QuillAudits

Audit Report
July, 2022

For

NUGEN COIN

Table of Content

Executive Summary

Project Name	Nugen Token
Overview	NUGEN is the indigenous coin of the NUGEN platform. It is a BEP-20 standard token built on the robust Binance Smart Chain network. NUGEN will serve as a collateralized token that facilitates cryptocurrency payments on the network.
Timeline	30 June, 2022 to 2 July, 2022
Method	Manual Review, Functional Testing, Automated Testing etc.
Scope of Audit	The scope of this audit was to analyse Nugen Token codebase for quality, security, and correctness.
Codebase	*https://testnet.bscscan.com/address/0x5659E4deef7537E57c475866E7F1FC8B1A1DEB5e#code*
Fixed In	*https://testnet.bscscan.com/address/0xad9648b6dc8cEeB132D32cD7C6749f96329d01d4#code*

BSC Mainnet Address 0xA62a8a65013F789367bE37e5C0afc83445F77Cc2

High Medium Low Informational

	High	Medium	Low	Informational
Open Issues	0	0	0	0
Acknowledged Issues	0	0	2	0
Partially Resolved Issues	0	0	0	0
Resolved Issues	0	0	1	2

Types of Severities

High

A high severity issue or vulnerability means that your smart contract can be exploited. Issues on this level are critical to the smart contract's performance or functionality, and we recommend these issues be fixed before moving to a live environment.

Medium

The issues marked as medium severity usually arise because of errors and deficiencies in the smart contract code. Issues on this level could potentially bring problems, and they should still be fixed.

Low

Low-level severity issues can cause minor impact and or are just warnings that can remain unfixed for now. It would be better to fix these issues at some point in the future.

Informational

These are severity issues that indicate an improvement request, a general question, a cosmetic or documentation error, or a request for information. There is low-to-no impact.

Types of Issues

Open

Security vulnerabilities identified that must be resolved and are currently unresolved.

Resolved

These are the issues identified in the initial audit and have been successfully fixed.

Acknowledged

Vulnerabilities which have been acknowledged but are yet to be resolved.

Partially Resolved

Considerable efforts have been invested to reduce the risk/impact of the security issue, but are not completely resolved.

Checked Vulnerabilities

- Re-entrancy
- Timestamp Dependence
- Gas Limit and Loops
- Exception Disorder
- Gasless Send
- Use of tx.origin
- Compiler version not fixed
- Address hardcoded
- Divide before multiply
- Integer overflow/underflow
- Dangerous strict equalities
- Tautology or contradiction
- Return values of low-level calls
- Missing Zero Address Validation
- Private modifier
- Revert/require functions
- Using block.timestamp
- Multiple Sends
- Using SHA3
- Using suicide
- Using throw
- Using inline assembly

Techniques and Methods

Throughout the audit of smart contract, care was taken to ensure:

- The overall quality of code.
- Use of best practices.
- Code documentation and comments match logic and expected behaviour.
- Token distribution and calculations are as per the intended behaviour mentioned in the whitepaper.
- Implementation of ERC-20 token standards.
- Efficient use of gas.
- Code is safe from re-entrancy and other vulnerabilities.

The following techniques, methods and tools were used to review all the smart contracts.

Structural Analysis

In this step, we have analysed the design patterns and structure of smart contracts. A thorough check was done to ensure the smart contract is structured in a way that will not result in future problems.

Static Analysis

Static analysis of smart contracts was done to identify contract vulnerabilities. In this step, a series of automated tools are used to test the security of smart contracts.

Code Review / Manual Analysis

Manual analysis or review of code was done to identify new vulnerabilities or verify the vulnerabilities found during the static analysis. Contracts were completely manually analysed, their logic was checked and compared with the one described in the whitepaper. Besides, the results of the automated analysis were manually verified.

Gas Consumption

In this step, we have checked the behaviour of smart contracts in production. Checks were done to know how much gas gets consumed and the possibilities of optimization of code to reduce gas consumption.

Tools and Platforms used for Audit

Remix IDE, Truffle, Truffle Team, Solhint, Mythril, Slither, Solidity statistic analysis.

Manual Testing

A. Contract - Nugen Token

High Severity Issues

No issues found

Medium Severity Issues

No issues found

Low Severity Issues

1. Centralization Risk

Function - setMaxTxPBEPent(), addBlacklist

Description
The max transfer percent and blacklist functionality of this contract poses a centralization risk because if the owner wants then they could simply halt all the transfers by setting the max transfer percent as 0. Moreover, the owner can blacklist or whitelist any account they wish.

However, it also poses a risk in a scenario where the owner loses their private key or renounces the ownership then certain functions could never be called in the contract. For example, if the private key is lost then no account could ever be removed from the blacklist or the max transfer percent could never be updated.

There is also the risk of private keys being stolen and if it happens then the owner could lose all the control of the contract.

Remediation
Blacklisting should be done via proper verification of the account to check for bots if necessary and Instead of a single owner of the contract, there can be multiple accounts that can be added and used. Lost keys or compromised accounts can be blacklisted/blocked.

Status

Acknowledged

Client's Comments: Blacklisting will be done only after proper verification of the account to check for bots from owner side.
We are using proper governance outside contract before blacklisting an account.
Lost key is not considering an issue as of now, it will happen only if we have some negligence in storing owner keys.

Auditor's Comments: "The issue we were posing out as the heading suggest is a "risk" and that risk of losing the keys will always be there in a centralized system."

2. Non-Compliant with BEP-20 standard

Description
BEP-20 contracts must be compliant to the BEP-20 standard and the Tokens which don't implement this method will never flow across the Binance Chain and Binance Smart Chain.
The given contract doesn't have a "getOwner" named function which is required in the BEP-20 method.
For reference: *https://github.com/bnb-chain/BEPs/blob/master/BEP20.md#5116-getowner*

5.1.1.6 getOwner

```
function getOwner() external view returns (address);
```

- Returns the bep20 token owner which is necessary for binding with bep2 token.
- NOTE - This is an extended method of EIP20. Tokens which don't implement this method will never flow across the Binance Chain and Binance Smart Chain.

Remediation
Add the "getOwner" function or rename the existing "owner" function

Status
Resolved

3. Transfer Ownership should be a two step process

Function - transferOwnership()

Description
The transfer of ownership is crucial functionality in token contracts and it should be done by a two step process so that the owner doesn't mistakenly transfer ownership to a false address.

Remediation
There should be two functions involved while transferring the ownership:
1. First function should take the proposed new owner's address as input and emit an event about the "proposed owner".
2. Then the proposed owner address should call the second function in order to transfer the ownership to the "proposed owner's" address.

Status

Acknowledged

Client's Comments: This may cause ownership open till the proposed owner claim ownership. No proper tested code available in openzeppelin or any other ERC 20 standards related to this till now.

Auditor's Comments: "The remediation we provided was actually one way of implementing the two way process and we would like to specify that Openzeppelin is not a standard, it's a library. However, we would still recommend a two way process because the transferOwnership function by openzeppelin poses the risk of losing the ownership and a two-way process reduce that risk by a big margin."

Informational Issues

4. Missing events

Description
Missing events don't pose any security risk. However, it makes it difficult to track chain events. It is advisable to emit an event whenever a significant action takes place on contract.

Remediation
Consider adding events to the following functions:
- addBlacklist
- removeBlacklist
- setMaxTxPBEPent
- setAntibotPaused

Status
Resolved

5. Unlocked pragma (pragma solidity ^0.8.0)

Description
Contracts should be deployed with the same compiler version and flags that they have been tested with thoroughly. Locking the pragma helps to ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

Remediation
Here all the in-scope contracts have an unlocked pragma, it is recommended to lock the same. Moreover, we strongly suggest not to use experimental Solidity features (e.g., pragma experimental ABIEncoderV2) or third-party unaudited libraries. If necessary, refactor the current code base to only use stable features.

Status
Resolved

6. General Recommendation

Description
In our audit we have concluded that the max transfer percent check can be bypassed by a whale in a scenario where the whale wants to sell their 50% of tokens and the max transfer is set to 10% then the whale could simply sell it via 5-6 different accounts at the same time.

Status

Acknowledged

Client's Comments: We have a blacklist option to freeze transfer on that wallet.

Auditor's Comments: "Whales can simply distribute their funds into multiple accounts and it'll be near to impossible to predict whether they will be selling or not. Hence, freezing transfer on a wallet without knowing the intention isn't feasible."

Functional Tests

- Should be able to mint and transfer the total supply to the owner while deployment.
- Should be able to burn tokens by owner account only
- Only owner should be able to transfer ownership
- Should be able to transfer tokens
- Only owner should be able to blacklist/whitelist addresses
- Only owner should be able to set max transfer percent
- Only owner should be able to renounce ownership
- Should revert if the transfer amount exceeds balance
- Should revert if the transfer amount exceeds allowance of non-owner accounts
- Should revert if the transfer address is a zero address.
- Should revert if the transfer amount is more than max transfer percent.
- Should set and get state of AntibotPaused
- Should set and get state of setMaxtxPBEPent

Automated Tests

No major issues were found. Some false positive errors were reported by the tools. All the other issues have been categorized above according to their level of severity.

Closing Summary

In this report, we have considered the security of the Nugen Token. We performed our audit according to the procedure described above.

Some issues of Low and Informational severity were found, Some suggestions and best practices are also provided in order to improve the code quality and security posture. At the End, Nugen Team fixed some Issues and Acknowledged others.

Disclaimer

QuillAudits smart contract audit is not a security warranty, investment advice, or an endorsement of the Nugen Token Platform. This audit does not provide a security or correctness guarantee of the audited smart contracts.

The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them. Securing smart contracts is a multistep process. One audit cannot be considered enough. We recommend that the Nugen Token Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

About QuillAudits

QuillAudits is a secure smart contracts audit platform designed by QuillHash Technologies. We are a team of dedicated blockchain security experts and smart contract auditors determined to ensure that Smart Contract-based Web3 projects can avail the latest and best security solutions to operate in a trustworthy and risk-free ecosystem.

500+
Audits Completed

$15B
Secured

500K
Lines of Code Audited

Follow Our Journey

Audit Report
July, 2022

For

NUGEN COIN

QuillAudits

Canada, India, Singapore, United Kingdom

audits.quillhash.com

audits@quillhash.com