of the ERC/BEP standard to change the allowed amount to 0 or from 0 (wait till transaction is mined and approved).  Token owner just needs to make sure that the first transaction actually changed allowance from N to 0, i.e., that the spender didn't manage to transfer some of N allowed tokens before the first transaction was mined. Such checking is possible using advanced blockchain explorers such as [Etherscan.io](https://etherscan.io/)  Another way to mitigate the threat is to approve token transfers only to smart contracts with verified source code that does not contain logic for performing attacks like described above, and to accounts owned by the people you may trust.



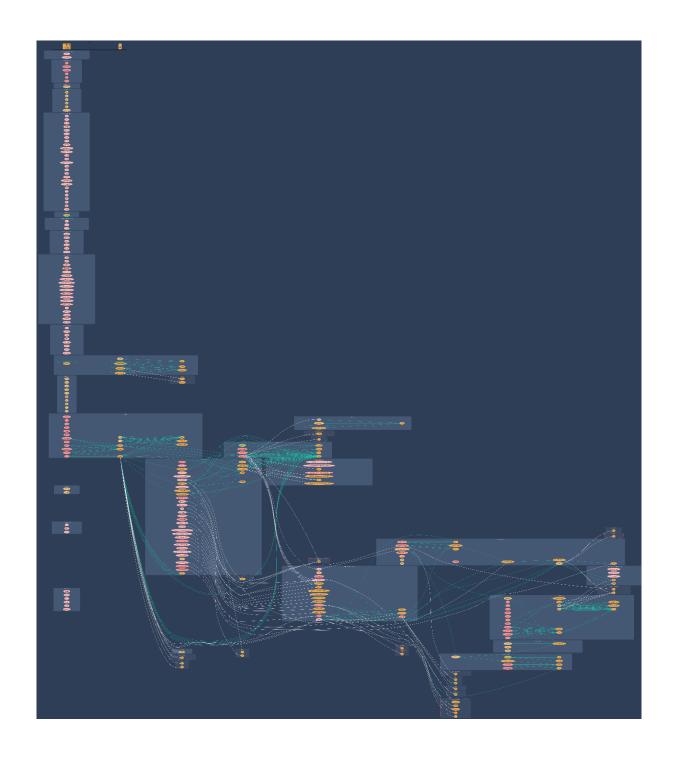
#### ⚠ Dangerous Unary Expressions (1 Item)



Function	Severity	Remedation
Mathematical operations in the smart contracts forms the base for all the arithmetic logic in the code. Care should be taken when implementing these because they may control critical function logic, tokens, ether, etc. The contract was found to be incorrectly implementing arithmetic expressions assuming the contract wanted to add the value of a to b and store it back in a. Correct usage: a += b; Incorrect usage: a =+b;	Severity : medium	Developers should exercise caution when writing these expressions because a simple mistake may cause loss of funds or compromise of the contract.  Make sure that the expressions used are in the correct format, i.e., a += b; and not a =+b;.

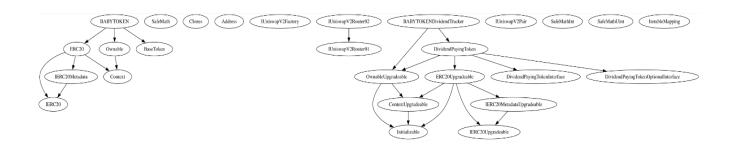


# Contract Flow Graph





## Inheritance Graph







# Contract Descriptions

Contract	File	Bases		
	<b>Function Name</b>	Visibility	Modifiers	
IERC20	Interface			
	totalSupply	External 🌡		NO[
	balanceOf	External [		NOÏ
	transfer	External 🌡		№Д
	allowance	External 🌡		NO
	approve	External 🌡		№Д
	transferFrom	External 🎚		NO[
IERC20Metadat a	Interface	IERC20		
	name	External 🌡		NO[
	symbol	External 🌡		NO[
	decimals	External 🌡		NO[
Contout	les els es setations			
Context	Implementation	Internal 🖺		
	_msgSender			
	_msgData	Internal 🖺		
ERC20	Implementation	Context, IERC20, IERC20Metadata		
		Public 🌡		NO[
	name	Public 🌡		NO
	symbol	Public 🌡		№[
	decimals	Public 🏻		№[
	totalSupply	Public 🌡		№
	balanceOf	Public 🌡		NO[
	transfer	Public 🌡		NO
	allowance	Public 🌡		NO[
	approve	Public 🌡		NO[
	transferFrom	Public 🌡		NO[
	increaseAllowanc e	Public 🎚	•	NO[



	decreaseAllowan ce	Public 🏿		Пои
	_transfer	Internal 🖺		
	_mint	Internal 🖺		
	_burn	Internal 🖺		
	_approve	Internal 🖺		
	_beforeTokenTra nsfer	Internal 🖺		
	_afterTokenTrans fer	Internal 🖺		
Ownable	Implementation	Context		
		Public 🌡		NO[
	owner	Public 🏿		NO
	renounceOwners hip	Public 🌡		onlyOwner
	transferOwnershi p	Public 🌡		onlyOwner
	_setOwner	Private 🖺		
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 🖺		
Clones	Library			
	clone	Internal 🖺		
	cloneDeterminist ic	Internal 🖺	•	
	predictDetermini sticAddress	Internal 🖺		



	predictDetermini sticAddress	Internal 🖺		
Address	Library			
	isContract	Internal 🖺		
	sendValue	Internal 🖺		
	functionCall	Internal 🖺		
	functionCall	Internal 🖺		
	functionCallWith Value	Internal 🖺		
	functionCallWith Value	Internal 🖺		
	functionStaticCal I	Internal 🖺		
	functionStaticCal I	Internal 🖺		
	functionDelegate Call	Internal 🖺		
	functionDelegate Call	Internal 🖺		
	verifyCallResult	Internal 🖺		
IUniswapV2Fact ory	Interface			
	feeTo	External 🌡		NO
	feeToSetter	External 🌡		№Д
	getPair	External 🌡		NO
	allPairs	External [		NO
	allPairsLength	External [		NO
	createPair	External [		NO
	setFeeTo	External [		NO
	setFeeToSetter	External [		NO
IUniswapV2Rou ter01	Interface			
	factory	External 🏻		NO
	WETH	External 🏻		NO
	addLiquidity	External 🌡		NO
	addLiquidityETH	External 🌡	<u>ED</u>	NO
	removeLiquidity	External 🎚		NO
	removeLiquidityE TH	External [		NO[



	removeLiquidity WithPermit	External 🌡		NO
	removeLiquidityE THWithPermit	External [		NO[
	swapExactToken sForTokens	External [		NO
	swapTokensForE xactTokens	External [		NO
	swapExactETHFo rTokens	External [	gip	NO
	swapTokensForE xactETH	External [		NO
	swapExactToken sForETH	External [		NO
	swapETHForExac tTokens	External [	<u>an</u>	NO
	quote	External 🌡		NO
	getAmountOut	External 🏻		NO
	getAmountIn	External 🏻		NO
	getAmountsOut	External 🏻		NO
	getAmountsIn	External 🌡		NOJ
IUniswapV2Rou ter02	Interface	IUniswapV2Rout er01		
	removeLiquidityE THSupportingFe eOnTransferToke ns	External 🌡		ПоМ
	removeLiquidityE THWithPermitSu pportingFeeOnTr ansferTokens	External 🌡		NO[
	swapExactToken sForTokensSupp ortingFeeOnTran sferTokens	External 🌡		№[
	swapExactETHFo rTokensSupporti ngFeeOnTransfer Tokens	External 🌡	<u>dia</u>	ПОЛ



	swapExactToken sForETHSupporti ngFeeOnTransfer Tokens	External 🌡	№
IERC20Upgrade able	Interface		
	totalSupply	External 🏻	NO[
	balanceOf	External 🏻	NO[
	transfer	External 🌡	NO
	allowance	External 🏻	NO[
	approve	External 🏻	NO
	transferFrom	External 🌡	NO
IERC20Metadat aUpgradeable	Interface	IERC20Upgradea ble	
	name	External 🌡	№
	symbol	External 🌡	NOÏ
	decimals	External 🌡	NO[
Initializable	Implementation		
ContextUpgrad eable	Implementation	Initializable	
	Context_init	Internal 🖺	initializer
	Context_init_u nchained	Internal 🖺	initializer
	_msgSender	Internal 🖺	
	_msgData	Internal 🖺	
ERC20Upgradea ble	Implementation	Initializable, ContextUpgrade able, IERC20Upgradea ble, IERC20Metadata Upgradeable	
	ERC20_init	Internal 🖺	initializer
	ERC20_init_unc hained	Internal 🖺	initializer



	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
	increaseAllowanc e	Public 🌡	•	NOÎ
	decreaseAllowan ce	Public 🌡		МО[
	_transfer	Internal 🖺		
	_mint	Internal 🖺		
	_burn	Internal 🖺		
	_approve	Internal 🖺		
	_beforeTokenTra nsfer	Internal 🖺		
	_afterTokenTrans fer	Internal 🖺		
OwnableUpgra deable	Implementation	Initializable, ContextUpgrade able		
	Ownable_init	Internal 🖺		initializer
	Ownable_init_u nchained	Internal 🖺		initializer
	owner	Public 🏻		NO
	renounceOwners hip	Public 🌡		onlyOwner
	transferOwnershi p	Public 🌡		onlyOwner
	_setOwner	Private 🖺		
IUniswapV2Pair	Interface			-
	name	External 🎚		NO[
	symbol	External 🎚		NO
	decimals	External 🌡		NO
	totalSupply	External 🌡		NO
	balanceOf	External 🌡		NO



	allowance	External 🌡	ио₿
	approve	External [	NO
	transfer	External [	NO
	transferFrom	External [	NO
	DOMAIN_SEPAR ATOR	External 🌡	№.
	PERMIT_TYPEHA SH	External 🌡	NO[
	nonces	External [	NO
	permit	External 🌡	NO
	MINIMUM_LIQUI DITY	External [	МОД
	factory	External 🏻	NO
	token0	External 🏻	 NO
	token1	External 🏻	NO
	getReserves	External 🌡	NO
	price0Cumulativ eLast	External 🌡	NO[
	price1Cumulativ eLast	External 🌡	NO
	kLast	External [	№
	mint	External [	NO
	burn	External [	NO
	swap	External [	NO
	skim	External 🌡	NO[
	sync	External 🌡	NO[
	initialize	External 🌡	NO[
SafeMathInt	Library		
	mul	Internal 🖺	
	div	Internal 🖺	
	sub	Internal 🖺	
	add	Internal 🖺	
	abs	Internal 🖺	
	toUint256Safe	Internal 🖺	
SafeMathUint	Library		
	toInt256Safe	Internal 🖺	
IterableMappin g	Library		
<u> </u>	get	Public 🌡	NO



	getIndexOfKey	Public 🌡	№
	getKeyAtIndex	Public 🎚	NO
	size	Public 🌡	NO[
	set	Public 🌡	NO[
	remove	Public 🌡	NO
DividendPaying TokenInterface	Interface		
	dividendOf	External 🌡	NO[
	withdrawDividen d	External 🌡	NO
DividendPaying TokenOptionall nterface	Interface		
	withdrawableDivi dendOf	External [	NO
	withdrawnDivide ndOf	External [	NO
	accumulativeDivi dendOf	External [	NO[
DividendPaying Token	Implementation	ERC20Upgradea ble, OwnableUpgrad eable, DividendPayingT okenInterface, DividendPayingT okenOptionalInt erface	
	DividendPayin gToken_init	Internal 🖺	initializer
	distributeCAKEDi vidends	Public 🎚	onlyOwner
	withdrawDividen d	Public 🌡	NO
	_withdrawDivide ndOfUser	Internal 🖺	



	dividendOf	Public 🎚	NO
	withdrawableDivi dendOf	Public 🌡	 NO
	withdrawnDivide ndOf	Public 🎚	МОД
	accumulativeDivi dendOf	Public 🎚	ио[
	_transfer	Internal 🖺	
	_mint	Internal 🖺	
	_burn	Internal 🖺	
	_setBalance	Internal 🖺	
BABYTOKENDiv idendTracker	Implementation	Ownable Upgrad eable, Dividend Paying T oken	
	initialize	External 🏻	initializer
	_transfer	Internal 🖺	
	withdrawDividen d	Public 🎚	МО[
	excludeFromDivi dends	External 🌡	onlyOwner
	isExcludedFromD ividends	Public 🎚	NO
	updateClaimWai t	External 🌡	onlyOwner
	updateMinimum TokenBalanceFor Dividends	External 🌡	onlyOwner
	getLastProcesse dIndex	External 🌡	МОД
	getNumberOfTo kenHolders	External 🌡	МОД
	getAccount	Public 🌡	NO
	getAccountAtInd ex	Public 🎚	NOÎ
	canAutoClaim	Private 🖺	
	setBalance	External [	onlyOwner
	process	Public [	NO
	processAccount	Public [	onlyOwner



BaseToken	Implementation			
BABYTOKEN	Implementation	ERC20, Ownable, BaseToken		
		Public 🌡	<b>ED</b>	ERC20
		External [	<u>a</u> D	NO
	setSwapTokensA tAmount	External 🌡		onlyOwner
	excludeFromFee s	External 🌡		onlyOwner
	excludeMultiple AccountsFromFe es	External 🌡		onlyOwner
	setMarketingWal let	External 🌡		onlyOwner
	setTokenReward sFee	External 🌡		onlyOwner
	setLiquiditFee	External 🎚		onlyOwner
	setMarketingFee	External 🌡		onlyOwner
	_setAutomatedM arketMakerPair	Private 🖺		
	updateGasForPr ocessing	Public 🌡		onlyOwner
	updateClaimWai t	External 🌡		onlyOwner
	getClaimWait	External 🌡		NO
	updateMinimum TokenBalanceFor Dividends	External 🌡		onlyOwner
	getMinimumTok enBalanceForDivi dends	External 🌡		NOÎ
	getTotalDividend sDistributed	External [		МО[
	isExcludedFromF ees	Public 🎚		NO[
	withdrawableDivi dendOf	Public 🌡		ио[
	dividendTokenBa lanceOf	Public 🏿		МОД
	excludeFromDivi dends	External 🏿		onlyOwner



is Excluded From Dividends	Public 🌡	ио≬
getAccountDivid endsInfo	External 🌡	МО[
getAccountDivid endsInfoAtIndex	External [	Мо[
processDividend Tracker	External [	Мо[
claim	External 🏻	№Д
getLastProcesse dIndex	External 🌡	МО[
getNumberOfDiv idendTokenHold ers	External 🌡	Мо[
_transfer	Internal 🖺	
swapAndSendTo Fee	Private 🖺	
swapAndLiquify	Private 🖺	
swapTokensForE th	Private 🖺	
swapTokensForC ake	Private 🖺	
addLiquidity	Private 🖺	
swapAndSendDi vidends	Private 🖺	

Function can modify state

**I** 

Function is payable

#### Source:

File Name SHA-1 Hash

c:\Solidity\babyltc.sol b4e69c7c64c78595cd52c4f5dba4be96d2bb96c4



#### 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 vulnaribilities in the code. Findings getting reported and improvements getting suggested.

#### Automatic and Manual Review

We are using automated tools to scan functions and weeknesses 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

