Smart Contract Security Audit

Audit details:

Audited project:	SafeStar
Deployer address	0x0fb3070f578f8205ba788528b1e3f5cd2fde2126
Client contacts:	@daddy_elon
Blockchain:	Binance Smart Chain
Project website:	https://safestar.me

April, 2021
TechRate

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report. In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions. We go into more detail on this in the below disclaimer below – please make sure to read it in full.

DISCLAIMER: By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and TechRate and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives) (TechRate) owe no duty of care towards you or any other person, nor does TechRate make any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and TechRate hereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report. Except and only to the extent that it is prohibited by law, TechRate hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against TechRate, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report.

The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been reviewed.

Background

TechRate was commissioned by SafeStar to perform an audit of smart contracts:

- *https://bscscan.com/address/0x3c00f8fcc8791fa78daa4a480095ec7d475781e2#code*

The purpose of the audit was to achieve the following:

- Ensure that the smart contract functions as intended.
- Identify potential security issues with the smart contract.

The information in this report should be used to understand the risk exposure of the smart contract, and as a guide to improve the security posture of the smart contract by remediating the issues that were identified.

Contracts details

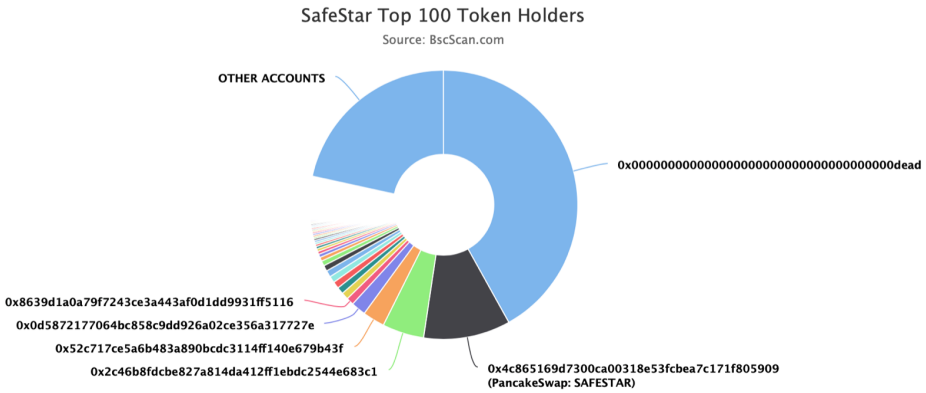
Token contract details for 07.04.2021.

Contract name:	SafeStar
Compiler version:	v0.6.12+commit.27d51765
Contract address:	0x3c00f8fcc8791fa78daa4a480095ec7d475781e2
Total supply:	1_000_000_000_000_000
Token ticker:	SAFESTAR
Decimals:	9
Token holders:	32947
Transactions count:	96859
Top 100 holders dominance:	78.4%
Contract deployer address:	0x0fb3070f578f8205ba788528b1e3f5cd2fde2126
Contract's current owner address:	0x0fb3070f578f8205ba788528b1e3f5cd2fde2126
Current liquidity fee:	5 percent
Current tax fee:	5 percent
Total fees:	185_813_752_360_408_077_791_926
Uniswap V2 pair:	0x4c865169d7300ca00318e53fcbea7c171f805909
Uniswap V2 router:	0x05ff2b0db69458a0750badebc4f9e13add608c7f
Max transaction amount:	10_000_000_000_000_000_000_000_000_000_000_000_000_000_000_000_000
Deployed at transaction:	0x533139072cabfe5484b6f532c2c5dea641ae6478c450a4466b929ca97738f3f8

SafeStar top 100 token distribution

The top 100 holders collectively own 78.41% (784,079,054,786,813.00 Tokens) of SafeStar

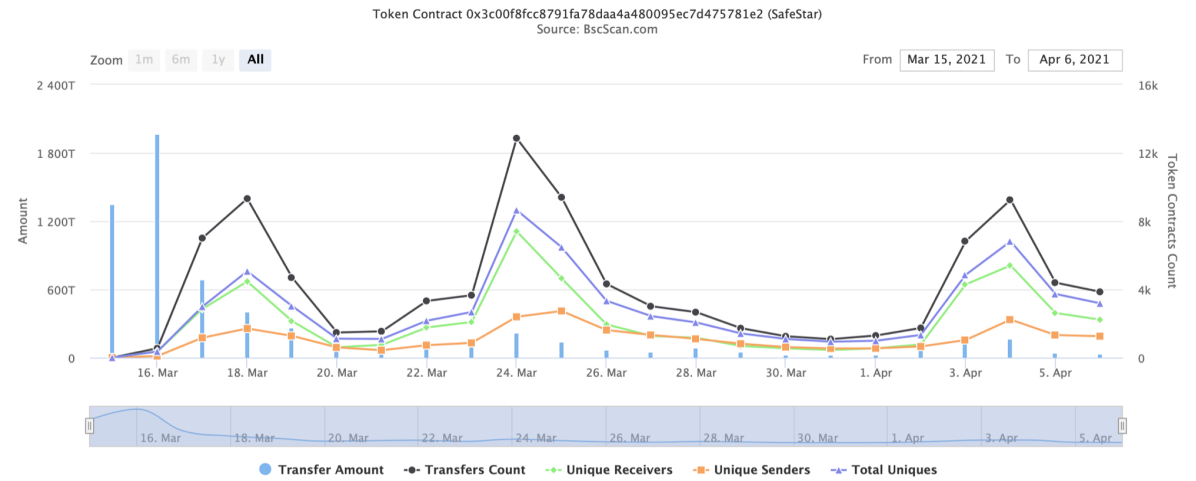
Token Total Supply: 1,000,000,000,000,000.00 Token | Total Token Holders: 32,948



SafeStar contract interaction details

Time Series: Token Contract Overview

Mon 15, Mar 2021 - Tue 6, Apr 2021



Contract functions details

Function	Return value	Who can call
name()	string	public
symbol()	string	public
decimals()	uint8	public
totalSupply()	uint256	public
balanceOf(address)	uint256	public
transfer(address, uint256)	bool	public
allowance(address, address)	uint256	public
approve(address, uint256)	bool	public
transferfrom(address, address, uint256)	bool	public
increaseAllowance(address, uint256)	bool	public
decreaseAllowance(address, uint256)	bool	public
isExcludedFromReward(address)	bool	public
totalFees()	uint256	public
deliver()	void	public
excludeFromReward(address)	void	owner
includeInReward(address)	void	owner
excludeFromFee(address)	void	owner
includeInFee(address)	void	owner
isExcludedFromFee(address)	bool	public
setTaxFeePercent(uint256)	void	owner
setLiquidityFeePercent(uint256)	void	owner
setMaxTxPercent(uint256)	void	owner
setSwapAndLiquifyEnabled(bool)	void	owner

Issues Checking Status

№	Issue description.	Checking status
1	Compiler errors.	Passed
2	Race conditions and Reentrancy. Cross-function race conditions.	Passed
3	Possible delays in data delivery.	Passed
4	Oracle calls.	Passed
5	Front running.	Passed
6	Timestamp dependence.	Passed
7	Integer Overflow and Underflow.	Passed
8	DoS with Revert.	Passed
9	DoS with block gas limit.	Low severity issues only
10	Methods execution permissions.	Passed
11	Economy model of the contract.	Passed
12	The impact of the exchange rate on the logic.	Passed
13	Private user data leaks.	Passed
14	Malicious Event log.	Passed
15	Scoping and Declarations.	Passed
16	Uninitialized storage pointers.	Passed
17	Arithmetic accuracy.	Passed
18	Design Logic.	Passed
19	Cross-function race conditions.	Passed
20	Safe Open Zeppelin contracts implementation and usage.	Passed
21	Fallback function security.	Passed

Security Issues

High Severity Issues

No high severity issues found.

Medium Severity Issues

No medium severity issues found.

Low Severity Issues

1. Out of gas

Issue:

- The function `includeInReward()` uses the loop to find and remove addresses from the `_isExcluded` list. Function will be aborted with `OUT_OF_GAS` exception if there will be a long excluded addresses list.

```
function includeInReward(address account) external onlyOwner() {
    require(_isExcluded[account], "Account is already excluded");
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_excluded[i] == account) {
            _excluded[i] = _excluded[_excluded.length - 1];
            _tOwned[account] = 0;
            _isExcluded[account] = false;
            _excluded.pop();
            break;
        }
    }
}
```

- The function `_getCurrentSupply` also uses the loop for evaluating total supply. It also could be aborted with `OUT_OF_GAS` exception if there will be a long excluded addresses list.

```
function _getCurrentSupply() private view returns(uint256, uint256) {
    uint256 rSupply = _rTotal;
    uint256 tSupply = _tTotal;
    for (uint256 i = 0; i < _excluded.length; i++) {
        if (_rOwned[_excluded[i]] > rSupply || _tOwned[_excluded[i]] > tSupply) return (_rTotal, _tTotal);
        rSupply = rSupply.sub(_rOwned[_excluded[i]]);
        tSupply = tSupply.sub(_tOwned[_excluded[i]]);
    }
    if (rSupply < _rTotal.div(_tTotal)) return (_rTotal, _tTotal);
    return (rSupply, tSupply);
}
```

Recommendation:

Use EnumerableSet instead of array or do not use long arrays.

Owner privileges

1. Owner privileges

- ❏ Owner can change the tax fee.

```
function setTaxFeePercent(uint256 taxFee) external onlyOwner() {
    _taxFee = taxFee;
}
```

- ❏ Owner can change the liquidity fee.

```
function setLiquidityFeePercent(uint256 liquidityFee) external onlyOwner() {
    _liquidityFee = liquidityFee;
}
```

- ❏ Owner can change the maximum transaction amount.

```
function setMaxTxPercent(uint256 maxTxPercent) external onlyOwner() {
    _maxTxAmount = _tTotal.mul(maxTxPercent).div(
        10**2
    );
}
```

- ❏ Owner can exclude from the fee.

```
    function excludeFromFee(address account) public onlyOwner {
    _isExcludedFromFee[account] = true;
}
```

Conclusion

Smart contracts do not contain any high severity issues! However, smart contracts contain owner privileges.

Techrate note:

Please check the disclaimer above and note, the audit makes no statements or warranties on business model, investment attractiveness or code sustainability. The report is provided for the only contract mentioned in the report and does not include any other potential contracts deployed by Owner.