AuditAce

FROM INCEPTION TO SUCCESS

NFT AUDIT

Smart Contract Audit Report

ABOUT AUDITACE

Audit Ace is built, to combat financial fraud in the cryptocurrency industry, a growing security firm that provides audits, Smart contract creation, and end-to-end solutions to all crypto-related queries.

Website - https://auditace.tech/

Telegram - https://t.me/Audit_Ace

Twitter - https://twitter.com/auditace_

Github - https://github.com/Audit-Ace

Overview

AUDITACE team has performed a line-by-line manual analysis and automated review of smart contracts. Smart contracts were analyzed mainly for common contract vulnerabilities, exploits, and manipulation hacks.

Audit Result: Passed with medium risk

Audit Date: January 13, 2023

KYC :Not Done till date of Audit

Audit Team: TEAM AUDITACE

Disclaimer

All the content provided in this document is for general information only and should not be used as financial advice or a reason to buy any investment. Team provides no guarantees against the sale of team tokens or the removal of liquidity by the project audited in this document. Always Do your own research and protect yourselves from being scammed. The Auditace team has audited this project for general information and only expresses their opinion based on similar projects and checks from popular diagnostic tools. Under no circumstances did Auditace receive a payment to manipulate those results or change the awarding badge that we will be adding in our website. Always Do your own research and protect yourselves from scams. This document should not be presented as a reason to buy or not buy any particular token. The Auditace team disclaims any liability for the resulting losses.

Used Tools

Manual Review -Manual Review – Forge (Foundry)

Audit Scope:

- **NFTBox.sol**
- **NFTClaim.sol**

Token Summary

Parameter	Result
Proxy Address	0xb4eEBB3a9Ee0DaA14303CAbc09301adE70Ceb742
Token Type	BEP 20
Contract Checksum	3a02f2284fd482886e2fd54898fea3001b3b246c3f7f3a6b1ce321437aeb91b4
Decimals	-
Proxy	Yes
Platform	Binance Smart Chain
Compiler	v0.8.2+commit.661d1103
Contract Name	-
Symbol	-
License Type	-
Language	Solidity

AUDIT METHODOLOGY

The auditing process will follow a routine as special considerations by Auditace:

- Review of the specifications, sources, and instructions provided to Auditace to make sure the contract logic meets the intentions of the client without exposing the user's funds to risk.

- Manual review of the entire codebase by our experts, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.

- Specification comparison is the process of checking whether the code does what the specifications, sources, and instructions provided to Auditace describe.

- Test coverage analysis determines whether the test cases are covering the code and how much code isexercised when we run the test cases.

- Symbolic execution is analysing a program to determine what inputs cause each part of a program to execute.

- Reviewing the codebase to improve maintainability, security, and control based on the established industry and academic practices.

Issues Checking Status

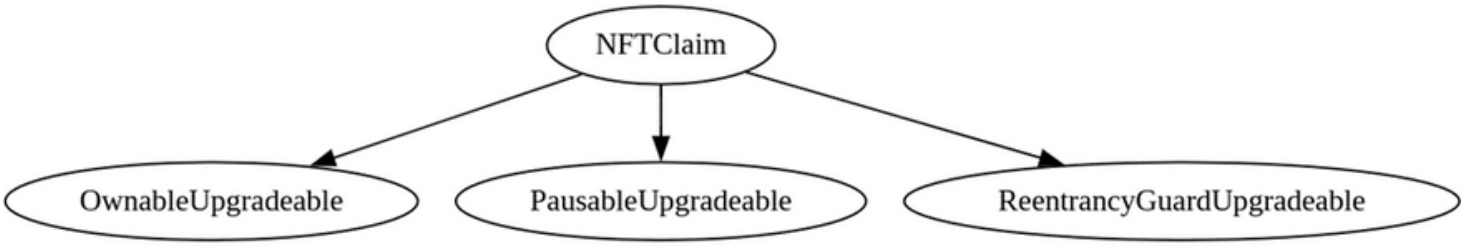
No	Issue Description	Checking Status
1	Compiler warnings.	Passed
2	Race conditions and Reentrancy. Cross-function race conditions.	Passed
3	Possible delays in data delivery.	Passed
4	Oracle calls.	Passed
5	Front running.	Passed
6	Timestamp dependence.	Passed
7	Integer Overflow and Underflow.	Passed
8	DoS with Revert.	Passed
9	DoS with block gas limit.	Passed
10	Methods execution permissions.	Passed
11	Design Logic.	Passed
12	Cross-function race conditions.	Passed
13	Safe Zeppelin module.	Passed
14	Malicious Event log.	Passed
15	Scoping and Declarations.	Passed
16	Fallback function security.	Passed
17	Arithmetic accuracy.	Passed

SWC ATTACK TEST

SWC ID	Description	Test Result
SWC-100	Function Visibility	Passed
SWC-101	Integer Overflow and Underflow	Passed
SWC-102	Outdated Compiler Version	Passed
SWC-103	Floating Pragma	Passed
SWC-104	Unchecked Call Return Value	Passed
SWC-105	Unprotected Ether Withdrawal	Passed
SWC-106	Unprotected SELFDESTRUCT Instruction	Passed
SWC-107	Re-entrancy	Passed
SWC-108	State Variable Default Visibility	Passed
SWC-109	Uninitialized Storage Pointer	Passed
SWC-110	Assert Violation	Passed
SWC-111	Use of Deprecated Solidity Functions	Passed
SWC-112	Delegate Call to Untrusted Callee	Passed
SWC-113	DoS with Failed Call	Passed
SWC-114	Transaction Order Dependence	Passed
SWC-115	Authorization through tx.origin	Passed
SWC-116	Block values as a proxy for time	Passed

SWC ID	Description	Test Result
SWC-117	Signature Malleability	Passed
SWC-118	Incorrect Constructor Name	Passed
SWC-119	Shadowing State Variables	Passed
SWC-120	Weak Sources of Randomness from Chain Attributes	Passed
SWC-121	Missing Protection against Signature Replay Attacks	Passed
SWC-122	Lack of Proper Signature Verification	Passed
SWC-123	Requirement Violation	Passed
SWC-124	Write to Arbitrary Storage Location	Passed
SWC-125	Incorrect Inheritance Order	Passed
SWC-126	Insufficient Gas Grieving	Passed
SWC-127	Arbitrary Jump with Function Type Variable	Passed
SWC-128	DoS With Block Gas Limit	Passed
SWC-129	Typographical Error	Passed
SWC-130	Right-To-Left-Override control character (U+202E)	Passed
SWC-131	Presence of unused variables	Passed
SWC-132	Unexpected Ether balance	Passed
SWC-133	Hash Collisions with Multiple Variable Length Arguments	Passed
SWC-134	Unencrypted Private Data On-Chain	Passed

Inheritance Tree



Summary

- NFTBuy: minting random NFTs (each one with different chance) by providing mint fee as nftToken
- NFTClaim: claiming NFTs if whitelisted by owner

Classification of Risks

Severity	Description
◆ **High-Risk**	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.
◆ **Medium-Risk**	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.
◆ **Low-Risk**	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.
◆ **Gas Optimization /Suggestion**	A vulnerability that has an informational character but is not affecting any of the code.

Findings

Severity	Found
◆ **High-Risk**	**2**
◆ **Medium-Risk**	**1**
◆ **Low-Risk**	**0**
◆ **Gas Optimization / Suggestions**	**4**

MANUAL AUDIT

HIGH RISK FINDINGS

Centralization - : approving proxy : NFTBuy contract requires buyer to approve proxy contract with the amount of mint fee tokens, if proxy contract is approved with more than necessary amount of tokens, then a malicious owner is able to upgrade the implementation contact and spend users nft token funds.

Recommendation:

make sure to approve proxy contract only with necessary amounts

Centralization : upgradeable contracts: contracts are upgradeable, meaning that new implementations may be totally different than what is audited here, this may open up new security risks (like the one described in previous issue)

MANUAL AUDIT

MEDIUM RISK FINDINGS

Centralization - lastSeed variable is never changed (always zero) , this can increase the possibability of predicting seeds

Suggestions - set lastSeed to latest calculated seed,

MANUAL AUDIT

Gas Optimizations

- set this variables as constant: PERCENT_DEVIDER
- use external keyword instead of public for setter functions
- redundant variable typeBox (its always 1)

Suggestions

- use up to 3 indexed arguments in events