

AUDIT

GUARDIANSALE AUDIT

"WWW.GUARDIANSALE.COM"




Guardiansale token is forked token of safemoon
so it has the same infrastructure.

guardiansale.com

SUMMARY



- 01 Summary** 
- 02 Understanding** 
- 03 Function Map** 
- 04 Safemoon Contract Fix** 
- 05 Resources & Social media** 
- 06 Renounce ownership and time lock** 

SUMMARY



PROJECT NAME:

GUARDIANSALE TOKEN

DESCRIPTION:

**THE GUARDIANSALE TOKEN
IS A MIXTURE OF RFI TOKEN
AND ADDED FUNCTION OF A
GENERATED PROTOCOL**

PLATFORM:

BSC

LANGUAGE:

SOLIDITY

CODEBASE:

CODEBASE:

AUDIT METHODOLOGY:

**STATIC ANALYSIS,
MANUAL REVIEW,
TESTNET**

UNDERSTANDING

OVERVIEW

The guardiansale protocol is decentralized “DeFi” token deployed on the Binance smart chain “BSC”. Guardiansale has 4 features in tax:

- Rewards for holders
- Liquidity
- Marketing
- LP buyback & Burn

guardiansale.com



@guardiansale

UNDERSTANDING

OVERVIEW

As guardiansale we have created the most detailed explanation of smart contracts. Our goal is making our contracts understandable and transparent as well as supporting developers community. In all DeFi market this is a first. Take this chance and look it up in our github and bscscan.

Each guardiansale transaction is taxed 4%.

- * %1 of this tax goes to all holders.

- * %1 of this tax goes for liquidity.

- * %1 of this tax goes for buyback & burn.

- * %1 of this tax goes for marketing.

LP ACQUISITION

The LP acquisition mechanism can be indirectly triggered by any normal transaction of the token as all transfers evaluate the set of conditions that triggers the mechanism. The main conditions of the mechanism are whether the sender is different than the LP pair and whether the accumulation threshold has been breached. Should these conditions be satisfied, the swapAndLiquify function is invoked with the current contract's Guardiansale token balance.

The swapAndLiquify function splits the contract's balance in two halves properly accounting for any truncation that may occur. The first half is swapped to BNB via the PancakeSwap Router using the Guardiansale-BNB pair and thus temporarily driving the price of the Guardiansale token down. Afterwards, the resulting BNB balance along with the remaining Guardiansale balance are supplied to the Guardiansale-BNB liquidity pool as liquidity via the Router.

STATIC REWARD (REFLECTION)

Balances in the Guardiansale token system are calculated in one of two ways. The first method, which most users should be familiar with, is a traditional fixed number of units being associated with a user's address. The second method, which is of interest to static rewards, represents a user's balance as a proportion of the total supply of the token. This method works similarly to how dynamic rebasing mechanisms work such as that of Ampleforth.



Whenever a taxed transaction occurs, the 1% meant to be re-distributed to token holders is deducted from the total "proportion" supply resulting in a user's percentage of total supply being increased. Within the system, not all users are integrated in this system and as such the 1% fee is rewarded to a subset of the total users of the Guardiansale token. The owner of the contract is able to introduce and exclude users from the dynamic balance system at will.

STATIC REWARD (REFLECTION)

The contract contains the following privileged functions that are restricted by the onlyOwner modifier. They are used to modify the contract configurations and address attributes. We grouped these functions below: Account management functions for inclusion and exclusion in the fee and reward system:

Account management functions for inclusion and exclusion in the fee and reward system:

- excludeFromReward(address account)
- includeInReward(address account)
- excludeFromFee(address account)
- includeInFee(address account)

Modification of liquidation, tax and max transaction percents of the system:

- function setTaxFeePercent(uint256 taxFee)
- function setLiquidityFeePercent(uint256 liquidityFee)
- function setMaxTxPercent(uint256 maxTxPercent)

Toggle feature of the LP acquisition mechanism:

- function setSwapAndLiquifyEnabled(bool _enabled)

FUNCTION MAP



ALL PROCESS WILL BE AUTOMATIC WITHOUT HUMAN INTERVENTION

TOKEN CONTRACT

- 6% token is transferred to presale
- 94 % of token is transferred to timelock contract .
- Exclude presale contract from tax fees
- Exclude timelock contract from rewards

PRESALE CONTRACT

- 6% TOKEN RECEIVE FROM TOKEN CONTRACT
- Each transaction will send BNB'S TO timelock directly

TIME LOCK CONTRACT

- 94 % of token is received from token contract.
- Pair 4 % of token with presale received BNB to open liquidity pool and receive cake LP without human intervention or leaving the lock contract.
- After 3 months 15 % of the tokens will be out for listing purposes.
- Rest of the token and cake LP token will be locked for a year.

PANCAKESWAP POOL

SAFEMOON CONTRACT FIX



Guardianpresale token contract is a forked contract of safemoon and we fixed the errors that spotted by certik. You can check the rest of the explanations on guardiansale bscscan, github, ethscan.

FIXED: INCORRECT MESSAGE CHANGED TO "ACCOUNT IS NOT EXCLUDED".

CERTIK

SafeMoon Security Assessment

SSL-01 | Incorrect error message

Category	Severity	Location	Status
Logical Issue	Minor	Safemoon.sol: 869	Acknowledged

Description

The error message in `require(!_isExcluded[account], "Account is already excluded")` does not describe the error correctly.

Recommendation

The message "Account is already excluded" can be changed to "Account is not excluded".

Alleviation

The team acknowledged the finding, and given the deployed contract cannot be updated, decided to retain the code base unchanged.

FIXED: ELSE IF CHANGED TO "ELSE".

CERTIK

SafeMoon Security Assessment

SSL-02 | Redundant code

Category	Severity	Location	Status
Logical Issue	Informational	Safemoon.sol: 1128	Acknowledged

Description

The condition `(!_isExcluded[sender] && !_isExcluded[recipient])` can be included in `else`.

Recommendation

The following code can be removed:

```
1 ... else if (!_isExcluded[sender] && !_isExcluded[recipient]) {
2   _transferStandard(sender, recipient, amount);
3 } ...
```

Alleviation

The team acknowledged the finding, and given the deployed contract cannot be updated, decided to retain the code base unchanged.

WE HAVE CREATED A FUNCTION THAT SENDS LEFTOVER BNB FROM SWAPANDLIQUIFY TO TIMELOCK CONTACT.

CERTIK

SafeMoon Security Assessment

SSL-03 | Contract gains non-withdrawable BNB via the `swapAndLiquify` function

Category	Severity	Location	Status
Logical Issue	Medium	Safemoon.sol: 1057	Acknowledged

Description

The `swapAndLiquify` function converts half of the `contract.balanceOf(address)` SafeMoon tokens to BNB. The other half of SafeMoon tokens and part of the converted BNB are deposited into the SafeMoon-BNB pool on pancakeswap as liquidity. For every `swapAndLiquify` function call, a small amount of BNB leftover in the contract. This is because the price of SafeMoon drops after swapping the first half of SafeMoon tokens into BNBs, and the other half of SafeMoon tokens require less than the converted BNB to be paired with it when adding liquidity. The contract doesn't appear to provide a way to withdraw those BNB, and they will be locked in the contract forever.

Recommendation

It's not ideal that more and more BNB are locked into the contract over time. The simplest solution is to add a `withdraw` function in the contract to withdraw BNB. Other approaches that benefit the SafeMoon token holders can be:

- Distribute BNB to SafeMoon token holders proportional to the amount of token they hold.
- Use leftover BNB to buy back SafeMoon tokens from the market to increase the price of SafeMoon.

Alleviation

The team acknowledged the finding, and given the deployed contract cannot be updated, decided to retain the code base unchanged.

SAFEMOON CONTRACT FIX



IN SAFEMOON CONTRACT GENERATED LP TOKEN GOES TO OWNER. WE HAVE CHANGED IT FROM OWNER TO TIMELOCK CONTRACT.

Safemoon Security Assessment

SSL-04 | Centralized risk in `addLiquidity`

Category	Severity	Location	Status
Centralization / Privilege	Major	Safemoon.sol: 1108	Partially Resolved

Description

```
1 // add the liquidity
2 uniswapV2Router.addLiquidityETH(value: ethAmount){
3     address(this),
4     tokenAmount,
5     0, // slippage is unavoidable
6     0, // slippage is unavoidable
7     owner(),
8     block.timestamp
9 };
```

The `addLiquidity` function calls the `uniswapV2Router.addLiquidityETH` function with the `to` address specified as `owner()` for acquiring the generated LP tokens from the `SafeMoon-BNB` pool. As a result, over time the `_owner` address will accumulate a significant portion of LP tokens. If the `_owner` is an EOA (Externally Owned Account), mishandling of its private key can have devastating consequences to the project as a whole.

Recommendation

We advise the `to` address of the `uniswapV2Router.addLiquidityETH` function call to be replaced by the contract itself, i.e. `address(this)`, and to restrict the management of the LP tokens within the scope of the contract's business logic. This will also protect the LP tokens from being stolen if the `_owner` account is compromised. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or via smart-contract based accounts with enhanced security practices, f.e. Multisignature wallets.


Indicatively, here are some feasible solutions that would also mitigate the potential risk:

- Time-lock with reasonable latency, i.e. 48 hours, for awareness on privileged operations;
- Assignment of privileged roles to multi-signature wallets to prevent single point of failure due to the private key;
- Introduction of a DAO / governance / voting module to increase transparency and user involvement.

SAFEMOON CONTRACT FIX



FIXED: TYPOS IN THE CONTRACT.

Safemoon Security Assessment

SSL-11 | Typos in the contract

Category	Severity	Location	Status
Coding Style	● Informational	Safemoon.sol: 746, 918	① Acknowledged

Description

There are several typos in the code and comments.

1. In the following code snippet, `tokensIntoLiquidity` should be `tokensIntoLiquidity`.

```
1 event SwapAndLiquify(  
2     uint256 tokensSwapped,  
3     uint256 ethReceived,  
4     uint256 tokensIntoLiquidity  
5 );
```

2. `recieve` should be `receive` and `swaping` should be `swapping` in the line of comment `//to recieve ETH from uniswapV2Router when swaping.`

Recommendation

We recommend correcting all typos in the contract.

Alleviation

The team acknowledged the finding, and given the deployed contract cannot be updated, decided to retain the code base unchanged.

RENOUNCE OWNERSHIP AND LOCK TOKEN

All tokens are locked before the presale.

We have renounced ownership before the presale.

RESOURCES & SOCIAL MEDIA



GUARDIANSALE OFFICAL SOCIAL ACCOUNTS

RESOURCES:



WWW.GUARDIANSALE.COM



GITHUB.COM/GUARDIANSALE

SOCIALS:



TWITTER.COM/GUARDIANSALE



FACEBOOK.COM/GUARDIANSALELAUNCHPAD



INSTAGRAM.COM/GUARDIANSALE



YOUTUBE.COM/GUARDIANSALE



TIKTOK.COM/@GUARDIANSALE



T.ME/GUARDIANSALEEN

guardiansale.com



@guardiansale