DCORP Token Contract Audit

by Hosho, October 2017

Table of Contents

Technical Summary

This document outlines the overall security of DCORP’s smart contract as evaluated by Hosho’s Smart Contract auditing team.

The scope of this audit was to analyze and document DCORP’s token contract codebase for quality, security, and correctness.

It should be noted that this audit is not an endorsement of the reliability or effectiveness of the contract, 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, we at Hosho recommend that the DCORP Team put in place a bug bounty program to encourage further and active analysis of the smart contract.

Auditing Strategy and Techniques Applied

The Hosho Team has performed a thorough review of the smart contract code as written and last updated on October 24, 2017. The following contract files and their respective SHA256 fingerprints were evaluated with the files altered from the previous audit are :

File	Fingerprint (SHA256)
infrastructure/authentication/IAuthenticationManager.sol	aa5bca0d4d3a1b1d89eb5c463481c69ba1e8ac25052308383c426081f398dc30
infrastructure/authentication/IAuthenticator.sol	035d60a7475d98b338c2ce4bdaf3c6f836432bd9a9d3994ddc59da4b4b4cedef
infrastructure/authentication/whitelist/IWhitelist.sol	374711d023619ab87eaf9b7befdee31a5c868aad03d732fc2c730cb161303c39
infrastructure/authentication/whitelist/Whitelist.sol	61dc564f663b08a3c71e9322c9fc5d4b39d89de896cf1036667d0d3bbfc4f837
infrastructure/behaviour/IObservable.sol	1fc3cf1ba8685f18b68801edc9d5c3887630c659ac6d97a80e871bc74ca673ef
infrastructure/behaviour/Observable.sol	924088fce2cccddd123c266f74c0b7ef0470df4c53b60656aa57b6928770bd67
infrastructure/modifier/InputValidator.sol	e9bca32a2fa6f62a1df0513ab5798320dfee9c6f3dae3ecc59556e24d41a6a0f
infrastructure/ownership/IMultiOwned.sol	68267d72649cea1fa57a892e2a0b6674f120d4bb3256330049c6f1c16009717a
infrastructure/ownership/MultiOwned.sol	122fee0e212376d872366810c3f81b0ed324500c3ee04dc7cff670b51bfc195d

infrastructure/ownership/IOwnership.sol	f804c1d4e333ef06fcc94f2745561039fbd5d659e16dc743e2b6d1b7eab85a6d
infrastructure/ownership/ITransferableOwnership.sol	b3e1551fb5392552ca57d9acc2d5a15eefc03315ee6b68492ca88ec4b8875d36
infrastructure/ownership/Ownership.sol	a82a3080498a66b66ff3f79ff96bfdd9883a8eb141faebfe8138f037031433e5
infrastructure/ownership/TransferableOwnership.sol	70817ebee945513eff45bfdacf7cb21934c11d51d6e2580f7e46f5d31c8a608f
infrastructure/state/IPausable.sol	f4e9fcf993f2266178034e5ef3e920e5c5f05edf09f804a5dadeb14f7c91ccb8
source/DRPSToken.sol	0df527ce12cf68c1ab318f0c6ca01ccdab864c3ccafaa384dd416bb3c5c39efe
source/DRPSTokenConverter.sol	e329992b918e1742dae3792eb58dc7bdd145a9e6a66be24c1f5459de73ccc1f5
source/DRPTokenChanger.sol	3a76443f6eba112e20b7cec7c667d0949d2275293865fe207b21f4d38235e805
source/DRPUToken.sol	93249b3f21f33dd47f3b6b36b081121cd27ae7a9b0f557f1c65c27f7f14fd72e
source/DRPUTokenConverter.sol	e9288ece7b26369f41731de3d36b6920f87f0b8d757f1e9dd176072dec17922c
source/DcorpProxy.sol	38ae3bf804b3243ed5483cb16ee5f295263e25c200c8fb2244ced4cfe07df6a6
source/token/IManagedToken.sol	e91e6efafeec796afb7f0b322af86b3ce7bc741e4ce23db351ecbaa66f039e6c
source/token/IToken.sol	ded577cfcfc70f27e9a3667ef5e2d99ecad68cd70f5767cee7f2eb8323678478
source/token/ManagedToken.sol	c95242f9f68692f122d6120aec331735f1821c421cf2a5c1a022111d0b0eafa9
source/token/Token.sol	ef2925d14ecaeae8fef55783ccc62e4422ae8126a96c96bbae0c9145b7e3300c
source/token/changer/ITokenChanger.sol	ee34526c3ce81c5d5f326fbd1a9841591f4ef91eb202dfcbbc3c8b3173b76c3b
source/token/changer/TokenChanger.sol	73abac16c86a1487fb0d2af9fee078f3c7e7c94cccef62564519ffe32b4e0078
source/token/observer/ITokenObserver.sol	cdebd8e12768bc6a29f05cc54c936d99c0d8809dc017430fdaf1bab9c01cbc95
source/token/observer/TokenObserver.sol	0cbed0e23ab6af26ee8471911dfccc92c66a5b791be87fa87ccfaf9ace6264af
source/token/retriever/ITokenRetriever.sol	21772c2180f2c6079ebc119b5d231c02a16bef5b4e8af03e4692514f7ac851c0
source/token/retriever/TokenRetriever.sol	05c0f48e52fe76009b42686fe328d42c012f32244699d09911c35e1268c257c6

After the initial audit, the following contract files were updated by DCORP and an additional full audit was completed on October 31, 2017:

infrastructure/behaviour/Observable.sol	`490211d4b8bc9c2a85472be83422558f20a4c62152499e5ec983651e8700c962`
infrastructure/ownership/MultiOwned.sol	`1f0a68c8dd2a800f38979aba1e57d5a0ddc46ab63658bbe4df9f38f7e5d44b0f`
source/token/changer/TokenChanger.sol	`b5b7e117a28b042299937665a3c99c90e1bda52057104dd4617fbfc842f8e865`
source/DRPTokenChanger.sol	`3ca5d572d875e4d34562f938dcb64ba12d9aa5d10e8f7aed767f1154bfd94f54`
source/DcorpProxy.sol	`cb24e8e737aa9cd817599f1c7467d609ca35c2cd52f8bac96d125f77e2955e57`

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

- Implements and adheres to existing ERC20 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; and
- Uses methods safe from reentrance attacks.

The Hosho Team has followed best practices and industry-standard techniques to verify the implementation of DCORP's token contract. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as they were discovered. Part of this work included 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.
5. Deploying the smart contract to testnet and production networks using multiple client implementations to run live tests.

Contract Analysis and Test Results

Summary

The DCORP token is a compliant ERC-20 Token with additional functionality added to adhere to the rules structured for their crowdsale.

Our follow up analysis showed that the DCORP team has successfully fixed the issues that we found in our original analysis.

The only small concern uncovered by the Hosho team could be in the `drpCrowdsaleRecordedBalance` system, where in order to load the `DCorpProxy` into a deployed state, the funds on the contract must be greater than or equal to the value in `drpCrowdsaleRecordedBalance`, which is the balance of the DRP Crowdsale at the time of the `init`. This should not be an issue. However, it may be worth adding a way that a small amount of additional funds could be added in case of any critical issue with the transfer from the DRP Crowdsale contract.

Aside from this, the Hosho team is pleased with the technical aspects of this contract and believe that these contracts are well written.

Coverage Report

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

Uncovered lines/branches are primarily extreme bounds checks that can not be hit in the live contract and for expediency were skipped. The `DCorpProxy` contract lines not covered include the `not_accepted_token modifier`, the calling function `retrieveTokens`, and the token transfer fail to revert calls. These all work properly in other test cases, and did not need to be covered, as they're all `super` calls.

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

File	% Statements	% Branches	% Functions	% Lines
infrastructure/authentication/IAuthenticationManager.sol	100.00%	100.00%	100.00%	100.00%
infrastructure/authentication/IAuthenticator.sol	100.00%	100.00%	100.00%	100.00%
infrastructure/authentication/whitelist/IWhitelist.sol	100.00%	100.00%	100.00%	100.00%
infrastructure/authentication/whitelist/Whitelist.sol	100.00%	100.00%	100.00%	100.00%
infrastructure/behaviour/IObservable.sol	100.00%	100.00%	100.00%	100.00%

infrastructure/behaviour/Observable.sol	100.00%	100.00%	100.00%	100.00%
infrastructure/modifier/InputValidator.sol	100.00%	50.00%	100.00%	100.00%
infrastructure/ownership/IMultiOwned.sol	100.00%	100.00%	100.00%	100.00%
infrastructure/ownership/MultiOwned.sol	100.00%	100.00%	100.00%	100.00%
infrastructure/ownership/IOwnership.sol	100.00%	100.00%	100.00%	100.00%
infrastructure/ownership/ITransferableOwnership.sol	100.00%	100.00%	100.00%	100.00%
infrastructure/ownership/Ownership.sol	100.00%	100.00%	100.00%	100.00%
infrastructure/ownership/TransferableOwnership.sol	100.00%	100.00%	100.00%	100.00%
infrastructure/state/IPausable.sol	100.00%	100.00%	100.00%	100.00%
source/DRPSToken.sol	100.00%	100.00%	100.00%	100.00%
source/DRPSTokenConverter.sol	100.00%	100.00%	100.00%	100.00%
source/DRPTokenChanger.sol	100.00%	50.00%	100.00%	100.00%
source/DRPUToken.sol	100.00%	100.00%	100.00%	100.00%
source/DRPUTokenConverter.sol	100.00%	100.00%	100.00%	100.00%
source/DcorpProxy.sol	96.43%	75%	90.91%	95.04%
source/token/IManagedToken.sol	100.00%	100.00%	100.00%	100.00%
source/token/IToken.sol	100.00%	100.00%	100.00%	100.00%
source/token/ManagedToken.sol	100.00%	75.00%	100.00%	100.00%
source/token/Token.sol	100.00%	70.00%	100.00%	100.00%
source/token/changer/ITokenChanger.sol	100.00%	100.00%	100.00%	100.00%
source/token/changer/TokenChanger.sol	100.00%	75%	100.00%	100.00%
source/token/observer/ITokenObserver.sol	100.00%	100.00%	100.00%	100.00%
source/token/observer/TokenObserver.sol	100.00%	100.00%	100.00%	100.00%

source/token/retriever/ITokenRetriever.sol	100.00%	100.00%	100.00%	100.00%
source/token/retriever/TokenRetriever.sol	100.00%	100.00%	100.00%	100.00%

Test Results

Contract: ERC-20 Compliant Token

✓ Should deploy with DRP Security as the name of the token (60ms)

✓ Should deploy with DRPS as the symbol of the token (39ms)

✓ Should deploy with 8 decimals

✓ Should deploy with 0 tokens (40ms)

✓ Should allocate tokens per the minting function, and validate balances (318ms)

✓ Should transfer tokens from 0x1bbb1269032bfd0b0fe0851235fc798af6bd3c9b to 0x42adbad92ed3e86db13e4f6380223f36df9980ef (138ms)

✓ Should not transfer negative token amounts (53ms)

✓ Should not transfer more tokens than you have (48ms)

✓ Should allow 0x3b44fa9f7511113a8c1a1528070d45b1d7cdd101 to authorize 0x341106cb00828c87cd3ac0de55eda7255e04933f to transfer 1000 tokens (85ms)

✓ Should not allow 0x3b44fa9f7511113a8c1a1528070d45b1d7cdd101 to authorize 0x341106cb00828c87cd3ac0de55eda7255e04933f to transfer an additional 1000 tokens once authorized, and authorization balance is > 0 (58ms)

✓ Should allow 0x3b44fa9f7511113a8c1a1528070d45b1d7cdd101 to zero out the 0x341106cb00828c87cd3ac0de55eda7255e04933f authorization (103ms)

✓ Should allow 0xdaef8d8c30eeb858b8c774a8d7d5e92a552bb0d9 to authorize 0x53353ef6da4bbb18d242b53a17f7a976265878d5 for 1000 token spend, and 0x53353ef6da4bbb18d242b53a17f7a976265878d5 should be able to send these tokens to 0x341106cb00828c87cd3ac0de55eda7255e04933f (386ms)

✓ Should not allow 0x53353ef6da4bbb18d242b53a17f7a976265878d5 to transfer negative tokens from 0xdaef8d8c30eeb858b8c774a8d7d5e92a552bb0d9 (62ms)

✓ Should allow the token owner to retrieve tokens (337ms)

✓ Should not accept ETH

Contract: ERC-20 Compliant Token

✓ Should deploy with DRP Utility as the name of the token

✓ Should deploy with DRPU 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 (347ms)

```
    ✓ Should transfer tokens from 0x1bbb1269032bfd0b0fe0851235fc798af6bd3c9b to
0x42adbad92ed3e86db13e4f6380223f36df9980ef (137ms)

    ✓ Should not transfer negative token amounts (50ms)

    ✓ Should not transfer more tokens than you have (52ms)

    ✓ Should allow 0x3b44fa9f7511113a8c1a1528070d45b1d7cdd101 to authorize
0x341106cb00828c87cd3ac0de55eda7255e04933f to transfer 1000 tokens (75ms)

    ✓ Should not allow 0x3b44fa9f7511113a8c1a1528070d45b1d7cdd101 to authorize
0x341106cb00828c87cd3ac0de55eda7255e04933f to transfer an additional 1000 tokens once
authorized, and authorization balance is > 0 (48ms)

    ✓ Should allow 0x3b44fa9f7511113a8c1a1528070d45b1d7cdd101 to zero out the
0x341106cb00828c87cd3ac0de55eda7255e04933f authorization (79ms)

    ✓ Should allow 0xdaef8d8c30eeb858b8c774a8d7d5e92a552bb0d9 to authorize
0x53353ef6da4bbb18d242b53a17f7a976265878d5 for 1000 token spend, and
0x53353ef6da4bbb18d242b53a17f7a976265878d5 should be able to send these tokens to
0x341106cb00828c87cd3ac0de55eda7255e04933f (278ms)

    ✓ Should not allow 0x53353ef6da4bbb18d242b53a17f7a976265878d5 to transfer negative
tokens from 0xdaef8d8c30eeb858b8c774a8d7d5e92a552bb0d9 (47ms)

    ✓ Should allow the token owner to retrieve tokens (241ms)

    ✓ Should not accept ETH


  Contract: ERC-20 Compliant Token

    ✓ Should allocate tokens per the minting function, and validate balances (443ms)

    ✓ Should burn tokens from an owned account as expected (127ms)

    ✓ Should not burn more tokens than you have (48ms)

    ✓ Should allow the token to be locked by an owner (68ms)

    ✓ Should not allow the token to be unlocked by a non-owner

    ✓ Should disallow transfers while locked

    ✓ Should allow the token to be unlocked by an owner (96ms)

    ✓ Should not allow the token to be locked by a non-owner


  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 (227ms)

    ✓ Should allow an owner to remove an observer, but only on the contract it's for - DRPS
(250ms)

    ✓ Should allow an owner to remove an observer, but only on the contract it's for (DRPU)
(240ms)

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

✓ Should not allow double adding/removing an observer (262ms)

Contract: Ownership

✓ Should return if someone is an owner or not - DRPS (161ms)

✓ Should return if someone is an owner or not - DRPU (71ms)

✓ Should let an owner add a new owner - DRPU (180ms)

✓ Should allow an owner to be removed - DRPU (123ms)

✓ Should allow not an owner to be removed twice - DRPU (78ms)

✓ Should let the owner of a contract transfer ownership, then deny non-owner - DRPS Converter (75ms)

✓ Should return the current owner of a contract - DRPS Converter (50ms)

Contract: DRP Whitelist

✓ Should allow someone to be added to the whitelist (167ms)

✓ Should allow someone to be verified against the whitelist

✓ Should not allow a non-owner to whitelist someone

✓ Should allow someone to be removed from to the whitelist (85ms)

✓ Should allow someone to be removed from to the whitelist even if they aren't on it (69ms)

✓ Should not allow a non-owner to remove a whitelist

✓ Should allow someone to be re-added to the whitelist (133ms)

Contract: DRP->DRPS Token Changer

✓ Should initialize with DRP as the left, and DRPS as the right (74ms)

✓ Should initialize with the correct settings for DRP -> DRPS (115ms)

✓ Should only let the owner pause and unpause

✓ Should only let the owner enable and disable authentication

✓ Should let the owner pause and unpause (120ms)

✓ Should let the owner enable and disable authentication (181ms)

✓ Should not accept ETH

✓ Should not allow anyone but the owner to retrieve tokens

✓ Should allow the owner to retrieve tokens (532ms)

✓ Should not allow the owner to retrieve left side tokens

✓ Should not allow a 0 token conversion

✓ Should not allow a conversion if authentication is required, and the account is not whitelisted (70ms)

✓ Should not transfer if there's no authorization on the token (103ms)

✓ Should not transfer if there's no authorization on the token (106ms)

✓ Should not transfer if there is a pause on the token (332ms)

✓ Should transfer tokens properly (449ms)

✓ Should not be able to transfer the left token back to the owner via retrieve (45ms)

Contract: DRP->DRPU Token Changer

✓ Should initialize with DRP as the left, and DRPU as the right (102ms)

✓ Should initialize with the correct settings for DRP -> DRPU (130ms)

✓ Should only let the owner pause and unpause

✓ Should only let the owner enable and disable authentication

✓ Should let the owner pause and unpause (136ms)

✓ Should let the owner enable and disable authentication (104ms)

✓ Should not accept ETH

✓ Should not allow anyone but the owner to retrieve tokens

✓ Should allow the owner to retrieve tokens (352ms)

✓ Should not allow the owner to retrieve left side tokens

✓ Should not allow a 0 token conversion

✓ Should not allow a conversion if authentication is required, and the account is not whitelisted (93ms)

✓ Should not transfer if there's no authorization on the token (108ms)

✓ Should not transfer if there's no authorization on the token (100ms)

✓ Should not transfer if there is a pause on the token (301ms)

✓ Should transfer tokens properly (407ms)

✓ Should not be able to transfer the left token back to the owner via retrieve

Contract: DRPS<->DRPU Token Changer

✓ Should initialize with DRPS as the left, and DRPU as the right (69ms)

✓ Should initialize with the correct settings for DRPS <-> DRPU (112ms)

✓ Should only let the owner pause and unpause

✓ Should let the owner pause and unpause (129ms)

✓ Should not accept ETH

✓ Should not allow anyone but the owner to retrieve tokens

✓ Should allow the owner to retrieve tokens (395ms)

✓ Should transfer tokens properly, DRPS -> DRPU (551ms)

✓ Should transfer tokens properly, DRPU -> DRPS (485ms)

✓ Should transferFrom tokens properly, DRPS -> DRPU (586ms)

✓ Should transferFrom tokens properly, DRPU -> DRPS (582ms)

Contract: DCorp Proxy

✓ Should start at the deploying stage (1627ms)

✓ Should only allow the DRPCrowdsale to send funds to the proxy

First transfer made

✓ Should allow the DRPCrowdsale to send funds to the proxy, and send it into the deployed state (305ms)

✓ Should not allow DRP Crowdsale to transfer funds to the proxy after deployment

✓ Should return the balances in the contract as 0 before anything is sent. Unknown addresses should return 0 (130ms)

✓ Should accept DRPU and DRPS and insert their balances appropriately, then allow the checking of the balances (431ms)

✓ Should let you withdraw DRPS and DRPU from the contract (830ms)

✓ Should not let you withdraw more DRPS than you have in the contract

✓ Should not let you withdraw more DRPU than you have in the contract

✓ Should not let you withdraw DRPS when locked (196ms)

✓ Should not let you withdraw more DRPU when locked (174ms)

✓ Should let you create a proposal, vote on it, attempting to execute before 7 days should fail. (3289ms)

✓ Should, after 7 days allow you to attempt to execute, on fail, block and time to start a new one (664ms)

Structure and Organization of 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:

- **Informational** - The issue has no impact on the contract's ability to operate.
- **Low** - The issue has minimal impact on the contract's ability to operate.
- **Medium** - The issue affects the ability of the contract to operate in a way that doesn't significantly hinder its behavior.
- **High** - The issue affects the ability of the contract to compile or operate in a significant way.
- **Critical** - The issue affects the contract in such a way that funds may be lost, allocated incorrectly, or otherwise result in a significant loss.

Complete Analysis

Resolved, Medium: No Protection To Require Owner

source/DRPSTokenConverter.sol; source/DRPUTokenConverter.sol

Explanation

The `setRate`, `setFee`, and `setPrecision`, do not have protection for requiring an owner.

Resolution

An `only_owner` modifier was added to these functions.

Informational: Potential Deployment Issue

`drpCrowdsaleRecordedBalance` system

Explanation

While this should not pose an issue during operation, in order to load the `DCorpProxy` into a deployed state, the funds on the contract must be greater than or equal to the value in `drpCrowdsaleRecordedBalance`, which is the balance of the DRP Crowdsale at the time of the init.

Suggested Resolution

Adding a way that a small amount of additional funds could be added in case of any critical issue with the transfer from the DRP Crowdsale contract.

Closing Statement

We are grateful to have been given the opportunity to work with the DCORP team and Frank Bonnet, the Solidity developer for their contracts.

The Hosho Team is pleased to state that this is a well-written contract. The DCORP team and Frank Bonnet have shown themselves to be forward-thinking in this new space and we look forward to working with them in the future.

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

YoSub Kwon