Guard

Smart contracts security assessment

Final report

Tariff: Standard

Buffies NFT Token

February 2022

0xguard.com

hello@0xguard.com

Contents

Introduction

The report has been prepared for the Buffies team.

The audited code has md5 hash-sum 04760AB9FC1C3A5B2B4DC64E7CC11F57. Users should check if they are interacting with the audited contract.

The audited contract is a standard ERC721 token contract with the mint function for which a commission is charged and without burn function. The mint() function is using whitelist. ERC721 interface is implemented with the use of OpenZeppelin libraries, which is considered the best practices.

Name	Buffies NFT Token
Audit date	2022-02-14 - 2022-02-14
Language	Solidity
Platform	Ethereum

Contracts checked

Name	Address
Buffies	

Procedure

We perform our audit according to the following procedure:

Automated analysis

- Scanning the project's smart contracts with several publicly available automated Solidity analysis tools
- Manual verification (reject or confirm) all the issues found by the tools

Manual audit

- Manually analyse smart contracts for security vulnerabilities
- Smart contracts' logic check

Known vulnerabilities checked

Title	Check result
Unencrypted Private Data On-Chain	passed
Code With No Effects	passed
Message call with hardcoded gas amount	passed
Typographical Error	passed
DoS With Block Gas Limit	passed
Presence of unused variables	passed
Incorrect Inheritance Order	passed
Requirement Violation	passed
Weak Sources of Randomness from Chain Attributes	passed
Shadowing State Variables	passed
Incorrect Constructor Name	passed
Block values as a proxy for time	passed
Authorization through tx.origin	passed
DoS with Failed Call	passed
Delegatecall to Untrusted Callee	passed
Use of Deprecated Solidity Functions	passed
Assert Violation	passed

State Variable Default Visibility	passed
Reentrancy	passed
Unprotected SELFDESTRUCT Instruction	passed
Unprotected Ether Withdrawal	passed
Unchecked Call Return Value	passed
Floating Pragma	not passed
Outdated Compiler Version	passed
Integer Overflow and Underflow	passed
Function Default Visibility	passed

Classification of issue severity

High severity	High severity issues can cause a significant or full loss of funds, change of contract ownership, major interference with contract logic. Such issues require immediate attention.
Medium severity	Medium severity issues do not pose an immediate risk, but can be detrimental to the client's reputation if exploited. Medium severity issues may lead to a contract failure and can be fixed by modifying the contract state or redeployment. Such issues require attention.
Low severity	Low severity issues do not cause significant destruction to the contract's functionality. Such issues are recommended to be taken into consideration.

Issues

High severity issues

No issues were found

Medium severity issues

No issues were found

Low severity issues

1. Different pragma version of libraries (Buffies)

'>=0.7.0<0.9.0' compiler version is used but libraries used '^0.8.0'

2. Floating pragma (Buffies)

Using pragma solidity >=0.7.0 <0.9.0 without SafeMath there is a risk of overflow uint variables.

Recommendation: Use for example the 0.8.6 version of the compiler or use SafeMath.

3. Require function without error message (Buffies)

In function mint(uint256 _mintAmount) uses require(!paused) without error message parameters, users may not understand the reason for reverting a transaction.

Recommendation: Initialize all the variables. If a variable is meant to be initialized to zero, explicitly set it to zero to improve code readability.

4. Gas optimization (Buffies)

The function isWhitelisted(address _user) iterates over the whitelistedAddresses[] in the for() loop, this can lead to high gas consumption when using the mint() function.

Recommendation: The best solution would be to use mapping(address => bool)

5. Not initialized local variable (Buffies)

uint256 i variable not initialized in function walletOfOwner().

Recommendation: Initialize all the variables. If a variable is meant to be initialized to zero, explicitly set it to zero to improve code readability.

Conclusion

Buffies NFT Token Buffies contract was audited. 5 low severity issues were found.

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 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 without 0xGuard prior written consent.

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 0xGuard 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.

Slither output

Buffies.walletOfOwner(address).i (eger.sol#1383) is a local variable never initialized

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#uninitialized-local-variables

ERC721._checkOnERC721Received(address,address,uint256,bytes) (eger.sol#1073-1094) ignores return value by IERC721Receiver(to).onERC721Received(_msgSender(),from,tokenId,_data) (eger.sol#1080-1090)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#unused-return

Buffies.constructor(string,string,string,string)._name (eger.sol#1304) shadows:

- ERC721._name (eger.sol#715) (state variable)

Buffies.constructor(string,string,string,string)._symbol (eger.sol#1305) shadows:

- ERC721._symbol (eger.sol#718) (state variable)

Buffies.walletOfOwner(address)._owner (eger.sol#1376) shadows:

- Ownable._owner (eger.sol#120) (state variable)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#local-variable-shadowing

ERC721._checkOnERC721Received(address,address,uint256,bytes) (eger.sol#1073-1094) has external calls inside a loop: IERC721Receiver(to).onERC721Received(_msgSender(),from,tokenId,_data) (eger.sol#1080-1090)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation/#calls-inside-a-loop

Variable 'ERC721._checkOnERC721Received(address,address,uint256,bytes).retval (eger.sol#1080)' in ERC721._checkOnERC721Received(address,address,uint256,bytes)

(eger.sol#1073-1094) potentially used before declaration: retval == IERC721Receiver.onERC721Received.selector (eger.sol#1081)

Variable 'ERC721._checkOnERC721Received(address,address,uint256,bytes).reason (eger.sol#1082)' in ERC721._checkOnERC721Received(address,address,uint256,bytes) (eger.sol#1073-1094) potentially used before declaration: reason.length == 0 (eger.sol#1083)

Variable 'ERC721._checkOnERC721Received(address,address,uint256,bytes).reason (eger.sol#1082)' in ERC721._checkOnERC721Received(address,address,uint256,bytes) (eger.sol#1073-1094) potentially used before declaration: revert(uint256,uint256)(32 + reason,mload(uint256)(reason)) (eger.sol#1087)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#pre-declaration-usage-of-local-variables

Address.isContract(address) (eger.sol#205-215) uses assembly

- INLINE ASM (eger.sol#211-213)

Address.verifyCallResult(bool,bytes,string) (eger.sol#374-394) uses assembly

- INLINE ASM (eger.sol#386-389)

ERC721._checkOnERC721Received(address,address,uint256,bytes) (eger.sol#1073-1094) uses assembly

- INLINE ASM (eger.sol#1086-1088)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#assembly-usage

Buffies.mint(uint256) (eger.sol#1335-1355) compares to a boolean constant:

-onlyWhitelisted == true (eger.sol#1347)

Buffies.tokenURI(uint256) (eger.sol#1389-1416) compares to a boolean constant:

-revealed == false (eger.sol#1401)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#boolean-equality

Different versions of Solidity is used:

- Version used: ['>=0.7.0<0.9.0', '^0.8.0']

- ^0.8.0 (eger.sol#7)

- ^0.8.0 (eger.sol#77)

- ^0.8.0 (eger.sol#104)

- ^0.8.0 (eger.sol#182)

- ^0.8.0 (eger.sol#402)

- ^0.8.0 (eger.sol#432)

- ^0.8.0 (eger.sol#460)

- ^0.8.0 (eger.sol#491)

- ^0.8.0 (eger.sol#636)

- ^0.8.0 (eger.sol#667)

- ^0.8.0 (eger.sol#696)

- ^0.8.0 (eger.sol#1122)

- >=0.7.0<0.9.0 (eger.sol#1285)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#different-pragma-directives-are-used

ERC721Enumerable._removeTokenFromAllTokensEnumeration(uint256) (eger.sol#1261-1279) has costly operations inside a loop:

- delete _allTokensIndex[tokenId] (eger.sol#1277)

ERC721Enumerable._removeTokenFromAllTokensEnumeration(uint256) (eger.sol#1261-1279) has costly operations inside a loop:

- _allTokens.pop() (eger.sol#1278)

ERC721Enumerable._removeTokenFromOwnerEnumeration(address,uint256) (eger.sol#1236-1254) has costly operations inside a loop:

- delete _ownedTokensIndex[tokenId] (eger.sol#1252)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#costly-operations-inside-a-loop

Address.functionCall(address,bytes) (eger.sol#258-260) is never used and should be removed

Address.functionCall(address,bytes,string) (eger.sol#268-274) is never used and should be removed

Address.functionCallWithValue(address,bytes,uint256) (eger.sol#287-293) is never used and should be removed

Address.functionCallWithValue(address,bytes,uint256,string) (eger.sol#301-312) is never used and should be removed

Address.functionDelegateCall(address,bytes) (eger.sol#347-349) is never used and should be removed

Address.functionDelegateCall(address,bytes,string) (eger.sol#357-366) is never used and should be removed

Address.functionStaticCall(address,bytes) (eger.sol#320-322) is never used and should be removed

Address.functionStaticCall(address,bytes,string) (eger.sol#330-339) is never used and should be removed

Address.sendValue(address,uint256) (eger.sol#233-238) is never used and should be removed

Address.verifyCallResult(bool,bytes,string) (eger.sol#374-394) is never used and should be removed

Context._msgData() (eger.sol#94-96) is never used and should be removed

ERC721._baseURI() (eger.sol#796-798) is never used and should be removed

ERC721._burn(uint256) (eger.sol#993-1005) is never used and should be removed

Strings.toHexString(uint256) (eger.sol#43-54) is never used and should be removed

Strings.toHexString(uint256,uint256) (eger.sol#59-69) is never used and should be removed

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#dead-code

Pragma version^0.8.0 (eger.sol#7) allows old versions

Pragma version^0.8.0 (eger.sol#77) allows old versions

Pragma version^0.8.0 (eger.sol#104) allows old versions

Pragma version^0.8.0 (eger.sol#182) allows old versions

Pragma version^0.8.0 (eger.sol#402) allows old versions

Pragma version^0.8.0 (eger.sol#432) allows old versions

Pragma version^0.8.0 (eger.sol#460) allows old versions

Pragma version^0.8.0 (eger.sol#491) allows old versions

Pragma version^0.8.0 (eger.sol#636) allows old versions

Pragma version^0.8.0 (eger.sol#667) allows old versions

Pragma version^0.8.0 (eger.sol#696) allows old versions

Pragma version^0.8.0 (eger.sol#1122) allows old versions

Pragma version>=0.7.0<0.9.0 (eger.sol#1285) is too complex

solc-0.8.11 is not recommended for deployment

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#incorrect-versions-of-solidity

Low level call in Address.sendValue(address,uint256) (eger.sol#233-238):

- (success) = recipient.call{value: amount}() (eger.sol#236)

Low level call in Address.functionCallWithValue(address,bytes,uint256,string) (eger.sol#301-312):

- (success,returndata) = target.call{value: value}(data) (eger.sol#310)

Low level call in Address.functionStaticCall(address,bytes,string) (eger.sol#330-339):

- (success,returndata) = target.staticcall(data) (eger.sol#337)

Low level call in Address.functionDelegateCall(address,bytes,string) (eger.sol#357-366):

- (success,returndata) = target.delegatecall(data) (eger.sol#364)

Low level call in Buffies.withdraw() (eger.sol#1450-1457):

- (os) = address(owner()).call{value: address(this).balance}() (eger.sol#1454)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#low-level-calls

Parameter ERC721.safeTransferFrom(address,address,uint256,bytes)._data (eger.sol#870) is not in mixedCase

Parameter Buffies.airDrop(address[],uint256)._to (eger.sol#1322) is not in mixedCase

Parameter Buffies.airDrop(address[],uint256)._id (eger.sol#1323) is not in mixedCase

Parameter Buffies.mint(uint256)._mintAmount (eger.sol#1335) is not in mixedCase

Parameter Buffies.isWhitelisted(address)._user (eger.sol#1358) is not in mixedCase

Parameter Buffies.setOnlyWhitelisted(bool)._state (eger.sol#1367) is not in mixedCase

Parameter Buffies.whitelistUsers(address[])._users (eger.sol#1371) is not in mixedCase

Parameter Buffies.walletOfOwner(address)._owner (eger.sol#1376) is not in mixedCase

Parameter Buffies.setCost(uint256)._newCost (eger.sol#1423) is not in mixedCase

Parameter Buffies.setmaxMintAmount(uint256)._newmaxMintAmount (eger.sol#1427) is not in mixedCase

Parameter Buffies.setNotRevealedURI(string)._notRevealedURI (eger.sol#1431) is not in mixedCase

Parameter Buffies.setBaseURI(string)._newBaseURI (eger.sol#1435) is not in mixedCase

Parameter Buffies.setBaseExtension(string)._newBaseExtension (eger.sol#1439) is not in mixedCase

Parameter Buffies.pause(bool)._state (eger.sol#1446) is not in mixedCase

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#conformance-to-solidity-naming-conventions

Buffies.maxSupply (eger.sol#1295) should be constant

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#state-variables-that-could-be-declared-constant

renounceOwnership() should be declared external:

- Ownable.renounceOwnership() (eger.sol#153-155)

transferOwnership(address) should be declared external:

- Ownable.transferOwnership(address) (eger.sol#161-164)

name() should be declared external:

- ERC721.name() (eger.sol#770-772)

symbol() should be declared external:

- ERC721.symbol() (eger.sol#777-779)

tokenURI(uint256) should be declared external:

- Buffies.tokenURI(uint256) (eger.sol#1389-1416)

- ERC721.tokenURI(uint256) (eger.sol#784-789)

approve(address,uint256) should be declared external:

- ERC721.approve(address,uint256) (eger.sol#803-813)

setApprovalForAll(address,bool) should be declared external:

- ERC721.setApprovalForAll(address,bool) (eger.sol#827-829)

transferFrom(address,address,uint256) should be declared external:

- ERC721.transferFrom(address,address,uint256) (eger.sol#841-850)

safeTransferFrom(address,address,uint256) should be declared external:

- ERC721.safeTransferFrom(address,address,uint256) (eger.sol#855-861)

tokenByIndex(uint256) should be declared external:

- ERC721Enumerable.tokenByIndex(uint256) (eger.sol#1169-1172)

airDrop(address[],uint256) should be declared external:

- Buffies.airDrop(address[],uint256) (eger.sol#1321-1331)

mint(uint256) should be declared external:

- Buffies.mint(uint256) (eger.sol#1335-1355)

setOnlyWhitelisted(bool) should be declared external:

- Buffies.setOnlyWhitelisted(bool) (eger.sol#1367-1369)

whitelistUsers(address[]) should be declared external:

- Buffies.whitelistUsers(address[]) (eger.sol#1371-1374)

walletOfOwner(address) should be declared external:

- Buffies.walletOfOwner(address) (eger.sol#1376-1387)

reveal() should be declared external:

- Buffies.reveal() (eger.sol#1419-1421)

setCost(uint256) should be declared external:

- Buffies.setCost(uint256) (eger.sol#1423-1425)

setmaxMintAmount(uint256) should be declared external:

- Buffies.setmaxMintAmount(uint256) (eger.sol#1427-1429)

setBaseExtension(string) should be declared external:

- Buffies.setBaseExtension(string) (eger.sol#1439-1444)

pause(bool) should be declared external:

- Buffies.pause(bool) (eger.sol#1446-1448)

withdraw() should be declared external:

- Buffies.withdraw() (eger.sol#1450-1457)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#public-function-that-could-be-declared-external

0x Guard