SMART CONTRACT AUDIT

ZOKYO.

Dec 8th, 2021 | v. 1.0

PASS

Zokyo Security Team has concluded that this smart contract passes security qualifications to be listed on digital asset exchanges.

TECHNICAL SUMMARY

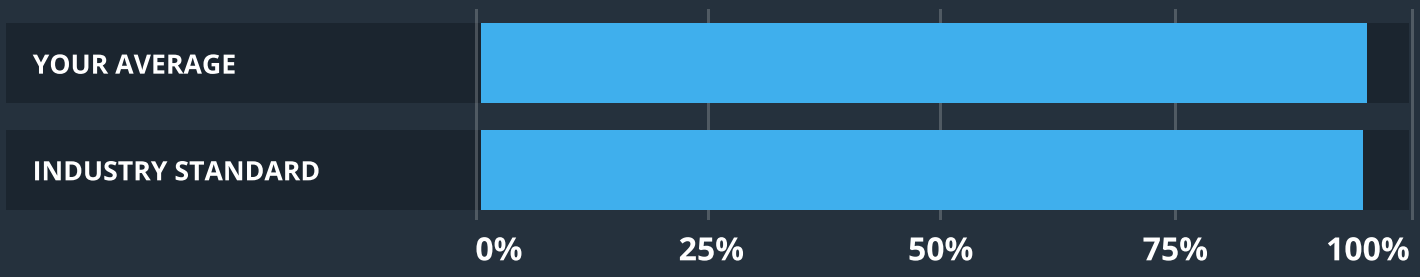
This document outlines the overall security of the Endemic smart contracts, evaluated by Zokyo's Blockchain Security team.

The scope of this audit was to analyze and document the Endemic smart contract codebase for quality, security, and correctness.

Contract Status

There were no critical issues found during the audit.

Testable Code



The testable code is 95.5%, which is above the industry standard of 95%.

It should be noted that this audit is not an endorsement of the reliability or effectiveness of the contract, rather limited to an assessment of the logic and implementation. In order to ensure a secure contract that's able to withstand the Ethereum network's fast-paced and rapidly changing environment, we at Zokyo recommend that the Endemic team put in place a bug bounty program to encourage further and active analysis of the smart contract.

TABLE OF CONTENTS

AUDITING STRATEGY AND TECHNIQUES APPLIED

The Smart contract's source code was taken from the Endemic archive file.

Repository:
https://github.com/Endemic-NFT/endemic-contracts/commit/bd78e0627583afe5bf944d6c0eae1e653390c000

Last commit:
bd78e0627583afe5bf944d6c0eae1e653390c000

Contracts under the scope:

- Endemic;
- EndemicMasterNFT;
- EndemicNFTFactory;
- EndemicNFT;
- EndemicVesting;
- EndemicTokenMining;
- Marketplace.

Throughout the review process, care was taken to ensure that the token contract:

- Implements and adheres to existing Token standards appropriately and effectively;
- Documentation and code comments match logic and behavior;
- Distributes tokens in a manner that matches calculations;
- Follows best practices in efficient use of gas, without unnecessary waste;
- Uses methods safe from reentrance attacks;
- Is not affected by the latest vulnerabilities;
- Whether the code meets best practices in code readability, etc.

Zokyo's Security Team has followed best practices and industry-standard techniques to verify the implementation of Endemic smart contracts. To do so, the code is reviewed line-by-line by our smart contract developers, documenting any issues as they are discovered. Part of this work includes writing a unit test suite using the Truffle testing framework. In summary, our strategies consist largely of manual collaboration between multiple team members at each stage of the review:

1	Due diligence in assessing the overall code quality of the codebase.
2	Cross-comparison with other, similar smart contracts by industry leaders.
3	Testing contract logic against common and uncommon attack vectors.
4	Thorough, manual review of the codebase, line-by-line.

SUMMARY

The Zokyo team has conducted a security audit of the given codebase. The contracts provided for an audit are in excellent condition and are written and structured. During the auditing process, there were no issues found.

Based on the conducted audit, we give a score of 100 to the aforementioned contracts.

Zokyo auditing team can state that the contracts are full production-ready and bear no security or operational risk.

STRUCTURE AND ORGANIZATION OF DOCUMENT

For ease of navigation, sections are arranged from most critical to least critical. Issues are tagged “Resolved” or “Unresolved” depending on whether they have been fixed or addressed. Furthermore, the severity of each issue is written as assessed by the risk of exploitation or other unexpected or otherwise unsafe behavior:

Critical

The issue affects the ability of the contract to compile or operate in a significant way.

High

The issue affects the ability of the contract to compile or operate in a significant way.

Medium

The issue affects the ability of the contract to operate in a way that doesn’t significantly hinder its behavior.

Low

The issue has minimal impact on the contract’s ability to operate.

Informational

The issue has no impact on the contract’s ability to operate.

COMPLETE ANALYSIS

During the auditing process (both manual part and testing part) no issues were identified.

	Endemic	EndemicNFT	EndemicVesting
Re-entrancy	Pass	Pass	Pass
Access Management Hierarchy	Pass	Pass	Pass
Arithmetic Over/Under Flows	Pass	Pass	Pass
Unexpected Ether	Pass	Pass	Pass
Delegatecall	Pass	Pass	Pass
Default Public Visibility	Pass	Pass	Pass
Hidden Malicious Code	Pass	Pass	Pass
Entropy Illusion (Lack of Randomness)	Pass	Pass	Pass
External Contract Referencing	Pass	Pass	Pass
Short Address/ Parameter Attack	Pass	Pass	Pass
Unchecked CALL Return Values	Pass	Pass	Pass
Race Conditions / Front Running	Pass	Pass	Pass
General Denial Of Service (DOS)	Pass	Pass	Pass
Uninitialized Storage Pointers	Pass	Pass	Pass
Floating Points and Precision	Pass	Pass	Pass
Tx.Origin Authentication	Pass	Pass	Pass
Signatures Replay	Pass	Pass	Pass
Pool Asset Security (backdoors in the underlying ERC-20)	Pass	Pass	Pass

	EndemicMasterNFT	EndemicNFTFactory
Re-entrancy	Pass	Pass
Access Management Hierarchy	Pass	Pass
Arithmetic Over/Under Flows	Pass	Pass
Unexpected Ether	Pass	Pass
Delegatecall	Pass	Pass
Default Public Visibility	Pass	Pass
Hidden Malicious Code	Pass	Pass
Entropy Illusion (Lack of Randomness)	Pass	Pass
External Contract Referencing	Pass	Pass
Short Address/ Parameter Attack	Pass	Pass
Unchecked CALL Return Values	Pass	Pass
Race Conditions / Front Running	Pass	Pass
General Denial Of Service (DOS)	Pass	Pass
Uninitialized Storage Pointers	Pass	Pass
Floating Points and Precision	Pass	Pass
Tx.Origin Authentication	Pass	Pass
Signatures Replay	Pass	Pass
Pool Asset Security (backdoors in the underlying ERC-20)	Pass	Pass

	EndemicTokenMining	Marketplace
Re-entrancy	Pass	Pass
Access Management Hierarchy	Pass	Pass
Arithmetic Over/Under Flows	Pass	Pass
Unexpected Ether	Pass	Pass
Delegatecall	Pass	Pass
Default Public Visibility	Pass	Pass
Hidden Malicious Code	Pass	Pass
Entropy Illusion (Lack of Randomness)	Pass	Pass
External Contract Referencing	Pass	Pass
Short Address/ Parameter Attack	Pass	Pass
Unchecked CALL Return Values	Pass	Pass
Race Conditions / Front Running	Pass	Pass
General Denial Of Service (DOS)	Pass	Pass
Uninitialized Storage Pointers	Pass	Pass
Floating Points and Precision	Pass	Pass
Tx.Origin Authentication	Pass	Pass
Signatures Replay	Pass	Pass
Pool Asset Security (backdoors in the underlying ERC-20)	Pass	Pass

CODE COVERAGE AND TEST RESULTS FOR ALL FILES

Tests written by Zokyo Security team

As part of our work assisting Endemic in verifying the correctness of their contract code, our team was responsible for writing integration tests using the Truffle testing framework.

Tests were based on the functionality of the code, as well as a review of the Endemic contract requirements for details about issuance amounts and how the system handles these.

Code Coverage

The resulting code coverage (i.e., the ratio of tests-to-code) is as follows:

FILE	% STMTS	% BRANCH	% FUNCS	% LINES	UNCOVERED LINES
Endemic.sol	100.00	100.00	100.00	100.00	
EndemicMasterNFT.sol	94.12	62.50	85.71	94.12	
EndemicNFT.sol	100.00	100.00	100.00	100.00	
EndemicNFTFactory.sol	90.91	100.00	66.67	100.00	
EndemicTokenMining.sol	85.00	87.50	80.00	85.00	
EndemicVesting.sol	98.50	76.32	100.00	99.00	
Marketplace.sol	100.00	100.00	100.00	100.00	
All files	**95.50**	**89.47**	**90.34**	**96.87**	

Test Results

```
Contract: Endemic
  Contract initialization
    ✓ SUCCESS: Should return ENDEMIC as the contract name
```

✓ FAILURE: Should not have ABCDEF as contract name
✓ SUCCESS: Should have END as contract symbol
✓ FAILURE: Should not return ABC as contract symbol
✓ SUCCESS: Should have correct owner address
✓ FAILURE: Owner address should not be undefined
✓ SUCCESS: contract should be able to add a new signer (40ms)
✓ SUCCESS: contract should be able to renounce signer (66ms)

Mint Function

✓ SUCCESS: Should mint token with tokenId = 1 (59ms)
✓ FAILURE: should not be able to mint same tokenId twice (143ms)
✓ SUCCESS: Should mint token with correct token uri (55ms)
✓ SUCCESS: Should emit Mint event with correct parameters (46ms)
✓ FAILURE: should not be able to mint if the address does not have signer role (39ms)

Base token uri

✓ SUCCESS: should update token uri (67ms)
✓ FAILURE: Nonowner account cannot update uri

Burn Function

✓ should successfully burn token by owner (98ms)
✓ FAILURE: Should fail if nonowner burn tokens (71ms)

Contract: EndemicMasterNFT

Contract initialization

✓ SUCCESS: should have name as Endemic Master Keys
✓ FAILURE: should have name as Endemic Master Keys
✓ SUCCESS: Should have correct owner address
✓ FAILURE: Should not have wrong owner address
✓ SUCCESS: Should have correct symbol as EMK
✓ FAILURE: Should not have wrong symbol as SSS

mintNFT Function

✓ SUCCESS: should mint nft if address is owner
✓ FAILURE: should fail to mint nft if address is not owner
✓ SUCCESS: Should not mint more than max supply (1294ms)
✓ SUCCESS: Should have correct tokenId (1651ms)

setBaseTokenUri

✓ Owner should update token uri (89ms)
✓ FAILURE: should fail to update token uri if not owner

Burn

✓ SUCCESS: should only burn token by owner (106ms)
✓ FAILURE: should not burn token if not owner (60ms)

Shares distribution
- ✓ should distribute shares to all token owners (194ms)
- ✓ should fail to distribute shares if a distributor is not a caller

publicMint Function
- ✓ SUCCESS: Should mint if sale is activated
- ✓ FAILURE: Should not mint if sale is not activated
- ✓ SUCCESS: Should mint for sufficient amount of eth (53ms)
- ✓ FAILURE: Should not mint for insufficient value

setSalePrice
- ✓ SUCCESS: should set sale price
- ✓ FAILURE: should not set sale price if called by nonowner address

Contract: EndemicNFT

Contract initialization
- ✓ SUCCESS: should have correct contract name
- ✓ FAILURE: should not have wrong contract name
- ✓ SUCCESS: should have correct owner address
- ✓ FAILURE: should not have wrong owner address
- ✓ SUCCESS: should have correct symbol
- ✓ FAILURE: should not have wrong symbol

Mint Function
- ✓ SUCCESS: should mint an NFT if caller is owner (46ms)
- ✓ FAILURE: should not mint an NFT if caller is not owner
- ✓ SUCCESS: should mint an NFT with correct uri (38ms)
- ✓ SUCCESS: should mint an NFT if address is approved (66ms)

setDefaultApproval Function
- ✓ SUCCESS: sets address as default approver

Contract: EndemicNFTFactory
- ✓ should have initial roles
- ✓ SUCCESS: should be able to add new minters if admin
- ✓ FAILURE: should not be able to add new minters if not admin (97ms)

createToken
- ✓ SUCCESS: should deploy a new contract correctly if minter (177ms)
- ✓ FAILURE: should fail to create token contract if caller is not minter (110ms)

Contract: EndemicTokenMining

claim Function
- ✓ SUCCESS: Should claim balance if signed by owner (92ms)

```
    ✓ FAILURE: Should not be able to claim balance if not signed by owner
    ✓ FAILURE: Should not claim balance if not in balances array
    ✓ FAILURE: Should not claim balance they don;t own (64ms)
    ✓ SUCCESS: Should claim all balances by an andress (94ms)

Contract: EndemicVesting
  ✓ FAILURE: Should revert if duration is less than cliff
  ✓ FAILURE: Should revert if duration is not in harmony with interval
  ✓ FAILURE: Should revert when cliff is more than 2 years
  addTokenGrant function
    ✓ SUCCESS: Should add token grant and emit GrantAdded event (55ms)
    ✓ FAILURE: Should revert when duration is more than 10 years
    ✓ FAILURE: Should revert if amountVestedPerDay > 0
  removeGrant
    ✓ SUCCESS: Should remove grant (88ms)
    ✓ SUCCESS: Should remove grant with vested amount (100ms)
  grantClaim
    ✓ SUCCESS: Should claim grant (251ms)
    ✓ SUCCESS: Should calculated correct token vested per day (51ms)
  changeMultiSig
    ✓ SUCCESS: Should change multisig wallet
    ✓ FAILURE: Should not change multsig

72 passing (23s)
```

We are grateful to have been given the opportunity to work with the Endemic team.

The statements made in this document should not be interpreted as investment or legal advice, nor should its authors be held accountable for decisions made based on them.

Zokyo's Security Team recommends that the Endemic team put in place a bug bounty program to encourage further analysis of the smart contract by third parties.

ZOKYO.