HashEx
Blockchain Security

Manufactory

smart contracts
final audit report

November 2021

hashex.org

contact@hashex.org

Contents

1. Disclaimer	3
2. Overview	5
3. Found issues	6
4. Contracts	7
5. Conclusion	9
Appendix A. Issues’ severity classification	10
Appendix B. List of examined issue types	11

1. Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice 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 disclaimer below – please make sure to read it in full.

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 HashEx and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers, and other representatives) (HashEx) owe no duty of care towards you or any other person, nor does HashEx 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 HashEx 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, HashEx hereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against HashEx, 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

2. Overview

HashEx was commissioned by the Manufactory team to perform an audit of their smart contract. The audit was conducted on 22-11-2021.

The purpose of this audit was to achieve the following:

- Identify potential security issues with smart contracts
- Formally check the logic behind given smart contracts.

Information in this report should be used for understanding the risk exposure of smart contracts, and as a guide to improving the security posture of smart contracts by remediating the issues that were identified.

The code is available at 0x87cb006d1d93c790bab0c8261809c8f25484454b.

2.1 Summary

Project name	Manufactory
URL	https://manufactory.gg
Platform	Binance Smart Chain
Language	Solidity

2.2 Contracts

Name	Address
BEP20Token	0x87cb006D1D93c790BAB0c8261809c8f25484454b

3. Found issues

2
Total issues

🟧	High	1 (50%)
■	Informational	1 (50%)

BEP20Token

ID	Title	Severity	Status
01	Excessive mint rights	🟧 High	Resolved
02	Gas optimization	■ Informational	Acknowledged

4. Contracts

4.1 BEP20Token

4.1.1 Overview

The token contract implementing the BEP-20 standard.

4.1.2 Issues

01. Excessive mint rights

■ High ✓ Resolved

If the ownership is not renounced (**L:321**), the owner has exclusive access to the *mint()* function (**L:502**). Thus losing control over the owner account would lead to uncontrolled emission and, therefore, token devaluation.

Recommendation

We recommend renouncing token ownership after initial minting or transferring it to a contract with a transparent minting policy.

Update

The ownership was renounced on 23-11-2021. The total supply of 500'000'000 tokens was held by a single EOA by that date.

02. Gas optimization

▪ Informational ⓘ Acknowledged

1. Variables *_decimals*, *_symbol*, *_name* (**L:353** - **L:355**) can be set as immutable.

```
uint256 private _totalSupply;
uint8 private _decimals;
string private _symbol;
string private _name;

constructor() public {
  _name = "MANUFACTORY";
  _symbol = "MNFT";
  _decimals = 12;
  _totalSupply = 5000000000000000000000;
  _balances[msg.sender] = _totalSupply;

  emit Transfer(address(0), msg.sender, _totalSupply);
}
```

2. The Internal method *_burnFrom()* (**L:594**) is not used by the contract and can be deleted. Since *_burn()* (**L:559**) is called only inside of *_burnFrom()* it becomes unused and also can be removed.

```
  function _burnFrom(address account, uint256 amount) internal {
    _burn(account, amount);
    _approve(account, _msgSender(), _allowances[account]
[_msgSender()].sub(amount, "BEP20: burn amount exceeds allowance"));
  }
```

3. *renounceOwnership()*, *transferOwnership()*, *increaseAllowance()*, *decreaseAllowance()*, and *mint()* functions could be declared external in order to save gas.

5. Conclusion

1 high severity issue was found in the audited code and was fixed with the ownership renouncing. The audited contract is a standard BEP-20 and ERC-20 token with an initial supply of 500M tokens with 12 decimals.

This audit includes recommendations on the gas usage reduction.

The audited contract is deployed to the mainnet of Binance Smart Chain: 0x87cb006d1d93c790bab0c8261809c8f25484454b.

Appendix A. Issues’ severity classification

Critical. Issues that may cause an unlimited loss of funds or entirely break the contract workflow. Malicious code (including malicious modification of libraries) is also treated as a critical severity issue. These issues must be fixed before deployments or fixed in already running projects as soon as possible.

High. Issues that may lead to a limited loss of funds, break interaction with users, or other contracts under specific conditions. Also, issues in a smart contract, that allow a privileged account the ability to steal or block other users' funds.

Medium. Issues that do not lead to a loss of funds directly, but break the contract logic. May lead to failures in contracts operation.

Low. Issues that are of a non-optimal code character, for instance, gas optimization tips, unused variables, errors in messages.

Informational. Issues that do not impact the contract operation. Usually, informational severity issues are related to code best practices, e.g. style guide.

Appendix B. List of examined issue types

- Business logic overview
- Functionality checks
- Following best practices
- Access control and authorization
- Reentrancy attacks
- Front-run attacks
- DoS with (unexpected) revert
- DoS with block gas limit
- Transaction-ordering dependence
- ERC/BEP and other standards violation
- Unchecked math
- Implicit visibility levels
- Excessive gas usage
- Timestamp dependence
- Forcibly sending ether to a contract
- Weak sources of randomness
- Shadowing state variables
- Usage of deprecated code

HashEx
Blockchain Security