CERTIK

Security Assessment

Nanobyte

Nov 16th, 2021

Table of Contents

Summary

This report has been prepared for Nanobyte to discover issues and vulnerabilities in the source code of the Nanobyte project as well as any contract dependencies that were not part of an officially recognized library. A comprehensive examination has been performed, utilizing Static Analysis and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line-by-line manual review of the entire codebase by industry experts.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards and industry practices. We suggest recommendations that could better serve the project from the security perspective:

- Enhance general coding practices for better structures of source codes;
- Add enough unit tests to cover the possible use cases;
- Provide more comments per each function for readability, especially contracts that are verified in public;
- Provide more transparency on privileged activities once the protocol is live.

Overview

Project Summary

Project Name	Nanobyte
Platform	bsc
Language	Solidity
Codebase	https://github.com/nanobytetoken/nbt/tree/f0751be848bb277563a1910fd235016886c41b26
Commit	• d134255be8d85724b0f0b4412e0f9bc0e1ab40bc • 3be3517a0529c1f972e4868eee2abd17883a3adf • e251901520d70fa4f4ff15248ab9f3761d7015fa - e3b3651cefb26614b74a2a9d541eddc401a2ffd8 • 99c3294b14278564caa2ddae1083cf8d1304f08e • aa4b370a12d60be20fad0718deccd7958c5bc866

Audit Summary

Delivery Date	Nov 16, 2021
Audit Methodology	Static Analysis, Manual Review
Key Components	NanoByteToken

Vulnerability Summary

Vulnerability Level	Total	ⓘ Pending	⊗ Declined	ⓘ Acknowledged	Partially Resolved	✓ Resolved
● Critical	0	0	0	0	0	0
● Major	1	0	0	0	1	0
● Medium	0	0	0	0	0	0
● Minor	0	0	0	0	0	0
● Informational	5	0	0	0	0	5
● Discussion	0	0	0	0	0	0

Audit Scope

ID	File	SHA256 Checksum
BEP	libs/BEP20.sol	9e94eda0da52608760be3dcc198dd3407654c76e12f3d83e0df5951ea043b545
BEC	libs/BEP20Capped.sol	7ad4e6fd6bec056d81927833adc615409d77bb75de6469b236e02bb523d046cc
IBE	libs/IBEP20.sol	2845b978d8c170f7cffbe83625924b261d65386a141d21dca2db66b7431a66a0
MBE	libs/MockBEP20.sol	94d88a846f9088b01a46d5861301ff2a21d0fe0bc2e10620d000539443e878c8
SBE	libs/SafeBEP20.sol	93d6a82e5aebaaba6c937cabffd9a5041a56e624da465f272adce82e8c653e5b
NBT	NanoByteToken.sol	4c5d658e40826165038db8d57e75435a09bffda8051ea7ac39187657259f01a3

Overview

NanoByteToken is a BEP20 token with extended functionality to allow token holders to delegate their votes as well as to query an account's total amount of votes at a given block number.

External Dependencies

The scope of the audit treats third-party entities as black boxes and assumes their functional correctness. However, in the real world, third parties can be compromised and this may lead to lost or stolen assets.

Here, the contract is serving as the underlying entity to interact with these third-party contracts:

- `Address.sol`;
- `SafeMath.sol`;
- `Ownable.sol`;
- `Context.sol`.

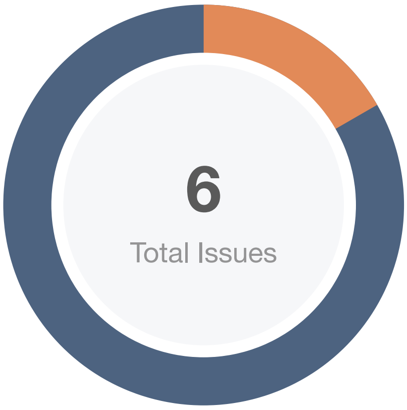
Privileged Functions

The contract `NanoByteToken` contains the following privileged functions that are restricted by the `owner`:

- `mint(uint256 _amount)` to mint tokens for the owner and move the delegation;
- `mint(address _to, uint256 _amount)` to mint tokens for the given address and move the delegation;
- `burnFrom()` to burn tokens from then given address and move the delegation.

To improve the trustworthiness of the project, dynamic runtime updates in the project should be notified to the community. Any plan to invoke the aforementioned functions should be also considered to move to the execution queue of the `Timelock` contract.

Findings



🟥 **Critical**	**0** (0.00%)
🟧 **Major**	**1** (16.67%)
🟨 **Medium**	**0** (0.00%)
🟨 **Minor**	**0** (0.00%)
🟦 **Informational**	**5** (83.33%)
🟩 **Discussion**	**0** (0.00%)

ID	Title	Category	Severity	Status
NBT-01	Centralization Risk	**Centralization / Privilege**	● **Major**	Partially Resolved
NBT-02	Not Declaring Return Types in Functions with Return Statements	Coding Style	● Informational	✓ Resolved
NBT-03	Storage Modification in the `require` Statement	Coding Style	● Informational	✓ Resolved
NBT-04	Improper Usage of Public and External Type	Gas Optimization	● Informational	✓ Resolved
NBT-05	Unlocked Compiler Version & Version Inconsistency	Language Specific	● Informational	✓ Resolved
NBT-06	Incorrect Comment	Coding Style	● Informational	✓ Resolved

NBT-01 | Centralization Risk

Category	Severity	Location	Status
Centralization / Privilege	● Major	projects/nanobyte/contracts/NanoByteToken.sol (031ae4b): 28, 18, 12	Partially Resolved

Description

In the contract `NanoByteToken`, the role `owner` has the authority over the following function:

- `mint(uint256 _amount)` to mint tokens for the owner and move the delegation;
- `mint(address _to, uint256 _amount)` to mint tokens for the given address and move the delegation;

Any compromise to the `owner` account may allow the hacker to take advantage of these functions.

Recommendation

We advise the client to carefully manage the `owner` account's private key to avoid any potential risks of being hacked. In general, we strongly recommend centralized privileges or roles in the protocol to be improved via a decentralized mechanism or smart-contract-based accounts with enhanced security practices, e.g., Multisignature wallets.

Indicatively, here is some feasible suggestions that would also mitigate the potential risk at the different level in term of short-term and long-term:

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

Alleviation

[NanoBytes]: The team acknowledged the issue and adapted to use the multi-sig and timelock solution to improve the problem.

The team chooses to use gnosis safe account for three owners, and the minting operation approval needs 2 out of 3.

Below is the timelock and multisig address:

nbt-timelock: 0x96c96feaB5007aa53E7216D6f5b6451b14a427E2 nbt-safe: 0x4f7bADBAD2D269B86c789A0e54494Be8d124Fef5

Multisig Addresses:

- 0x3b8eA037356CfD867c2191FD11614FA97BAB2772 (EOA)
- 0xA03455e5F9BDdbbD92bfd3D1C12b77404fE7dD62 (EOA)
- 0xeeF478A5c7C038850d913de7864e3D6Bbd13B6C4 (EOA)
- 0x3F39674D0d0c8c23bEe64B76dF4ebe1244648aC0 (EOA)

NBT-02 | Not Declaring Return Types in Functions with Return Statements

Category	Severity	Location	Status
Coding Style	● Informational	projects/nanobyte/contracts/NanoByteToken.sol (031ae4b): 95~96, 147	✓ Resolved

Description

Functions `delegate` and `delegateBySig` include return statements even though they do not explicitly declare return types in their definitions.

Recommendation

We recommend the client either remove the return statements in `delegate` and `delegateBySig` or explicitly declare return types and change the return value.

Alleviation

[NanoBytes]: The team addressed the issue and reflected in the commit hash d134255be8d85724b0f0b4412e0f9bc0e1ab40bc

NBT-03 | Storage Modification in the `require` Statement

Category	Severity	Location	Status
Coding Style	● Informational	projects/nanobyte/contracts/NanoByteToken.sol (031ae4b): 145~146	✓ Resolved

Description

In the `require` statement on L145, it compares `nounce` against `nounces[signatory]` before incrementing `nounces[signatory]`. Had the `require` statement failed, it would have been caused by `nounce != nonces[signatory]` instead of `nounce != nonces[signatory]++`. This could lead to confusion in the future in events of code change.

Recommendation

We recommend the client to increment the `nounces[signatory]` after the comparison as follows:

```
143  address signatory = ecrecover(digest, v, r, s);
144  require(signatory != address(0), "NBT::delegateBySig: invalid signature");
145  require(nonce == nonces[signatory], "NBT::delegateBySig: invalid nonce");
146
147  //ADDED CODE
148  nonces[signatory]++;
149
150  require(block.timestamp <= expiry, "NBT::delegateBySig: signature expired");
```

Alleviation

[NanoBytes]: The team addressed the issue and reflected in the commit hash e3b3651cefb26614b74a2a9d541eddc401a2ffd8

NBT-04 | Improper Usage of Public and External Type

Category	Severity	Location	Status
Gas Optimization	● Informational	projects/nanobyte/contracts/NanoByteToken.sol (031ae4b): 28, 18, 23, 12	✓ Resolved

Description

Public functions that are never called by the contract could be declared `external`. The following public functions that are never called by the contract in the following functions:

- `mint(uint256 _amount)`
- `mint(address _to, uint256 _amount)`
- `burn()`
- `burnFrom()`

Recommendation

Consider using the `external` attribute for functions never called from the contract.

Alleviation

[NanoBytes]: The team addressed the issue and reflected in the commit hash 99c3294b14278564caa2ddae1083cf8d1304f08e

NBT-05 | Unlocked Compiler Version & Version Inconsistency

Category	Severity	Location	Status
Language Specific	● Informational	projects/nanobyte/contracts/NanoByteToken.sol (031ae4b): 3~4	✓ Resolved

Description

The contract has unlocked and inconsistent compiler versions. For instance, contract `NanoByteToken` uses `solidity ^0.8.4`, yet `BEP20` and `BEP20Capped` use `solidity ^0.8.0`.

An unlocked compiler version in the source code of the contract permits the user to compile it at or above a particular version. This, in turn, leads to differences in the generated bytecode between compilations due to differing compiler version numbers. This can lead to ambiguity when debugging as compiler-specific bugs may occur in the codebase that would be hard to identify over a span of multiple compiler versions rather than a specific one.

Recommendation

We recommend locking the compiler at the lowest possible version that supports all the capabilities wished by the codebase. This will ensure that the project utilizes a compiler version that has been in use for the longest time and as such is less likely to contain yet-undiscovered bugs. Additionally, we also recommend using compiler versions that are consistent among contracts.

Alleviation

[NanoBytes]: The team addressed the issue and reflected in the commit hash e251901520d70fa4f4ff15248ab9f3761d7015fa

NBT-06 | Incorrect Comment

Category	Severity	Location	Status
Coding Style	● Informational	projects/nanobyte/contracts/NanoByteToken.sol (031ae4b): 79	✓ Resolved

Description

The comment bears little logical relation to the code block that follows it. The comment describes the function is to "delegate votes from `msg.sender` to `delegatee`", however, the function merely returns an associated delegatee given the delegator.

Recommendation

We advise that the comment to be properly adjusted to reflect the true functionality of the `delegates()` function.

Alleviation

[NanoBytes]: The team addressed the issue and reflected in the commit hash 3be3517a0529c1f972e4868eee2abd17883a3adf

Appendix

Finding Categories

Centralization / Privilege

Centralization / Privilege findings refer to either feature logic or implementation of components that act against the nature of decentralization, such as explicit ownership or specialized access roles in combination with a mechanism to relocate funds.

Gas Optimization

Gas Optimization findings do not affect the functionality of the code but generate different, more optimal EVM opcodes resulting in a reduction on the total gas cost of a transaction.

Language Specific

Language Specific findings are issues that would only arise within Solidity, i.e. incorrect usage of private or delete.

Coding Style

Coding Style findings usually do not affect the generated byte-code but rather comment on how to make the codebase more legible and, as a result, easily maintainable.

Checksum Calculation Method

The "Checksum" field in the "Audit Scope" section is calculated as the SHA-256 (Secure Hash Algorithm 2 with digest size of 256 bits) digest of the content of each file hosted in the listed source repository under the specified commit.

The result is hexadecimal encoded and is the same as the output of the Linux "sha256sum" command against the target file.

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Services Agreement, or the scope of services, and terms and conditions provided to you ("Customer" or the "Company") in connection with the Agreement. This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes, nor may copies be delivered to any other person other than the Company, without CertiK's prior written consent in each instance.

This report is not, nor should be considered, an "endorsement" or "disapproval" of any particular project or team. This report is not, nor should be considered, an indication of the economics or value of any "product" or "asset" created by any team or project that contracts CertiK to perform a security assessment. This report does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors, business, business model or legal compliance.

This report should not be used in any way to make decisions around investment or involvement with any particular project. This report in no way provides investment advice, nor should be leveraged as investment advice of any sort. This report represents an extensive assessing process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. CertiK's position is that each company and individual are responsible for their own due diligence and continuous security. CertiK's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies, and in no way claims any guarantee of security or functionality of the technology we agree to analyze.

The assessment services provided by CertiK is subject to dependencies and under continuing development. You agree that your access and/or use, including but not limited to any services, reports, and materials, will be at your sole risk on an as-is, where-is, and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives, and other unpredictable results. The services may access, and depend upon, multiple layers of third-parties.

ALL SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF ARE PROVIDED "AS IS" AND "AS

AVAILABLE” AND WITH ALL FAULTS AND DEFECTS WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED UNDER APPLICABLE LAW, CERTIK HEREBY DISCLAIMS ALL WARRANTIES, WHETHER EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS. WITHOUT LIMITING THE FOREGOING, CERTIK SPECIFICALLY DISCLAIMS ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AND ALL WARRANTIES ARISING FROM COURSE OF DEALING, USAGE, OR TRADE PRACTICE. WITHOUT LIMITING THE FOREGOING, CERTIK MAKES NO WARRANTY OF ANY KIND THAT THE SERVICES, THE LABELS, THE ASSESSMENT REPORT, WORK PRODUCT, OR OTHER MATERIALS, OR ANY PRODUCTS OR RESULTS OF THE USE THEREOF, WILL MEET CUSTOMER’S OR ANY OTHER PERSON’S REQUIREMENTS, ACHIEVE ANY INTENDED RESULT, BE COMPATIBLE OR WORK WITH ANY SOFTWARE, SYSTEM, OR OTHER SERVICES, OR BE SECURE, ACCURATE, COMPLETE, FREE OF HARMFUL CODE, OR ERROR-FREE. WITHOUT LIMITATION TO THE FOREGOING, CERTIK PROVIDES NO WARRANTY OR UNDERTAKING, AND MAKES NO REPRESENTATION OF ANY KIND THAT THE SERVICE WILL MEET CUSTOMER’S REQUIREMENTS, ACHIEVE ANY INTENDED RESULTS, BE COMPATIBLE OR WORK WITH ANY OTHER SOFTWARE, APPLICATIONS, SYSTEMS OR SERVICES, OPERATE WITHOUT INTERRUPTION, MEET ANY PERFORMANCE OR RELIABILITY STANDARDS OR BE ERROR FREE OR THAT ANY ERRORS OR DEFECTS CAN OR WILL BE CORRECTED.

WITHOUT LIMITING THE FOREGOING, NEITHER CERTIK NOR ANY OF CERTIK’S AGENTS MAKES ANY REPRESENTATION OR WARRANTY OF ANY KIND, EXPRESS OR IMPLIED AS TO THE ACCURACY, RELIABILITY, OR CURRENCY OF ANY INFORMATION OR CONTENT PROVIDED THROUGH THE SERVICE. CERTIK WILL ASSUME NO LIABILITY OR RESPONSIBILITY FOR (I) ANY ERRORS, MISTAKES, OR INACCURACIES OF CONTENT AND MATERIALS OR FOR ANY LOSS OR DAMAGE OF ANY KIND INCURRED AS A RESULT OF THE USE OF ANY CONTENT, OR (II) ANY PERSONAL INJURY OR PROPERTY DAMAGE, OF ANY NATURE WHATSOEVER, RESULTING FROM CUSTOMER’S ACCESS TO OR USE OF THE SERVICES, ASSESSMENT REPORT, OR OTHER MATERIALS.

ALL THIRD-PARTY MATERIALS ARE PROVIDED “AS IS” AND ANY REPRESENTATION OR WARRANTY OF OR CONCERNING ANY THIRD-PARTY MATERIALS IS STRICTLY BETWEEN CUSTOMER AND THE THIRD-PARTY OWNER OR DISTRIBUTOR OF THE THIRD-PARTY MATERIALS.

THE SERVICES, ASSESSMENT REPORT, AND ANY OTHER MATERIALS HEREUNDER ARE SOLELY PROVIDED TO CUSTOMER AND MAY NOT BE RELIED ON BY ANY OTHER PERSON OR FOR ANY PURPOSE NOT SPECIFICALLY IDENTIFIED IN THIS AGREEMENT, NOR MAY COPIES BE DELIVERED TO, ANY OTHER PERSON WITHOUT CERTIK’S PRIOR WRITTEN CONSENT IN EACH INSTANCE.

NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING

MATERIALS AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH SERVICES, ASSESSMENT REPORT, AND ANY ACCOMPANYING MATERIALS.

THE REPRESENTATIONS AND WARRANTIES OF CERTIK CONTAINED IN THIS AGREEMENT ARE SOLELY FOR THE BENEFIT OF CUSTOMER. ACCORDINGLY, NO THIRD PARTY OR ANYONE ACTING ON BEHALF OF ANY THEREOF, SHALL BE A THIRD PARTY OR OTHER BENEFICIARY OF SUCH REPRESENTATIONS AND WARRANTIES AND NO SUCH THIRD PARTY SHALL HAVE ANY RIGHTS OF CONTRIBUTION AGAINST CERTIK WITH RESPECT TO SUCH REPRESENTATIONS OR WARRANTIES OR ANY MATTER SUBJECT TO OR RESULTING IN INDEMNIFICATION UNDER THIS AGREEMENT OR OTHERWISE.

FOR AVOIDANCE OF DOUBT, THE SERVICES, INCLUDING ANY ASSOCIATED ASSESSMENT REPORTS OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

About

Founded in 2017 by leading academics in the field of Computer Science from both Yale and Columbia University, CertiK is a leading blockchain security company that serves to verify the security and correctness of smart contracts and blockchain-based protocols. Through the utilization of our world-class technical expertise, alongside our proprietary, innovative tech, we're able to support the success of our clients with best-in-class security, all whilst realizing our overarching vision; provable trust for all throughout all facets of blockchain.