SOLIDITY FINANCE

MetaMate

Smart Contract Audit Report

AUDIT SUMMARY

MetaMate is building a new BEP-20 token with a Liquidity Generation Event.

For this audit, we reviewed the project team's MTMToken contract at 0x3244478Da3F518B33b99D5929Dd0bC3396C56981 on the Binance Smart Chain Mainnet.

AUDIT FINDINGS

> *Please ensure trust in the team prior to investing as they have some control in the ecosystem and currently own 100% of the total supply.*
> *Date: March 21st, 2022.*

Description: *The* `setFee` *function is declared public, but is never called internally.*

Recommendation: *This function should be declared external for additional gas savings on each call.*

CONTRACT OVERVIEW

- *The total supply of the token is set to 1 billion $MTM [1,000,000,000].*
- *No mint or burn functions are present though the circulating supply can be decreased by sending tokens to the 0x..dead address.*
- *At the time of writing this report, 100% of the total supply belongs to the owner.*

- *The contract supports a Liquidity Generation Event consisting of one or many rounds as determined by the owner.*
- *Each round has a set duration, a maximum contribution amount per user, and a whitelist of addresses that are able to participate; these values are determined by the LGE Whitelister address, which is set by the owner.*
- *After the rounds are created and the Pair address is set, anyone may kick off the Liquidity Generation Event by transferring tokens to the Pair address.*
- *Only whitelisted users may participate in the Liquidity*

may be a different set of whitelisted users for every round.

- *During the Liquidity Generation Event, the amount of purchased tokens is recorded to ensure it does not exceed the maximum allowed amount per user per round.*
- *The LGE Whitelister address can set the Pair address, delete the round data, modify the round duration, maximum contribution amount, and list of whitelisted users for any round at any time.*
- *The LGE Whitelister address can transfer their role to any address at any time; initially, the owner is set as the LGE Whitelister.*

- *The contract enforces a maximum transaction amount (determined by the owner) when selling tokens via Pancakeswap.*
- *There is a transfer fee charged on all buy and sell trades with Pancakeswap where neither the sender nor the recipient is the contract address.*
- *The tokens collected through the transfer fee are swapped for BNB and sent to the team's Marketing wallet.*
- *The owner can set both the buy fee and sell fee to any values up to 10%.*
- *The owner can set the team's Marketing wallet to any address at any time.*
- *The owner can set the maximum transaction amount to any value greater than 200,000.*

> - *The contract utilizes the SafeMath library to prevent overflows/underflows.*
> - *The contract complies with the BEP-20 token standard.*

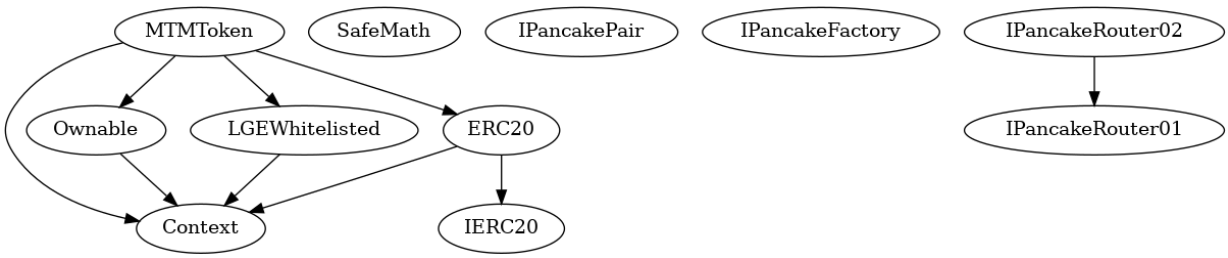
AUDIT RESULTS

Vulnerability Category	Notes	Result
Arbitrary Jump/Storage Write	N/A	PASS
Centralization of Control	The owner currently owns 100% of the total supply.	PASS
Compiler Issues	N/A	PASS
Delegate Call to Untrusted Contract	N/A	PASS
Dependence on Predictable Variables	N/A	PASS

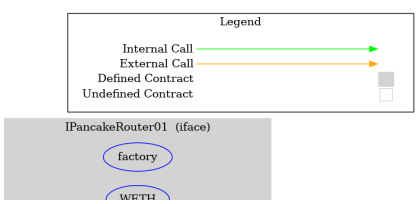
Vulnerability Category	Notes	Result
Ether/Token Theft	N/A	PASS
Flash Loans	N/A	PASS
Front Running	N/A	PASS
Improper Events	N/A	PASS
Improper Authorization Scheme	N/A	PASS
Integer Over/Underflow	N/A	PASS
Logical Issues	N/A	PASS
Oracle Issues	N/A	PASS
Outdated Compiler Version	N/A	PASS
Race Conditions	N/A	PASS

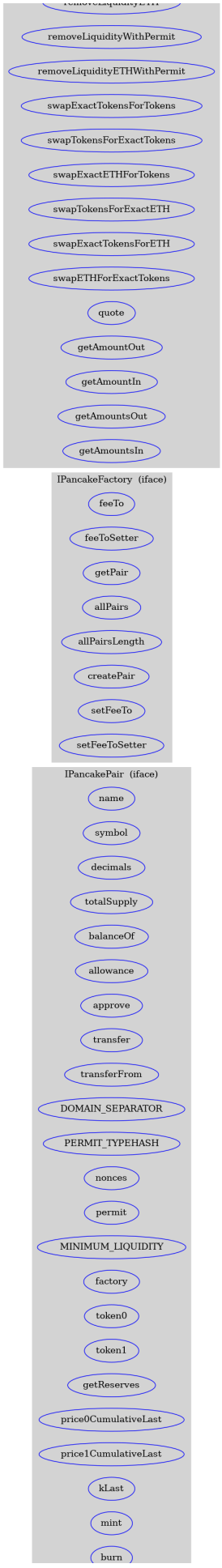
Vulnerability Category	Notes	Result
Signature Issues	N/A	PASS
Unbounded Loops	N/A	PASS
Unused Code	N/A	PASS
Overall Contract Safety		PASS

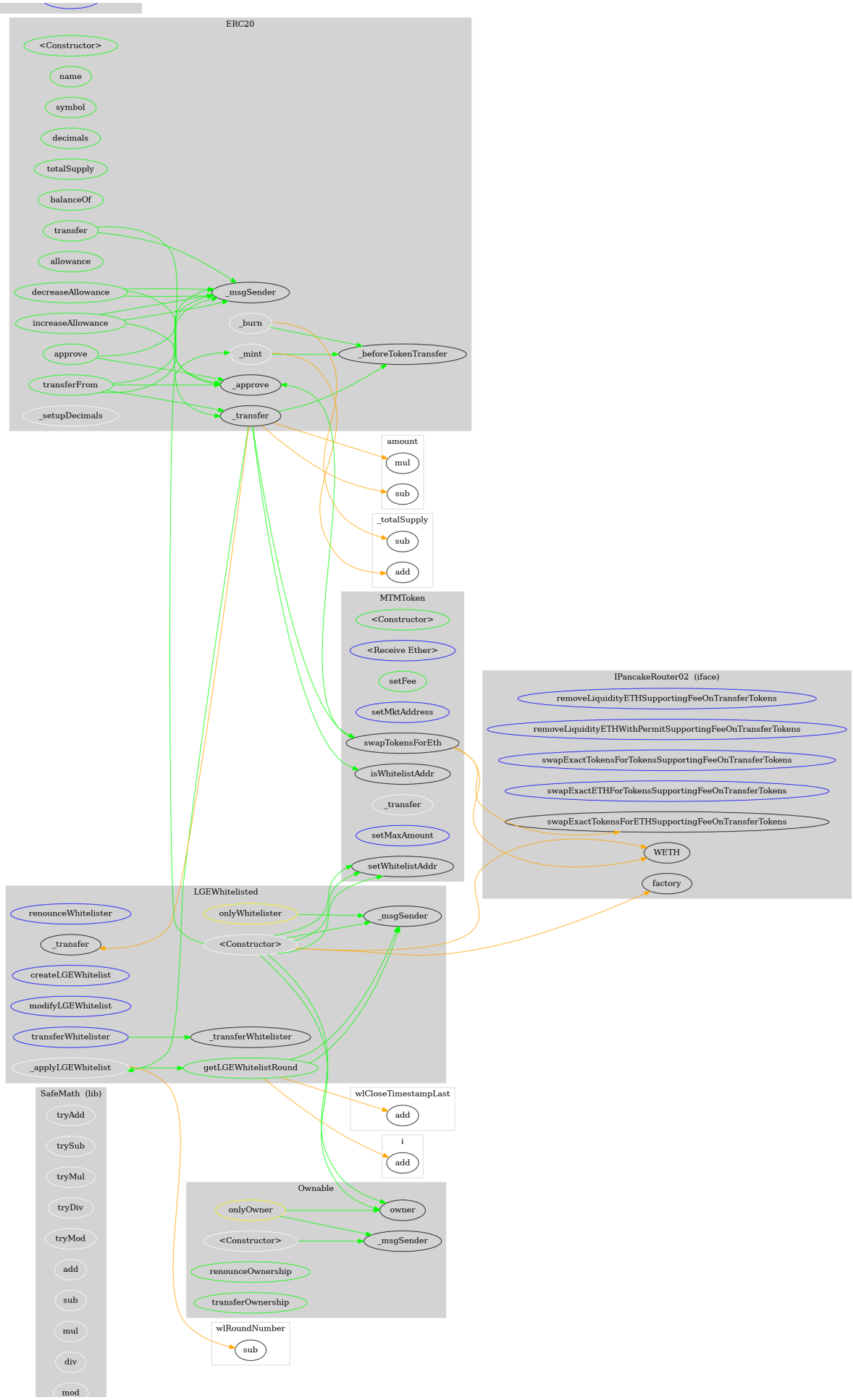
INHERITANCE CHART

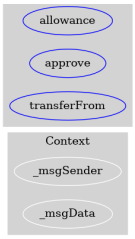


FUNCTION GRAPH









FUNCTIONS OVERVIEW

```
($) = payable function
# = non-constant function

Int = Internal
Ext = External
Pub = Public

+  Context
   - [Int] _msgSender
   - [Int] _msgData

+ [Int] IERC20
   - [Ext] totalSupply
   - [Ext] balanceOf
   - [Ext] transfer #
   - [Ext] allowance
   - [Ext] approve #
   - [Ext] transferFrom #

+ [Lib] SafeMath
   - [Int] tryAdd
   - [Int] trySub
```

```
    - [Int] tryMod
    - [Int] add
    - [Int] sub
    - [Int] mul
    - [Int] div
    - [Int] mod
    - [Int] sub
    - [Int] div
    - [Int] mod

+   ERC20 (Context, IERC20)
    - [Pub]  #
    - [Pub] name
    - [Pub] symbol
    - [Pub] decimals
    - [Pub] totalSupply
    - [Pub] balanceOf
    - [Pub] transfer #
    - [Pub] allowance
    - [Pub] approve #
    - [Pub] transferFrom #
    - [Pub] increaseAllowance #
    - [Pub] decreaseAllowance #
    - [Int] _transfer #
    - [Int] _mint #
    - [Int] _burn #
    - [Int] _approve #
    - [Int] _setupDecimals #
    - [Int] _beforeTokenTransfer #

+   Ownable (Context)
    - [Int]  #
```

```
        - modifiers: onlyOwner
    - [Pub] transferOwnership #
        - modifiers: onlyOwner

+ [Int] IPancakePair
    - [Ext] name
    - [Ext] symbol
    - [Ext] decimals
    - [Ext] totalSupply
    - [Ext] balanceOf
    - [Ext] allowance
    - [Ext] approve #
    - [Ext] transfer #
    - [Ext] transferFrom #
    - [Ext] DOMAIN_SEPARATOR
    - [Ext] PERMIT_TYPEHASH
    - [Ext] nonces
    - [Ext] permit #
    - [Ext] MINIMUM_LIQUIDITY
    - [Ext] factory
    - [Ext] token0
    - [Ext] token1
    - [Ext] getReserves
    - [Ext] price0CumulativeLast
    - [Ext] price1CumulativeLast
    - [Ext] kLast
    - [Ext] mint #
    - [Ext] burn #
    - [Ext] swap #
    - [Ext] skim #
    - [Ext] sync #
    - [Ext] initialize #
```

```
    - [Ext] feeTo
    - [Ext] feeToSetter
    - [Ext] getPair
    - [Ext] allPairs
    - [Ext] allPairsLength
    - [Ext] createPair #
    - [Ext] setFeeTo #
    - [Ext] setFeeToSetter #

+ [Int] IPancakeRouter01
    - [Ext] factory
    - [Ext] WETH
    - [Ext] addLiquidity #
    - [Ext] addLiquidityETH ($)
    - [Ext] removeLiquidity #
    - [Ext] removeLiquidityETH #
    - [Ext] removeLiquidityWithPermit #
    - [Ext] removeLiquidityETHWithPermit #
    - [Ext] swapExactTokensForTokens #
    - [Ext] swapTokensForExactTokens #
    - [Ext] swapExactETHForTokens ($)
    - [Ext] swapTokensForExactETH #
    - [Ext] swapExactTokensForETH #
    - [Ext] swapETHForExactTokens ($)
    - [Ext] quote
    - [Ext] getAmountOut
    - [Ext] getAmountIn
    - [Ext] getAmountsOut
    - [Ext] getAmountsIn

+ [Int] IPancakeRouter02 (IPancakeRouter01)
    - [Ext] removeLiquidityETHSupportingFeeOnTransferTok
```

```
    - [Ext] swapExactETHForTokensSupportingFeeOnTransfer
    - [Ext] swapExactTokensForETHSupportingFeeOnTransfer

+   LGEWhitelisted (Context)
    - [Int]  #
    - [Ext] renounceWhitelister #
       - modifiers: onlyWhitelister
    - [Ext] transferWhitelister #
       - modifiers: onlyWhitelister
    - [Int] _transferWhitelister #
    - [Ext] createLGEWhitelist #
       - modifiers: onlyWhitelister
    - [Ext] modifyLGEWhitelist #
       - modifiers: onlyWhitelister
    - [Pub] getLGEWhitelistRound
    - [Int] _applyLGEWhitelist #

+   MTMToken (Context, ERC20, Ownable, LGEWhitelisted)
    - [Pub]  #
       - modifiers: ERC20
    - [Ext]  ($)
    - [Pub] setFee #
       - modifiers: onlyOwner
    - [Ext] setMktAddress #
       - modifiers: onlyOwner
    - [Pub] setWhitelistAddr #
       - modifiers: onlyOwner
    - [Pub] isWhitelistAddr
    - [Int] _transfer #
    - [Ext] setMaxAmount #
       - modifiers: onlyOwner
    - [Prv] swapTokensForEth #
```

ABOUT SOLIDITY FINANCE

Solidity Finance was founded in 2020 and quickly grew to have one of the most experienced and well-equipped smart contract auditing teams in the industry. Our team has conducted 1000+ solidity smart contract audits covering all major project types and protocols, securing a total of over $10 billion U.S. dollars in on-chain value.
Our firm is well-reputed in the community and is trusted as a top smart contract auditing company for the review of solidity code, no matter how complex. Our team of experienced solidity smart contract auditors performs audits for tokens, NFTs, crowdsales, marketplaces, gambling games, financial protocols, and more!

Contact us today to get a free quote for a smart contract audit of your project!

WHAT IS A SOLIDITY AUDIT?

Typically, a smart contract audit is a comprehensive review process designed to discover logical errors, security vulnerabilities, and optimization opportunities within code. A *Solidity Audit* takes this a step further by verifying economic logic to ensure the stability of smart contracts and highlighting privileged functionality to create a report that is easy to understand for developers and community members alike.

HOW DO I INTERPRET THE FINDINGS?

Each of our Findings will be labeled with a Severity level. We always recommend the team resolve High, Medium, and Low severity findings prior to deploying the code to the mainnet. Here is a breakdown on what each Severity level means for the project:

- **High** severity indicates that the issue puts a large number of users' funds at risk and has a high probability of exploitation, or the smart contract contains serious logical issues which can prevent the code from operating as intended.
- **Medium** severity issues are those which place at least some users' funds at risk and has a medium to high probability of exploitation.
- **Low** severity issues have a relatively minor risk association; these issues have a low probability of occurring or may have a minimal impact.

practices.

GO HOME