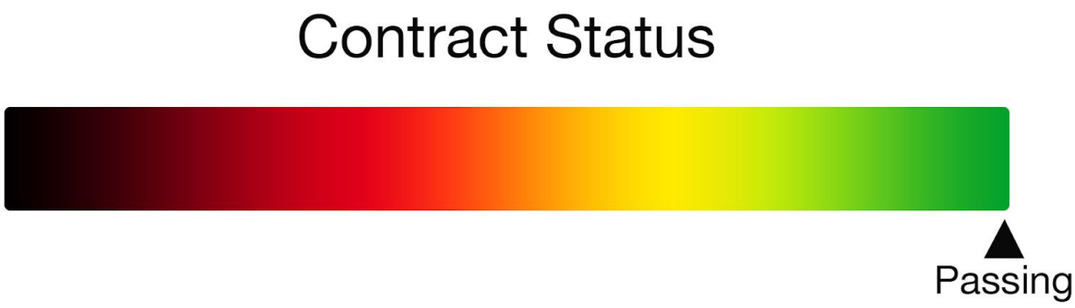
Unchained Capital Contract Audit

Prepared by Hosho
August 30th, 2018

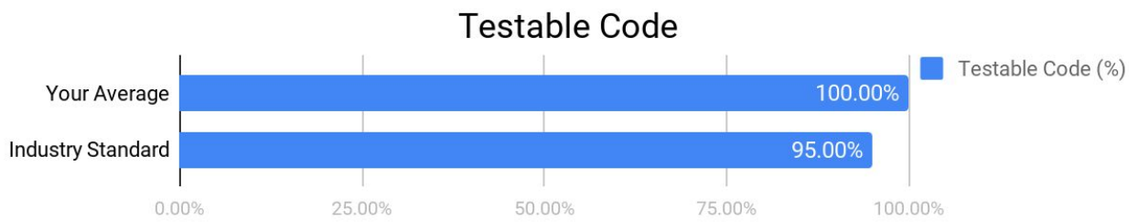
Report Version: 3.0

Executive Summary

This document outlines the overall security of Unchained Capital's smart contract as evaluated by Hosho's Smart Contract auditing team. The scope of this audit was to analyze and document Unchained Capital's contract codebase for quality, security, and correctness.



These contracts have passed the rigorous auditing process performed by the Hosho team. (See Complete Analysis)



Testable code is 100.00% which is greater than the industry standard of 95%. (See Coverage Report)

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 Hosho recommend that the Unchained Capital team put in place a bug bounty program to encourage further and active analysis of the smart contract.

Table Of Contents

1. Auditing Strategy and Techniques Applied

The Hosho team has performed a thorough review of the smart contract code, the latest version as written and updated on August 23rd, 2018. All main contract files were reviewed using the following tools and processes. (See All Files Covered)

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;
- Manages tokens in the intended manner;
- Follows best practices in efficient use of gas, without unnecessary waste;
- Uses methods safe from reentrance attacks; and
- Is not affected by the latest vulnerabilities.

The Hosho team has followed best practices and industry-standard techniques to verify the implementation of Unchained Capital's contract. To do so, the code is reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as they are discovered. Part of this work includes writing a unit test suite using the Meadow 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.
5. Deploying the smart contract to testnet and production networks using multiple client implementations to run live tests.

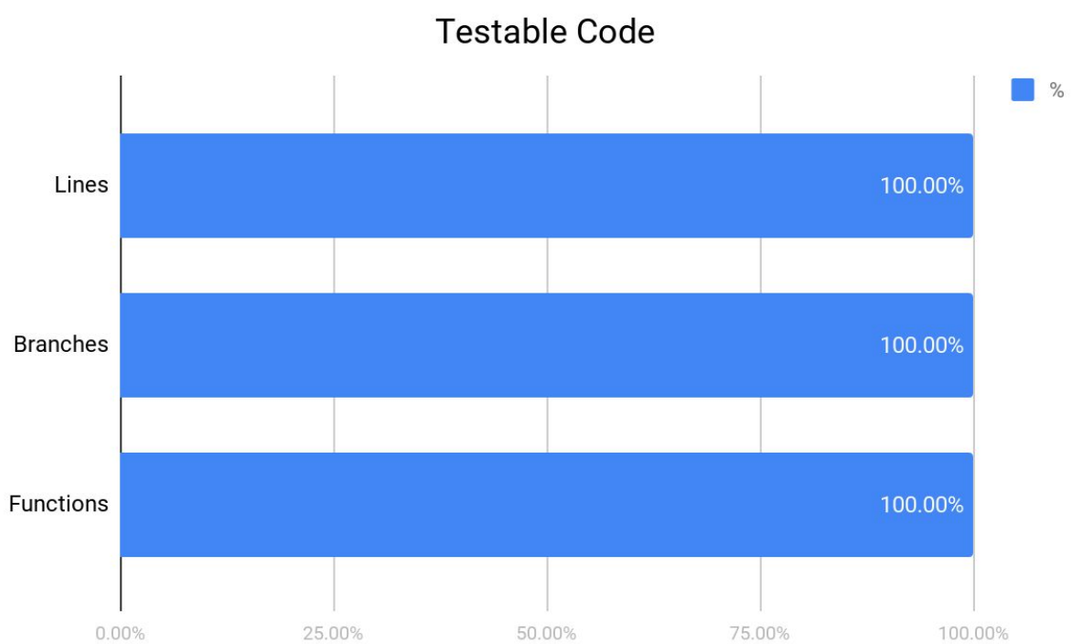
2. Structure Analysis and Test Results

2.1. Summary

The Unchained Capital multisignature wallet is a well constructed piece of Solidity code, designed to work with Trezor and Ledger hardware wallets. It is also capable of working with other software that is able to generate the correct signature from private keys, allowing utilization of standard ETH wallets. As the contract is able to return the internally generated hash for various arguments, it also becomes simple to integrate with 3rd party wallets, given they have the ability generate the required security hashes. Private keys, however, are still required as they must be signed.

2.2 Coverage Report

As part of our work assisting Unchained Capital in verifying the correctness of their contract code, our team was responsible for writing a unit test suite using the Meadow testing framework.



For each file see Individual File Coverage Report

2.3 Failing Tests

No failing tests.

See Test Suite Results for all tests.

3. Complete Analysis

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 still need addressing. 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, and is meant only as additional information.

3.1 Informational: Nonce Overflow

Contract: MultiSig2of3

Explanation

There is no protection from allowing the nonce to `overflow` and reset to 0. While this is highly unlikely, as it would require `(2^256) -1` separate transactions, it would allow replay attacks on all transactions from the contract address.

Resolution

The Unchained Capital team has acknowledged their awareness of this informational item, and also to the extreme improbability of it ever occurring. Per Unchained Capital: “...if you could execute a million transactions per second, it would take `10^63` years to make the nonce overflow, and it would cost more money than exists. Given that the sun will explode in 5 billion years, we think we're probably safe.” The Hosho team agrees with this reasoning and acknowledges this to be realistically impossible, given current restraints on gaslimits, Bremermann's computational limit, and stellar age estimation.

---

4. Closing Statement

The Hosho team is grateful to have been given the opportunity to work with the Unchained Capital team.

The team of experts at Hosho, having backgrounds in all aspects of blockchain, cryptography, and cybersecurity, can say with confidence that the Unchained Capital contract is free of any critical issues.

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.

We at Hosho recommend that the Unchained Capital team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

5. Appendix A

Test Suite Results

HoshoAudit.BasicTests

✓ MultisigConstructor_FirstAndSecondOwnersSame_Revert (268ms)
✓ MultisigConstructor_FirstAndThirdOwnersSame_Revert (226ms)
✓ MultisigConstructor_FirstOwnerZero_Revert (482ms)
✓ MultisigConstructor_SecondAndThirdOwnersSame_Revert (228ms)
✓ MultisigConstructor_SecondOwnerZero_Revert (268ms)
✓ MultisigConstructor_ThirdOwnerZero_Revert (269ms)
✓ MultiSigFallback_SendEth_AcceptEth (62ms)
✓ MultiSigFallback_SendEth_EmitFunded (32ms)

HoshoAudit.SpendingTests

✓ MultiSig_GenerateMessage_GeneratesValidKeccakValue (7ms)
✓ MultiSig_GenerateMessageForTxToSelf_Revert (64ms)
✓ MultiSig_Spend_AllowMultipleSpendsWithIncreasingNonce (1ms)
✓ MultiSig_Spend_IncreaseNonceValidSpend (146ms)
✓ MultiSig_Spend_RequireSignersAreOwners (7ms)
✓ MultiSig_Spend_RequireUniqueSignatures (7ms)
✓ MultiSig_SpendFirstSignerNotOwner_Revert (43ms)
✓ MultiSig_SpendOverContractBalance_Revert (30ms)
✓ MultiSig_SpendSecondSignerNotOwner_Revert (60ms)
✓ MultiSig_SpendWithInvalidSigners_Revert (124ms)
✓ MultiSig_SpendWithValidSigners_EmitSpent (116ms)
✓ MultiSig_SpendWithValidSigners_TransferFunds (115ms)

6. Appendix B

All Contract Files Tested

Commit Hash: f40143e00a378addfc5559ff743f1c8a7ca7fae3

File	Fingerprint (SHA256)
MultiSig2of3.sol	4F436F84AB192BB664AEE206EB7ED80138481B46C9BBA7EC5C70C2774752CEDF

7. Appendix C

Individual File Coverage Report

File	% Lines	% Branches	% Functions
MultiSig2of3.sol	100.00%	100.00%	100.00%
All files	**100.00%**	**100.00%**	**100.00%**