SMART CONTRACT CODE REVIEW AND SECURITY ANALYSIS REPORT

Customer: Red Fox
Date: June 14th, 2022

Document

Name	Smart Contract Code Review and Security Analysis Report for Red Fox.
Approved By	Evgeniy Bezuglyi | SC Department Head at Hacken OU
Type	ERC721 token; ERC1155 token; Token sale
Platform	EVM
Language	Solidity
Methods	Architecture Review, Functional Testing, Computer-Aided Verification, Manual Review
Website	https://www.rfox.com/
Timeline	16.05.2022 - 14.06.2022
Changelog	24.05.2022 - Initial Review 14.06.2022 - Second Review

Table of contents

Introduction

Hacken OÜ (Consultant) was contracted by Red Fox (Customer) to conduct a Smart Contract Code Review and Security Analysis. This report presents the findings of the security assessment of the Customer's smart contracts.

Scope

The scope of the project is smart contracts in the repository:

Initial review scope

Repository:
https://github.com/RFL-NFTPlatform/nft-factory
Commit:
6a2464ffd5ef95cae612a05c36885b4814fffe34
Technical Documentation: No
JS tests: Yes
https://github.com/RFL-NFTPlatform/nft-factory/tree/master/test
Contracts:

```
File: ./contracts/erc1155/factory/RFOXFactoryStandard1155.sol
SHA3: 5ffbdd392aff51451573fea8341f52f1e4c6eab5addca63c36ae14e9cfaa55cc

File: ./contracts/erc1155/factory/RFOXFactoryStandardBotPrevention1155.sol
SHA3: 328716b3eb2f45273f3b139a2d49e00f78159781c6e12d8034ae48c35fa57c3d

File: ./contracts/erc1155/factory/RFOXFactoryWhitelist.sol
SHA3: 7a3a329f99912a997c57521eb6f26ba25dd8571bcbd5f99a094d5e6dde0d893c

File: ./contracts/erc1155/factory/RFOXFactoryWhitelistBotPrevention1155.sol
SHA3: 6c39b12d82185e231d2279a3fd0bbbeb3e5615f259214b2e76ba8791ba2f5f57

File: ./contracts/erc1155/lib/base/BaseRFOXNFT1155.sol
SHA3: 505809965ac712ab87b98cfa71c02c0e14e87e6a503d3bf8381fa34dc4be0ba1

File: ./contracts/erc1155/lib/base/BaseRFOXNFTPresale1155.sol
SHA3: 0862d53a3ae4d3523062c6fd01bb0dec2b4abd1c7d208589dd6065c9ce5bd67c

File: ./contracts/erc1155/lib/RFOXNFTPresale1155.sol
SHA3: bb7f6794d1450e399959772887baac188aa1a20c992adbe927366b02a27af2ee

File: ./contracts/erc1155/lib/RFOXNFTSale1155.sol
SHA3: a76191d919111bffab443565fc162c107b862876389416e72ba6ae6da416f81d

File: ./contracts/erc1155/lib/RFOXNFTSignaturePresale1155.sol
SHA3: 5ebce1932150e7ddb33db796421d68da87807bc759664e6d7548bf22ef5fb625

File: ./contracts/erc1155/lib/RFOXNFTSignatureSale1155.sol
SHA3: 0ed5be1cd5c75e3f95222dba5e4310ceb5c6d318af006fae814d9c32c0b799f3

File: ./contracts/erc1155/RFOXNFTStandard1155.sol
SHA3: a95c7ff5698770211906343b3c314b9103db7f73c4880a7d67cef3bf4f6204c1

File: ./contracts/erc1155/RFOXNFTStandardBotPrevention1155.sol
SHA3: b7f94329b07ce711b84da134eb4a52f73dc88273e8900990307d449ae4120f80
```

```
File: ./contracts/erc1155/RFOXNFTWhitelist1155.sol
SHA3: 225f2d5d1e9c076a81aadd3a52d75e3cea68592e1323e1f2c9b4662e32574552

File: ./contracts/erc1155/RFOXNFTWhitelistBotPrevention1155.sol
SHA3: e93fe27d954b9ee552a5e17352a31ee36feb683cb2d6871ece73b439f630586e

File: ./contracts/erc1155/structs/ParamStructs1155.sol
SHA3: 44ef0a0cccbe769fee31be588605e5433037bdc07425aca2d57d2969fc79ba97

File: ./contracts/erc1155/structs/TokenStructs.sol
SHA3: 920249d080b525e380a7df4cd54c2bdb667bec043f8e5aeeb75b63773c0a99e2

File: ./contracts/erc721/factory/RFOXFactoryStandard.sol
SHA3: 543eb5aba00c992ec62e606dba90a740a7b996035d4b9300bbdc54fc4fd63ec2

File: ./contracts/erc721/factory/RFOXFactoryStandardBotPrevention.sol
SHA3: 1d809ce2716fb5885ae1768a0966c0cbdf2131ccef66c738a464f8e808479dd3

File: ./contracts/erc721/factory/RFOXFactoryWhitelist.sol
SHA3: 8e05bb6ffd49a6550cff24486b531c40df4913637ebff170effb97bdfde09bc2

File: ./contracts/erc721/factory/RFOXFactoryWhitelistBotPrevention.sol
SHA3: 0a00118ec4be20896ae4790d85fd30b693dd0d22b611c97290b7c06c9fd41148

File: ./contracts/erc721/lib/base/BaseRFOXNFT.sol
SHA3: 42b5e12a8b1ea64e53469747529fc42314ab078a217ccff9b9feafcba0c746d6

File: ./contracts/erc721/lib/base/BaseRFOXNFTPresale.sol
SHA3: d24925887ea6a44aa5bc3ef5e54150ebe78d3a02e8a8c07771ce984bcbc7fc07

File: ./contracts/erc721/lib/RFOXNFTPresale.sol
SHA3: 40629f7560b2f53e96181612caf6d702ed1c47748330908a112db88ce7593bc6

File: ./contracts/erc721/lib/RFOXNFTSale.sol
SHA3: a510649845a35b772ae8ef7e3fb1f6fd02dccfbd9104b4840c5977d6d65e93c5

File: ./contracts/erc721/lib/RFOXNFTSignaturePresale.sol
SHA3: d459a3c08c1b380fcb1df6bcb49de5ac4395ae809429ab077faf5f7f6d5ee91c

File: ./contracts/erc721/lib/RFOXNFTSignatureSale.sol
SHA3: 61229c94c24b12b2cca9671b3ef7a457bda886e5e0f3526343ac21f047318524

File: ./contracts/erc721/RFOXNFTStandard.sol
SHA3: 891487d7cf42bf8f4878998c9c8d263803001fadcf81421caeb2aeaa7515a570

File: ./contracts/erc721/RFOXNFTStandardBotPrevention.sol
SHA3: 5ecfe92030536959cc2260454fa3ff2fc87698536f3b8b4e6502d0e78d8f7c7b

File: ./contracts/erc721/RFOXNFTWhitelist.sol
SHA3: 0cec62e19a16c5826bef78e907fdda3eb284595e00cbc29512bc492bc23d2cde

File: ./contracts/erc721/RFOXNFTWhitelistBotPrevention.sol
SHA3: daac86c7ea033731bc85a79df29e7d135cfecc754aa364a7d24304bdbaf4c20c

File: ./contracts/erc721/structs/ParamStructs.sol
SHA3: 357dd3076ef58b09f22d1c7e73fe207dffddd0552411dbe3239bd6bdd801e7a0

File: ./contracts/interfaces/IRFOXFactory.sol
SHA3: 8f834b87104e14159f855566309d904250f446b833aadaf43beff598dba55bd4
```

Second review scope

Repository:
https://github.com/RFL-NFTPlatform/nft-factory
Commit:
731ccbdb6df349432a57f997383d51860c82a4b2
Technical Documentation: No
JS tests: Yes
Contracts:

```
File: ./contracts/erc1155/factory/RFOXFactoryStandard1155.sol
SHA3: 7ac1edf1eadce66980ff0d4c90d09f863be3eff1618c22332902b68248477b2d

File: ./contracts/erc1155/factory/RFOXFactoryStandardBotPrevention1155.sol
SHA3: c82091de3c77c83e7243b7a4171cc9e720753c56abf61922f4fadeff6990a477

File: ./contracts/erc1155/factory/RFOXFactoryWhitelist.sol
SHA3: 8a9ce16948355fb978efaaea6336a43ef9dfe52d053a82cbcc2335d9984889b6

File: ./contracts/erc1155/factory/RFOXFactoryWhitelistBotPrevention1155.sol
SHA3: 29f5885ba094c50fff0b101ee808c97b5c70327b42d69926dd318a32adb123ca

File: ./contracts/erc1155/lib/base/BaseRFOXNFT1155.sol
SHA3: aae680b4f999065714660ab7be05f7d7d0c4dcb86a15511489e70373431346f0

File: ./contracts/erc1155/lib/base/BaseRFOXNFTPresale1155.sol
SHA3: b263c71deb68ebe73fb929a8463008a4bda326740666159bd368b2fbf6e62e66

File: ./contracts/erc1155/lib/RFOXNFTPresale1155.sol
SHA3: 703e769be8b6720f11dd2d1d9c940fe4dc93309669f11365ab61af0a1fb113f9

File: ./contracts/erc1155/lib/RFOXNFTSale1155.sol
SHA3: 5004bd3c582d715fe3d2f845c9d27cdf79b998118edd833a766a38b7ad2654f3

File: ./contracts/erc1155/lib/RFOXNFTSignaturePresale1155.sol
SHA3: 4b8904879fd3f6cb99d623ca52e31aef72879dd9a0ea45f3a5d98e75a916af9d

File: ./contracts/erc1155/lib/RFOXNFTSignatureSale1155.sol
SHA3: f0f57866928aefea86aa5a376743bae826f9897d501f6b4059991201f9a60504

File: ./contracts/erc1155/RFOXNFTStandard1155.sol
SHA3: f4c82573d66eab87c3f9e68a2408edb6b467607bd46d87bf6e6532bc97e4fdb9

File: ./contracts/erc1155/RFOXNFTStandardBotPrevention1155.sol
SHA3: b92eff165bc79a9b3c16299ddd137326e8c8982f852672499b66bd2c7bc3c23e

File: ./contracts/erc1155/RFOXNFTWhitelist1155.sol
SHA3: 0a290a17191781b6393e3ffd74fa79142b61a0b22e32c9234609614d17dcfb8b

File: ./contracts/erc1155/RFOXNFTWhitelistBotPrevention1155.sol
SHA3: ee33bb06bb2cbbb9c059e0a29a1044bf4ca545dddc8de30c3f126c24ce2c8a7a

File: ./contracts/erc1155/structs/ParamStructs1155.sol
SHA3: 030490872abcb0c77165be99445c8913b51c92bbfac062d30b4c4240b3fce68b

File: ./contracts/erc1155/structs/TokenStructs.sol
SHA3: bbba3a5ebed1bf053bc088e8e9b9487f2a0ad06adae688549d0d7394964070ec

File: ./contracts/erc721/factory/RFOXFactoryStandard.sol
SHA3: de329d0230a013772d0509f3903e8c2a60112f32b9e966cf375e8489d445fe1f

File: ./contracts/erc721/factory/RFOXFactoryStandardBotPrevention.sol
SHA3: 0aa35d965e8480eba49508a1e799969a27207c111ec745a921f661c73ce176dc
```

```
File: ./contracts/erc721/factory/RFOXFactoryWhitelist.sol
SHA3: 95cb79511c439c55b35550fa37a0e2b3dc96dd39f8ab463a673f8fae53a23c53

File: ./contracts/erc721/factory/RFOXFactoryWhitelistBotPrevention.sol
SHA3: e660b929a13ef295c0107c0514201b1901384bfcafd510c0e612fe13a193e8d3

File: ./contracts/erc721/lib/base/BaseRFOXNFT.sol
SHA3: 554a4367bcdcecb55fab294e09fd1dde842ef22d5197dc279fb8cbfd777504c4

File: ./contracts/erc721/lib/base/BaseRFOXNFTPresale.sol
SHA3: 86f0daa769b80dbb85521b73bc8c5eec87795b162bd137c4e1058182374a35d6

File: ./contracts/erc721/lib/RFOXNFTPresale.sol
SHA3: cae69962f269ae48eb4f38e4b5f67d3c507c3a79dea2cc9f0d677fe8ab6d4b1e

File: ./contracts/erc721/lib/RFOXNFTSale.sol
SHA3: 5058de75839c39086833fad7bbf7898a22bda551d0ff0106060a61b43defb3ce

File: ./contracts/erc721/lib/RFOXNFTSignaturePresale.sol
SHA3: cc1cffc00c44ec096d3c523bbdec7ab00430751aa4d993923cd5cea6423e2631

File: ./contracts/erc721/lib/RFOXNFTSignatureSale.sol
SHA3: 38d537e1b04880e2f2721c52c066f4c11d65da509aeed6b1a4cf5e714550a85f

File: ./contracts/erc721/RFOXNFTStandard.sol
SHA3: 019fa42eb58f9e408df8e96bd2605f26b908218adacd85b8e2f8a868501044b1

File: ./contracts/erc721/RFOXNFTStandardBotPrevention.sol
SHA3: 96de7ee562b37ca559c89cd9fb887808941b1ad7108bd781830ec83850688fd0

File: ./contracts/erc721/RFOXNFTWhitelist.sol
SHA3: 9448df083f2547b8516b1bcbbece82ebf2b3ff2a798bfa659ba50898d3612f09

File: ./contracts/erc721/RFOXNFTWhitelistBotPrevention.sol
SHA3: 8e76da7abb2ec6e917879d4c6652a45bb8ac6876f9e2f27909a696fbfbffa358

File: ./contracts/erc721/structs/ParamStructs.sol
SHA3: e155bea6530a491c2fd57912b89bf4b0b41c363d29ab9df78742703bdbdcb23e

File: ./contracts/interfaces/IRFOXFactory.sol
SHA3: bb236d1afc89629041ebb0e1f31b5bd52608673e856d46c8167e836ff0504930
```

Severity Definitions

Risk Level	Description
Critical	Critical vulnerabilities are usually straightforward to exploit and can lead to assets loss or data manipulations.
High	High-level vulnerabilities are difficult to exploit; however, they also have a significant impact on smart contract execution, e.g., public access to crucial functions.
Medium	Medium-level vulnerabilities are important to fix; however, they cannot lead to assets loss or data manipulations.
Low	Low-level vulnerabilities are mostly related to outdated, unused, etc. code snippets that cannot have a significant impact on execution.

Executive Summary

The score measurement details can be found in the corresponding section of the methodology.

Documentation quality

The Customer provided superficial functional requirements and did not provide technical requirements. Technical documentation is available in code. The total Documentation Quality score is **6** out of **10**.

Code quality

The total CodeQuality score is **7** out of **10**. Code violates the order of functions and maximum line length defined in the style guide. Unit tests were provided.

Architecture quality

The architecture quality score is **10** out of **10**. Code use best practices.

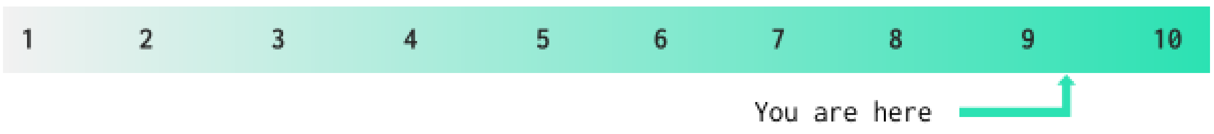
Security score

As a result of the audit, the code contains no issues. The security score is **10** out of **10**.

All found issues are displayed in the “Findings” section.

Summary

According to the assessment, the Customer's smart contract has the following score: **9.3**



Checked Items

We have audited provided smart contracts for commonly known and more specific vulnerabilities. Here are some of the items that are considered:

Item	Type	Description	Status
Default Visibility	SWC-100 SWC-108	Functions and state variables visibility should be set explicitly. Visibility levels should be specified consciously.	Passed
Integer Overflow and Underflow	SWC-101	If unchecked math is used, all math operations should be safe from overflows and underflows.	Passed
Outdated Compiler Version	SWC-102	It is recommended to use a recent version of the Solidity compiler.	Passed
Floating Pragma	SWC-103	Contracts should be deployed with the same compiler version and flags that they have been tested thoroughly.	Passed
Unchecked Call Return Value	SWC-104	The return value of a message call should be checked.	Passed
Access Control & Authorization	CWE-284	Ownership takeover should not be possible. All crucial functions should be protected. Users could not affect data that belongs to other users.	Passed
SELFDESTRUCT Instruction	SWC-106	The contract should not be destroyed until it has funds belonging to users.	Not Relevant
Check-Effect-Interaction	SWC-107	Check-Effect-Interaction pattern should be followed if the code performs ANY external call.	Passed
Uninitialized Storage Pointer	SWC-109	Storage type should be set explicitly if the compiler version is < 0.5.0.	Not Relevant
Assert Violation	SWC-110	Properly functioning code should never reach a failing assert statement.	Not Relevant
Deprecated Solidity Functions	SWC-111	Deprecated built-in functions should never be used.	Passed
Delegatecall to Untrusted Callee	SWC-112	Delegatecalls should only be allowed to trusted addresses.	Not Relevant
DoS (Denial of Service)	SWC-113 SWC-128	Execution of the code should never be blocked by a specific contract state unless it is required.	Passed

Race Conditions	SWC-114	Race Conditions and Transactions Order Dependency should not be possible.	Passed
Authorization through tx.origin	SWC-115	tx.origin should not be used for authorization.	Passed
Block values as a proxy for time	SWC-116	Block numbers should not be used for time calculations.	Passed
Signature Unique Id	SWC-117 SWC-121 SWC-122	Signed messages should always have a unique id. A transaction hash should not be used as a unique id.	Passed
Shadowing State Variable	SWC-119	State variables should not be shadowed.	Passed
Weak Sources of Randomness	SWC-120	Random values should never be generated from Chain Attributes.	Passed
Incorrect Inheritance Order	SWC-125	When inheriting multiple contracts, especially if they have identical functions, a developer should carefully specify inheritance in the correct order.	Passed
Calls Only to Trusted Addresses	EEA-Level-2 SWC-126	All external calls should be performed only to trusted addresses.	Passed
Presence of unused variables	SWC-131	The code should not contain unused variables if this is not justified by design.	Passed
EIP standards violation	EIP	EIP standards should not be violated.	Passed
Assets integrity	**Custom**	Funds are protected and cannot be withdrawn without proper permissions.	Passed
User Balances manipulation	**Custom**	Contract owners or any other third party should not be able to access funds belonging to users.	Passed
Data Consistency	**Custom**	Smart contract data should be consistent all over the data flow.	Passed
Flashloan Attack	**Custom**	When working with exchange rates, they should be received from a trusted source and not be vulnerable to short-term rate changes that can be achieved by using flash loans. Oracles should be used.	Not Relevant
Token Supply manipulation	**Custom**	Tokens can be minted only according to rules specified in a whitepaper or any other documentation provided by the customer.	Not Relevant

Gas Limit and Loops	**Custom**	Transaction execution costs should not depend dramatically on the amount of data stored on the contract. There should not be any cases when execution fails due to the block Gas limit.	Passed
Style guide violation	**Custom**	Style guides and best practices should be followed.	Failed
Requirements Compliance	**Custom**	The code should be compliant with the requirements provided by the Customer.	Not Relevant
Repository Consistency	**Custom**	The repository should contain a configured development environment with a comprehensive description of how to compile, build and deploy the code.	Passed
Tests Coverage	**Custom**	The code should be covered with unit tests. Test coverage should be 100%, with both negative and positive cases covered. Usage of contracts by multiple users should be tested.	Passed

System Overview

Red Fox is ERC721 and ERC1155 NFT system with the following contracts:

- RFOXFactoryStandard - factory contract to create new RFOXNFTStandart and store their addresses.
- RFOXFactoryStandardBotPrevention - factory contract to create new RFOXNFTStandartBotPrevention and store their addresses.
- RFOXFactoryWhitelist - factory contract to create new RFOXFactoryWhiteList and store their addresses.
- RFOXFactoryWhitelistBotPrevention - factory contract to create new RFOXFactoryWhiteListBotPrevention and store their addresses.
- BaseRFOXNFT - base contract with functionality to work with the other project`s contracts.
- BaseRFOXNFTPresale - base contract for presale and whitelist mechanism.
- RFOXNFTPresale - contract for implementation of the presale of NFT.
- RFOXNFTSale - contract with public NFT selling function.
- RFOXNFTSignaturePresale - contract with a signature presale function.
- RFOXNFTSignatureSale - contract with the extension for the base contract, adding the signature mechanism.
- ParamStructs - contract with parameters for another project`s contract.
- RFOXNFTStandard – contract with the initializing function of the standard RFOX NFT.
- RFOXNFTStandardBotPrevention – contract with the initializing function of the standard RFOX NFT with bot prevention.
- RFOXNFTWhitelist – contract with the initializing function of the standard RFOX NFT with a presale for whitelist.
- RFOXNFTWhitelistBotPrevention – contract with the initializing function of the standard RFOX NFT with a presale for whitelist and bot prevention.
- RFOXFactoryStandard1155 - factory contract to create new RFOXNFTStandart1155 and store their addresses.
- RFOXFactoryStandardBotPrevention1155 - factory contract to create new RFOXNFTStandartBotPrevention1155 and store their addresses.
- RFOXFactoryWhitelist1155 - factory contract to create new RFOXFactoryWhiteList1155 and store their addresses.
- RFOXFactoryWhitelistBotPrevention1155 - factory contract to create new RFOXFactoryWhiteListBotPrevention1155 and store their addresses.
- BaseRFOXNFT1155 - base contract with functionality to work with the other project`s ERC1155 contracts.
- BaseRFOXNFTPresale1155 - base contract for presale and whitelist mechanism for ERC1155 contracts.
- RFOXNFTPresale1155 - contract for implementation of the presale of ERC1155 tokens.
- RFOXNFTSale1155 - contract with public ERC1155 tokens selling function.

- RFOXNFTSignaturePresale1155 - contract with a signature presale function.
- RFOXNFTSignatureSale1155 - contract with the extension for the base contract, adding the signature mechanism.
- ParamStructs1155 - contract with parameters for another project`s ERC1155 contracts.
- RFOXNFTStandard1155 – contract with the initializing function of the standard RFOX NFT and a function for updating token settings.
- RFOXNFTStandardBotPrevention1155 – contract with the initializing function of the standard RFOX NFT with bot prevention and a function for updating token settings.
- RFOXNFTWhitelist1155 – contract with the initializing function of the standard RFOX NFT with a presale for whitelist and a function for updating token settings.
- RFOXNFTWhitelistBotPrevention1155 – contract with the initializing function of the standard RFOX NFT with a presale for whitelist and bot prevention and a function for updating token settings.
- IRFOXFactory - interface for factory contracts.

Privileged roles

- The Owner - can mint tokens, withdraw funds, update token`s data and price, call *createNFT* function in factory contracts, set base URI and maximum number of tokens per transaction, pause and unpause transactions, change authorized signer address, activate and deactivate whitelist feature and update Merkle root.

Findings

🟥🟥🟥🟥 Critical

No critical severity issues were found.

🟨🟨🟨 High

1. **Owner can stop the project`s transactions.**

   The owner can pause and unpause token buying functions.

   This can lead to token selling manipulation.

   Contracts: RFOXNFTPresale.sol, RFOXNFTSale.sol, RFOXNFTSignaturePresale.sol, RFOXNFTSignatureSale.sol

   Functions: buyNFTsPresale, buyNFTsPublic

   Recommendation: Add highly permissive functionality to the documentation.

   Status: Fixed (revised commit: 731ccbd; Documentation)

2. **Highly permissive owner access.**

   The owner can mint tokens to a certain address, change the price of the tokens, change the maximum tokens number per transaction and change presale values, all after the start of sales.

   This can lead to token manipulation.

   Contracts: BaseRFOXNFT.sol, BaseRFOXNFTPresale.sol, BaseRFOXNFT1155.sol

   Functions: safeMint, setMaxTokensPerTransaction, setTokenPrice, updateMaxMintedPresalePerAddress, setTokenPricePresale

   Recommendation: Add highly permissive functionality to the documentation.

   Status: Fixed (revised commit: 731ccbd; Documentation)

🟩🟩 Medium

1. **Multisig wallets will be rejected.**

   Project`s architecture has a restriction on buying any tokens for contracts, which can not be recommended due to the fact that smart contracts, such as multi-sig, often can be big buyers.

   This can lead to a loss of profit

   Contracts: BaseRFOXNFT.sol, BaseRFOXNFT1155.sol

   Functions: callerIsUser

Recommendation: Refactor the restriction functions.

Status: Mitigated (with customer notice)

Customer notice: A feature that we build in to prevent smart contracts/bots from interacting with our contracts.

Low

1. Floating pragma.

The project`s contracts use floating pragma ^0.8.0

Locking the pragma helps 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.

Contracts: RFOXFactoryStandard.sol, RFOXFactoryStandardBotPrevention.sol, RFOXFactoryWhitelist.sol, RFOXFactoryWhitelistBotPrevention.sol, BaseRFOXNFT.sol, BaseRFOXNFTPresale.sol, RFOXNFTPresale.sol, RFOXNFTSale.sol, RFOXNFTSignaturePresale.sol, RFOXNFTSignatureSale.sol, ParamStructs.sol, RFOXNFTStandard.sol, RFOXNFTStandardBotPrevention.sol, RFOXNFTWhitelist.sol, RFOXNFTWhitelistBotPrevention.sol, RFOXFactoryStandard1155.sol, RFOXFactoryStandardBotPrevention1155.sol, RFOXFactoryWhitelist.sol, RFOXFactoryWhitelistBotPrevention1155.sol, RFOXNFTPresale1155.sol, RFOXNFTSale1155.sol, RFOXNFTSignaturePresale1155.sol, RFOXNFTSignatureSale1155.sol, ParamStructs1155.sol, TokenStructs.sol, RFOXNFTStandard1155.sol, RFOXNFTStandardBotPrevention1155.sol, RFOXNFTWhitelist1155.sol, RFOXNFTWhitelistBotPrevention1155.sol, IRFOXFactory.sol, MockContractBuyer.sol, MockContractBuyer1155.sol, MockERC20.sol, MockReceiver.sol, MockReceiver1155.sol

Recommendation: Consider locking the pragma version whenever possible and avoid using floating pragma in the final deployment.

Status: Fixed (revised commit: 731ccbd)

Disclaimers

Hacken Disclaimer

The smart contracts given for audit have been analyzed by the best industry practices at the date of this report, with cybersecurity vulnerabilities and issues in smart contract source code, the details of which are disclosed in this report (Source Code); the Source Code compilation, deployment, and functionality (performing the intended functions).

The audit makes no statements or warranties on the security of the code. It also cannot be considered a sufficient assessment regarding the utility and safety of the code, bug-free status, or any other contract statements. While we have done our best in conducting the analysis and producing this report, it is important to note that you should not rely on this report only — we recommend proceeding with several independent audits and a public bug bounty program to ensure the security of smart contracts.

Technical Disclaimer

Smart contracts are deployed and executed on a blockchain platform. The platform, its programming language, and other software related to the smart contract can have vulnerabilities that can lead to hacks. Thus, the audit cannot guarantee the explicit security of the audited smart contracts.