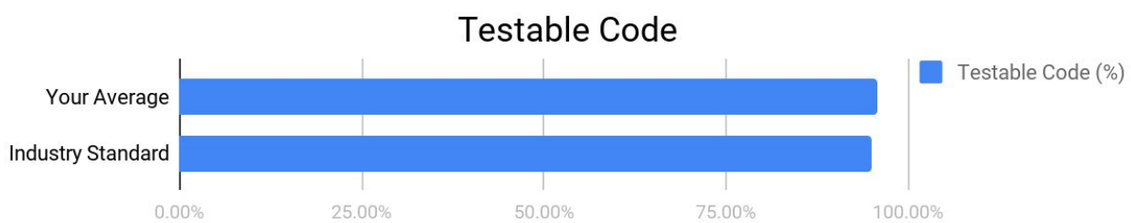
Eximchain Token Contract Audit

by Hosho, February 2018

Executive Summary

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

There remains a single suggestion to be properly implemented. See Complete Analysis.



The testable code is on par with industry standard. 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 Eximchain 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 **3**

2. Structure Analysis and Test Results **4**

2.1. Summary 4

2.2 Coverage Report 4

2.3 Failing Tests 4

3. Complete Analysis **5**

7.1. Resolved, Medium: Event Missing 5

Explanation 5

Resolution 5

7.2. Resolved, Low: Question Regarding Transfer to 0x0 5

Explanation 5

Resolution 5

7.3. Resolved, Low: Non-Needed Checks 6

Explanation 6

Resolution 6

7.4. Unresolved, Informational: Lack of Token Trap/Escape 6

Explanation 6

Update 6

4. Closing Statement **7**

5. Test Suite Results **8**

6. All Contract Files Tested **10**

7. Individual File Coverage Report **12**

1. Auditing Strategy and Techniques Applied

The Hosho Team has performed an initial review of the code on December 7, 2017 and completed a thorough follow-up review of the smart contract code as written and last updated on January 20, 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 Eximchain'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.

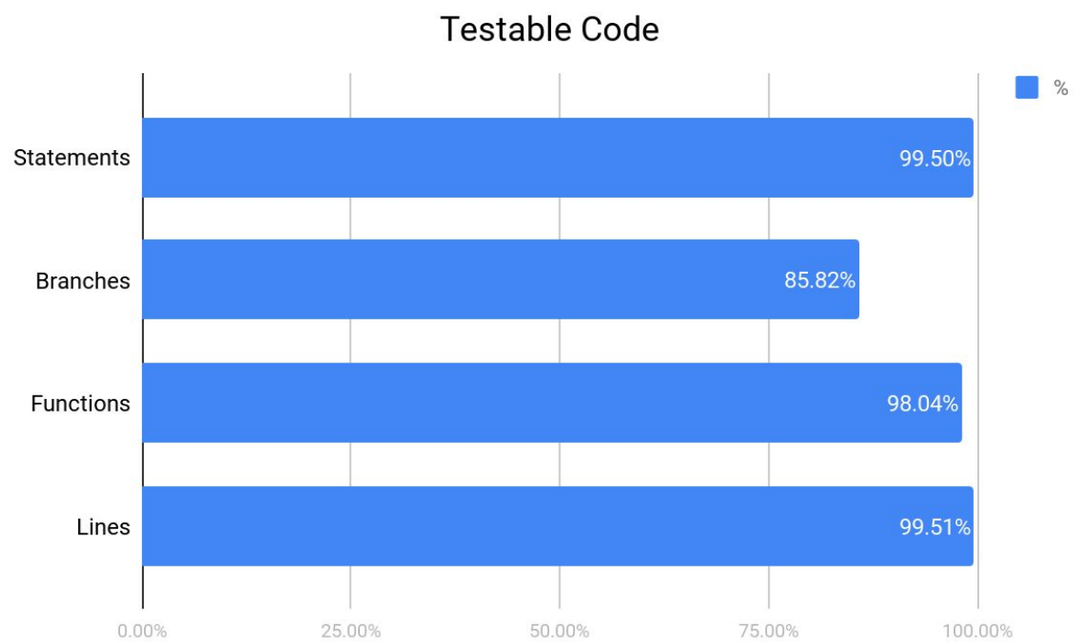
2. Structure Analysis and Test Results

2.1. Summary

The Eximchain Token and associated Token Sale contracts are well written and properly functioning contracts. The low level issues discovered have been verified as intentional by the Eximchain team. Both of these contracts do not contain major issues and are free of code that functions contrary to what is documented. There remains only the single suggestion of a token trap that was implemented but incorrectly. The relatively low level of coverage over branches is due to the number of non-executable checks which are caused by the lockouts from other portions of the contract, in particular, deployments and whitelisting.

2.2 Coverage Report

As part of our work assisting Eximchain 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

- Contract: ERC-20 Tests for EximchainToken. Should not transfer tokens to 0x0 (See Issue 7.2.)
- Contract: Extended tests for EximchainToken. Should be able to transfer away ERC-20 tokens: (See Issue 7.4)

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, Medium: Event Missing

EximchainToken.sol

Explanation

All token transfer occurrences require a transfer event. In the case of the burn functionality starting on line 32, there is no transfer event issued.

Resolution

In discussions with the Eximchain Team, they have verified that this is intended. Due to this, we are able to mark this as passing since it is not causing the contract to break or operate in a manner different than intended.

7.2. Resolved, Low: Question Regarding Transfer to 0x0

ERC20Token.sol

Explanation

On line 38 there is a Transfer that occurs to address 0x0. Given that the contract contains a dedicated burn function for this purpose, is there a reason for keeping this transfer ability?

Resolution

This is as intended per discussions with the Eximchain Team.

7.3. Resolved, Low: Non-Needed Checks

FlexibleTokenSale.sol

Explanation

Based on the design of the whitelist, none of the addresses that are being validated against have the ability to send or receive tokens and ETH for purchase, rendering the following checks unnecessary:

```
require(_beneficiary != address(0));

require(_beneficiary != address(this));

require(_beneficiary != address(token));

require(msg.sender != address(walletAddress));
```

Unnecessary checks cause code clutter, rendering the codebase less readable and creates code paths that are more difficult to follow.

Resolution

The Eximchain Team has opted to keep these in place.

7.4. Unresolved, Informational: Lack of Token Trap/Escape

EximchainToken.sol

Explanation

The Hosho team suggests adding an escape function for trapped tokens that are not issued by the contract. There are an increasing number of ERC-20 and ERC-223 tokens getting trapped forever in contracts, so it is valuable to have a function that can return these tokens to a contract issuer or owner for refund.

Update

The Eximchain Team added code for this, but it is incorrect. This should be calling to a remote ERC-20 token and executing the transfer() function on the remote token rather than on the local contract as is currently implemented.

---

4. Closing Statement

We are grateful to have been given the opportunity to work with the Eximchain Team.

The Eximchain Token and associated Token Sale contracts are well written and properly functioning contracts. The low level issues discovered have been verified as intentional by the Eximchain team. Both of these contracts are free of major issues and code that functions contrary to what is documented. There remains only the single suggestion of a token trap that was implemented but incorrectly.

As a small team of experts, having backgrounds in all aspects of blockchain, cryptography, and cybersecurity, we can say with confidence that the Eximchain 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 Eximchain 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: ERC-20 Tests for EximchainToken

✓ Should deploy a token with the proper configuration (118ms)

✓ Should allocate tokens per the minting function, and validate balances (234ms)

✓ Should transfer tokens from 0xd86543882b609b1791d39e77f0efc748dfff7dff to 0x42adbad92ed3e86db13e4f6380223f36df9980ef (67ms)

✓ Should not transfer negative token amounts (44ms)

✓ Should not transfer more tokens than you have (49ms)

✓ Should allow 0xa3883a50d7d537cec8f9bad8e8404aa8ff3078f3 to authorize 0x341106cb00828c87cd3ac0de55eda7255e04933f to transfer 1000 tokens

✓ Should allow 0xa3883a50d7d537cec8f9bad8e8404aa8ff3078f3 to zero out the 0x341106cb00828c87cd3ac0de55eda7255e04933f authorization

✓ Should allow 0x667632a620d245b062c0c83c9749c9bfadf84e3b to authorize 0x53353ef6da4bbb18d242b53a17f7a976265878d5 for 1000 token spend, and 0x53353ef6da4bbb18d242b53a17f7a976265878d5 should be able to send these tokens to 0x341106cb00828c87cd3ac0de55eda7255e04933f (128ms)

✓ Should not allow 0x53353ef6da4bbb18d242b53a17f7a976265878d5 to transfer negative tokens from 0x667632a620d245b062c0c83c9749c9bfadf84e3b (55ms)

✓ Should not allow 0x53353ef6da4bbb18d242b53a17f7a976265878d5 to transfer tokens from 0x667632a620d245b062c0c83c9749c9bfadf84e3b to 0x0 (39ms)

X Should not transfer tokens to 0x0

✓ Should not allow 0x53353ef6da4bbb18d242b53a17f7a976265878d5 to transfer more tokens than authorized from 0x667632a620d245b062c0c83c9749c9bfadf84e3b (39ms)

Contract: Extended tests for EximToken

✓ Should allow funds transfers back to the owner, before the token is finalized (103ms)

✓ Should not allow any user <-> user funds transfers before finalization (89ms)

✓ Should allow a proper transfer of ownership, blocking 0x0 and the contract itself to become possible owners (194ms)

✓ Should allow the ops address to be set properly by the owner (158ms)

✓ Should allow tokens to be burned at any time, even before finalization (50ms)

✓ Should only be able to be finalized once (58ms)

✓ Should allow tokens to be burned at any time (138ms)

X Should be able to transfer away ERC-20 tokens

✓ Should let the owner freeze the tokens (75ms)

Contract: Configuration/Management tests for EximchainToken

✓ Should allow the contract to be initialized by a proper token (266ms)

✓ Should allow the wallet address to be changed by the owner, or the operator (isOwnerOrOps testing) (175ms)

✓ Should allow the owner to configure the contract as needed (451ms)

✓ Should allow the contracts to be suspended and resumed (141ms)

✓ Should allow the owner and operators to setup whitelists via updateWhitelist and updateWhitelistBatch (420ms)

Contract: Token purchase tests for EximchainToken

✓ Should not allow purchase until minimum requirements are met, including that the sender and receiver are whitelisted (1601ms)

✓ Should cleanly handle token reclaim (172ms)

6. All Contract Files Tested

Original Files December 7, 2017

File	Fingerprint (SHA256)
contracts/EximchainToken.sol	2ddf0dc3380a40111bcb2fc420d29d8194239171c63ec878248593701e80eaef
contracts/EximchainTokenConfig.sol	4cb2045ae587e5bdf461e5c090dc420b7ea5e2f53bf64d8ea5a0af5fb29eefc0
contracts/EximchainTokenSale.sol	42597048bbaedb05c03b7f908480d5575644ebd5037155c28098cb78e6bcc8eb
contracts/EximchainTokenSaleConfig.sol	b83935aaf69961eca81e10984ba6d472c0ea5d10d2b61f25217d7c69c46a4ff3
contracts/Enuma/ERC20Interface.sol	616ca49fc0aabde267feb1a2d5ff8ec01fd97d2c4d146bcf8c01df35599d0ea1
contracts/Enuma/ERC20Token.sol	c9e5c0beb8b966b7809a0640c388b528deadd9668544d67fd20b615b74e0cc7c
contracts/Enuma/Finalizable.sol	8acccc5d0b5439008f406fb3c62dacea3c544417dac3be9993e747b4d728d620
contracts/Enuma/FinalizableToken.sol	b7076acc9c0fb6b04149914b0d20d688f83d8fc3c1a8ada7c2f23e8fa290bf54
contracts/Enuma/FlexibleTokenSale.sol	4af346d07ccbb0afcb7917cdc95d7c931ef4f1048cd968055aec7e7855149e52
contracts/Enuma/Math.sol	8fbe832fb320fe84d96a3ffa88d50e714916d7808b8147a26e29cbaa1dc128d7
contracts/Enuma/OpsManaged.sol	d473fcb6e820be2e3734a68f3f229f93200098b63e0867d919ab8c101eeb2f51
contracts/Enuma/Owned.sol	a974fe0f2bc4fe3d8ac0404e50f0ee6cbb519dfd766f4ee663fc9f79aa9c62de

Updated Files January 20, 2018

File	Fingerprint (SHA256)
contracts/EximchainToken.sol	638e80dc82e86c97ecfe999a14a819ee5e562ea13652de36cad62a324b46fe0b
contracts/EximchainTokenConfig.sol	4cb2045ae587e5bdf461e5c090dc420b7ea5e2f53bf64d8ea5a0af5fb29eefc0
contracts/EximchainTokenSale.sol	42597048bbaedb05c03b7f908480d5575644ebd5037155c28098cb78e6bcc8eb
contracts/EximchainTokenSaleConfig.sol	b83935aaf69961eca81e10984ba6d472c0ea5d10d2b61f25217d7c69c46a4ff3
contracts/Enuma/ERC20Interface.sol	616ca49fc0aabde267feb1a2d5ff8ec01fd97d2c4d146bcf8c01df35599d0ea1

contracts/Enuma/ERC20Token.sol	c9e5c0beb8b966b7809a0640c388b528deadd9668544d67fd20b615b74e0cc7c
contracts/Enuma/Finalizable.sol	8acccc5d0b5439008f406fb3c62dacea3c544417dac3be9993e747b4d728d620
contracts/Enuma/FinalizableToken.sol	b7076acc9c0fb6b04149914b0d20d688f83d8fc3c1a8ada7c2f23e8fa290bf54
contracts/Enuma/FlexibleTokenSale.sol	4af346d07ccbb0afcb7917cdc95d7c931ef4f1048cd968055aec7e7855149e52
contracts/Enuma/Math.sol	2b2ec25bdb355ff0d26ad96aba6131a8a38ebe1af5fa1c3a7679d78338c9f280
contracts/Enuma/OpsManaged.sol	d473fcb6e820be2e3734a68f3f229f93200098b63e0867d919ab8c101eeb2f51
contracts/Enuma/Owned.sol	a974fe0f2bc4fe3d8ac0404e50f0ee6cbb519dfd766f4ee663fc9f79aa9c62de

7. Individual File Coverage Report

Original Files December 7, 2017

File	% Statements	% Branches	% Functions	% Lines
contracts/EximchainToken.sol	100.00%	100.00%	100.00%	100.00%
contracts/EximchainTokenConfig.sol	100.00%	100.00%	100.00%	100.00%
contracts/EximchainTokenSale.sol	100.00%	100.00%	100.00%	100.00%
contracts/EximchainTokenSaleConfig.sol	100.00%	100.00%	100.00%	100.00%
contracts/Enuma/ERC20Interface.sol	100.00%	100.00%	100.00%	100.00%
contracts/Enuma/ERC20Token.sol	100.00%	100.00%	100.00%	100.00%
contracts/Enuma/Finalizable.sol	100.00%	100.00%	100.00%	100.00%
contracts/Enuma/FinalizableToken.sol	100.00%	100.00%	100.00%	100.00%
contracts/Enuma/FlexibleTokenSale.sol	99.04%	80.49%	92.86%	99.06%
contracts/Enuma/Math.sol	100.00%	66.67%	100.00%	100.00%
contracts/Enuma/OpsManaged.sol	100.00%	100.00%	100.00%	100.00%
contracts/Enuma/Owned.sol	100.00%	90.00%	100.00%	100.00%
All files	**99.50%**	**85.82%**	**98.04%**	**99.51%**

Updated Files January 20, 2018

File	% Statements	% Branches	% Functions	% Lines
contracts/EximchainToken.sol	100.00%	100.00%	100.00%	100.00%
contracts/EximchainTokenConfig.sol	100.00%	100.00%	100.00%	100.00%
contracts/EximchainTokenSale.sol	100.00%	100.00%	100.00%	100.00%
contracts/EximchainTokenSaleConfig.sol	100.00%	100.00%	100.00%	100.00%
contracts/Enuma/ERC20Interface.sol	100.00%	100.00%	100.00%	100.00%

contracts/Enuma/ERC20Token.sol	100.00%	100.00%	100.00%	100.00%
contracts/Enuma/Finalizable.sol	100.00%	100.00%	100.00%	100.00%
contracts/Enuma/FinalizableToken.sol	100.00%	100.00%	100.00%	100.00%
contracts/Enuma/FlexibleTokenSale.sol	99.04%	80.49%	92.86%	99.06%
contracts/Enuma/Math.sol	100.00%	66.67%	100.00%	100.00%
contracts/Enuma/OpsManaged.sol	100.00%	100.00%	100.00%	100.00%
contracts/Enuma/Owned.sol	100.00%	90.00%	100.00%	100.00%
All files	**99.50%**	**85.82%**	**98.04%**	**99.51%**