0x Guard

Smart contracts security assessment

Final report

Tariff: Standard

Kulfy

January 2022

0xguard.com

hello@0xguard.com

Contents

Introduction

Standard ERC721 token contract with an added author tip feature. ERC7211 interface is realized with the use of the OpenZeppelin library that is considered the best practice. The contract can be considered as an NFT service as it has the open mint function allowing anyone to create their own token.

Name	Kulfy
Audit date	2022-01-12 - 2022-01-12
Language	Solidity
Platform	Ethereum

Contracts checked

Name	Address
KulfyV3	

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 analyze 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	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Outdated Compiler Version	passed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. Gas optimisation - FIXED (KulfyV3)

a. `mintNFT()` function visibility can be changed to external.

b. `tipKulfyAuthor()` function visibility can be changed to external.

2. No error message in require statement - FIXED (KulfyV3)

The requirement in the L:102 checks the validity of the passed `_id` parameter. If the passed token id exceeds the max id value, the transaction will be reverted with an empty message. The absence of a revert explanation may confuse users.

Recommendation: Add "Invalid token id" error message.

3. Excessive ERC721 import - FIXED (KulfyV3)

The list of imports includes the ERC721 contract in the L:9. It is unnecessary since ERC721URIStorage is inherited from ERC721 and consequently imports it under the hood.

Recommendation: Remove the L:9.

4. Floating pragma - FIXED (KulfyV3)

Fixing pragma to the intended version of the contract's deployment is standard procedure. A contract deployed with an outdated compiler version is more likely to encounter bugs as new solidity releases arrive.

Recommendation: Fix pragma to a particular version.

5. Not all fallback functions can be supported - FIXED (KulfyV3)

When `tipKulfyAuthor()` is called, all provided ETH is forwarded to an author and is added to the total tip amount. ETH is sent to the author with the `transfer()` method which has a limitation on 2300 gas and throws an error in case of an unsuccessful transfer. However, some authors may want to have special logic for incoming tips. For example, emit certain events or record the benefactor's address. They won't be able to do that because of gas restrictions.

Recommendation: We advise using the `call()` method with a reasonable gasLimit. Also, we suggest placing the gasLimit parameter to a variable and having a setter for it if the price of operations changes over time in hard forks. An increase of gasLimit enables reentrancy to `tipKulfyAuthor()` function, which leads to an incorrect `tipAmount` calculation. So, `tipKulfyAuthor()` function should comprise `nonReentrant()` modifier from ReentrancyGuard contract in order to prevent this inaccuracy.

Conclusion

Kulfy KulfyV3 contract was audited. 5 low severity issues were found.

Disclaimer

Static code analysis result

KulfyV3.changeGasLimit(uint256) (contracts/KulfyV3.sol#139-142) should emit an event for:

- gasLimit = newGasLimit (contracts/KulfyV3.sol#141)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-events-arithmetic

KulfyV3.tipKulfyAuthor(uint256,uint256)._author (contracts/KulfyV3.sol#116) lacks a zero-check on :

- (sent,data) = _author.call{gas: _gas,value: msg.value}() (contracts/KulfyV3.sol#119)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#missing-zero-address-validation

Reentrancy in KulfyV3.tipKulfyAuthor(uint256,uint256) (contracts/KulfyV3.sol#105-137):

External calls:

- (sent,data) = _author.call{gas: _gas,value: msg.value}() (contracts/KulfyV3.sol#119)

Event emitted after the call(s):

- KulfyTipped(_id,_kulfy.tokenURI,_kulfy.kid,_kulfy.tipAmount,_author,msg.sender) (contracts/KulfyV3.sol#129-136)

Reference: https://github.com/crytic/slither/wiki/Detector-Documentation#reentrancy-vulnerabilities-3

Low level call in KulfyV3.tipKulfyAuthor(uint256,uint256) (contracts/KulfyV3.sol#105-137):

- (sent,data) = _author.call{gas: _gas,value: msg.value}() (contracts/KulfyV3.sol#119)

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

Parameter KulfyV3.mintNFT(address,string,string,string)._tokenURI (contracts/KulfyV3.sol#68) is not in mixedCase

Parameter KulfyV3.mintNFT(address,string,string,string)._assetURI (contracts/KulfyV3.sol#69) is not in mixedCase

Parameter KulfyV3.mintNFT(address,string,string,string)._kid (contracts/KulfyV3.sol#70) is not in mixedCase

Parameter KulfyV3.tipKulfyAuthor(uint256,uint256)._id (contracts/KulfyV3.sol#105) is not in mixedCase

Parameter KulfyV3.tipKulfyAuthor(uint256,uint256)._gas (contracts/KulfyV3.sol#105) is not in mixedCase

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

tipKulfyAuthor(uint256,uint256) should be declared external:

- KulfyV3.tipKulfyAuthor(uint256,uint256) (contracts/KulfyV3.sol#105-137)

changeGasLimit(uint256) should be declared external:

- KulfyV3.changeGasLimit(uint256) (contracts/KulfyV3.sol#139-142)

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

Guard