SMART CONTRACT AUDIT

ZOKYO.

January 11th, 2022 | v. 1.0

PASS

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

TECHNICAL SUMMARY

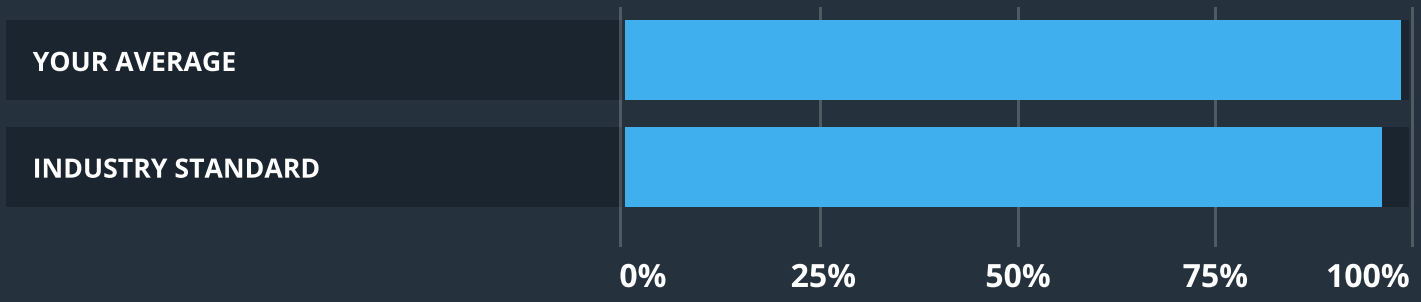
This document outlines the overall security of the Tarantino smart contracts, evaluated by Zokyo's Blockchain Security team.

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

Contract Status

There were some no critical and medium issues found during the audit.

Testable Code



The testable code is 100% 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 security of the contract we at Zokyo recommend that the Tarantino 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 Tarantino smart contract's source code was taken from the repository provided by the Tarantino team

Repository
https://github.com/NFTrade/tarantino-contracts/commit/775d623f2e82ad888b7514119c6d161eff702b24
Last commit
775d623f2e82ad888b7514119c6d161eff702b24

Within the scope of this audit Zokyo auditors have reviewed the following contract(s):

- Tarantino.sol

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

- Implements and adheres to existing standards appropriately and effectively;
- Documentation and code comments match logic and behavior;
- Follows best practices in efficient use of resources, 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 Tarantino 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. 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.	3	Testing contract logic against common and uncommon attack vectors.
2	Cross-comparison with other, similar smart contracts by industry leaders.	4	Thorough, manual review of the codebase, line-by-line.

EXECUTIVE SUMMARY

There were no critical issues found during the audit. During the audit Zokyo team has found some informational issues only.

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. Issues tagged “Verified” contain unclear or suspicious functionality that either needs explanation from the Customer’s side or it is an issue that the Customer disregards as an issue. 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 contract in such a way that funds may be lost, allocated incorrectly, or otherwise result in a significant loss.

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

LOW | RESOLVED

The addSigners function can allow address(o) to be added as a signer

Recommendation:
Ensure address(0) can not be made a signer

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

CODE COVERAGE AND TEST RESULTS FOR ALL FILES

Tests written by Tarantino team

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

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

```
Contract: Tarantino
  Auction
    ✓ Add to signers and sign (387ms)
    ✓ adds sale (138ms)
    ✓ bid (255ms)
    ✓ not on whitelist (842ms)
    ✓ bid on same amount (165ms)
    ✓ bid on less amount than current highest offer (148ms)
    ✓ bid on different amount but same bidder (141ms)
    ✓ goes to target price (1405ms)
    ✓ extend saleTotalTime to 12.5 hours when bid on last 30 min (248ms)
  9 passing (4s)
```

FILE	% STMTS	% BRANCH	% FUNCS	% LINES	% UNCOVERED LINES
contracts/	79.59	75	68.75	80.39	
Tarantino.sol	79.59	75	68.75	80.39	115,120,151
All files	79.59	75	68.75	80.39	

CODE COVERAGE AND TEST RESULTS FOR ALL FILES

Tests written by Zokyo Secured team

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

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

```
Contract: Tarantino
    ✓ Should be initialized  correctly (335ms)
    ✓ Should allow new sale to be added then edited and toggled  (327ms)
    ✓ Should toggle contract validity (102ms)
    ✓ Should get sale total time (238ms)
    ✓ Should get sale total time based on different bids (170ms)
    ✓ Should confirm sale is live (100ms)
    ✓ Should add signers and show that they are whitelisted (92ms)
    ✓ Should be able to remove signers (102ms)
    ✓ Should allow new bids to be added (122ms)
    ✓ Should return previous highest bid to the  bidder (170ms)
    ✓ Should prevent bids from a contract (150ms)
    ✓ Should be able to get the highest offer (135ms)
    ✓ Should allow owner to withdraw funds  (126ms)
    ✓ Should prevent bids from been added if parameters are incorrect  (319ms)
  14 passing (2s)
```

FILE	% STMTS	% BRANCH	% FUNCS	% LINES	% UNCOVERED LINES
contracts/	100	100	100	100	
Tarantino.sol	100	100	100	100	
All files	100	100	100	100	

We are grateful to have been given the opportunity to work with the Tarantino 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 Tarantino team put in place a bug bounty program to encourage further analysis of the smart contract by third parties.

ZOKYO.