PALADIN
BLOCKCHAIN SECURITY

Smart Contract Security Assessment

For Thoreum Finance

21 June 2021

paladinsec.co info@paladinsec.co

Table of Contents

Disclaimer

1 Overview

This report has been prepared for Thoreum Finance. Paladin provides a user-centred examination of the smart contracts to look for vulnerabilities, logic errors or other issues from both an internal and external perspective.

1.1 Summary

Project Name	Thoreum Finance
URL	https://thoreum.finance
Platform	Binance Smart Chain
Language	Solidity

1.2 Contracts Assessed

Token	https://bscscan.com/address/0x580dE58c1BD593A43DaDcF0A739d504621817c05
Masterchef	https://bscscan.com/address/0xF4168CD3C00799bEeB9a88a6bF725eB84f5d41b7#code
Referral	https://bscscan.com/address/0xaB7EC1C6A86D12C9Ea64c817f421465cdDDF28F4#code

1.3 Issues

	Found	Resolved	Partially Resolved	Acknowledged (no change made)
🔴 **High Severity**	4	3	0	1
🟠 **Medium Severity**	6	4	0	1
🟡 **Low Severity**	3	0	1	2
🟣 **Informational**	5	2	1	2
Total	**18**	**9**	**2**	**6**

2 Findings

2.1 ThoreumToken

2.1.1 Token Overview

This contract is a fork of RFI, with some modifications of features made.

Address	https://bscscan.com/address/0x580dE58c1BD593A43DaDcF0A739d504621817c05
Token Supply	5 billion (5,000,000,000)
Decimal Places	18
Transfer Maximum Size	There is a maximum amount per transaction, originally set to 1,000,000 THOREUM. If a transfer greater than this amount is made, and the sender or recipient is not whitelisted from the antiwhale, the transaction will revert and not successfully complete.

Transfer Fees	Besides the reflection fee, there is also a liquidity fee. 2% is reflected, 8% is converted into BNB for future automatic buyback.
	
	When a token transfer is done, if the sender or recipient is not in `_isExcludedFromFee`, it will do the following to deduct fees calculated:
	
	`_takeLiquidity`
	`_reflectFee`
	
	If the to or from address for that transfer is whitelisted in `_isExcludedFromFee`, there will be no fees charged for that transaction.

SwapTokens	When the THOREUM contract's balance of THOREUM is greater than or equals to 50000 THOREUM, as defined by `minimumTokensBeforeSwap`. Everytime the total of this transfer tax exceeds 50,000 THOREUM, the contract will automatically sell 50,000 THOREUM for BNB and will be added to Thoreum's contract. This balance will be used for buybacks.
Buybacks	If buyback is enabled, a portion of the BNB in the token contract will be used to buy THOREUM, which will be sent to the burn address. This causes a deflationary supply of THOREUM over time. This is done when the following criteria are met in a transfer: • The to address of a transfer `isIncludedInThoreumLpList` • The contract's BNB balance is greater than or equals to `minimumBalanceRequired` • The transfer amount is greater than or equals to `minimumSellOrderAmount` The buy back amount is 1% of the contract's BNB balance or `buyBackUpperLimit`, whichever is smaller. THOREUM tokens bought back are sent to the burn address.

2.1.2 Privileged Roles

The owner of the THOREUM contract is an EOA. The following `onlyOwner` functions can be called by the owner address:

```
renounceOwnership()
transferOwnership(address newOwner)
excludeFromReward(address account)
includeInReward(address account)
setExcludedFromAntiWhale(address _account, bool _isExcludedOrNot)
setIncludeInThoreumLpList(address _address, bool _isIncludedOrNot)
setMinimumBalanceRequired(uint256 _newAmount)
setMinimumSellOrderAmount(uint256 _newAmount)
excludeFromFee(address account)
includeInFee(address account)
setTaxFeePercent(uint256 taxFee)
setLiquidityFeePercent(uint256 liquidityFee)
setMaxTxAmount(uint256 maxTxPercent)
setNumTokensSellToAddToLiquidity(uint256 _minimumTokensBeforeSwap)
setBuybackUpperLimit(uint256 buyBackLimit)
setSwapAndLiquifyEnabled(bool _enabled)
setBuyBackEnabled(bool _enabled)
prepareForPreSale()
afterPreSale()
```

2.1.3 Operations

The tokens for rewards should be transferred to masterchef before farming begins, and before `afterPresale` is called.

This contract contains novel functionality, specifically the buyback and sell transactions. This functionality must be tested thoroughly, specifically also with the setup where the pair and Masterchef are excluded as would be the case in production.

2.1.4 Issues & Recommendations

Issue	**`prepareForPreSale` can be called after presale has ended**
Severity	● HIGH SEVERITY
Description	`prepareForPreSale()` can be called by the owner again after afterPresale is called. This sets the `_maxTxAmount` to 0, thus disallowing all non-whitelisted addresses to make any transfers.
Recommendation(s)	There should be a check to ensure that `prepareForPreSale` and `afterPreSale` can only be called once.
Resolution	✓ RESOLVED `prepareForPreSale` and `afterPreSale` functions have been removed.

Issue	Lack of upper and lower limits for setting sensitive variables
Severity	● HIGH SEVERITY

Description	The owner can use the following setter functions to modify sensitive state variables without any limitations, and cause unexpected behavior to users of the token:
	
	`setTaxFeePercent(uint256 _taxFee)`
	
	The `taxFee` can be set to an extremely high value, up to 100% as there is no maximum limit check.
	
	The owner can use the following setter functions to modify sensitive state variables without any limitations, and cause unexpected behavior to users of the token:
	
	`setLiquidityFeePercent(uint256 _liquidityFee)`
	
	The `liquidityFee` can be set to an extremely high value, up to 100% as there is no maximum limit check.
	
	`setMaxTxAmount(uint256 maxTxAmount)`
	
	The `maxTxAmount` can be set to 0, disallowing any transfers except by whitelisted addresses. This value not only has to be non-zero, but have a reasonable minimum amount (e.g 0.001% of the total supply).

Recommendation(s)	Add reasonable upper and lower limit checks for the above mentioned setter functions. Additionally, if there is no need to change the values of these state variables, the owner of the token contract can be set to a proxy contract which does not contain any way to interact with these functions.
Resolution	✓ RESOLVED A maximum limit of 10% each for tax and liquidity fee is set and checked in the setter functions. `setMaxTxAmount` now has a minimum limit of 1 THOREUM.

Issue	Unused Functions
Severity	● INFORMATIONAL
Description	`transferToAddressETH` is an unused function. Deliver function can also be removed. `addLiquidity` is an unused function.
Recommendation(s)	Remove those functions if unused.
Resolution	● PARTIALLY RESOLVED `transferToAddressETH` and `addLiquidity` has been removed. deliver still remains in the code.

Issue	Contract deployer is excluded from fees and antiwhale in the constructor
Severity	● LOW SEVERITY
Description	When the contract is created, the owner address is added to the `_isExcludedFromFee` mapping in the constructor. As this address is an externally owned address, the address will be able to make transfers and trade THOREUM tokens without any fees.
Recommendation(s)	Include contract deployer address after presale is done.
Comments	● ACKNOWLEDGED Owner address will be included after official launch.

Issue	Farming contract needs to be excluded from antiwhale
Severity	INFORMATIONAL
Description	If a user who is farming Thoreum tokens has a pending harvest which exceeds the maxTxAmount limit, the user will be unable to harvest.
Recommendation(s)	Exclude farming contracts like masterchef from antiwhale.
Comments	ACKNOWLEDGED Thoreum will exclude the Masterchef address on deployment.

Issue	`includeInReward` causes a previously excluded address to gain reflection rewards at the expense of other addresses' reflection fees
Severity	MEDIUM SEVERITY
Description	As described in this blog article (https://perafinance.medium.com/safemoon-is-it-safe-though-a-detailed-explanation-of-frictionless-yield-bug-338710649846), a previously excluded address would gain some reflection fees at the expense of other addresses' reflection fees. This could cause a loss of some balances of users. This is done because `_rOwned` is not updated in `includeInFee`. This function is only callable by the contract's owner.
Recommendation(s)	The `_rOwned` of the address to be included should be updated based on the address' `_tOwned`.
Resolution	RESOLVED The `_rOwned` of the address to be included will update based on the address' `_tOwned`.

Issue	Lack of events for functions that change sensitive variables
Severity	● LOW SEVERITY
Description	Functions that affect the status of sensitive variables such as whitelists and fee percentages should emit events as notifications.
Recommendation(s)	Add events for the setter functions which modify sensitive state variables.

Resolution	● PARTIALLY RESOLVED
	
	Events have been added for `setTaxFeePercent` and `setLiquidityFeePercent`, but not for other setter functions like `setMaxTxAmount`.
	
	The following do NOT have event emits (full list, not just state variables):
	
	`setExcludedFromAntiWhale`
	`setIncludeInThoreumLpList`
	`setMinimumBalanceRequired`
	`setMinimumSellOrderAmount`
	`setNumTokensSellToAddToLiquidity`
	`setBuybackUpperLimit`
	`setDevAddress`
	`setFeeAddress`
	`setThoreumReferral`
	`setReferralCommissionRate`

Issue	Sensitive state variable modifying functions can be changed instantaneously
Severity	HIGH SEVERITY
Description	Owner is not a timelock contract and can make sensitive variable changes without delay. All `onlyOwner` functions can be called without any delay, causing changes to the behavior of the contract.
Recommendation(s)	Set the owner as a timelock contract to allow some delay before sensitive state changes are done.
Comments	ACKNOWLEDGED Thoreum plans to have a Timelock for the Masterchef in time, but not for the token yet as they’re still monitoring the early stages of the project.

Issue	Masterchef address and LP contract address has to be excludeFromReward
Severity	MEDIUM SEVERITY
Description	As the Masterchef will be holding all the reward tokens as well as staked tokens, rewarded tokens will not be accounted for in the harvests or user balances. The LP contract can have tokens skimmed out if the balance increases due to reflection rewards.
Recommendation(s)	Add Masterchef and LP address to `excludeFromReward`
Comments	ACKNOWLEDGED Thoreum will exclude Masterchef address and LP contract address on deployment.

Issue	Buyback is done before user’s selling is done (tokenomics)
Severity	● INFORMATIONAL
Description	The buyback is done before the actual token transfer by the user is completed when selling THOREUM to the LP contract. This could provide the user with a better selling price, resulting in tokenomical waste.
Recommendation(s)	The buyback will always be done before the user’s sell if it is in the transfer, so to change this behavior, the tokenomics has to be reconsidered.
Comments	● ACKNOWLEDGED Acknowledged, but no change to be made.

Issue	LP pair should not be removable from `_includeInThoreumLpList`
Severity	● MEDIUM SEVERITY
Description	If the Thoreum/WBNB LP pair is removed intentionally or by accident from the `_includeInThoreumLpList`, it could result in vicious cycles of the addLiquidity functionality.
Recommendation(s)	Within the setIncludeInThoreumLpList function, add the following requirement: `require(_address != ouniswapV2Pair || _isIncludedOrNot == true, "Cannot disable uniswapV2Pair”);`
Resolution	✓ RESOLVED The proposed `require` statement has been added.

Issue	Automated buybacks may not be sufficient to use all accumulated BNB (tokenomics)
Severity	● LOW SEVERITY
Description	Currently, if the user transfers a large enough amount, a single transaction will trigger both a sell (the 8% of the transfer that is sold off) and a buyback. These two transactions are exact opposites and thus the only result is that money is wasted in PancakeSwap transaction fees.
Recommendation(s)	Consider this scenario and how often it occurs. If it occurs sufficiently often, consider using else if instead of else. Another solution would be to not do the buyback in the transfer itself, as discussed in the previous tokonomics issue.
Comments	● ACKNOWLEDGED Acknowledged, but no change to be made.

Issue	Doing both the sell and buyback in a single transfer is wasteful (tokenomics)
Severity	● INFORMATIONAL
Description	Current buybacks are only done in sell transactions, and may not be sufficient to provide the equilibrium of the `SwapTokens`.
Recommendation(s)	Add an `onlyOwner` function that allows the same buyback functionality to allow manual buy backs if required.
Resolution	✓ RESOLVED A manual buyback function only callable by the owner has been added.

2.2 Masterchef

Contract address: `0xF4168CD3C00799bEeB9a88a6bF725eB84f5d41b7`

2.2.1 Privileged Roles

There are 2 privileged roles: Owner and Operator.

The owner of the Masterchef contract is `0x383d56631aae8341ffe40a14b22f09e49d34004b`, which is an EOA.

The owner is able to call the following owner-only functions:

- `add`
- `set`
- `renounceOwnership`
- `transferOwnership`
- `setThoreumReferral`

The operator of the Masterchef contract is `0x383d56631aae8341ffe40a14b22f09e49d34004b`, which is an EOA.

The operator is able to call the following operator-only functions:

- `transferOperator`
- `setReferralCommissionRate`
- `updateAllocPoint`
- `emergencyWithdrawAllRewards`
- `emergencyWithdrawRewards`

2.2.2 Tokenomics

The specified emission per block is 1000 THOREUM. This value has been initialized in the constructor upon contract creation. `updateEmissionRate` can be called to change the emission per block to any amount.

Token rewards are pre-minted and deposited into the Masterchef contract, unlike the usual Masterchef behavior which mints tokens when `updatePool` is called.

10% of `thoreumReward` is calculated when `updatePool` is called, and transferred to the `devAddress`.

The pre-minted amount accounts for the 10% dev rewards and 3% referral commission.

2.2.3 Deposits and Withdrawals

Handling Fee on Transfer Tokens

As there are pools with tokens that have a fee on transfer, the actual amount used for deposit calculations is the following:

```
_amount = afterDeposit - beforeDeposit
```

This ensures that the token amount received by the Masterchef contract is used as the deposit amount, not the amount specified in the deposit function call.

Deposit Fees

Pools can have a deposit fee of up to 10%. This deposit fee is set at the time of adding a pool, and can be modified from any value between 0% to 10%.

When depositing into a pool with a deposit fee, the fee percentage will be subtracted from the deposit amount and sent to the `feeAddress`. The remaining amount after fee deduction will be added as the user's deposited amount into the pool.

The nonReentrant modifier is used for `deposit`, `withdraw`, and `emergencyWithdraw`.

Cached Balances

Unlike the original MasterChef which uses the balance of tokens in the contract as the `lpSupply`, this contract uses cached balances which are updated on each `deposit`, `withdraw` and `emergencyWithdraw`.

Besides the `totalLp` for each pool, there is also a state variable `totalThoreumInPools` used to keep track of the amount of staked Thoreum. This amount is used to ensure that the reward token and staked token balances are kept track of separately.

When a deposit is done, the `totalLp` for the pool is increased by the deposit amount after deducting the deposit fee. Also, if the pool's token is Thoreum, `totalThoreumInPools` is increased by the deposit amount.

When `withdraw` or `emergencyWithdraw` is called, the `totalLp` for the pool is decreased by the withdrawal amount. Also, if the pool's token is Thoreum, `totalThoreumInPools` is decreased by the withdrawal amount.

Harvests

When a user has a pending reward balance and `deposit` or `withdraw` is called, `payOrLockupPendingThoreum` is called to check if the user is eligible for harvesting rewards. This is based on the user's `nextHarvestUntil`.

If the user does not have a `nextHarvestUntil` set and the `startBlock` has passed, it is set to the current block + the pool's `harvestInterval`. Otherwise, it will check if the user's `nextHarvestUntil` has passed. If passed, it will transfer all pending rewards to the user, and update `nextHarvestUntil` to the current block + `harvestInterval`. Else, the user's pending rewards will be updated.

`harvestInterval` is different for each pool, and can be up to 14 days.

Referral Commission

When a user deposits and specifies a valid referrer (a non-zero address that is not the user's address), the referrer address for the user will be set. This address will receive a commission rate, initially set at 3%. This commission rate can be set up to a max of 10%.

When a user with a referrer address does a harvest, the commission rate will be used to calculate the commission, which will be transferred to the referrer.

Referrer logic is dependent on an external `ThoreumReferral` contract, which can be changed by the owner. If the contract address is changed to an invalid address that is not a contract, or does not support the ABI, calls that depend on `ThoreumReferral` will fail. This includes adding a referrer during deposit and harvesting.

Others

The operator has the ability to withdraw Thoreum tokens, but only up to the balance of the total Thoreum balance - `totalThoreumInPools`. This ensures that deposited Thoreum tokens can only be removed by the users that deposited them.

2.2.4 Issues & Recommendations

Issue	**Privileged functions can be called without delay**
Severity	● MEDIUM SEVERITY
Description	There are privileged functions in the Masterchef contract which can be called by the owner and operator. As these roles are held by EOAs, the privileged functions can be called instantaneously.
Recommendation(s)	Privileged roles should be set to a timelock contract with a reasonable delay (e.g. 12 hours).
Resolution	N/A

Issue	**Lack of maximum limit for** `updateEmissionRate`
Severity	● MEDIUM SEVERITY
Description	`updateEmissionRate` can be called by the owner to set the emission per block to an arbitrary amount.
Recommendation(s)	`updateEmissionRate` should have a hard cap to prevent setting it to an amount that is excessively high.
Resolution	✓ RESOLVED There is an added check for a maximum of 2000 THOREUM per block for the emissions in `updateEmissionRate`.

Issue	Differing code for Thoreum transfers
Severity	● INFORMATIONAL
Description	When Thoreum is transferred for rewards, there is different code used. In `payOrLockupPendingThoreum`, `safeThoreumTransfer` is used to transfer out Thoreum rewards. However, in other functions (e.g. `payReferralCommission`) that deal with transfers from the Thoreum rewards balance, `canTransferReward` is used.
Recommendation(s)	The code for Thoreum transfers should be standardized if the same behavior is expected.
Resolution	✓ RESOLVED All Thoreum transfers have been changed to use `safeThoreumTransfer`.

Issue	`updatePoolWithoutSendingReward` **should be internal instead of public**
Severity	● HIGH SEVERITY
Description	Currently, anyone can call the function to update a pool without sending the dev rewards.
Recommendation(s)	Set `updatePoolWithoutSendingRewards` to internal.
Resolution	✓ RESOLVED `updatePoolWithoutSendingRewards` has been removed.

2.3 ThoreumReferral

Contract address: `0xaB7EC1C6A86D12C9Ea64c817f421465cdDDF28F4`

2.3.1 Privileged Roles

There are 2 privileged roles: Owner and Operator.

The owner of the `ThoreumReferral` contract is `0x6a963573b9a7aaef9cb9eee747e1b6b2c21b0520`, which is an EOA.

The owner is able to call the following owner-only functions:

- `updateOperator`
- `drainBEP20Token`
- `renounceOwnership`
- `transferOwnership`

There can be more than 1 operator address, which can be set by the owner. Under intended circumstances, there should only be 1 operator, the Masterchef address.

The operator is able to call the following operator-only functions:

- `recordReferral`
- `recordReferralCommission`

Others

If an address does not have a referrer address, the operator can set an arbitrary user address. If an address already has a referrer address set, it cannot be changed.

`drainBEP20Token` allows the owner to transfer any tokens from the Referral contract. Under normal circumstances, there should be no tokens contained in this contract. Tokens could be mistakenly sent to this address, so this function allows the owner to help retrieve those tokens.

2.3.2 Issues & Recommendations

Issue	**`ThoreumReferral` can have multiple operators**
Severity	● MEDIUM SEVERITY
Description	Having other operators that are non Masterchef can allow arbitrary setting of referral addresses for other users. There should only be 1 operator address in `ThoreumReferral`, and that should only be the Masterchef address.
Recommendation(s)	Allow only a single address to be set as the operator.
Resolution	✓ RESOLVED The code has been changed to use `_masterOperator`, which is a single address, for the operator.

PALADIN

BLOCKCHAIN SECURITY