Moxy-One Contract Audit

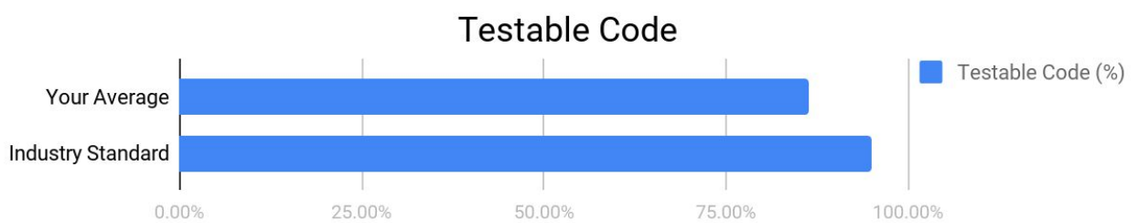
by Hosho, January 2018

NOTICE: *This document is a draft and is not representative of a completed audit.*

Executive Summary

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

The major issues have been resolved but a low level issue remains. See Complete Analysis.



While the testable code is slightly lower than industry standard, all code paths have been manually verified. See Coverage Report.

It should be noted that this audit is not an endorsement of the reliability or effectiveness of the contract; it is merely an assessment of its 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, the Hosho Team recommends that the Moxy-One staff 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 an initial review thorough review of the smart contract code as written and last updated on January 5, 2018. All of the 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 token contract:

- Implements and adheres to existing ERC-20 Token standard 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

The Hosho Team has followed best practices and industry-standard techniques to verify the proper implementation of Moxy-One's token contract. Our staff of expert pentesters and smart contract developers reviewed the contract line by line, documenting any issues as they were discovered. Part of this work included writing a code-specific unit test suite using the Truffle testing framework. As demonstrated, our strategies consist largely of manual collaboration between multiple team members at each stage of the review, including:

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

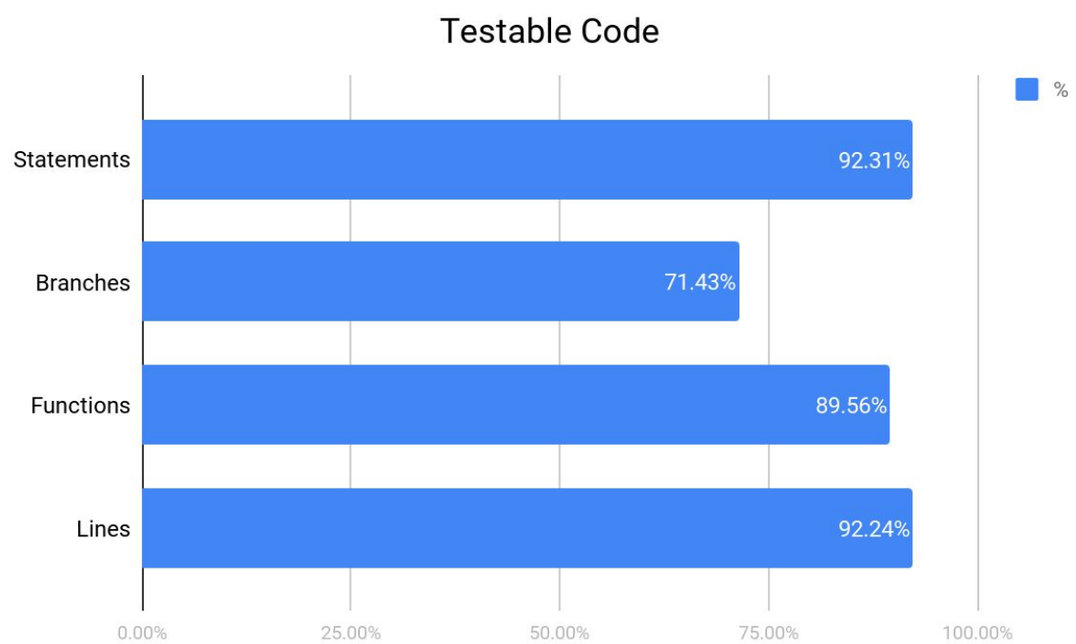
2.1. Summary

The Moxy-One contract is functionally made up of a large, highly customizable CrowdSale contract combined with a well written ERC-20 token. The sale contract is made up of a number of smaller pieces, including multiple proxy contracts for proxy purchases, these are primarily internal callers that call upstream into parent codebases. Despite the overall soundness of the code and the updates made by Frank Bonnet, there still exists a low level risk regarding contract ownership .

Automated code coverage is lower than normal due to the time-intensive functions and high number of checks within the code, however, all paths were manually tested and verified by the Hosho Team.

2.2 Coverage Report

As part of our work assisting Moxy-One in verifying the correctness of their contract code, our team was responsible for writing a unit test suite using the Truffle testing framework.



For individual files see Additional 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 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 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.

7.1. Resolved, High: Pause Security Concern

MoxyOneCrowdsale.sol

Explanation

For the pause and resume functionality, the modifier OnlyOwner is not used, neither are there any checks verifying the user role attempting to execute this function. This makes it possible for any user to pause the sale which is a major security concern.

Resolution

Frank Bonnet has added the OnlyOwner modifier preventing standard users from executing these functions.

7.2. Resolved, Medium: Unused Variables

PersonalCrowdsaleProxyDispatcher.sol

Explanation

The constraints `beneficiary` and `passphraseHash` are defined, but not utilized. Similar contracts use these parameters for security and validation purposes. Without knowing what these functions are for or why they are not called throughout the contract, we cannot be sure as to whether they were intended to be implemented or removed, posing a functionality risk to the contracts.

Resolution

As discussed with the Moxy-One Team and Frank Bonnet, this is as intended for `delegateCall`.

7.3. Unresolved, Low: Ownership Issues

Explanation

Due to the chaining design of the proxy systems, several of the child contracts become owned by their parent contracts. While the `retrieveToken` functionality has been added to these contracts, they are guarded by the `only_owner` check and the parent contracts cannot cause the child to execute the `retrieveToken` function. This causes unreachable code paths and potentially unintended behavior.

---

4. Closing Statement

We are grateful to have been given the opportunity to work with the Moxy-One Team and Frank Bonnet.

Overall, the crowdsale and token contracts are very well written and adhere to ERC-20 guidelines. Automated code coverage is lower than normal due to the time-intensive functions and high number of checks within the code, however, all paths were manually tested and verified by the Hosho Team. While the high and medium level issues have been resolved, there remains a low level issue that requires attention for these contracts to be declared sound.

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 Moxy-One Team put in place a bug bounty program to encourage further analysis of the smart contract by other third parties.

5. Test Suite Results

Contract: SpendToken Burning

✓ Should allocate tokens per the minting function, and validate balances (1682ms)

✓ Should handle burning of tokens via transfer or `transferFrom` (241ms)

✓ Should not allow ETH to be sent to the contract

✓ Should transfer external tokens from itself and the token contract to the owner (76ms)

Contract: Moxy-One Crowdsale

✓ Should the amount of wei raised

✓ Should have a start that is before the end

✓ Should have the presale timing set correctly

✓ Should be not in presale phase before the Beneficiary is set

✓ Should not allow the crowdsale to be sent funds in a non InProgress stage

✓ Should allocate tokens, and update amount raised (388ms)

✓ Should be in presale phase when starting to send funds

✓ Should transfer external tokens from itself and the token contract to the owner (101ms)

✓ Should transfer external tokens from itself with the child being at 0 tokens (72ms)

✓ Should not transfer any tokens if both balances are 0. (53ms)

✓ Should not allow the crowdsale to be sent funds from non-whitelisted addresses

✓ Should not allow the crowdsale to be sent funds under the minimum amount required during presale (62ms)

✓ Should allow the crowdsale to init and set balances (1256ms)

✓ Should not allow the crowdsale to be sent funds that would send it over the maximum presale amount during presale. (244ms)

✓ Should return correct balances for eth and tokens for the various stakeholders

✓ Should not issue a refund until the crowdsale is over

✓ Should permit authorized pause/resume (95ms)

✓ Should permit authorized authentication changes (94ms)

Contract: ERC-20 Compliant Token

✓ Should deploy with Spend Token as the name of the token

✓ Should deploy with SPEND as the symbol of the token

✓ Should deploy with 8 decimals

✓ Should deploy with 0 tokens

✓ Should allocate tokens per the minting function, and validate balances (1499ms)

✓ Should transfer tokens from 0x1bbb1269032bfd0b0fe0851235fc798af6bd3c9b to 0x64a91312c9386f6e8031973b36bda59e224e6b26 (59ms)

✓ Should not transfer negative token amounts

✓ Should not transfer more tokens than you have

✓ Should allow 0x3b44fa9f7511113a8c1a1528070d45b1d7cdd101 to authorize 0x341106cb00828c87cd3ac0de55eda7255e04933f to transfer 1000 tokens

✓ Should not allow 0x3b44fa9f7511113a8c1a1528070d45b1d7cdd101 to authorize 0x341106cb00828c87cd3ac0de55eda7255e04933f to transfer an additional 1000 tokens once authorized, and authorization balance is > 0

✓ Should allow 0x3b44fa9f7511113a8c1a1528070d45b1d7cdd101 to zero out the 0x341106cb00828c87cd3ac0de55eda7255e04933f authorization

✓ Should allow 0xdaef8d8c30eeb858b8c774a8d7d5e92a552bb0d9 to authorize 0x53353ef6da4bbb18d242b53a17f7a976265878d5 for 1000 token spend, and 0x53353ef6da4bbb18d242b53a17f7a976265878d5 should be able to send these tokens to 0x341106cb00828c87cd3ac0de55eda7255e04933f (129ms)

✓ Should not allow 0x53353ef6da4bbb18d242b53a17f7a976265878d5 to transfer negative tokens from 0xdaef8d8c30eeb858b8c774a8d7d5e92a552bb0d9

✓ Should not allow 0x53353ef6da4bbb18d242b53a17f7a976265878d5 to transfer more tokens than permitted from 0xdaef8d8c30eeb858b8c774a8d7d5e92a552bb0d9

Contract: Observable

✓ Should not allow a non-owner to add an observer

✓ Should not allow a non-owner to remove an observer

✓ Should allow an owner to add an observer, but only on the contract it's for (54ms)

✓ Should allow an owner to remove an observer, but only on the contract it's for (79ms)

✓ Should allow you to get an observer at the numerical index

✓ Should not allow double adding/removing an observer (67ms)

Contract: Moxy-One Ownership

✓ Should return if someone is an owner or not

✓ Should return the contract owner

✓ Should return the contract owner

✓ Should return the number of owners

✓ Should allow the adding/removing of owners (152ms)

✓ Should allow the transfer of ownership if single-owned (Whitelist)

Contract: Spend Token

✓ Should not allow ETH to be sent to the contract

✓ Should allow the crowdsale to init and set balances (1478ms)

✓ Should be locked against transfers to start

✓ Should not permit transfers while locked

Contract: Moxy-One Crowdsale Time-Shifting Tests

✓ Should instantiate proxy factory with correct crowdsale and token

✓ Should not allow the crowdsale to be sent funds in a non InProgress stage

✓ Should create a new CrowdsaleProxy for msg.sender (778ms)

✓ Should not let the beneficiary self-destruct the contract before 2 years are up

✓ Should create a new CrowdsaleProxy for a beneficiary (38ms)

✓ Should createPersonalDepositAddress for msg.sender

✓ Should createPersonalDepositAddressFor for msg.sender (450ms)

✓ createPersonalDepositAddressFor tests (136ms)

✓ Should instantiate pool factory with correct crowdsale and token

✓ Should createProxyPool and register pool as observer (916ms)

✓ Should createAccumulatingPool and register pool as observer (844ms)

✓ Should allocate tokens, and update amount raised during presale (410ms)

✓ Should transfer external tokens from itself and the token contract to the owner (42ms)

✓ Should exit presale once the first time-marker is hit

✓ Should set whitelists in preparation to send funds (116ms)

✓ Should not be able to finalize the crowdsale if the amount raised is too low

✓ Should not allow excessively small payments inside of the ICO.

✓ Should add refundable funds for a payment made after the first stage, and show as refundable (472ms)

✓ Should not allow transactions while the contract is paused (60ms)

✓ Should be able to finalize the crowdsale and rate should become 0 (499ms)

✓ Should return correct balances for eth and tokens for the various stakeholders

✓ Should allow presale ETH to be withdrawn by the stakeholders immediately after the crowdsale has ended (218ms)

✓ Should not have any more eth to send after being drained (242ms)

✓ Should not have any tokens available for immediate withdraw by the stakeholders immediately after the crowdsale has ended (44ms)

✓ Should allow all token releases after 150 days (203ms)

✓ Should be able to manage tokens for proxyPool (341ms)

✓ Should manage tokens for accumulating pool (209ms)

✓ Should transfer external tokens from the new pools to the owner (46ms)

✓ Should manage tokens for failed pool (201ms)

✓ Should not allow transactions after the crowdsale is over

✓ Should let the beneficiary self-destruct the contract after 2 years (185ms)

6. All Contract Files Tested

File	Fingerprint (SHA256)
contracts/ERC20Generic.sol	461c55cf32ca771a84810c7d266ae3cf61121f0c9139d9c25bd9e7c1fada3ecb
contracts/infrastructure/authentication/whitelist/Whitelist.sol	1d8031d3a590994630d80014de60ac512e3a1b391fce18a80d450a73454bbaa0
contracts/infrastructure/behaviour/Observable.sol	73e35f93d0fd67dd045fe1bdd28d74e352e818d986e906df1898e20e2b2de768
contracts/infrastructure/dispatcher/Dispatchable.sol	f56687c67e569b0c07e8cc39cc4297a6f4a7aa07fb78c1ce9611412c8ccf81ac
contracts/infrastructure/dispatcher/SimpleDispatcher.sol	4fc7afa40e920e888e24152d9bb1717ee05972de0e8c3e9601152a806933625f
contracts/infrastructure/modifier/InputValidator.sol	d73361152f92d94cf0b30363ab892cf8ae533551ac287b93cfd6ea790bf6079d
contracts/infrastructure/ownership/MultiOwned.sol	199c2086344b0f1ef69a4028167ac1ec0cf13f1ff1075384e44f557a18049c22
contracts/infrastructure/ownership/Ownership.sol	eda033ed2df78b99b6b4bb169f973801cf0ade35b781a45382e98244034f191d
contracts/infrastructure/ownership/TransferableOwnership.sol	19845764245e86ed64182ea636d62150489558c8f2e2442623c2821d72c92526
contracts/source/MoxyOneCrowdsale.sol	165300a36115d4f92e8db98ebb56ebf508387f2242192d050389eec34cf3ce67
contracts/source/SpendToken.sol	8f3ecb3f340cbb0fd4795b76228bcb2ad04d6f4cac209bde7467282499907666
contracts/source/SpendTokenBurner.sol	0f441994e6cb59c87eda61160e0dcc29abc3a2024b485dbbe9c07c14fb75f4f3
contracts/source/crowdsale/Crowdsale.sol	914e06270844dc9c37a2430e6dd5b2aa2ee9073643cbe90f8f32e528cfa51458
contracts/source/crowdsale/proxy/CrowdsaleProxy.sol	9110cbf66d722940511a8706f06046f897cf687b6e89a456677c5edf13ecbf00
contracts/source/crowdsale/proxy/factory/CrowdsalePoolFactory.sol	32739a7e923ba89f6a681347edbf93db8bc621e380324124538f8765753849a4
contracts/source/crowdsale/proxy/factory/CrowdsaleProxyFactory.sol	19c14c2b5c5b5bbf95e58eb5b6d84f2f8275af2c11bd879eb178f6a7f4fcdb85

contracts/source/crowdsale/proxy/personal/PersonalCrowdsaleProxy.sol	ccae9d03a6a2f2f844538653ceaab19b8ac02c18acac60a87527518dd01c2a85
contracts/source/crowdsale/proxy/personal/PersonalCrowdsaleProxyDispatcher.sol	98d2f921417e3afc3cc28fef4dd1fd094297e992a9e0b037bc871f5a837ec772
contracts/source/crowdsale/proxy/pool/CrowdsalePool.sol	51ccbc3ba77fabf5a9b894d1bdfb26411b7d6e82c55deda97ed8ca2a77c369fb
contracts/source/crowdsale/proxy/pool/CrowdsalePoolAccumulator.sol	8d2b59b3917354c27a6f8218d01b005d9de010ab2dc747918cd0721fe5759498
contracts/source/crowdsale/proxy/pool/CrowdsalePoolProxy.sol	a8846d173b61cf5dc0ee2e88bdda81b6a50a49f693eaf10600b993be9dc290af
contracts/source/depository/Depository.sol	da8aa823e392bfe5227b8e72d332c8283c94cd427a000666434644addfedcfad
contracts/source/depository/DepositoryRecipient.sol	5a784912cae737fade4abc487677a2ccc2182190f093ae2c894b91f3b108ea53
contracts/source/token/ManagedToken.sol	118c97ff4565a6990a54cdf2a33b4e2eb436542996c18ea062103a8c5564d940
contracts/source/token/Token.sol	d35e821089e5c9a2a54dd2c4f6afe3a8882bdd18c791d846055ddde94fff591d
contracts/source/token/burner/TokenBurner.sol	758502fa66db059bceca4f933b4433c253968c1b66871771477l6d20e83d66f4
contracts/source/token/observer/TokenObserver.sol	6199172cf54913f23f4898fc49cfdfb80312782ce6a91501b55e90024e4dac60
contracts/source/token/retriever/TokenRetriever.sol	43c3dc8b9a586ca1230089df6107dfad5900a4067dad9438c0d7912212617cb6
contracts/test/artifacts/ProxyCrowdsale.sol	8dddd88ec0245b98548d52df3496250b8ea95b47142000d04aac4d41bc94ef20
contracts/test/artifacts/ProxyToken.sol	a5dcfb6f2969df6c2a9c2c64e798bccf368a64fc3335288faf4f8eacc0d11b05
contracts/test/artifacts/ProxyTokenBurner.sol	a1d6ef343242d1b1d152cfe4f057de4aa09358c0eb718332eb21e30a95eeaa5e

7. Individual File Coverage Report

File	% Statements	% Branches	% Functions	% Lines
contracts/ERC20Generic.sol	100.00%	100.00%	100.00%	100.00%
contracts/infrastructure/authentication/whitelist/Whitelist.sol	100.00%	100.00%	100.00%	100.00%
contracts/infrastructure/behaviour/Observable.sol	100.00%	100.00%	100.00%	100.00%
contracts/infrastructure/dispatcher/Dispatchable.sol	100.00%	100.00%	100.00%	100.00%
contracts/infrastructure/dispatcher/SimpleDispatcher.sol	100.00%	100.00%	100.00%	100.00%
contracts/infrastructure/modifier/InputValidator.sol	100.00%	50.00%	100.00%	100.00%
contracts/infrastructure/ownership/MultiOwned.sol	100.00%	100.00%	100.00%	100.00%
contracts/infrastructure/ownership/Ownership.sol	100.00%	100.00%	100.00%	100.00%
contracts/infrastructure/ownership/TransferableOwnership.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/MoxyOneCrowdsale.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/SpendToken.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/SpendTokenBurner.sol	100.00%	66.67%	100.00%	100.00%
contracts/source/crowdsale/Crowdsale.sol	90.37%	72.83%	90.00%	90.67%
contracts/source/crowdsale/proxy/CrowdsaleProxy.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/crowdsale/proxy/factory/CrowdsalePoolFactory.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/crowdsale/proxy/factory/CrowdsaleProxyFactory.sol	100.00%	100.00%	100.00%	100.00%

contracts/source/crowdsale/proxy/personal/PersonalCrowdsaleProxy.sol	100.00%	50.00%	90.00%	92.31%
contracts/source/crowdsale/proxy/personal/PersonalCrowdsaleProxyDispatcher.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/crowdsale/proxy/pool/CrowdsalePool.sol	68.00%	35.71%	75.00%	66.67%
contracts/source/crowdsale/proxy/pool/CrowdsalePoolAccumulator.sol	67.39%	38.24%	84.21%	71.70%
contracts/source/crowdsale/proxy/pool/CrowdsalePoolProxy.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/depository/Depository.sol	0.00%	100.00%	0.00%	0.00%
contracts/source/depository/DepositoryRecipient.sol	0.00%	100.00%	0.00%	0.00%
contracts/source/token/ManagedToken.sol	90.91%	62.50%	90.00%	91.30%
contracts/source/token/Token.sol	100.00%	80.00%	100.00%	100.00%
contracts/source/token/burner/TokenBurner.sol	100.00%	50.00%	100.00%	100.00%
contracts/source/token/observer/TokenObserver.sol	100.00%	100.00%	100.00%	100.00%
contracts/source/token/retriever/TokenRetriever.sol	100.00%	100.00%	100.00%	100.00%
contracts/test/artifacts/ProxyCrowdsale.sol	100.00%	100.00%	0.00%	100.00%
contracts/test/artifacts/ProxyToken.sol	100.00%	100.00%	0.00%	100.00%
contracts/test/artifacts/ProxyTokenBurner.sol	100.00%	100.00%	0.00%	100.00%
All files	**92.31%**	**71.43%**	**89.56%**	**92.24%**