
Goatcoin

Security Code Review

<https://twitter.com/VidarTheAuditor> - 1 March 2021



Overview

Project Summary

Project Name	Goatcoin
Description	BSC token
Platform	BSC, Solidity
Contracts	https://bscscan.com/address/0x7c67dccb04b67d4666fd97b2a00bb6d9b8d82e3f#code https://bscscan.com/address/0x7044326135a8f416dd4a1d48bc47f808879ed425#code

Executive Summary

Binance Smart Chain contracts were provided.

We have run extensive static analysis of the codebase as well as standard security assessment utilising industry approved tools.

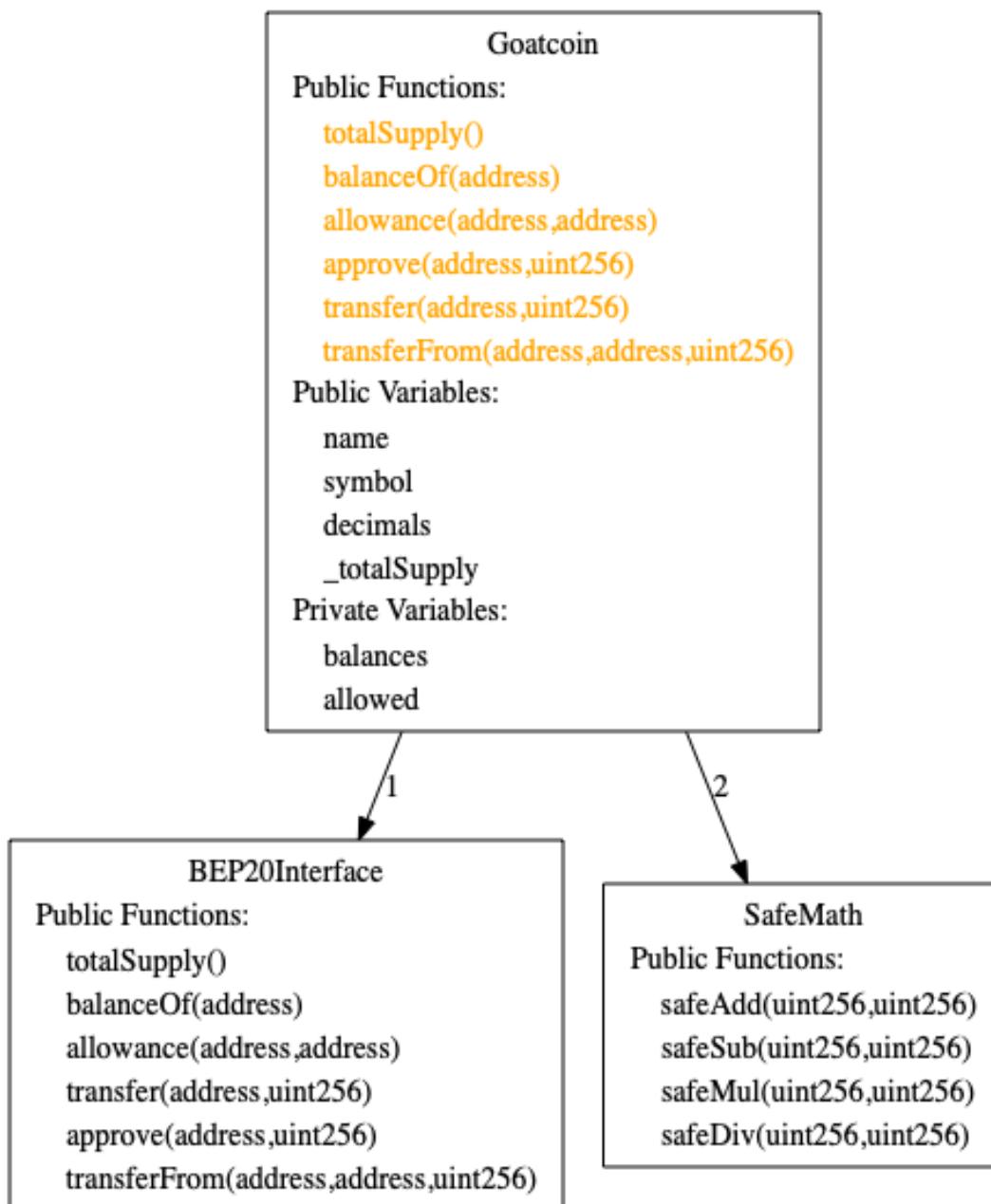
We have not found any critical vulnerabilities arising from a third party involvement.

Some recommendations were issued regarding the ownership structure of the currently deployed contract (more info can be found in [Deployment](#) section).

Disclaimer: The analysis did not include any tokenomics analysis (e.g. APY rates etc).

Architecture & Standards

Please find below the calling architecture of the reviewed contracts.



Goatcoin contracts are fully BEP-20 compatible.

```
# Check Goatcoin

## Check functions
[✓] totalSupply() is present
    [✓] totalSupply() -> () (correct return value)
    [✓] totalSupply() is view
[✓] balanceOf(address) is present
    [✓] balanceOf(address) -> () (correct return value)
    [✓] balanceOf(address) is view
[✓] transfer(address,uint256) is present
    [✓] transfer(address,uint256) -> () (correct return value)
    [✓] Transfer(address,address,uint256) is emitted
[✓] transferFrom(address,address,uint256) is present
    [✓] transferFrom(address,address,uint256) -> () (correct return value)
    [✓] Transfer(address,address,uint256) is emitted
[✓] approve(address,uint256) is present
    [✓] approve(address,uint256) -> () (correct return value)
    [✓] Approval(address,address,uint256) is emitted
[✓] allowance(address,address) is present
    [✓] allowance(address,address) -> () (correct return value)
    [✓] allowance(address,address) is view
[✓] name() is present
    [✓] name() -> () (correct return value)
    [✓] name() is view
[✓] symbol() is present
    [✓] symbol() -> () (correct return value)
    [✓] symbol() is view
[✓] decimals() is present
    [✓] decimals() -> () (correct return value)
    [✓] decimals() is view

## Check events
[✓] Transfer(address,address,uint256) is present
    [✓] parameter 0 is indexed
    [✓] parameter 1 is indexed
[✓] Approval(address,address,uint256) is present
    [✓] parameter 0 is indexed
    [✓] parameter 1 is indexed

[ ] Goatcoin is not protected for the ERC20 approval race condition
```

Findings

Number of contracts: 6 (including inherited ones)

Use: SafeMath

Name	# functions	ERCS	ERC20 info	Complex code	Features
Goatcoin	17	ERC20	No Minting Approve Race Cond.	No	

Name	# functions	ERCS	ERC20 info	Complex code	Features
Math	3			No	
SafeMath	8			No	
IERC20	6	ERC20	No Minting Approve Race Cond.	No	
Address	3			No	Send ETH Assembly
SafeERC20	6			No	Tokens interaction
StakingRewards	30			No	Tokens interaction

For more info see [Deployment](#) section.

Static Analysis Findings

High issues: None

Medium issues: None

Low/Informational issues:

Old Solidity version:

`Pragma version^0.5.0 (goatcoin.sol#5) allows old versions`

[Recommendation] Use newer version of Solidity.

Dynamic Tests

We have run fuzzing / property-based testing of Solidity smart contracts. It was using sophisticated grammar-based fuzzing campaigns based on a contract ABI to falsify user-defined predicates or Solidity assertions.

There were also dynamic tests run on EVM byte code to detect common vulnerabilities including integer underflows, owner-overwrite-to-Ether-withdrawal, and others.

The analysis was completed successfully. No issues were detected.

No Issues were found.

Manual Checks

Goatcoin is fully BEP20 compatible contract.

StakingRewards is a fork of <https://github.com/Synthetixio/synthetix/blob/develop/contracts/StakingRewards.sol>

Function `notifyRewards` (that sets the rewards rate) is controlled by deployer address.

```
712 |     function notifyRewardAmount(uint256 reward↑) external onlyRewardsDistribution updateReward(address(0)) {
```

Please see [Deployment](#) section for recommendations.

Automatic Tests

The project lacks any automatic testing and tests scripts. We did not run any functional tests provided by the team, due to lack of such scripts provided. Hence the full business logic functionality was not tested.

[Recommendation]: Create comprehensive test cases and implement them as scripts or mocha tests using the hardhat infrastructure.

[Disclaimer] There were no tests conducted testing full system functionality due to lack of proper test cases and/or test scripts.

Deployment & Contract Ownership

The contracts are currently deployed on BSC Mainnet:

- <https://bscscan.com/address/0x7c67dccb04b67d4666fd97b2a00bb6d9b8d82e3f#code>
- <https://bscscan.com/address/0x7044326135a8f416dd4a1d48bc47f808879ed425#code>

Recommendations:

- Deploy a Timelock contract to control only RewardsDistribution role of the StakingRewards

Disclaimer

The information appearing in this report is for general purposes only and is not intended to provide any legal security guarantees to any individual or entity. As one review is not enough to provide 100% security against any attacks or bugs, it is **strongly advisable** to conduct **more reviews or/and audits**.

The report does not provide personalised investment advice or recommendations, especially does not provide advice to conclude any transactions and it does not provide investment, financial, legal or tax advice.

We are not responsible or liable for any loss which results from the report.

The report should not be considered as an investment advice.